**UNIVERSIDAD MIGUEL HERNÁNDEZ**
**Programa de Doctorado en Tecnologías Industriales y de Telecomunicación**


# Creation of Hybrid Hierarchical Models by Using Omnidirectional Vision and Machine Learning Techniques.

**Autor:** Sergio Cebollada López
**Director:** Dr. Ing. Óscar Reinoso García
**Codirector:** Dr. Ing. Luis Payá Castelló

**Tesis Doctoral presentada en la Universidad Miguel Hernández para la obtención del título de Doctor del Programa de Doctorado en Tecnologías Industriales y de Telecomunicación**

**2020**

La presente Tesis Doctoral está sustentada por un compendio de trabajos previamente publicados en revistas de impacto, indexadas según JCR Science Edition. El cuerpo de dicha tesis queda constituido por los siguientes artículos, cuyas referencias bibliográficas completas se indican a continuación:

- Mapping and localization module in a mobile robot for insulating building crawl spaces. [41]
  S. Cebollada, L. Payá, M. Juliá, M. Holloway, O. Reinoso
  **Automation in Construction.** Vol 87, pp. 248-262 (March 2018)
  ISSN:0926-5805. Ed. Elsevier
  **JCR-SCI Impact Factor: 4.313**, Quartile **Q1**
  Web: https://doi.org/10.1016/j.autcon.2017.11.007
  DOI: 10.1016/j.autcon.2017.11.007

- Evaluation of Clustering Methods in Compression of Topological Models and Visual Place Recognition Using Global-Appearance Descriptors. [44]
  S. Cebollada, L. Payá, W. Mayol, O. Reinoso
  **Applied Sciences.** Vol 9(3), 377 (2019)
  ISSN:2076-3417. Ed. MDPI
  **JCR-SCI Impact Factor: 2.474**, Quartile **Q2**
  Web: https://doi.org/10.3390/app9030377
  DOI: 10.3390/app9030377

- Hierarchical Localization in Topological Models Under Varying Illumination Using Holistic Visual Descriptors. [45]
  Sergio Cebollada, Luis Payá, Vicente Román, Oscar Reinoso
  **IEEE Access.** Vol 7(1), pp. 49580-49595 (2019)
  ISSN:2169-3536. Ed. IEEE
  **JCR-SCI Impact Factor: 3.745**, Quartile **Q1**
  Web: https://doi.org/10.1109/ACCESS.2019.2910581
  DOI: 10.1109/ACCESS.2019.2910581

- A State-Of-The-Art Review on Mobile Robotics Tasks Using Artificial Intelligence and Visual Data. [40]
  Sergio Cebollada, Luis Payá, Maria Flores, Adrián Peidró and Oscar Reinoso
  **Expert Systems with Applications.** pp. 114195. (2020)
  ISSN:0957-4174. Ed. Elsevier
  **JCR-SCI Impact Factor: 11.0 (2019)**, Quartile **Q1**
  Web: http://www.sciencedirect.com/science/article/pii/S095741742030926X
  DOI: 10.1016/j.eswa.2020.114195

UNIVERSITAS
Miguel Hernández

# AUTORIZACIÓN DE PRESENTACIÓN DE TESIS DOCTORAL POR UN CONJUNTO DE PUBLICACIONES

Director: Dr. Ing. Óscar Reinoso García
Codirector: Dr. Ing. Luis Payá Castelló

Título de la tesis: ***Creation of hybrid hierarchical models by using omnidirectional vision and machine learning techniques.***

Autor: Sergio Cebollada López

Departamento de Ingeniería de Sistemas y Automática
Universidad Miguel Hernández de Elche

El director y codirector de la tesis reseñada AUTORIZAN SU PRESENTACIÓN EN LA MODALIDAD DE CONJUNTO DE PUBLICACIONES.

En Elche, a 14 de diciembre de 2020

OSCAR|REINOSO|GARCIA
Firmado digitalmente por OSCAR|REINOSO|GARCIA
Fecha: 2020.12.15 10:12:22 +01'00'

Fdo: Dr. D. Óscar Reinoso García

LUIS|PAYA|CASTELLO
Firmado digitalmente por LUIS|PAYA|CASTELLO
Fecha: 2020.12.14 17:24:41 +01'00'

Fdo: Dr. D. Luis Payá Castelló

PROGRAMA DE DOCTORADO EN TECNOLOGÍAS INDUSTRIALES

Y DE TELECOMUNICACIÓN

Dr. D. Óscar Reinoso García, Coordinador del Programa de Doctorado en Tecnologías Industriales y de Telecomunicación de la Universidad Miguel Hernández de Elche.

## Certifica

Que el trabajo realizado por D. Sergio Cebollada López titulado ***Creation of hybrid hierarchical models by using omnidirectional vision and machine learning techniques*** ha sido dirigido por el Dr. D. Óscar Reinoso García y codirigido por el Dr. D. Luis Payá Castelló y se encuentra en condiciones de ser leído y defendido como Tesis Doctoral ante el correspondiente tribunal en la Universidad Miguel Hernández de Elche.

Lo que firmo para los efectos oportunos en Elche, a 14 de diciembe de 2020

OSCAR| REINOSO| GARCIA

Firmado digitalmente por OSCAR|REINOSO|GARCIA
Fecha: 2020.12.14 20:14:42 +01'00'

Fdo.: Dr. D. Óscar Reinoso García
Coordinador del Programa de Doctorado en Tecnologías Industriales y de Telecomunicación

**DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA**
Universidad Miguel Hernández de Elche. Ed. Innova. Avda. de la Universidad s/n, 03202 Elche
Telf: 96 665 8616

# Abstract

Over the past few years, the presence of mobile robots has significantly increased. Nowadays, they can be used for a wide range of applications and they can be found in diverse kinds of environments, such as industrial, household, educational and healthcare. Regarding mobile autonomous robots, these systems need a high degree of autonomy to develop their tasks. This means that they must be able to localize themselves and to navigate through environments, which are a priori unknown. Therefore, the robot will have to carry out the mapping task, which consists in obtaining information from the environment and creating a model. Once this task is done, the robot will be able to address the localization task, i.e., estimating its position within the environment with respect to a specific reference system.

This thesis presents the analysis and design of mapping and localization methods in indoor environments. On the one hand, the thesis presents a work that focuses on solving these problems in underfloor voids with the aim of tackling a spray foam insulation task. On the other hand, a hierarchical localization framework is proposed and evaluated, considering severe visual effects that can influence the accuracy of the proposed method.

The present thesis carries out a work, which focuses on solving the mapping and localization problems in voids between floor and foundations. Solving these tasks in such environments is especially challenging concerning visual information because the environment is dark and the terrain is uneven as stones, bricks fragments or sand are often present. Within these environments, the robot should be able to localize itself and apply insulation foam to the underside of the floor. Hence, the localization process is solved by estimating the position of the robot with respect to previously known position. This is done by using the alignment between point clouds (depth information). The robot is equipped with a 2D laser sensor, which permits building point clouds from several positions of the underfloor environment. This thesis describes several algorithms to obtain robustly the alignment between two positions. The proposed algorithms are tested with a set of point clouds captured with a laser scan under real working conditions. The results show that the localization problem can be solved and the accuracy obtained is enough to develop the insulation task.

The present thesis also proposes the study of the hierarchical visual models to address the mapping and localization tasks. This work is based on the use of omnidirectional images obtained from indoor environments such as offices, corridors, bathrooms, etc. Therefore, this thesis proposes new methods for obtaining the hierarchical maps and solving the localization.

The first proposed work focuses on studying the compression of topological models. Two clustering methods are tested with the aim of knowing their utility to build a compressed model of the environment and to solve the localization task. Omnidirectional images are characterized through global-appearance descriptors, which are used to build a compact model and to estimate the robot

position. The results collected from this work confirm that compression of visual information contributes to a more efficient localization process through saving computation time and keeping a relatively good accuracy.

The second work proposed is a method for addressing the localization task hierarchically. This approach is solved by calculating global-appearance descriptors of the omnidirectional images and the position of the robot is estimated by comparing these descriptors with the information contained in a topological visual model. The proposed localization method is evaluated with some sets of images, captured in large indoor environments under real operating conditions, including illumination changes that affect substantially the appearance of the scenes. The results show a reasonable tradeoff computation time-accuracy when the localization is addressed in a hierarchical way.

The third proposed work focuses on analysing two machine-learning techniques to carry out hierarchical localization tasks. On the one hand, three classifiers are trained with three different global-appearance description methods and then these classifiers are evaluated to retrieve the area or room where a query omnidirectional image was captured. On the other hand, data fitting neural networks are used to calculate the position where the image was captured within the selected area. The results show that these methods introduce an efficient alternative to conduct hierarchical localization regarding computing time and accuracy.

Finally, the fourth proposed work focuses on the use of deep learning techniques. Two main works are developed in this line. First, two deep learning techniques are proposed to calculate global-appearance descriptors. That is, intermediate layers of pre-trained convolutional neural networks and autoencoders are used to obtain holistic descriptors by introducing panoramic images into the network. That is, the robot estimates its position through a nearest neighbour search by comparing the obtained descriptor with the visual model contained in the retrieved room. These description methods are compared with the analytic methods used during the last few years for visual mobile robotics. The results show that deep learning based descriptors can provide also an interesting solution to address visual localization tasks. Second, a deep learning convolutional neural network is developed from scratch to carry out the room retrieval task within an indoor environment. Moreover, the developed network is also used to calculate global-appearance descriptors from intermediate layers and to tackle the position retrieval where the image was captured within the room. The blend of the room retrieval method and the position retrieval method constitutes a novel hierarchical localization approach. The results obtained from the experiments show that the proposed deep learning technique and the novel hierarchical localization method constitute a successful solution to carry out the visual localization task.

# Resumen

Durante los últimos años, la presencia de robots móviles ha crecido sustancialmente y hoy en día se utilizan para un amplio espectro de aplicaciones. Dichos robots se pueden encontrar en diversos entornos como industriales, familiares, de ámbito educativo y de salud. En cuanto a los robots móviles autónomos, éstos requieren un alto grado de autonomía para poder desarrollar la tarea para la cual han sido desarrollados. Esto significa que deben ser capaces de localizarse y de navegar por un escenario que a priori es desconocido. Por lo tanto, el robot tendrá que llevar a cabo la tarea de mapeo, que consiste en obtener información del entorno y crear un modelo del mismo. Una vez se ha realizado dicha tarea, el robot será capaz de localizarse, esto es, estimar su posición dentro del entorno con respecto a un sistema de referencia.

Esta tesis presenta el análisis y diseño de métodos de mapeo y localización en entornos de interior. Por un lado, la tesis presenta un trabajo que se centra en resolver dichas tareas en subsuelos de edificios con el objetivo de llevar a cabo la tarea de rociado de aislante térmico. Por otro lado, se propone un desarrollo de localización jerárquica y se evalúan varios efectos visuales que pueden repercutir en la precisión de dicho método.

En cuanto al trabajo que se centra en resolver el mapeo y la localización en huecos entre el suelo y los cimientos de edificios, la presente tesis parte de la consideración de que resolver estas tareas en los entornos propuestos presenta una dificultad añadidita desde el punto de vista visual. Eso es debido a que estos entornos suelen ser oscuros y el terreno suele presentar superficies desniveladas con presencia de piedras, trozos de ladrillos y/o arena. Dentro de estos entornos, se propone que un robot, de manera autónoma, se localice y posteriormente aplique aislante térmico sobre la parte inferior del suelo del edificio. Por tanto, se propone que el proceso de localización se resuelva estimando la posición del robot con respecto a posiciones anteriores conocidas. Esto se realiza utilizando algoritmos de alineación entre nubes de puntos (información de profundidad). Para esto, el robot está equipado con un sensor laser 2D, el cual permite construir nubes de puntos desde diferentes posiciones del entorno. Esta tesis propone distintos algoritmos para realizar de manera robusta el alineamiento entre nubes de puntos. Los algoritmos propuestos son evaluados a través de un conjunto de nubes de puntos capturadas con un láser bajo condiciones reales de trabajo. Los resultados recogidos muestran que el problema de localización se puede resolver con la precisión necesaria para poder llevar a cabo la tarea de aislamiento.

La presente tesis también propone el estudio de modelos jerárquicos visuales para resolver las tareas de mapeo y localización. Este trabajo se basa en el uso de imágenes omnidireccionales, las cuales se obtienen en entornos de interior tales como oficinas, pasillos, servicios, etc. Por tanto, esta tesis propone nuevos métodos y técnicas que mejoren la obtención de mapas jerárquicos y también que mejoren la tarea de localización.

En esta línea, el primer trabajo se centra en estudiar la compresión de modelos topológicos visuales. De este modo, dos métodos de agrupamiento (*clustering*)

son evaluados con la finalidad de conocer su utilidad para construir modelos compactos del entorno y llevar a cabo la tarea de localización. Para ello, las imágenes omnidireccionales son caracterizadas a través de descriptores de apariencia global, los cuales se utilizan para construir los modelos compactos y también para estimar la posición del robot. Los resultados recogidos de este trabajo confirman que la compresión de los modelos visuales aporta métodos de localización más eficientes, ahorrando tiempo de cómputo y manteniendo una precisión relativamente buena.

El segundo trabajo propuesto se basa en un método para realizar la tarea de localización jerárquica. Esta estrategia es desarrollada mediante el cálculo de descriptores de apariencia global calculados a partir de imágenes omnidireccionales. La posición del robot se estima comparando los descriptores con la información contenida en el modelo visual. El método propuesto se evalúa a través de conjuntos de imágenes que han sido capturados en entornos grandes bajo condiciones reales de trabajo, incluyendo cambios de iluminación, los cuales afectan considerablemente la apariencia de las escenas. Los resultados muestran que existe una compensación entre tiempo de cálculo y precisión cuando se aplica la localización jerárquica.

El tercer trabajo analiza el uso de dos técnicas de aprendizaje máquina (*machine learning*) para poder realizar la localización jerárquica. Por un lado, tres clasificadores son entrenados con tres métodos de descripción global y tras esto, dichos clasificadores son utilizados para recuperar el área o la estancia donde una imagen omnidireccional fue capturada. Por otro lado, se utiliza una red neuronal de ajuste de datos para calcular la posición de captura de la imagen dentro del área seleccionada. Los resultados muestran que las técnicas propuestas introducen una alternativa eficiente para realizar la tarea de localización jerárquica en cuanto a tiempo de cálculo y precisión.

Por último, el cuarto trabajo relacionado con la línea de modelos jerárquicos se centra en aplicar técnicas de aprendizaje profundo (*deep learning*). En concreto, se han desarrollado dos líneas de trabajo. La primera, propone el uso de técnicas de aprendizaje profundo para calcular descriptores de apariencia global. Esto es, usar capas intermedias de redes neuronales convolucionales y de autoencoders para calcular descriptores mediante la introducción de imágenes panorámicas a dichas redes. De esta forma, el robot estima su posición a través de un método de búsqueda del vecino más cercano mediante la comparación del descriptor obtenido con los descriptores que componen el modelo visual del entorno. Estos métodos de descripción son comparados con los métodos analíticos que han sido utilizados comúnmente durante los últimos años para realizar tareas relacionadas con la robótica móvil visual. Los resultados han demostrado que los descriptores basados en aprendizaje profundo también pueden proporcionar soluciones interesantes en cuanto a la tarea de localización visual. La segunda línea de trabajo trata del desarrollo de una red neuronal desde cero para realizar la recuperación de estancia en un entorno de interior. Además, dicha red también se utiliza para calcular descriptores de apariencia global de las capas intermedias y llevar a cabo la recuperación de la posición de captura de una imagen dentro de la estancia. La combinación de la recuperación de estancia y la recuperación de posición dentro de la estancia forman un novedoso método de localización

jerárquica. Los resultados obtenidos muestran que las técnicas de aprendizaje profundo propuestas y el nuevo método de localización jerárquica constituyen una solución satisfactoria para realiza la tarea de localización visual.

# Agradecimientos

Me gustaría dar las gracias en primer lugar a mis directores de tesis, Luis Payá y Óscar Reinoso. Gracias por su infinita paciencia, por dedicar tanto tiempo en mí y por compartir sus conocimientos conmigo. Solo espero haber estado a la altura.

También quiero dar las gracias al resto de miembros del grupo de ARVC: Luis Miguel, David Valiente, Arturo, José María, David Úbeda, Mónica y Adrián, por todos sus consejos y colaboraciones cuando las he necesitado. A María José, por su ayuda cuando he tenido problemas de papeleo y a José Antonio, que me ha ayudado con mis problemas con el ordenador.

Quiero dar las gracias en especial a mis principales compañeros por el mundo del doctorado: Yerai y Vicente. Con ellos he compartido infinitos momentos y anécdotas que siempre quedarán en mi memoria como una maravillosa etapa de mi vida. Yerai, nunca olvidaré cuando a principios de 2015 te pusiste en contacto conmigo para saber si estaba interesado en trabajar en ARVC. Gracias a ti, abandoné el desapacible mundo laboral en el que me encontraba sumergido para empezar a trabajar con un grupo de personas muy válidas tanto en lo profesional como en lo personal. Gracias a eso, mi vida cambió radicalmente y empecé a crear una vida en la cual me siento muy a gusto. Vicente, nuestros caminos se cruzaron gracias a un amor desconocido por las imágenes “omnis”, pero a partir de esto hemos construido una amistad forjada con tardes de Bierwinkel, carreras *survivor*, y un loca y rara estancia en el Reino Unido.

Tampoco quiero olvidarme de todos los compañeros de laboratorio que he tenido durante todos mis años en ARVC: Valerio, Cristóbal, Héctor, Julio, Óscar Soriano, María, Pablo, Amalia, Orlando, Antonio, Juanjo, … Gracias por crear un entorno de trabajo tan enriquecedor y ameno. Me encanta poder decir que os conocí trabajando en la universidad pero que ahora somos amigos. Solo espero que la tradición de almorzar los viernes perdure.

Mi agradecimiento a mi responsable durante las estancia en Münster (Alemania) y a todos sus compañeros que me permitieron acelerar en mis investigaciones. Sus conocimientos sobre *deep learning* produjeron un punto de inflexión en mi carrera investigadora. También he de agradecer a mi responsable durante la estancia en Newcastle por haberme acogido en su grupo y haberme permitido colaborar en tareas tan interesantes.

Fuera del ámbito universitario, quiero dar las gracias a mi familia. A mis hermanas, Paola y Claudia, porque, aunque siempre nos estamos *chinchando* (como buenos hermanos que somos) y siempre me estáis quitando las estrellas del “Alumno 5 estrellas”, estoy seguro que en el fondo estáis orgullosas de mí, casi tanto como lo estoy yo de vosotras. A mis abuelos, los cuales no entienden mucho que hace su nieto, pero ellos están igual de orgullosos. En especial, a mi abuela Carmen, quien hace unos meses nos dejó. Me habría hecho mucha ilusión que me hubiera visto convertirme en doctor.

A mis padres, Pilar y José Luis, quienes a día de hoy tampoco tienen muy claro a qué se dedica su hijo, que no terminan de entender mi frase “trabajar para vivir y nunca vivir para trabajar” y tampoco terminan de entender que no voy a ser doctor *honoris causa* (aunque se lo haya explicado mil veces). Pero de lo que sí estoy seguro es que están profundamente orgullosos de su hijo. Vuestro hijo también está orgulloso de los padres que le han tocado. En especial, gracias por inculcarme los valores de trabajo, esfuerzo y humildad.

Tampoco quiero olvidarme de las amistades: Borja, Mario, Javi, Cristian, Josemi, el grupo dels “Serpaets” y mis amigos de la carrera, “los telecos”. A mis amigos de Erasmus, en especial, Dani, José y Manu (aunque este último es un amigo ‘transversal’). Gracias por llenar mi memoria de recuerdos únicos y extraordinarios.

Por último, quiero dar las gracias a Silvia. Gracias por haberme acompañado en esta locura de viaje. Tú me has sostenido cuando pensaba que me iba a rendir. Tanto tú como tu familia me habéis transmitido todo vuestro apoyo y admiración cuando más lo he necesitado. Creo que tu apoyo durante este largo camino ha sido la clave del éxito. Gracias por nuestros infinitos viajes. Gracias por compartir tu vida conmigo. Gracias por haberme hecho querer ser mejor persona. Gracias por haber sido el broche de oro a estos excelentes años. Gracias por ser como eres.

Y en general, gracias a todas las personas que han contribuido de alguna manera, en mayor o menor medida, a que haya logrado alcanzar esta meta de la mejor de las maneras posibles.

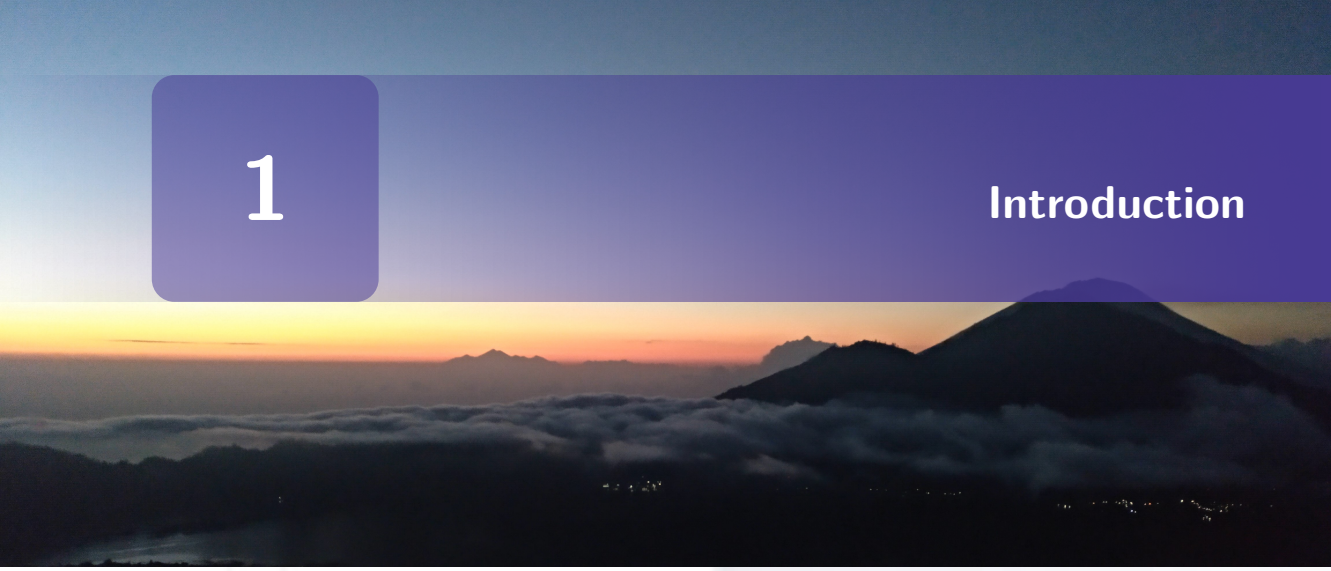
Esta tesis ha sido cofinanciada por la Unión Europea a través del Programa Operativo del Fondo Social Europeo (FSE) de la Comunitat Valenciana 2014-2020 a través de su beca predoctoral ACIF/2017/146.

“What we know is a drop,
what we don't know is an ocean.”
- Isaac Newton

# Contents

# List of Tables

# List of Figures



# 1 Introduction

## 1.1 Motivation

The present thesis focuses on autonomous mobile robots, a field that has substantially increased during the past few years. Robotics are systems that replace humans to develop mechanical, routine or dangerous tasks and need a high degree of autonomy to develop their tasks. In the case of autonomous mobile robots, it means that they must be able to localize themselves within a map and to navigate through environments which are unknown. Therefore, the robot has to tackle the mapping task, which consists in obtaining information from the environment and creating a representative model. Once this task is done, the robot is able to address the localization task, i.e., estimating its position within the environment with respect to the model created. Many sensors have been proposed to tackle the mapping and localization tasks in mobile robotics. For example, Bloesch *et al.* [28] use the kinematic information from the encoder together with an IMU (Inertial Measurement Unit) to estimate the state for legged robots, Kim *et al.* [126] use GPS and odometry data by using the framework of extended Kalman filter to localize a mobile robot. Lingemann*et al.* [157] propose a laser-based approach for tracking the pose of a high-speed mobile robot.

Regarding vision sensors, they have been successfully used to tackle the mapping as well as the localization tasks. For instance, Häne *et al.* [98] use fisheye cameras for 3D mapping, visual localization and obstacle avoidance, Pire *et al.* [219] propose a visual SLAM system based on stereo cameras to solve real-time localization for mobile robots, Arth *et al.* [9] introduce an instant outdoor SLAM based on 2.5D maps and Fuentes-Pacheco *et al.* [81] present a broad review of the state-of-the-art of visual SLAM algorithms. Nevertheless, these approaches present drawbacks in environments,

such as their sensitivity to changes of lighting conditions. For instance, underfloor environments may be completely dark and the illumination is only provided by light sources installed either on the robot and/or in a specific position of the environment. Hence, the shadows will generate inaccuracies both in the keypoints detected and in their descriptors calculated.

Considering the disadvantages of a purely visual approach, this thesis also considers a possibility that consists in using the depth data obtained from laser scanners. For instance, Lingemann *et al.* [158] propose a laser-based approach for indoor and outdoor localization for fast mobile robots, Nguyen *et al.* [202] present an broad evaluation of different line extraction algorithms on 2D laser scans for localization in indoor environments. These devices can capture point clouds from specific poses of the environment. Additionally, there are some authors who use both visual and depth data to estimate the position of the robot. This method consists in using the visual information to obtain an initial estimation and the using the depth information to get more accurate results. For example, Endres *et al.* [71] used either SURF, SIFT or ORB for Pairwise Feature Matching. Henry *et al.* [102] also used SIFT. However, instead of ICP point-to-point, they used ICP point-to-plane. More recently, dos Santos *et al.* [68] use visual information (SIFT) for the coarse alignment. However, instead of ICP they use SLIC (Simple Linear Iterative Clustering) in the fine alignment.

Depending on the number of cameras and the field of view, different configurations have been proposed. Some authors (such as Okuyama *et al.* [203]) have used monocular configurations. Others proposed stereo cameras by using binocular (such as Yong-Guo *et al.* [311] or Gwinner *et al.* [94]) or even trinocular systems (such as Jia *et al.* [115]). Although stereo cameras can measure the depth of images, these systems have a limitation related to their field of view. In addition, to obtain complete information about the environment, several images must be captured. In this sense, omnidirectional cameras are a good alternative. They can provide a lot of information with a 360 degree field of view around it and its cost is relatively low compared to other types of sensors. Additionally, omnidirectional vision systems present further advantages. For example, the characteristics of the images are more stable (because they remain longer as the robot moves) and allow to estimate both the position and the orientation of the robot. Different authors have successfully used omnidirectional cameras for mapping and localization [283],[21], [271], [191], [179]. Payá *et al.* [211] conducted an extensive study, presenting a state of the art of the most relevant localization and mapping algorithms developed with omnidirectional visual information. Fig. 1.1(a) shows an example of a mobile robot having an omnidirectional camera mounted on it and fig. 1.1(b) shows an example of an omnidirectional image.

Visual mapping and localization have been resolved primarily using two main approaches to extract the most relevant information from the scenes; either by detecting, describing, and tracking some relevant landmarks, or by working with holistic algorithms, that is, by constructing a unique descriptor per image.

In the related literature, two main frameworks have been proposed to carry out the mapping task: metric maps, which represent the environment with geometric

(a) (b)

**Figure 1.1:** (a) Example of a Pioneer P3-AT ® robot equipped with an omnidirectional vision system and a laser rangefinder. (b) Example of an omnidirectional image captured from an outdoor environment. Image obtained from the Fukuoka Datasets for Place Categorization [174].

accuracy; and topological maps that describe the environment as a graph containing a set of locations with related links. Concerning the second option, some authors have proposed to arrange the information on the map hierarchically, in a set of layers. The way in which a robot efficiently solves the task of locating on hierarchical maps is as follows: first, an approximate but rapid localization is carried out using high-level layers; second, a fine localization in a local area is addressed using low-level layers. Hence, to address the problem of mapping and localization, hierarchical maps are an efficient alternative (as shown in works [207], [96] and [110]), since they can provide less computational time while maintaining similar accuracy.

In the light of the previous information, in this thesis, the use of hierarchical models is proposed to solve the localization task efficiently. In this sense, compression methods are used as a solution to generate the high-level layers of the hierarchical model. Some authors have used clustering algorithms to perform the compression task. For example, Zivkovic *et al.* [324] use spectral clustering to obtain higher-level models that improve route planning efficiency. Grudic and Mulligan [92] build topological maps through the use of an unsupervised learning algorithm which works with spectral clustering. Valgren *et al.* [282] tackle an online topological mapping through the use of incremental spectral clustering. Štimec *et al.* [264] use an unsupervised clustering based on the multiple eigenspace algorithm to perform topological mapping hierarchically using omnidirectional images. More recently, Shi *et al.* [250] have proposed the use of a differential clustering method to improve the compression of telemetry data.

Additionally, in recent years, machine learning techniques have contributed to solve a variety of problems in mobile robotics. For example, Gonzalez *et al.* [88] use machine learning to detect different levels of slip for robotic missions in Mars. Dymczyk *et al.* [70] propose the use of a boosted classifier to classify landmark observations and carry out the localization task in a more robust fashion.

The purpose of the present thesis is to introduce and test the performance of some machine learning techniques in mapping and localization, and to carry out a comparative evaluation between different global-appearance descriptors and machine learning approaches. The efficiency of these tools will be evaluated through their ability to estimate robustly the position of the robot using the information stored in the map. The approach evaluated consists in the use of a variety of classifiers, neural networks and clustering algorithms, used in combination with global-appearance visual descriptors, to solve the localization problem.

## 1.2 Objectives

The main objective of the present thesis is to optimize the mapping and localization tasks in mobile robotics by reducing the localization error and the computing time through using omnidirectional vision, depth data and machine learning tools. To reach this purpose, the following goals were established:

- **Localization through alignment between point clouds.** Visual features are not enough to carry out the localization task under visually complex environments. For example, estimating the movement of the robot just using visual information in underfloor environments can be inaccurate. In order to avoid important drawbacks introduced in this type of environments, it is necessary to use depth information provided by laser sensors. Hence, to carry out the localization task through this type of information, an alignment algorithm between point clouds is developed and evaluated.
- **Comparison between methods to create hierarchical maps.** In order to create hierarchical models based on visual information, compression techniques are proposed and afterwards evaluated. Hence, the obtained maps are analyzed according to compression parameters.
- **Evaluation of localization through hierarchical maps.** After obtaining different hierarchical maps based on several compression tools, the obtained maps are used to carry out the localization task and to evaluate its performance regarding accuracy and time. This performance is solved as an image retrieval problem and considers also how these methods are affected by changes of illumination. This effect is very common in real-operation conditions, then, the methods have to be robust enough to reach successful results.
- **Developing of global-appearance descriptors through machine learning tools and evaluation of these descriptors to carry out the localization**

**task.** In order to get more efficient localization methods, one important key is the visual information used. This thesis not only uses global-appearance descriptors based on analytic methods such as *gist* or HOG, but novel methods based on machine learning techniques are also developed and evaluated. This work creates novel global-appearance descriptors departing from auto encoders and also from intermediate layers of Convolutional Neural Networks which have been already pre-trained to solve classification tasks.

- **Evaluation of machine learning tools to optimize the hierarchical localization.** Once evaluated the efficiency of the localization task by means of hierarchical methods, this work focuses on optimizing the processes involved. Therefore, different classifiers and neural networks are evaluated to carry out the rough as well as the fine localization steps. The objective is to improve the efficiency regarding computing time, accuracy and robustness against changes of illumination.

- **Obtaining Convolutional Neural Networks for mapping and developing a novel global-appearance descriptor to solve the localization task.** After analyzing the machine learning tools to carry out the hierarchical localization step and also to obtain global-appearance descriptors, the aim of this work is to go one step ahead by using deep learning. Therefore, a Convolutional Neural Network (CNN) is developed, trained and evaluated to carry out localization-related tasks. This network has panoramic images as input and predicts the room where the input image was captured within the environment. Afterwards, the CNN is also used to obtain global-appearance descriptors from intermediate layers. This way, the aim of using this technique is to generate holistic descriptors to solve the localization task more efficiently.

## 1.3 Framework of this Thesis

The present thesis has been developed under a framework supported by different collaborations, grants and research projects, as detailed next.

### 1.3.1 Grants and Awards

The development of this thesis has been supported by a ACIF grant from Conselleria de Educación, Investigación, Cultura y Deporte confunded by the European Social Fund (ESF). This grant, whose reference number is ACIF/2017/146, has supported financially the author of the present thesis during three years (from December 2017 to December 2020), in order to develop this thesis during this period. This grant was extended up to 100 days by Real Decreto-ley 11/2020, de 31 de marzo, with the aim of palliating the effects produced by the COVID-19 crisis.

Also, another two grants were conceded to the author of this thesis in order to do short research stays at foreign universities, as described in the next subsection. The first one was conceded from the Miguel Hernandez University in 2018 and the second

one was conceded from the Conselleria de Educación, Investigación, Cultura y Deporte and the European Social Fund (ESF) in 2020.

### 1.3.2 Research Stays and Collaborations

From September to November 2018, the author of this thesis spent three months collaborating at the Department of Computer Science of the University of Münster (Germany). The objective of this research stay, which was supervised by Prof. Dr. Xiaoyi Jiang, was to investigate the use of classifiers to tackle the rough localization step within a hierarchical localization method, as well as to investigate and learn about deep learning tools to develop more efficient global-appearance descriptors for localization tasks. This research stay was supported by the Miguel Hernandez University.

Additionally, from September to December 2020, the author of this thesis spent four months collaborating at the Computer Science Department in the Northumbria University and the Computer Science department at Durham University (United Kingdom). The objective of this research stay, which was supervised by Prof. Dr. Hubert P. H. Shum, was to investigate the creation of 3D maps and robots localization through the use of artificial intelligence techniques with the aim to estimate also the height. This research stay was supported by the Generalitat Valenciana and the European Social Fund (ESF).

### 1.3.3 Projects

During the development of the present thesis, the author has participated in several research projects as detailed next.

- **"Creación de modelos jerárquicos y localización robusta de robots móviles en entornos sociales".** Project supported by the Generalitat Valenciana from January 2019 until December 2020.
- **"Creación de Mapas Mediante Métodos de Apariencia Visual para la Navegación de Robots ".** Project supported by Ministerio de Ciencia e Innovación from January 2017 until December 2019.
- **"Navegación de Robots en Entornos Dinámicos Mediante Mapas Compactos con Información Visual de Apariencia Global ".** Project supported by Ministerio de Ciencia e Innovación from September 2014 until May 2017.
- **"Localización y Creación de Mapas Visuales para Navegación de Robots con 6 GDL".** Project supported by the Generalitat Valenciana from January 2015 until December 2016.
- **"Creación de mapas topológicos a partir de la apariencia global de un conjunto de escenas".** Project supported by Generalitat Valenciana from January 2015 until December 2016.

## 1.4 Publications

### 1.4.1 Quality Evidences

The main contributions are supported by four articles which were published in JCR-indexed journals. Three journals were categorized in the first quartile (Q1) of their respective categories and one was categorized in the second quartile (Q2) of its respective journal. These publications, whose metadata are provided next, are directly aligned with the purpose and aim of this thesis.

- Mapping and Localization Module in a Mobile Robot for Insulating Building Crawl Spaces. [41]
  S. Cebollada, L. Payá, M. Juliá, M. Holloway, O. Reinoso
  **Automation in Construction.** Vol 87, pp. 248-262 (March 2018)
  ISSN:0926-5805. Ed. Elsevier
  **JCR-SCI Impact Factor: 4.313**, Quartile **Q1**
  Web: https://doi.org/10.1016/j.autcon.2017.11.007
  DOI: 10.1016/j.autcon.2017.11.007

- Evaluation of Clustering Methods in Compression of Topological Models and Visual Place Recognition Using Global-Appearance Descriptors. [44]
  S. Cebollada, L. Payá, W. Mayol, O. Reinoso
  **Applied Sciences.** Vol 9(3), 377 (2019)
  ISSN:2076-3417. Ed. MDPI
  **JCR-SCI Impact Factor: 2.474**, Quartile **Q2**
  Web: https://doi.org/10.3390/app9030377
  DOI: 10.3390/app9030377

- Hierarchical Localization in Topological Models Under Varying Illumination Using Holistic Visual Descriptors. [45]
  Sergio Cebollada, Luis Payá, Vicente Román, Oscar Reinoso
  **IEEE Access.** Vol 7(1), pp. 49580-49595 (2019)
  ISSN:2169-3536. Ed. IEEE
  **JCR-SCI Impact Factor: 3.745**, Quartile **Q1**
  Web: https://doi.org/10.1109/ACCESS.2019.2910581
  DOI: 10.1109/ACCESS.2019.2910581

- A State-Of-The-Art Review on Mobile Robotics Tasks Using Artificial Intelligence and Visual Data. [40]
  Sergio Cebollada, Luis Payá, Maria Flores, Adrián Peidró and Oscar Reinoso
  **Expert Systems with Applications.** pp. 114195. (2020)
  ISSN:0957-4174. Ed. Elsevier
  **JCR-SCI Impact Factor: 11.0 (2019)**, Quartile **Q1**
  Web: http://www.sciencedirect.com/science/article/pii/S095741742030926X

DOI: 10.1016/j.eswa.2020.114195

The first article proposes two novel algorithms to robustly obtain the alignment between two point clouds. The alignment between point clouds is used to estimate the position of the robot with respect to previously known positions. The proposed methods, which are developed in chapter 3, are tested and validated to tackle the localization task in challenging underfloor voids through the use of depth information. They firstly consist in downsampling the point cloud information and removing the top and bottom information, after that, the registration is carried out assuming four initial rotations and the transformation matrix is obtained. Secondly, through the second contribution method, a novel algorithm is developed to solve the cases in which the alignment between consecutive poses was not possible. The results presented in this work show the robustness and effectiveness of the proposed methods and their ability to cope with such challenging underfloor environments.

The second article proposes an exhaustive study about the compression of topological models in indoor environments. For this aim, two clustering methods are tested in order to know their utility both to build a model of the environment and to solve the localization task. To evaluate the goodness of the proposed clustering algorithms, which is shown in chapter 4, several datasets were considered. They are composed of either panoramic or omnidirectional images captured in several environments, under real-operation conditions. The results collected proved that the use of clustering methods to tackle the compression step are more efficient than carrying out a direct downsampling process of the images from the database.

The third article proposes a hierarchical localization framework using just omnidirectional cameras as source of information. Several holistic descriptors are obtained and compared between them to tackle the hierarchical localization task proposed. Moreover, a new global-appearance descriptor based on a intermediate layer of a pre-trained CNN is also proposed. The evaluation also considers changes of illumination during the localization process. The results obtained are shown in chapter 4 and they prove that the compaction proposed throughout the present thesis is suitable to solve the localization task efficiently.

The fourth article presents a survey about the use of artificial intelligence and visual data to address mobile robotics tasks. This manuscript, which is presented in chapter 2, focuses on showing how researchers have addressed relevant tasks in mobile robotics by means of artificial intelligence tools and visual information; and how these approaches have evolved in recent years. Among these tasks, it its worth citing mapping (building a robust model of the environment; (b) localization (estimating the position in the environment; (c) SLAM (Simultaneous Localization and Mapping); (c) navigation (planning a path and controlling the movement of the robot towards a target point) and exploration (exploring the free space of an environment). These abilities are essential to address any high-level task by using mobile autonomous robots. Hence, the manuscript carries out a state of the art review of them by using AI and visual information.

These articles are appended in Appendix A.

### 1.4.2 Other Works Related to this Thesis

This subsection presents other works developed within the framework of the thesis that have been sent and published in conferences and other journals, as a consequence of the tasks carried out within the framework of this thesis in its different lines of research.

- A Novel Method to Estimate the Position of a Mobile Robot in Underfloor Environments Using RGB-D Point Clouds. [208]
  Cristóbal Parra, Sergio Cebollada, Luis Payá, Mathew Holloway, Oscar Reinoso
  **IEEE Access.** Vol. 8, pp. 9084-9101 (2020)
  ISSN:2169-3536. Ed. IEEE
  **JCR-SCI Impact Factor: 4.098**, Quartile **Q1**
  Web: https://doi.org/10.1109/ACCESS.2020.2964317
  DOI: 10.1109/ACCESS.2020.2964317

- S. Cebollada, L. Payá, M. Flores, V. Román, A. Peidró, O. Reinoso. A Deep Learning Tool to Solve Localization in Mobile Autonomous Robotics. ICINCO 2020, 17th International Conference on Informatics in Control, Automation and Robotics (Lieusaint-Paris, France, 7-9 July, 2020). Ed. INSTICC [43].

- S. Cebollada, L. Payá, D. Valiente, X. Jiang, O. Reinoso. An evaluation between global-appearance descriptors based on analytic methods and Deep Learning techniques for localization in Autonomous Mobile Robots. ICINCO 2019, 16th International Conference on Informatics in Control, Automation and Robotics (Prague, Czech Republic, 29-31 July, 2019). Ed. INSTICC ISBN:978-989-758-380-3 ISSN:2184-2809. Volume 2, pp. 284-291 [237].

- L. Payá, W. Mayol, S. Cebollada, O. Reinoso. Compression of topological models and localization using the global-appearance of visual information. ICRA 2017. IEEE International Conference on Robotics and Automation (Singapur, 29 de mayo a 3 de junio de 2017). Ed. IEEE ISBN:978-1-5090-4632-4 [212].

- S. Cebollada, V. Román, L. Payá, M. Flores, L.M. Jiménez, O. Reinoso. Uso de técnicas de machine learning para realizar mapping en robótica móvil. XL JORNADAS DE AUTOMATICA (El Ferrol (Spain), 4-6 September 2019). Ed. CEA-IFAC ISBN:978-84-9749-716-9. pp. 686-693 [46].

- S. Cebollada, L. Payá, A. Peidró, L.M. Jiménez, O. Reinoso. Estudio de descriptores holísticos basados en métodos analíticos y técnicas de deep learning para localización con robots móviles. Jornadas Nacionales de Robótica 2019 (Alicante (Spain), 13-14 June 2019).Ed. CEA ISBN:978-84-09-12133-5. pp. 1-8 [42].

- S. Cebollada, C. Parra, M. Juliá, M. Holloway, L.M. Jiménez, O. Reinoso. Alineamiento 3D desde posiciones no cercanas de un robot para trabajos en interiores a partir de imágenes RGB-D. XXXVII Jornadas de Automática (Madrid (Spain), 7-9 September 2016). Ed. CEA-IFAC ISBN:978-84-617-4298-1 - pp. 374-381 [238].

Additionally, there are some results of the present thesis which have not been published yet. They are expected to be submitted for publication in other journals or conferences in the near future. These pending publications are:

- A paper about the use of machine learning tools to carry out the hierarchical localization task, related to Chapter 5.
- A paper about the use of deep learning to solve the localization task, related to Chapter 6.

## 1.5 Structure of this Thesis

This document has been organized as follows:

- Chapter 2 presents an up-to-date literature review about autonomous mobile robots and computer vision used to solve the mapping and localization tasks, discussing the advantages and drawbacks of using these techniques. After reviewing several works tackled in the last few years, this chapter also reviews some of the artificial intelligence tools which have been related to solve computer vision tasks.
- Chapter 3 presents a study of techniques used to solve the localization in underfloor voids. On the one hand, novel algorithms based on local features over images were considered whereas, on the other hand, novel algorithms to carry out the alignment between point cloud data was also proposed. Concerning this, the techniques proposed were evaluated in order to study their effectiveness regarding the error made for each of the methods proposed.
- Chapter 4 presents an exhaustive study regarding the compression of topological models to tackle an efficient localization task. For this purpose, the study considers several items which can play an important role. In this respect, the developed studies considered: different global-appearance descriptor techniques, different compression methods, different map dispositions and different distance metrics to calculate similitude between descriptors. Furthermore, this work also considers to study how the changes of illumination can affect the localization carried out through the hierarchical maps created.
- Chapter 5 deals several machine learning techniques to enhance the localization task. In this respect, auto encoders and CNN intermediate layers are proposed to create novel global-appearance descriptors which outputs more efficient localization tasks. Besides this, an exhaustive study was developed to study different layers from a pre-trained CNN (Convolutional Neural Network) in order to obtain global-appearance descriptors which suit the localization under indoor environments and also present robustness against changes of illumination conditions. At the same time, this chapter also presents the study of different classifiers to carry out the low level map and the use of fitting neural networks for the high

level maps. Both tools are developed with the aim to obtain more efficient and robust hierarchical localization tasks.

- Chapter 6 presents a work related to the use of deep learning tools. This work creates a novel CNN departing from an already created one, but adapted and re-trained to solve a room classification task within an indoor environment. Further this, some intermediate layers will be analyzed with the objective to obtain global-appearance descriptor that present optimal solutions to solve the localization in the environment trained or even in different environments but with similar characteristics.
- Last, chapter 7 summarizes the main contributions presented along this thesis, and also remarks possible future research works derived from these contributions.

## 1.6 Summary of Materials, Methods, and Discussion of Results

This section presents a summary of the main materials and methods used for developing the research lines presented in the present thesis. Additionally, this section exposes briefly a discussion about the main results obtained in each chapter.

### 1.6.1 Materials

The following list details the materials used to tackle the research works:

- A 2D laser sensor and a monocular data used to obtain colored point cloud data.
- Vision-based database with omnidirectional and panoramic images captured at the Miguel Hernández University [209].
- Vision-based database with omnidirectional images under different illumination conditions captured at three indoor laboratories which are located in three cities: the Autonomous Intelligent Systems Laboratory at the University of Freiburg, Germany; the Visual Cognitive Systems Laboratory at the University of Ljubljana, Slovenia; and the Language Technology Laboratory at the German Research Center for Artificial Intelligence in Saarbrücken, Germany [222].
- PC with a CPU Intel Core i7-7700 ® at 3.6 GHz with a GPU NVIDIA GEFORCE GTX 1080TI ®.

#### 1.6.1.1 Methods

The following list details the methods used to tackle the research works:

- Methods to treat the point cloud data through the PCL (Point Cloud Library): VoxelGrid, downsampling, planar segmentation. These tools were used in chapter 3.

- Matehmatical tools: Fourier Signature, gist descriptor, HOG descriptor, Euclidean distance, cosine distance, correlation distance, Manhattan distance, ICP (Iterative Closest Point) algorithm and spectral clustering.
- Several clustering algorithms (k-mean, spectral clustering, etc.) to compress the visual information which composes the topological maps. These tools were used in chapter 4.
- Several hierarchical localization algorithms used to estimate the current position of a robot in a visual hierarchical model. These tools were used in chapter 4, chapter 5 and chapter 6.
- Different machine learning tools: auto encoders and pre-trained CNNs (Convolutional Neural Networks) used to obtain global-appearance descriptors and classifiers and nerual networks to tackle the hierarchical localization. These tools were used in chapter 5.
- Data augmentation: technique that enables to significantly increase the diversity of data available for training models, without actually collecting new data. In this case, the images obtained from the databases were increased by applying effects such as changes of illumination, Gaussian noise, random rotation, blur effect, etc. This technique was used in the Chapters chapter 5 and chapter 6.
- Deep learning tools to develop, train and evaluate a CNN able to solve the hierarchical localization in an indoor environment. This tool was used in chapter 6.

#### 1.6.1.2 Results and Discussion

Finally, this subsection summarizes and discusses the results obtained from the research works described in each chapter. These results have been published either in national and international conferences or in JCR-indexed journals. These publications have been listed in 5.7.

- Chapter 3: The results presented in this chapter show the robustness and effectiveness of the two methods proposed to solve the registration task between poses and their ability to cope with such challenging underfloor environments.
  - The first contribution is an algorithm that uses point cloud information captured by a robot from different poses. After obtaining the point clouds, they are downsampled and the information in the top and bottom planes is removed in order to obtain clouds with less points and more robust results. At the end of this algorithm, the transformation matrix is calculated. This method works well in environments where the characterization using regular algorithms is difficult. Most of the current registration algorithms do not work well in this kind of environments due to the fact that they are not able to extract reliable features. Additionally, although the registration algorithms are commonly based on a coarse alignment (using visual information), a fine alignment (using depth information) and an optional

optimization, the method proposed here is able to find accurate enough results just using the depth information.

- As the second contribution, a novel algorithm is proposed to solve the cases that the alignment between consecutive poses was not possible. This algorithm tries to align the poses which were not well aligned with other poses within the environment. The results show that this method successfully aligns the majority of cases. Hence, this can be useful for online localization processes to ensure that no information is lost in order to determine the exact path followed by the robot within the environment. Although extra computing time is required, no pose information is lost and an accurate global map can be created.

- Chapter 4: This chapter focuses on studying the mapping and localization tasks through compression techniques.

  - Regarding the mapping task, this work proposes two different methods to compact topological maps. During the experiments, with the aim of compacting the information of the environment, the number of instances was reduced to a value in the interval from 10 to 100. That means a reduction of instances up to between 1.1% and 11.5% of the original number. The proposed methods are (1) spectral clustering and (2) Self-Organizing Maps. Moreover, three global-appearance descriptors are used since they present a good solution for environments whose data size is high. The work shows that it is possible to reduce considerably the visual information from the original model. Among these combinations of method-descriptor, spectral clustering together with *gist* descriptor has been proved to be the best choice to compact the model.

  - As for the localization task, an evaluation is carried out with the aim of measuring the goodness of the localization task through the use of compact maps and global-appearance descriptors. In this case, three descriptors and two indoor environments were evaluated. Furthermore, a mixture between indoor environments was also created with the aim of evaluating whether it is possible, first, to detect the right environment and second, estimate the position of the instance. From this study, HOG is the description method whose localization results are the best. Additionally, *gist* presents the most successful results in order to select the correct environment of a test instance from a combined dataset. Additionally, this work also analyzes the utility to solve the localization task hierarchically when substantial illumination changes are present. A comparative evaluation between four methods to globally describe images has been carried out: FS, HOG, *gist* and a CNN-based descriptor. The work showed that it is possible to keep a good localization error departing from a compact model. The CNN-based descriptor and cosine distance has been proved to be the best choice. Nevertheless, this work also proved the efficiency of this localization framework under severe changes of illumination. Moreover, it has proved that the test

images under sunny conditions affect more negatively the results than the night conditions.

- Chapter 5: In this chapter, on the one hand, we have evaluated the use of machine learning tools to carry out hierarchical localization task with mobile robotics. On the other hand, a study was also tackled regarding the use of global-appearance descriptors based on deep learning techniques for localization.

  - Regarding the use of deep learning techniques to obtain global-appearance descriptors for localization, this task was solved as an image retrieval problem. Five global-appearance descriptors were evaluated: two based on analytic methods (HOG and *gist*), two based on auto encoders and one based on CNN intermediate layers. As conclusion, the minimum localization error is obtained through the CNN-based descriptor option. The CNN-based descriptor introduces also the best option regarding the computing time to calculate the descriptor. Nevertheless, regarding the time to estimate the pose of the robot, HOG is the fastest. Notwithstanding, using an auto encoder which has been trained with images that belong to the environment outputs good-enough accuracy results to solve efficiently the localization. As for the use of CNN-based descriptors, we have proved that prior layers can output interesting descriptors despite they are not fully connected layers (typically proposed to obtain descriptors), because the obtained descriptors produce optimal localization solutions among all the methods evaluated: size of descriptor relatively small, low computing time to calculate the descriptor and very accurate localization.
  - As for the use of machine learning tools to improve the hierarchical localization task, the experiments were carried out with an indoor dataset which contains omnidirectional images and presents dynamic changes, blur effects and different illumination conditions. The work shows that most of the machine learning techniques proposed provide good localization results departing from a compact model. First, the classifiers have been validated as a tool to perform the rough localization. SVM (Support Vector Machine) and shallow neural network classifiers together with global-appearance descriptors (*gist* and CNN-fc7) provide high hit ratio to retrieve the corresponding room. Second, a data fitting neural network was proposed for the fine localization step and it works well for most of the cases. Moreover, these techniques present robustness against changes of illumination. Third, through the use of spectral clustering, the localization results are improved in comparison to the ones provided by the ground truth labeling information.

- Chapter 6: In this chapter, first, we have developed and evaluated a deep learning tool for visual classification. Second, a study was also tackled regarding the use of global-appearance descriptors based on the developed deep learning tool for localization.

- Regarding the development of the deep learning tool, this chapter presents the process to train and also to evaluate a Convolutional Neural Network (CNN) implemented for classification. In this case, the CNN is designed to estimate the room within the environment where a test image was captured. First, an architecture is chosen among common architectures developed by experts, after that, the visual data from an indoor environment is selected and augmented to train the network. Once the network has been trained, it is evaluated with new data from the same environment. As conclusion, the CNN is properly trained to carry out the classification, since it presents few wrong predictions. Moreover, through analyzing the confusion matrix obtained from the test dataset, the erroneous predictions are presented in images whose correct room is adjacent to the wrong room predicted. This CNN is proposed to be used as tool to tackle the rough localization step of a hierarchical localization method.
- As for the use of this deep learning tool to obtain holistic descriptors, this technique basically consists in introducing a test image into the CNN and obtaining a global-appearance descriptor from a layer of the network. This description method is evaluated to carry out the localization task. In this sense, descriptors are obtained from 5 different layers of the CNN ($conv_4$, $conv_5$, $fc_6$, $fc_7$ and $fc_8$) and their efficiency for localization is measured by calculating the average localization error and the average computing time to solve the localization task in an indoor environment. The results obtained proved to be more efficient than the results obtained by mean of classical analytic description methods (*gist* and HOG). Afterwards, two whole hierarchical localization methods are proposed. The first one consists in using the CNN in the rough step to predict the room where the test image was captured. After that, the holistic descriptor is obtained from a layer of the CNN and this descriptor is compared through a nearest neighbour search with all the descriptors contained in the room estimated. In this way, the localization results keep the accuracy whereas the computing time is reduced, since less comparisons are conducted. The two hierarchical localization method proposed is quite similar to the previous one but introducing a confidence threshold. This is due to the fact that the results obtained from the hierarchical localization method proved that there is a light worsening of the localization error, since the CNN occasionally fails to estimate the room. Through analyzing the results obtained from the CNN to estimate the room, we appreciate different likelihood values of the classification layer depending whether the CNN success or fails. Hence, this hierarchical localization method establishes a threshold regarding the likelihood values of the classification layer. In this way, when the CNN is not "completely sure" about which room is the correct one, the fine localization step is carried out with all the data contained in the most likely rooms. Through this second method, the localization error is reduced in comparison to the results obtained with the first hierarchical localization method proposed.

# 2 State of the Art

## 2.1 Mobile Autonomous Robots

The presence of autonomous robots in many kinds of environments has increased substantially during the last few years. The aim of these systems is to provide a service regarding a specific task. Robots have been commonly used to carry out either hazardous, repetitive or highly accurate tasks. For example, Baldawi [14] proposes the use of a robot to carry out duties of firefighters, Longo and Muscato [163] propose a climbing robot to inspect non-porous vertical walls, some authors have proposed the use of the da Vinci Surgical System, which is controlled by a surgeon with the objective to facilitate complex surgeries ([29] and [146]). Regarding the autonomous mobile robots, they are designed to carry out a specific task throughout traveling along an environment of work [253]. This type of robot eases a wide variety of tasks such as floor cleaning [322], exploration within hazardous environments [263], or planetary exploration [300] (see fig. 2.1).

In order to develop the task desired, they must be able to localize themselves and to navigate through environments that are beforehand unknown. Hence, the robot has to carry out the mapping task, which consists in obtaining information from the environment and creating a model which represents it. Once this task is done, the robot is able to address the localization task, i.e., estimating its position within the environment with respect to a specific reference system (see fig. 2.2). Concerning navigation, this task basically involves solving the problem to find out how the robot can get to other places from its current position [152]. That is, make a trajectory to reach a certain place. Mapping, localization, and navigation are the classic problems of mobile robotics. They have attracted a great attention and nowadays continue

(a) (b)

**Figure 2.1:** (a) Example of the Mars Exploration Rover. Image extracted by NASA JPL Cornell University, Maas Digital LLC. (b) Mopping vacuum cleaner Robot. Image obtained by Mamirobothk [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0)].

being a prominent research area, since a robust solution to these problems is critical to increase the autonomy of mobile robots and subsequently expand their use for other applications. Several authors have proposed different methods for performing these tasks. Some relevant examples are presented in the following subsections.

### 2.1.1 Mapping

To perform mobile robotics tasks, the robot must be provided with relevant information about the environment. To do this, the robots are equipped with sensors that allow them to obtain information from the environment. Subsequently, robots need to process the data captured from the environment and transform it into useful information for their tasks. The related literature highlights two main frameworks: metric and topological maps. On the one hand, metric maps are a grid-based representation, that is, they represent the environment with geometric accuracy. On the other hand, topological maps lead a graphical representation, that is, the environment is described as graphs that contain locations and relate to each other with links.

A wide range of works have been recently addressed regarding this task. For example, Markom *et al.* [172] solve the mapping task through using a low cost laser rangefinder. Sünderhau *et al.* [267] propose a system based on convolutional network which learns to recognise new semantic classes online while the robot is navigating along an environment. Kuric *et al.* [140] developed a software which uses the metric form of space representation to get a multilayer map system suitable for different tasks of mobile robot. More recently, Rao *et al.* [227] propose a work to create a map of an indoor environment through using Lidar and other sensors and perform autonomous navigation with using capabilities like dynamic obstacle avoidance, speech recognition and video streaming. Many other examples can be found regarding this task. Some of them can be found in the work presented by Kostavelis and Gasteratos [134], which presents an exhaustive survey about semantic mapping for mobile robotics tasks.

Regarding the map typology for visual mapping, in the related literature, two main frameworks have been commonly proposed: the metric and the topological maps. As for the metric maps, they represent the environment with geometric accuracy by representing some features from the environment regarding a reference system. For example, Gil *et al.* [87] propose an approach to the SLAM task by using a team of autonomous vehicles equipped with vision sensors. The metric map is built by the position of the robots in the movement plane and local features extracted with vision systems. Regarding the topological maps, they describe the environment as a graph that contains a set of locations with the related links among them. For instance, Fraundorfer *et al.* [78] create an online topological representation during robot exploration based on visual information. Blochliger *et al.* [26] propose a framework which simplifies the navigation task by transforming a sparse feature-based map from a visual SLAM system into a three-dimensional topological map. Garcia-Fidalgo and Ortiz [85] presented a review about the main approaches considered to carry out topological mapping and localization through visual information in the last years. More recently, da Silva *et al.* [57] propose a localization and navigation approach for mobile robots using topological maps and using CNN to obtain descriptors from omnidirectional images.

Apart from these options, arranging the information hierarchically is an efficient alternative. This framework consists of creating a map that consists of several layers with a hierarchical structure. The high-level layers present a relatively compact amount of information, allowing for an approximate but rapid localization. Low-level layers usually have more information and are used to refine the position. Hence, in order to address the mapping and localization issue, hierarchical maps constitute an efficient alternative, like the works [207], [96] and [110] show. A good example of this issue was developed by Štimec *et al.* [264], who proposed an unsupervised hierarchical mapping method. Kuipers *et al.* [138] propose a hierarchically hybrid map, which consists in using a metrical map to build local maps of small-scale space and topological maps to represent the structure of large-scale space. This approach is proposed to solve the SLAM task in an environment with multiple nested large-scale loops.

### 2.1.2 Localization

To perform the localization task, the environment must be modeled in advance. Therefore, in this way, first the robot performs the mapping task and then, once the map is available, the localization can be performed. Recently related literature presents a wide variety of works. For instance, Memon *et al.* [177] propose a model which addresses localization of two fire detecting mobile robots on a round lattice in a dynamic domain. Nazemzadeh *et al.* [198] propose a technique to fuse odometry and gyroscope data with position and heading measurements based on quick response (QR) code landmark recognition to carry out an optimal localization task which minimizes the error. Sahdev and Tsotsos [239] present a method for mobile robots which allow them to learn from experience and then to recognise previously observed places in known environments and also to categorize previously unseen places in new environments. A review of localization-related works is presented by Chen *et al.* in [50].

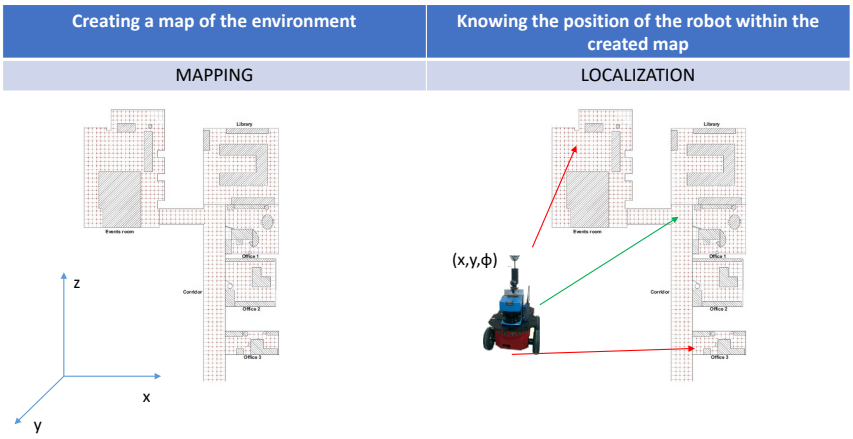



**Figure 2.2:** Mapping and localization tasks overview. On the left side (a), a representation of the environment in a reference system is built by obtaining information. On the right side (b), once the model is built, the robot has to be able to estimate its position within the environment through comparing the data acquired in its current position and the data contained in the model.

Moreover, the related literature has also studied a combination of both tasks that can be performed at the same time. This concept is known as Simultaneous Localization And Mapping (SLAM). This process basically consists in knowing the environment as the robot moves around it and, at the same time, estimating its position and orientation. For example, DaMota *et al.* [56] present a work in which a SLAM task is conducted through a petri net (PN) and radio-frequency identification (RFID) technology. Li *et al.* [153] introduce a technique that uses visual circle and corner features as landmarks in the scene and improves the SLAM process stability using saliency measurement. A wide survey of visual SLAM algorithms is presented by Fuentes-Pacheco *et al.* in [81].

## 2.2 Data Acquisition Systems

With regard to the mapping and localization process, the robot requires one or several sensors to capture information from the surroundings. Over the last years, many authors have proposed many different sensors to use to solve the tasks. For instance, Shen *et al.* [249] propose a mobile robot system using LIDAR (Light Detection and Ranging) to solve the SLAM task in indoor and unknown environments. Doi *et al.* [64] use GPS (Global Positioning Systems) with an extended Kalman filter (EKF) to carry out a path planning method for off-road mobile robots. Silva Almeida *et al.* [254] have developed an exhaustive analysis regarding the use of omnidirectional SONAR (SOund Navigation And Ranging) along with machine learning techniques to tackle the localization in an unknown environment. The present thesis focuses on the use of

depth data acquired from laser scanners and fundamentally on the use of vision data acquired from omnidirectional cameras.

### 2.2.1 Depth Data

Regarding the use of depth data obtained from laser scanners in mobile robotics, these devices can capture point clouds from specific poses of the environment. For example, Yang *et al.* [309] have proposed series of approaches to solve the problem of the mobile robot motion control and autonomous navigation through using 3-D laser scanning in large-scale outdoor for environments where GPS is not available. The localization task is usually solved by registration between two point clouds captured in different poses of the robot. Registration basically consists in estimating the translation and the rotation between one fixed point cloud and a moving point cloud. The most common registration algorithm is ICP (Iterative Closest Point) [23], however, many variations of this method have been proposed, such as the one presented in [246], where plane-plane matchesare considered instead of point-to-point, or [10], where SVD (*Singular Value Decomposition*) is used to calculate the transformation matrix. Jost and Hügli use a heuristic approach to find the nearest points and thus reduce complexity and speed up the process [118]. Despite the robustness of this information, there are some authors who extract key points (presented in subsection 2.4) from the visual information and then use in-depth data to estimate the position of the robot. Fig 3.5 shows this process. For example, Endres *et al.* [71] use SURF, SIFT, or ORB to match features in pairs and then a point-to-point registration algorithm. Henry *et al.* [102] also use SIFT, but use point-to-point ICP instead of point-to-point ICP. More recently, Cappelletto *et al.* [36] use color information to weight the distances between matching points in the ICP algorithm. For more information, an extensive comparison can be found on the methods presented in [301].

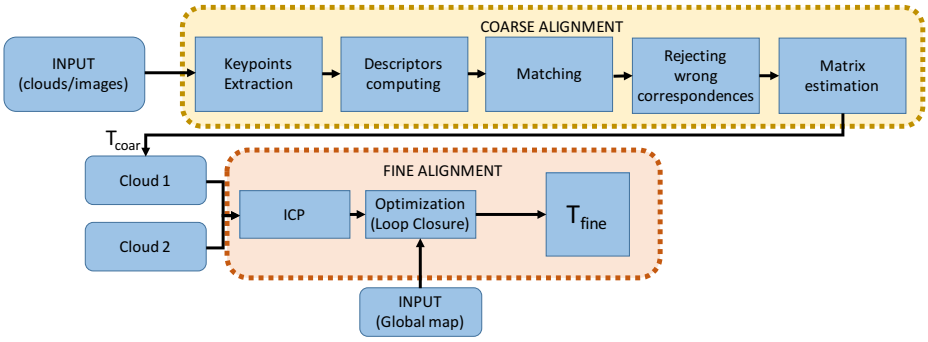



**Figure 2.3:** Schematic overview of the registration approach based on visual and depth data. A coarse alignment algorithm is tackled by using visual descriptors and, after that, a registration algorithm is tackled between two point clouds and using the initial transformation matrix obtained from the previous step.

### 2.2.2 Vision Data

These types of sensors have been widely used to address the tasks of mobile robotics. Many authors have proposed using visual information to solve mapping and localization tasks, as vision sensors are usually cheaper than other sensors such as laser or GPS scanners. In this way, Reinoso and Payá [229] present a special issue on some of the possibilities offered by vision systems, focusing on the different configurations that can be used and new applications, from mapping to navigation of mobile robots until the recognition of objects or scene reconstruction. The vision-based robot systems provide a "see and move" fashion or in real-time mode. By using these types of sensors, the amount of information collected may be enough to address most of the problems related to mobile robotics. In addition, these sensors present a relatively low cost alternative as they maintain a good enough accuracy. Nonetheless, these approaches have disadvantages, such as their sensitivity to changes in lighting conditions. For instance, underground environments can be completely dark and lighting is only provided by sources installed in a specific position in the environment and/or in the robot itself. For this reason the shadows will probably provoke inaccuracies [41, 208].

As for the type of cameras and the different configurations proposed, omnidirectional cameras are a good alternative. They are capable of providing a high quantity of information about the environment around them, with a 360-degree field of view, with a single snapshot. Their cost is also relatively low compared to other types of sensors. In addition, omnidirectional vision systems have additional advantages. For example, the characteristics of the images are more stable (because they remain longer as the robot moves) and allow to estimate both the position and the orientation of the robot. As mentioned previously, omnidirectional cameras have been successfully used by different authors for mapping and localization [284],[213], [271], [191], [179]. A wide state of the art review was carried out by Payá *et al.* [211], which presents the most relevant localization and mapping algorithms developed with omnidirectional visual information.

## 2.3 Artificial Intelligence Tools

In recent years, various tools and techniques based on artificial intelligence (AI) have been shown to be capable of addressing a variety of problems with a deep data treatment. The use of these techniques has become very popular among mobile robotics. The present thesis has studied the use of different Artifical Intelligence (AI) tools to carry out more efficiently the visual mapping and localization tasks.

Regarding **machine learning** methods, they try to automate the construction of analytic models from data analysis. These methods are based on the idea that the systems can learn to identify patterns departing from the data. Common machine learning techniques include decision trees, support vector machines and ensemble methods. Numerous works regarding the use of machine learning tools together with visual information can be found in the related literature. For example, Zhang and Wu [316] use a method based on kSVM (Kernel Support Vector Machine) with the aim of classifying images of fruits; Murthy and Jadon [193] use feed-forward neural networks

to detect hand gestures; and Fan *et al.* [73] use a feed-forward neural network with back-propagation to predict the texture characteristics from food surface images. Furthermore, in recent years, machine learning techniques have contributed to solve a variety of problems in mobile robotics. For instance, Triebel [277] proposes the Informative Vector Machine (IVM) classifier for semantic mapping in autonomous mobile robotics; Duguleana and Mogan [69] propose a path planning algorithm based on the use of Q-learning and artificial neural networks for mobile robots obstacle avoidance; Gonzalez *et al.* [88] use machine learning to detect different levels of slip for robotic missions in Mars; and Dymczyk *et al.* [70] propose the use of a boosted classifier to classify landmark observations and carry out the localization task in a more robust fashion.

As for **deep learning**, this branch belongs to the Machine learning and these methods try to construct automatically high level data models through architectures that allow linear, non-linear, multiple and iterative transformations [20] from the initial data matrices. The idea is to train the architecture to reach a model that is capable of creating representations which best define the inputs. Many deep learning tools have been applied over fields such as computer vision [32], speech recognition [142] or audio-visual recognition [3] and they have proved to performance cutting-edge results for many tasks. The present thesis studies the use of deep learning to address the mapping and localization tasks by using computer vision. A wide review can be found in the work presented by Voulodimos *et al.* in [289].

### 2.3.1 Definition and Areas of Use

Some definitions of artificial intelligence can be highlighted among the definitions found in the related literature. For example, Minsky [184] defined AI as "the science of making machines do things that would require intelligence if done by men". Charniak *et al.* [48] defined it as "the study of mental faculties through the use of computational model". Within the recent past, Schalkoff [243], defines AI as "a field of study that seeks to explain and emulate intelligent behavior in terms of computational process". As for the birth of AI, many researchers believe this happened during World War II, when scientist Alan Turing worked to decipher the 'Enigma' code used by German forces to send messages securely. Turing and his team created the Bombe machine, which was used to decipher Enigma messages. Both Enigma and Bombe Machines are considered the foundations of artificial intelligence [228]. Focusing on the use of AI to solve robotics tasks, this science can be defined as a set of techniques that are applied in computer programming to solve problems whose difficulty requires a certain degree of intelligence.

AI has been widely used in different areas. Depending on the type of manipulation, we can establish two categories: physical manipulation and thinking manipulation. In terms of physical manipulation, it covers the fields of artificial vision, robotics, and control systems. For instance, Vyborny *et al.* [290] have successfully used computer vision along with AI in mammograms to detect or characterize abnormalities in digital images. With the help of this approach, radiologists are capable of detecting abnormalities better and thus, errors in the interpretation of mammograms are greatly reduced.

Another example of computer vision and AI is proposed by Wachs *et al.* [291], who propose vision-based hand gesture recognition for human interaction with the computer based on an artificial neural network, fuzzy logic, and genetics algorithms. Concerning the use of Artificial Intelligence for Robotics, Singh and Parhi [256] propose a neural network to solve the problem of route and time optimization of mobile robots. The inputs to the proposed neural controller consist of distances to the obstacles with respect to the position of the robot and the angle of the target. The output of the neural network is the angle of direction. De Momi and Ferrigno [60] propose a back-propagation algorithm in the healthcare field with the aim of assisting surgeons with a robotic system controlled by a high-level intelligent controller capable of collecting and integrating surgeon information, diagnostic imaging, and a series of sensors in the field. As for the use of control systems, Wong *et al.* [302] propose a novel modeling and optimization approach for the tuning of the performance in permanent and transient regime of an engine at idle. Regarding electrical control, a genetic algorithm and particle swarm optimization are applied to obtain an optimal fit of the control unit automatically. Gadoue *et al.* [82] present a comparison between four different speed controller design strategies based on AI techniques; two are based on the fit of conventional PI (Proportional-Integral) controllers, the third makes use of a fuzzy logic controller and the last is based on hybrid fuzzy slider control theory.

In terms of thought manipulation, this branch covers fields such as Natural Language Processing, Data Mining, Neural Networks, Machine Learning, and Pattern Recognition. A wide amount of works can be found in the related literature. To cite a few examples, Kohavi and Quinlan [129] present data mining tasks by using decision-tree discovery for classification. Xing *et al.* [305] propose a learning-based framework for robust and automatic kernel segmentation with shape preservation.. Krittanawong *et al.* [135] review recent applications of AI in cardiovascular clinical care and discuss its potential role in facilitating precision cardiovascular medicine. Calderon *et al.* [33] carry out the design and development of the system architecture to recognition of Electromyography signal patterns by using Feedforward-backpropagation Artificial Neural Network. Minaei-Bidgoli *et al.* [183] introduce an approach to classify students in order to predict their final grade based on features extracted from big data recorded in a web-based education system and also propose the use of a genetic algorithm for learning an adequate weighting of characteristics.

### 2.3.2 Applications of AI

After introducing some AI definitions and examples of use, this subsection focuses on the main applications of AI that have been developed during the past few years.

**Self-driving** or autonomous navigation means that a vehicle can plan its path and execute it without human intervention. This task is addressed by using data captured from sensors on board the vehicle and sometimes by using remote navigation aids [181]. This application is quite common in vehicles such as cars and trucks, but also for robots and UAVs (Unmanned Aerial Vehicles). Interest in such applications has increased in recent years due to the desire to develop a fully autonomous vehicle with

the main goal of reducing traffic accidents caused by humans. These systems basically consist of a mobile platform that integrates a set of sensors. The data collected by the sensors provide the perception of the environment. This information can be processed through AI algorithms with the aim of tackling the task of path-planning to move around the environment with minimal human intervention. Autonomous navigation also considers tasks such as move-on-route, obstacle detection and avoidance and leader/follower capabilities.

As for the use of AI for this application, a considerable number of works have been developed in recent years. Li *et al.* [155] introduce a fully autonomous navigation system for a smart microvehicle with a CCD camera (Charge-Coupled Device), an AI planner, and a magnetic field generator. The AI planner is divided into three functional modules: a computer vision module to track the microvessel and detect obstacles in its environment; a motion planner that generates an optimal unobstructed route between the point of departure and the destination; and a magnetic motion controller for manipulating the movement of the microvessel along a predesigned trajectory. Polvara *et al.* [220] propose a path planning and collision detection methods for autonomous Unmanned Surface Vehicles by using artificial neural networks and evolutionary algorithms. Sharma *et al.* [248] apply artificial intelligence to protect wireless communications from connected vehicles, which facilitates the exchange of security messages to prevent collisions in autonomous vehicles. The AI system learns to increase its ability to discern and recognize its environment. Badue *et al.* [13] conduct a survey on the state of the art in standalone performances focusing on published work since the birth of the DARPA (Defense Advanced Research Projects Agency) challenges. This survey focuses on perception systems and decision-making systems based on methods that make use of AI.

As for**face detection**, this is the preliminary step for face recognition, and it basically consists of detecting faces in images. These tasks have played an important role in robotics issues such as surveillance [122] and home service robots [116]. The face detection task can be addressed in two steps [308]. The first step is to find out if there is any face in a given image or not. The second step, if there is any face within the image, is to calculate where it is located. In the related literature, many works in this field can be found. Romdhani *et al.* [231] propose a face detection algorithm that is based on running an observation window in all possible positions and using SVM to determine if a face is contained within the window. Ahuja *et al.* [122] propose local Binary Patterns (LBPs) to detect the ROI (Region Of Interest) of the face within the image and a Haar feature-based cascade classifiers for developing the face recognition. However, the revolution in this task comes when Viola and Jones [288] introduced the real-time face detector, which is capable of detecting faces in real time with high accuracy. The three main contributions of this work are the introduction of a new image representation that allows fast calculations, a simple and efficient classifier built through the AdaBoost learning algorithm [79] to select a small number of critical visual features from a very large set of potential features, and a method that combines cascading classifiers, allowing you to quickly reject background regions and spend more calculations on promising face-shaped regions. This task has been widely used to

identify multiple appearances in smartphone cameras [95]. Currently, face detection is usually proposed in any type of storage system or social media such as Google or Facebook, which use face detection in images uploaded to social media. [223].

Subsequently, face recognition is presented as a system for verifying the identity of a person identity between a set of candidates using as input a facial image and a database of facial images of known people [113]. This task has aroused enormous interest in the automatic processing of digital images in order to solve a variety of applications such as biometric authentication or surveillance [114]. Face recognition has been proposed during the past few years as an identification system in the same way that fingerprint and iris were proposed before. According to Abate *et al.* [2], face recognition systems fall into two categories: verification and identification. As for face verification, it is a one-to-one match that compares an image of a face, whose identity has to be recovered, against a template face. Furthermore, concerning face identification, it is a one-to-many problem that compares a candidate face image with all image templates that are contained in a face database with the goal to determine the identity of the candidate face. This application has been addressed for several purposes. For example, Kim [125] a security system that performs an automatic recognition for verification between the passport photo and the face of the individual; this work proposes a clustering algorithm that creates adaptive clusters to variations in input patterns and applies them to the extracted areas for recognition. In terms of surveillance, CCTVs (Closed-circuit television)can be used to search for someone. Wang *et al.* [296] use face recognition in real-world surveillance. They propose a CNN which is trained with a labeled dataset and subsequently proposes to recognize people on campus surveillance system.

Concerning **objects recognition and categorization**, these tasks have played an important role in robotics regarding building object-based representations of the environment and object manipulation. Object recognition basically involves detecting an object instance and object categorization involves classifying a specific object (such as a pen or a keyboard). [161]. Gao *et al.* [83] propose an object classification method using RGB-D data to train a convolutional neural network with the goal of detecting and categorizing common objects in an autonomous vehicle environment such as pedestrians, cyclists, trucks or even other cars. In a similar way, Zhu *et al.* [321] introduce a CNN to detect and classify traffic signs. In addition, there are several jobs that use this application to additionally perform a pose estimate of detected objects. Kanezaki *et al.* [121] propose CNNs to categorize objects from multi-view images and estimate their position. Wei *et al.* [297] developed an end-to-end Mask-CNN model that selects deep convolutional descriptors for fine-grained object recognition. Zaki *et al.* [313] propose a multi-scale feature representation based on a convolutional hypercube pyramid (HP-CNN) that is capable of performing a categorization of invariant semantic scenes and objects from the point of view.

**Object manipulation** or manipulation planning is a task related to the motion planning, but the focus is not on the movement of the robot, but on the objects to be manipulated. This task basically consists of changing the position and/or orientation of a specific object (or set of objects), avoiding collisions or breaking the objects

involved [117]. Interest in this application has increased substantially in recent decades with the aim of replacing human workers in challenging (due to the required accuracy) or dangerous tasks, especially in industrial, health and domestic environments [258]. Many works on manipulations and planning can be found in the bibliography that are based on AI tools. For example, Boularias *et al.* [30] introduce a robot system to catch objects in a dense clutter by using depth images. To do this, the robot learns to manipulate objects by trial and error using a decision-making problem based on a reinforcement learning framework. Yang *et al.* [310] propose a system that learns the action of manipulation by processing Internet videos using two CNNs, one for classifying the type of manual capture and the other for object recognition. Matas *et al.* [175] present a combination of state-of-the-art deep reinforcement learning algorithms to solve the problem of handling deformable objects.

### 2.3.3 Frameworks Commonly Proposed for Mapping and Localization

After presenting some definitions and common applications of AI in the field of robotics, this subsection presents some of the most popular AI tools used to address mapping and localization in mobile robotics.

#### 2.3.3.1 Machine Learning Classifiers

Classification is a task that predicts the class or category to which an 'object' belongs. The object is also known as a pattern and is assumed to belong to a single class among a set of categories. Each pattern is represented by a set of measures known as characteristics, which must provide sufficient class discriminatory information to predict the category of the pattern with high probability. [273]. Normally, $n$ feature variables, $x_1, ..., x_n$, are selected and arranged in a feature vector, $x \in \mathbb{R}^n$. The goal is to train a classifier whose function (or set of functions) $f(x)$ in $\mathbb{R}^n$ is capable of estimating the class which the pattern belongs to.

This technique has been widely used to solve a variety of problems. For instance, Atkinson and Campos [12]propose a feature-based emotion recognition model using a multi-class SVM with EEG-based brain-computer interfaces. Narudin *et al.* [195] use different machine learning classifiers to detect malware on mobile phones using the anomaly-based approach. As for the field of computer vision, there is a wide range of work using classifiers. For example, Korytkowski *et al.* [133] introduce a fuzzy classifier with local image features to do object classification. Zhang *et al.* [314] propose an automatic method for detecting defective apples by using a weighted relevance vector (RVM) machine classifier. Aguilar *et al.* [4] propose a pedestrian detector for UAVs based on a combination of Haar-LBP features with Adaboost and using cascading classifiers with Meanshift.

#### 2.3.3.2 Clustering

Classifiers, as described in the previous section, is a technique based on supervision, i.e. you need properly labeled data to carry out the training process. In this case,

clustering is an unsupervised technique, where class labeling of training patterns is not available. Therefore, the main goal is to know the organization of patterns into clusters (also known as groups). To organize specific data into clusters, a clustering (or several) criterion must be established. Then, each pattern is categorized into its respective group and each group is characterized by the common attributes of the data that belongs to it. [274].

This artificial intelligence tool has been commonly proposed in a wide range of fields related to robotics and computer vision. For instance, Dhanachandra *et al.* [62] propose image segmentation by means of k-means clustering. Schroff *et al.* [245] propose a system based on clustering and face recognition approach that directly learns a measure of facial similarity. Fan *et al.* [74] an unsupervised progressive learning method based on pedestrian pooling and fine-tuning a CNN to transfer previously trained deep representations to invisible domains. Wang *et al.* [293] propose an improvement in 3D object recognition by introducing a view clustering and pooling layer based on dominant sets. Additionally, clustering methods have been proposed to compress the information to obtain the high-level layers of hierarchical maps. This type of approaches have proved to be more effective than the traditional mapping methods when the data size is high. Furthermore, the Spectral Clustering developed by Ng *et al.* [200] confirmed to be a good solution to compress global-appearance description information [281, 264].

#### 2.3.3.3 Deep Feedforward Networks

Deep feedforward networks, also known as feedforward neural networks or multilayer perceptrons (MLPs), are models of deep learning whose goal is to approach some function $f^*$. This network defines a mapping $y = f(x;\phi)$ where $x$ and $y$ are respectively the input and output data.This network learns the value of the $\phi$ parameters that best approximate the function $f$ [90]. These models perform a flow through the $f$ function evaluating from $x$ to output $y$. However, these models do not provide feedback connections, that is, the model results are not fed back into the model itself. During training, the goal is to make $f(x)$ match $f^*(x)$:training data is provided and $f^*(x)$ is evaluated with this data. In addition, a label $y$ is included with each example $x$ to achieve $y \approx f^*(x)$.

These networks are extremely important in machine learning, as they are the basis of many applications. For example, many object recognition approaches are based on such models. Like the work of Mostajabi *et al.* [188], who propose a feedback architecture for semantic segmentation to address a rich representation of features that is used for object recognition.

#### 2.3.3.4 Autoencoders

An autoencoder is a neural network architecture composed basically of an encoder and decoder system whose goal is to find a compressed representation of the given input data. The process involves finding a representation or code to make useful transformations about the input data. Traditionally, autoencoders were proposed for dimensionality reduction or even for feature learning [90]. Denoising autoencoders try to find code

that can convert noisy data into clean data. In addition, autoencoders are also used to perform coloring, function arithmetic, detection, tracking, and segmentation, among others. As for the encoder, it transforms the input data $x$ into a latent low-dimensional representation $h = f(x)$. This latent representation is a low-dimensional vector. The encoder learns to extract the most important features from the input data. As for the decoder, it retrieves the input data from the latent representation, $r = g(h)$ with the goal that $g(f(x)) = r$, with $r$ being the closest possible at $x$. Overall, the encoder and decoder are non-linear functions and the dimension of the latent representation $h$ is considerably smaller than the input dimensions. Like other neural networks studied, the autoencoder attempts to minimize a loss function during the training process. The loss function is set to measure how different input $x$ and rebuilt input $r$ are. For example, the Mean Squared Error (MSE) is used for this purpose.

$$L(x, r) = MSE = \frac{1}{m} \sum_{i=1}^{i=m} (x_i - r_i) \tag{2.1}$$

where $m$ is the output dimensions ($m = width \times height \times channels$).

Autoencoders have been a successful tool for dimensional reduction and also for information retrieval. In terms of dimensionality reduction, this tool has provided reconstructions with a lower error rate than with other techniques such as PCA (Principal Component Analysis) [104]. Hence, by improving the representations of smaller dimensions, other related tasks have also been improved. First, in sorting tasks, autoencoders provide a model with fewer memory requirements and computational time consumption [168]. Second, by reducing dimensionality, information retrieval can be tackled more efficiently. With the use of autoencoders and their related dimensionality reduction, exhaustive search becomes more efficient. For instance, Pfeiffer *et al.* [218] present a study on classification learning approaches and refining queries for information retrieval in the pharmacogenomic domain. Zhu *et al.* [321] propose using an autoencoder for learning features from 2D images with the goal of performing 3D shape retrieval. In addition, autoencodershave been widely proposed to produce binary and low-dimensional encodings. This way, entries of a database can be stored in a hash table and information retrieval can be performed by returning all entries that have the same binary code as the query. To cite one example of this approach, Carreira-Perpinán, and Raziperchikolaei [37] introduce a fast search in image databases with binary hashing, where each high-dimensional, real-valued image is mapped with an autoencoder into a binary vector of low dimension and the search is performed in this binary space. In addition to these examples, many others can be found in the related literature on the use of autoencoders for mobile robotics. Sergeant *et al.* [247] tackle a navigation task by using a deep autoencoder that learns to navigate from sensory data stored in a dataset. Wang *et al.* [294] propose an autoencoder for the fusion and extraction of multiple visual characteristics of different sensors with the aim of carrying out a movement planning based on deep reinforcement learning. Fig. 5.5 shows the architecture design of the autoencoders for this purpose.

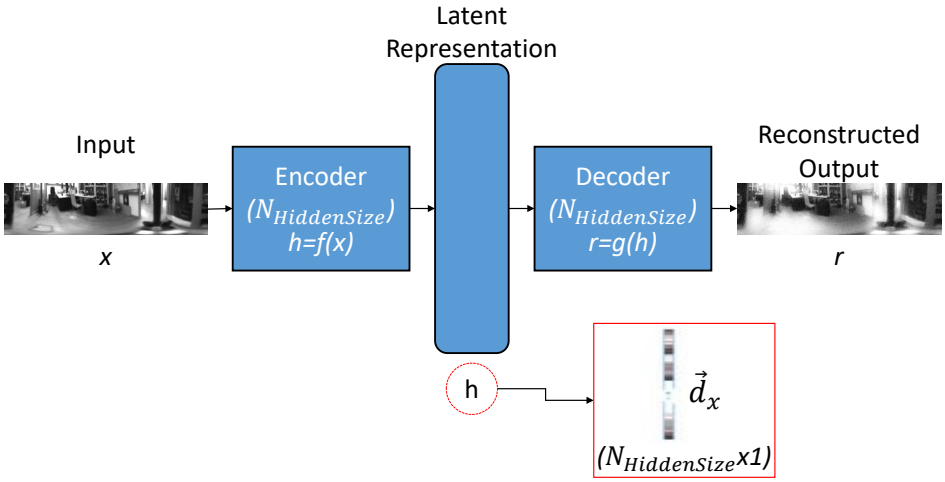



**Figure 2.4:** Autoencoder architecture design and extraction of features departing from the latent representation.

#### 2.3.3.5 Convolutional Neural Networks

Convolutional Neural Networks tool, commonly known for its acronym (CNNs), are currently the most popular tool among the deep learning techniques developed, as they have resulted in successful results in many practical applications. They are a type of specialized neural network for processing data that has an already known topology. These networks are commonly designed to receive images as input and have different applications such as classification or object detection. These types of networks are based on the use of convolutions, which is a specialized type of linear mathematical operation [90]. This means that while traditional neural networks use matrix multiplication with a separate parameter that describes the interaction between inputs and outputs, CNNs use specific and significant features obtained from the input data. CNNs consist of local connections between neurons and hierarchically organized transformations of data. Basically, CNNs are made up of three types of neural layers: convolutional layers, pooling layers, and fully connected layers. Each layer transforms the input and generates an output according to the established parameters. This process is addressed through several layers until the last layer is reached, which is a fully connected layer that generates a 1D feature vector that provides the most likely prediction.

Well-known CNN architectures have been used as a starting point to develop new artificial vision tasks. For example, AlexNet was presented by Krizhevsky *et al.* [136]. As shown in fig. 2.5, this network consists of eight layers (five convolutional layers and three fully connected layers) with a final 1000-way softmax and three grouping layers. The input image is $227 \times 227 \times 3$ and the network was trained to classify 1000 categories of objects, such as keyboards, pencils, and a variety of animals. GoogLeNet was proposed by Szegedy *et al.* [268]. This network has 22 layers, is also trained for

object classification, but uses 12 times fewer parameters than AlexNet. An extensive review of the top CNNs can be found in [206]. In addition, table 2.1 shows a summary chart of the most popular CNNs.

Moreover, there are other options that permit reusing robust CNNs that have yielded successful results, to solve different input image problems. On the one hand, the **transfer learning** technique involves the process of retraining a pre-trained network to classify a new set of images, that is, reusing the architecture, weights, and parameters of a CNN. which already works properly as a starting point for building a new CNN for a different purpose. The main idea is to take advantage of most of the intermediate layers, as their parameters have been fine-tuned with a large number of images. The problem, then, comes down to changing the final layers (to readjust to the proposed new task) and perhaps the initial layers (if the size of the images does not match the size used previously). Once the new network is established, the training process begins by using the new input data. The components needed for transfer learning are: pre-trained network layers, training data, and algorithm options. Therefore, this technique can save a considerable amount of time for training and even generate better results from creating a new network from scratch. This idea has been used by many authors. For instance, Han *et al.* [97] use CNN transfer learning along with augmented data to overcome good solutions despite the small size of the data sets used. In addition, as mentioned above, Wozniak *et al.* [303] use the transfer learning technique to retrain the VGG-F network and retrain it to classify places among 16 rooms acquired by a humanoid robot. On the other hand, many authors have also proposed the use of intermediate layers to generate global-appearance descriptors of the input image. In this sense, once the network is properly available to tackle the desired task, the hidden layers perform a vector description that can be used to characterize the input data. This idea has been exploited by some authors such as Arroyo *et al.* [8], who use a CNN that automatically learns to generate robust visual descriptors in the face of station changes, to perform a robust topological localization. Wozniak *et al.* [303] also use the features extracted from the $fc_6$ layer to train a linear classifier SVM. Mancini *et al.* [170] use this visual information to perform site categorization with a Naïve Bayes classifier.

**Table 2.1:** Summary of the most popular CNNs developed in recent years.

| CNN | Year | Developed by | No. of convolutional layers | No. of parameters |
|---|---|---|---|---|
| LeNet | 1998 | LeCun *et al.* [145] | 5 | 60,000 |
| AlexNet | 2012 | Krizhevsky *et al.* [136] | 8 | 60 million |
| GoogLeNet | 2014 | Szegedy *et al.* [268] (Google company) | 22 | 4 million |
| VGG Net | 2014 | Simonyan and Zisserman [255] | 19 | 138 million |
| Inception | 2015 | Szegedy *et al.* [268] | 65 | 5 million |
| ResNet | 2016 | He *et al.* [101] | 152 | 25.6 million |
| Xception | 2017 | Chollet [52] | 42 | 23 million |

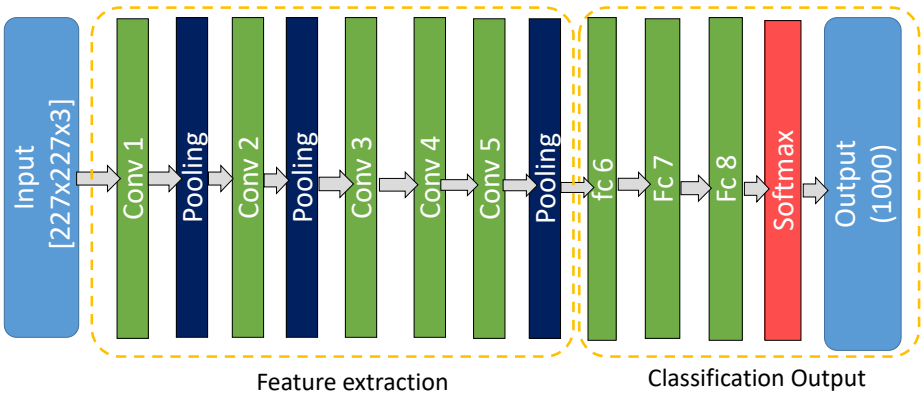



**Figure 2.5:** AlexNet architecture. Input images are $227 \times 227 \times 3$ and output can classify objects into 1000 categories.

As for the use of CNN to solve robotics tasks through visual information, there are many works that have yielded successful results when using this technique. For example, Sinha *et al.* [257] propose a CNN to process data from a monocular camera and address precise relocalization of the robot in indoor and outdoor environments without GPS. Wozniak *et al.* [303] use a transfer learning technique to retrain an existing CNN and retrain it to classify places among 16 rooms the image data was acquired by a humanoid robot. Payá *et al.* [213] propose using CNN-based descriptors to create hierarchical visual models for mobile robots localization. Chaves *et al.* [49] propose a CNN to construct a semantic map by means of using the network to detect objects in images, and then the results are placed within a geometric map of the environment. Xu *et al.* [306] propose a multi-sensor-based global indoor localization system that integrates visual localization with the help of CNN-based image retrieval with a Monte Carlo probabilistic approach.

#### 2.3.3.6 Regression neural networks

Regression neural networks are among the most popular deep learning tools. Deep neural networks are well known for classification problems, where the goal is to predict a single discrete label of input data. However, the regression problem is to obtain a continuous value. Hence, this type of network has been commonly proposed for continuous predictions like forecasting. Bilgili and Sahin [25] propose an analysis of regression neural network models to predict wind speed; and Kumar *et al.* [139] introduce a study on regression neural networks to estimate monthly mean global solar radiation. These models have also been proposed for other types of predictions, such as medical diagnoses. For example, Kayaer and Yıldırım [123] propose using a general regression neural network to diagnose diabetes. Ferreira *et al.* [76] use a general regression neural network to build the foundation of an adaptive neuro-fuzzy system and to tackle a foot control of an autonomous bipedal robot. Regarding mobile robotics, the related literature also presents different application examples. Wang *et al.* [295]

use a general regression neural network to approximate the functional relationship between high-dimensional map features and robot states. Rahman *et al.* [226] propose a location estimation algorithm using a generalized regression neural network and a wireless sensor network (WSN). Dezfoulian *et al.* [61] propose a method to interpret the data from various types of two-dimensional range sensors and a regression of the neural network to perform the navigation task.

## 2.4 Visual Description Methods

As presented in subsection 2.2.2, vision sensors have been widely used for mobile robotics purposes. However, images are highly dimensional data and they also change for several reasons apart from the movement of the robot, such as change of illumination or position of some objects that constitute the environment. Hence, the way to work with these data consists commonly in extracting the most relevant and invariant information from scenes. In this regard, two main approaches have been commonly proposed to extract the most relevant information from scenes; either by detecting, describing and tracking some relevant landmarks, or by working with global-appearance algorithms, that is, by building a unique descriptor for the image. On the one hand, methods based on local characteristics consist of extracting some highlights from each scene and creating a descriptor for each point, using the information around it. The most popular description methods used for this purpose are SIFT (Scale Invariant Feature Transform) [164] and SURF (Speeded-Up Robust Features) [17]. More recently, descriptors such as BRIEF (Binary Robust Independent Elementary Features) [34] or ORB (Oriented FAST and Rotated BRIEF) [234] have been proposed, trying to overcome some drawbacks such as the computational time and invariance against rotation. These descriptors have become very popular in visual mapping and localization and many authors have proposed methods that use them. For example, Angeli *et al.* employ SIFT [6] and propose this descriptor to perform a visual topological SLAM; Murillo *et al.* [191] use SURF to perform a localization with omnidirectional images; Campos *et al.* [35] propose the fusion of local features to perform a classification for the visual localization of robots. However, these methods have some disadvantages. For example, to obtain reliable landmarks, environments must be rich in detail. In addition, the detection of keypoints is not always robust to changes in environments (e.g., changes in lighting conditions) and sometimes, the description is not entirely invariant to changes in the position of the robot. In addition, these approaches can be computationally complex. Hence, in these cases, it would not be possible to build models in real time.

On the other hand, methods based on the global appearance of scenes consist of treating each image as a whole. This approach consists of working with the image as a whole, that is, without extracting any local information. For this type of approaches, each image is represented by a unique descriptor, which contains information about its global-appearance [211]. With regard to mobile robotics, this method of description has advantages in dynamic and poorly structured environments, where the extraction of stable local characteristics can be difficult. In addition, these methods lead to simpler localization and mapping algorithms, due to the fact that each scene is described by a single descriptor [5, 22].

Therefore, mapping and localization can be done simply by storing and comparing the descriptors in pairs. Besides, they could be more robust in dynamic, unstructured environments. Nonetheless, as drawbacks, these methods present a lack of metric information (commonly used to construct topological maps), visual aliasing can also have a negative impact on mapping and localization tasks, as environments interiors are prone to present repetitive visual structures. In addition, modeling large environments requires a large number of images, and this can present serious problems when these techniques have to be used in real-time applications. Therefore, using holistic descriptors is an intuitive alternative to solve the mapping and localization problem, but its robustness against these issues must be tested. Many authors have addressed mapping and localization using global-appearance descriptors. For example, Menegatti *et al.* [178] use the Fourier Signature to construct a visual memory of a relatively small environment from a set of panoramic images. Liu *et al.* [159] propose a descriptor based on color characteristics and geometric information for the recognition of scenes with omnidirectional vision in topological maps. Through the use of this type of descriptors, topological maps can be constructed using this type of descriptor. For instance, Payá *et al.* [210] propose a mapping method from global-appearance and solve the localization in a probabilistic way using a Monte Carlo approach. In addition, they develop a comparative analysis of some description methods. Rituerto *et al.* [230] propose the use of the descriptor *gist* [150, 204] to create topological maps from omnidirectional images. More recently, Berenguer *et al.* [21] propose the Radon transformation as a holistic descriptor of omnidirectional images and a hierarchical localization method. By this method, in the first place, an approximate localization is obtained; after that, a local topological map of a region is created and used to refine the localization of the robot.

### 2.4.1 Local Features by Using AI

Since the emerge of SIFT a considerable number of methods of extraction and description of local features have been developed. Many later developments focused on reducing their computational requirements or improving invariability for other purposes. For example, SURF has a lower computational cost and greater robustness compared to image transformation and BRIEF is designed to be used in real time at the expense of lower tolerance to image distortion and transformations. All of these examples are known as traditional or hand-crafted features, as they are based on the detection of visual structures. An profound study of these tools can be found in [211] and Mukherjee *et al.* [190] carried out an exhaustive comparative experimental study.

As for the development of local features based on AI techniques, they are widely known as learned features and, similar to the hand-crafted, AI methods generally consist of detecting or describing features, or even both (detect and describe). FAST (Features From Accelerated Segment Test) [232] was one of the first successful methods and is designed for high-speed corner detection. Despite being built primarily for speed purposes, this method was also shown to outperform existing corner detectors. Subsequently, was proposed to optimize the parameters of the FAST detector to achieve image repeatability [233]. This work shows that the use of machine learning produces

significant improvements in repeatability, speed, and quality. The first attempts were based on genetic algorithms. In this sense, Trujillo and Olague [278] present an approach to automatically extract low-level features through the application of genetic programming. These authors present an implementation of genetic programming that is capable of discovering a modified version of a feature operator that addresses a high level of performance. This work also highlights the balance between genetic programming and domain knowledge experience for results that improve hand-crafted solutions. More recently, machine learning tools have typically been used in feature detection to mimic and/or accelerate previously defined methods. Šochman and Matas [260] propose a faster version of binary decision algorithms by using a WaldBoost classifier, which learns to minimize the decision time of the classifier while guaranteeing a predefined accuracy. Holzer *et al.* [108] carry out the Interest Point (IP) as a regression problem through the use of machine learning. A regression forest (RF) model learns to detect if there is an IP in the center of a given image patch.Other researchers use machine learning to reduce the size of the descriptor, such as Strecha *et al.* [265], who propose metric learning to reduce the size of descriptors by representing them as short binary strings. In summary, they map the descriptor vectors in the Hamming space, which is used to compare the resulting representations. In this way, the size of the descriptors is reduced by representing them as short binary strings. Simonyan *et al.* [255] develop a descriptor of local features learned through the use of convex optimization. This work shows that learning clustering regions for the descriptor can be formulated as a convex optimization problem. It also shows a reduction in the dimensionality of the descriptor by using the regularization of the nuclear norm of the Mahalanobis matrix. Both formulations are based on high-margin discriminatory learning restrictions.

As for features learned based on deep learning techniques, they have often been used to improve rather than replace hand-crafted local features. For example, they have been used to learn detectors of invariant covariant features in the face of unsupervised point of view changes. For instance, Lenc and Vedaldi [147] propose a general machine learning formulation for covariant feature detectors. In addition, many other improvements can be made, such as the explicit inclusion of confidence in model detection, the prediction of multiple features in a patch, or the joint training of detectors and descriptors. Mishkin *et al.* [185] introduce a method for learning local related covariant regions. The proposed similar shape estimator is trained considering the loss function, the type of descriptor, the geometric parametrization, and so on. In addition, the training process does not require geometrically accurate aligned patches.

However, despite the widespread use of deep convolutional networks, locally invariant features based on hand-crafted techniques continue to play an important role in applications such as image retrieval and motion. Recently, Lenc and Vedaldi [148] have conducted an profound evaluation of local feature detectors by evaluating a range of state-of-the-art local feature detectors. Through this study, they concluded that machine learning-based detectors help to improve lighting invariance, that traditional methods are still competitive, and also suggest that significant progress can be made with respect to machine-based detectors in deep learning.

### 2.4.2 Holistic Description by Using AI

A wide range of works have been proposed during the past few years to develop holistic descriptors by using AI techniques. Known by some authors as feature engineering, this is a way to take advantage of human inventiveness and prior knowledge to compensate for weakness of current learning algorithms [20]. One of the main objectives of developing learning descriptors is to address faster solutions to proposed AI problems. In addition, AI applications have been shown to be capable of understanding the environment that sorrounds the camera, thanks to its ability to identify interesting information and reject unprofitable information from sensory data. Highlighting holistic descriptors based on deep architectures, they are generally effective at training robust models and introduce two advantages in this topic: first, deep architectures promote feature reusal and second, they lead to more abstract layered features superiors that are typically invariant to local variations. A deep study was carried out in [20] about unsupervised feature learning techniques.

Among the first proposed techniques, **PCA** [128] was one of the first alternatives that presented robustness. PCA basically performs a linear transformation $h = f(x) = W^T x + b$ of the input $x \in \mathbb{R}^n$ and the results are $d_h$ features that are the first components of the representation $h$. Similar to PCA, Independent Component Analysis (**ICA**) performs a linear analysis to obtain distinctive features based on linear generative models with non-Gaussian independent variables. Like sparse coding, ICA and its variants have also been used to obtain non-linear features as in [19, 120, 144].

Successful feature learning algorithms and related applications are used in many works using a variety of approaches such as **RBMs** (Restricted Boltzmann Machines). For instance, Hinton *et al.* [103] propose a technique of stacking previously trained RBMs into deep belief networks (DBNs), where the top layer is interpreted as an RBM and the bottom layers as a directed sigmoid belief network. This work has been shown to provide a better digit classification than discriminative learning algorithms. Salakhutdinov and Hinton [240] propose to combine the parameters from RBM to DBM (Deep Boltzmann Machines) by halving the RBM weights to obtain the DBM weights and train them with an approximate maximum probability. In this way, this work shows that DBM learns good generative models and performs well in visual object and handwritten digit recognition tasks. Larochelle *et al.* [143] conduct an empirical study on the use of different RBM input unit distributions. This study confirms the hypothesis that the greedy unsupervised layered training strategy improves optimization by initializing weights in a region close to a good local minimum and also leads to better input generalization.Another important perspective on holistic descriptors is based on the **manifold learning**, a geometric notion the premise is based on the concentration of high-dimensional input space in the vicinity of a manifold $M$ of lower dimensionality. Most methods based on this technique lead to a nonparametric approach based on neighbour graphs. Belkin and Niyogi [18] propose a geometric algorithm to represent high-dimensional data that provides a computationally efficient dimensionality reduction. This reduction has preservation properties of the locality and a natural connection to clustering. Donoho and Grimes [67] propose a Hessian-based local linear embedding method to retrieve the underlying parameterization of scattered data.

Weinberger and Saul [298] introduce an algorithm for unsupervised learning of image manifolds by semidefinite programming. The algorithm calculates a low-dimensional representation of each image so that the distances between nearby images are preserved. Van der Maaten [285] proposes variants of the Barnes-Hut algorithm with the t-SNE algorithm (t-distributed Stochastic Neighbor Embedding) to learn embedding data sets with millions of objects. More recently, some authors have proposed using free energy functions, that is, without explicit latent variables. For example, Ngiam *et al.* [201] use a Monte Carlo hybrid to train the free energy function. In summary, they propose the use of deep-feeding neural networks to model the energy landscapes that define probabilistic models. The lower layers of the model adapt the training of the upper layers and therefore this produces better generative models. Using this method, all layers of the model are trained simultaneously and efficiently. Kingma and Cun [127] propose to eliminate noise from the coincidence of scores. The differentiation of the loss with respect to the parameters of the model is automated with an extended version of a double backpropagation algorithm.

In addition to the techniques mentioned above, the use of deep neural networks, especially CNNs, to obtain global-appearance descriptors, as several studies have shown that these networks can learn more transferable features for the adaptation of domains and produce successful results in a wide range of scenarios and applications. For example, Donahue *et al.* [65] propose the use of features extracted from the activation of a fully supervised deep convolutional network trained in a large, fixed set of object recognition tasks and use them for a completely different task. The works focuses on investigating the semantic clustering of deep convolutional features with respect to a variety of tasks, as scene recognition, domain adaptation, and detailed recognition. The study addresses an efficiency comparison that relies on several levels of network to define a fixed function. Yosinski *et al.* [312] conduct an profound study of how transferable features are in deep neural networks. They conclude that the characteristics obtained from the initial layers do not seem to be specific to a particular data set or task and the characteristics become more specific as the selected layer approaches the last one. In addition, they conclude that the initialization of a network with characteristics transferred from almost any number of layers can drive generalization. Long *et al.* [162] propose a Deep Adaption Network (DAN) architecture that generalizes deep CNNs to the domain adaptation scenario. The DAN architecture learns transferable features and can scale linearly using an unbiased estimate of kernel incorporation. Arandjelovic *et al.* [7] solve the problem of image retrieval by developing a CNN and using it to obtain holistic descriptors. This network incorporates a new layer which is inspired by the "Vector of Locally Aggregated Descriptors" (VLAD) image representation, which is widely proposed to address image retrieval tasks. Gordo *et al.* [91] introduce a method that employs a region proposal network to find out which regions need to be grouped to form the final global descriptor. This approach produces a global image representation in a single step forward. Very recently, Xu *et al.* [306] propose a transfer learning method based on a previously trained model to transform general characteristics into special characteristics, which are adapted to the desired task. The previously trained Faster R-CNN model is used to extract the high-dimensional convolution features from the images.

## 2.5 Mobile Robotics Tasks by Using Vision and AI

This section presents a review of recent work related to solving mapping, localization, and SLAM tasks in robotics using visual information and AI tools.

### 2.5.1 Map Building

As explained above, mapping basically involves creating a map, model, or representation of the environment using the data provided by the sensors mounted on the robot. Thrun [275] presented an exhaustive of the concept of robotic mapping. As for the use of vision systems together with AI for mapping, a wide range of work has been proposed in recent years. For instance, Tanzmeister *et al.* [270] propose an approach that estimates a uniform, low-level, grid-based global model that includes dynamic and static objects. Da Silva *et al.* [57] propose a localization and navigation approach for mobile robots using topological maps and using CNN to obtain omnidirectional image descriptors. Kuipers *et al.* [138] address a mapping process by means of hierachical models, they propose a hierarchically hybrid map, which consists of using a metric map to construct local small-scale space maps and topological maps to represent the structure of space in large scale. This approach is proposed to solve the SLAM task in an environment with multiple large-scale nested loops.

Regarding mapping by means of visual data information, the models are built by either using local or global features. Furthermore, the use of AI with vision systems has contributed to the emergence of new paradigms for creating visual maps. Zivkovic *et al.* [323] build a hierarchical model based on omnidirectional images and the characterization of the data is done by means of local characteristics (SIFT) and a cluster algorithm is proposed to address the graph partitioning, and define the map hierarchically. Peretroukhin *et al.* [215] propose the use of Bayesian Convolutional Neural Networks (BCNN) to train and implement a sun detection model from a single RGB image to incorporate global orientation information from the sun into a visual odometry pipeline. They also propose an uncertainty associated with each prediction by using a Monte Carlo dropout scheme. Clark *et al.* [53] perform a mapping and subsequent relocalization task by feeding an LSTM network with holistic descriptors obtained from a CNN.The proposed model estimates the current position within an environment from short sequences of monocular frames. Similar to this work, on the use of CNN to obtain holistic descriptors, many authors have proposed this strategy. For example, Iyer *et al.* [112] propose an estimation approach based on self-supervised visual odometry. The approach first obtains holistic descriptors from the fully connected layer of CNN VGG-11. After that, an LSTM network is used to make a pose transformation regression between sequences of pairs of monocular frames. Kopitkov and Indelman [131] propose an approach to estimate robot position through holistic CNN descriptors and the use of neural networks to learn a generative model depending on the point of view of CNN characteristics given the position of the robot and approximate this model by a spatially varying Gaussian distribution. In addition, once the proposed model is developed, it is used within a Bayesian framework of probabilistic inference to solve the localization task. Sarlin *et al.* [242] propose a hierarchical model by means of a CNN.

This network simultaneously predicts local characteristics and holistic descriptors that are used for an accurate 6-DOF localization. Once the model is constructed, the coarse localization is solved using global retrieval through a k-nearest neighbour algorithm and holistic descriptors. Fine localization is solved by evaluating the matching points of the local characteristics.

Another widely developed strategy is the use of neural networks to model a system that is capable of estimating position directly from raw data. For instance, Kuse *et al.* [141] propose a deep residual network to model the representation of the environment. Naseer and Burgard [196] developed a model that allows 6-DOF localization using a regression neural network and a single monocular RGB image. The resulting map size is constant with respect to the size of the dataset and during the localization task, the time complexity is also constant and independent of the size of the dataset. Walch *et al.* [292] introduce a CNN + LSTM model to estimate the pose in both indoor and outdoor environments. The raw data is entered into the network and trained so that CNN layers learn the appropriate local characteristics and then are used by LSTM layers to improve the pose estimate. In this way, the whole network learns how to optimize the localization task. Brahmbhatt *et al.* [31] propose a mapping model based on a regression neural network, which enables learning a data-driven map representation. In addition,the proposed network can be updated with unlabelled data. Sinha *et al.* [257] also propose a mapping and a subsequent localization task based on regression neural networks. The proposed method first trains a CNN that takes RGB images from a monocular camera as input and performs regression for robot pose estimation. It then incorporates the relocalization output of the CNN in an Extended Kalman Filter to tackle the localization task. Moolan-Feroze *et al.* [187] propose the deployment of a model to map the environment that surrounds wind turbines. For this purpose, a CNN is trained to extract an estimate of the projection of the 3D skeleton representation departing from monocular images. After that, the localization task is solved by means of a pose graph optimization that uses the 3D representation outputs from the CNN.

### 2.5.2 Localization

As it was denoted in 2.1.2, localization is the task that attempts to estimate the current position and orientation of the robot within a model. Filliat and Meyer in [77] presented a profound review about strategies for addressing localization on mobile robots. Concerning the use of vision systems together with AI, a wide range of works have been proposed in recent years. For instance, Kendall *et al.* [124] present a robust and real-time monocular 6-DOF relocalization system. The proposed system trains CNN to retrieve the position of the 6-DOF camera from a single RGB image from one end to the other without additional graphics optimization. Neto [199] proposes a topological localization system based on monocular images, learning classification systems, and self-organized maps (SOM). The entire system performs a localization task by detecting and avoiding obstacles using both local and holistic features. Meng *et al.* [180] carry out the localization problem by using random forest-based methods that directly estimate 3D positions with SIFT features as input. Li *et al.* [154] introduce an

indoor localization approach serving a dual-flow regression CNN by entering color and depth data from monocular images. This system is tested in night lighting conditions and also with blur effects. As with mapping, there is also a wide range of work proposed over the past few years that proposes and evaluates the use of intermediate layers from various CNNs to obtain local features and holistic descriptors. For example, Sünderhauf *et al.* [259] present a real-time localization recognition algorithm using different layers of CNN to perform localization on large maps by integrating a variety of existing optimization techniques, such as semantic partitioning of search space.

Cascianelli *et al.* [38] propose a subsequent mapping and localization strategy based on the use of a CNN to obtain local features that are robust to appearance variations. Similarly, Unicomb *et al.* [280] also use a CNN to extract local features; in this case, they extract the map edges of the earth and then estimate a 6-DOF position using an EKF (Extended Kalman Filter) algorithm. Moolan-Feroze and Calway [186] present a framework that uses CNN to predict feature points of objects that are out of view in the input image. These feature points are then fed to estimate more robustly the position of the robot within the environment. Holliday and Dudek [105] propose a combination of deep-learning-based hierarchical object features and SIFT characteristics. These points are used to perform more robust localization tasks.Regression networks have been also widely proposed to directly estimate position within map. For instance, Sommer *et al.* [261] perform a 6-DOF localization task by developing a regression CNN by applying transfer learning over a previously trained CNN (Google Inception-V4). Xu *et al.* [306] introduce a multi-sensor-based global indoor localization approach that uses visual localization with the help of CNN-based image retrieval with a probabilistic Monte Carlo method. Cattaneo *et al.* [39] develop a regression network, which learns to localize an RGB-D image of a scene on a map constructed from LIDAR (Laser Image Detection and Range) data. Similarly, Weinzaepfel *et al.* [299] introduce a CNN-based regression strategy for visual localization from a single RGB image that is based on densely matching a set of objects of interest. Given a query image, the network model detects the object, segments it, and finds a dense set of 2D-2D matches between each detected object and its corresponding reference image. Given these 2D-2D matches, a Perspective-n-Point problem is used to estimate the pose.

### 2.5.3 Simultaneous Localization and Mapping

n addition to the mapping and subsequent localization task, SLAM presents a combined alternative. This process involves continuously building a map and updating it as the robot simultaneously estimates its position within the model. Fuentes-Pacheco *et al.* [81] presented a profound review about strategies for conducting SLAM. The related literature shows that not many approaches have been proposed to solve this task by using visual information and artificial intelligence tools. Apart from some of the examples discussed in subsections 2.5.1 and 2.5.2, which propose mapping or localization tasks with the aim of developing later SLAM, there are other examples that propose complete SLAM systems. For example, Wu and Qin [304] propose a SLAM algorithm based on omnidirectional images. This algorithm uses incremental learning of the appearance of reference points to provide a later probability distribution

to estimate the position of the robot in a particle filtering frame. The main contribution of the work is to represent the subsequent robot pose estimation by using incremental probabilistic PCA, which can be incorporated into the particle filtering algorithm for SLAM task. Lu *et al.* [165] propose a machine learning tool known as multi-task point retrieval to develop a regression model based on local characteristics of 3D points extracted from monocular images. Garg *et al.* [86] address an unsupervised deep convolutional network that behaves like an autoencoder, as it does not require ground truth information. This network is trained to predict the depth map of the source image. During the training step, a couple of images (source and target) are fed into the network. Schmidt *et al.* [244] introduce a CNN to produce robust local features and then use them to estimate dense correspondence and solve the SLAM task. Gao and Zhang [84] developed a work in which they perform a loop detection by training an autoencoder that calculates local characteristics from monocular images. Tateno *et al.* [272]propose an approach consisting of two CNNs based on monocular RGB images. The first network is trained to predict depth. CNN-predicted dense depth maps are naturally fused together with depth measurements obtained from direct monocular SLAM, based on a scheme that privileges depth prediction in image localization and the second network is trained to address segmentation semantics. Once the information is obtained from CNN, it is merged with more data to address the SLAM task in a highly accurate way. Mukasa *et al.* [189] introduce a SLAM framework that integrates geometric measurements obtained from a monocular vision system with predicted depth information using a CNN. Tang *et al.* [269] introduce the use of CNN powered by visual information and human voice commands with the goal of successfully solving the SLAM task on a mobile robot. Focusing on visual information, raw data is entered into two CNNs, the first network is used to produce accurate localization updates, and the second is used to perform an object recognition task. Recognized objects are used to reinforce the mapping task. Zhang *et al.* [315] propose a loop closure detection framework based on CNNs. This way, the images are introduced into a previously trained CNN model to extract global-appearance descriptors, and after that, these descriptors are preprocessed with PCA.

Very recently, Milz *et al.* [182] conducted an exploration of the deep learning tools that can be used to improve visual SLAM. On the one hand, they propose the use of CNN to perform depth estimation. they propose the use of CNN to address an end-to-end approach to learning feature matching. Since this technique can learn diversity and distribution instead of choosing the superior features of high textured features. Zhong *et al.* [317] carries out SLAM and object detection by using a Single Shot multi-box object Detector (SSD). RGB-D information is entered into the SSD and detects moving and static objects within the image. After that, the detected moving objects are removed and the rest of the data is used to construct a semantic map composed of all the static objects detected in the mapping thread. At the same time, dynamic objects are used to update the local tracking and mapping thread. Liang *et al.* [156] address visual navigation and SLAM tasks in outdoor environments by mean of a CNN based on panoramic images with information from 360 degrees to perform. Bloesch *et al.* [27] perform a compact but dense representation of scene geometry based on a deep autoencoder. This method is suitable for solving a dense monocular SLAM task

based on keyframes Liu *et al.* [160] propose a visual SLAM based on features by means of using a CNN to get more robust object localization information. Lu and Lu [166] propose a SLAM approach that uses a regression CNN for estimating pose without ground truth.

## 2.6 Publications Related to this Chapter

The main results presented in this chapter are related to the following publications:

- S. Cebollada, L. Payá, M. Flores, A. Peidró and O. Reinoso. A State-Of-The-Art Review on Mobile Robotics Tasks Using Artificial Intelligence and Visual Data. Expert Systems with Applications. Ed. Elsevier. pp. 114195 (November 2020) [40] **JCR-SCI Impact Factor (2019): 11.0**, Quartile **Q1**.

  - This paper presents a state-of-the-art review that focuses on how researchers have addressed relevant tasks in mobile robotics through the use of artificial intelligence and visual information; and how these approaches have evolved in recent years. This work focuses in the main tasks that should be addressed for mobile autonomous robotics, that is, mapping, localization, SLAM, navigation and exploration. The review is divided in three main blocks. First, relevant AI tools in the fields of mobile robotics and computer vision are outlined, among which Neural Networks stand out. Second, the review presents a state of the art of the works developed during the past few years regarding the description of the visual information by using AI tools. Finally, a state of the art of the works developed during the past few years to address mapping, localization, SLAM, exploration and navigation by using AI and visual sensors is presented.

# 3 Alignment of Point Clouds

## 3.1 Introduction

There are many buildings in Europe and around the world that have voids between foundations and floor due to building techniques. This type of uninsulated suspended timber floors can be a critical factor in heat loss, as some studies show. [99] [1]. This includes conductive heat loss to the ground and also infiltration of cold air through the underfloor environment and wooden flooring. Given this fact, the energy efficiency of these existing buildings can be improved by applying insulation under the floor. This process usually involves removing the carpet and floor boards, applying rigid panels or insulation rolls, and rebuilding everything. Therefore, this causes a lot of inconvenience to the occupants of the building as they often have to leave the premises during installation. To make this process less disruptive and faster, a robotic vehicle can be used. This robotic vehicle must be able to access gaps, move autonomously and apply foam insulation. An autonomous mobile robot that can maneuver around the void and apply insulation where necessary. Within this group, we can distinguish between three main types: robots with legs, with wheels and aerials. When choosing one of these three types, two main requirements must be considered. First, an umbilical hose must be connected to the robot to transmit energy and supply the robot with foam insulation. To meet these requirements, the hose would weigh about 3.5 kg per linear meter [106]. Second, a sprinkler filter must be mounted on the robot to expel the foam to the bottom of the floor. During this process, it will exert pressure on the robot. Therefore, the robot must remain stable. For these reasons, neither aerial nor leg robots would work properly. In addition, an aerial robot would also create too much dust and disturbance. Hence, a platform with wheels is used in this work.

This work was conducted in collaboration with the company Q-bot. This company has developed a robot to address the insulation task. As it is shown in the fig. 3.1, the robot consists of 4 small wheels, a horizontal front laser and a laser-powered camera system (3D scanner). Moreovoer, it has a spray nozzle that expels insulating foam. The vendor website[1] as well as the work presented in [106] provide further information about the robot specifications.

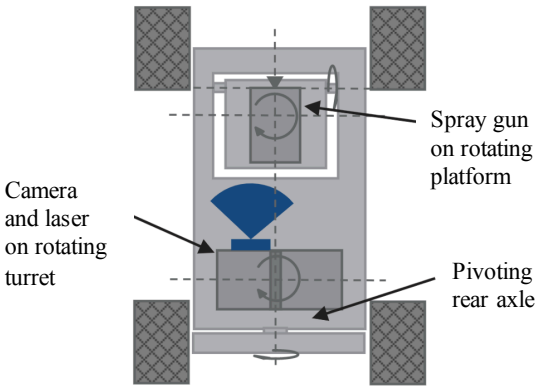



**Figure 3.1:** Bird eye's view of the robot with the main components.

The insulation task has been commonly developed through teleoperated assistance, i.e. the robot vehicle is driven by an expert human operator [119]. When a mobile robot is used to address this task, the robot needs to move around the environment in order to apply foam insulation in the required areas. When assisting teleoperated, the human operator recognizes and interprets the environment and makes decisions about moving the robot and the tasks to be tackled. However, despite the successful use of teleoperated robots, the use of autonomous robots would improve performance and speed while reducing costs. They could complete the task without the continuous supervision of skilled workers.

If autonomous development is desired, many problems arise and must be addressed accurately. First, access to the environment can be done by making an access hatch and placing the robot in the underground environment. Once the robot is inside, a number of challenges must be overcome. The main problems arise because the terrain tends to be extremely rugged, as there are often stones, fragments of bricks or sand. In this way, the robot must move along irregular 3D paths, also taking into account the presence of unknown obstacles. Figure 3.2 shows two sample images of typical underground environments. The robot must tackle the insulation task in such challenging and previously unknown environments.

To carry out this task in an autonomous way, the robot must be firstly able to map the environment with accurately with the aim of recognizing the areas where

[1] http://www.q-bot.co

**Figure 3.2:** Images obtained from typical underground environments where the registration task is difficult to address due to their characteristics.

insulation is required. In addition, the robot bust be capable of estimating its position within the map with accuracy. If both processes are tackled at the same time, this is known as *Simultaneous Localization And Mapping* (SLAM). To address mapping and localization, it is necessary to use one or more sensors to obtain some information about the environments. On the one hand, the SLAM task has traditionally been addressed using range sensors, such as the laser, which measures distance to the environment and usually leads to models showing occupied and unoccupied areas. [54] [80]. On the other hand, vision systems can also be used for this purpose and many researches has been proposed on mapping and localization using locally based methods such as SIFT, SURF, FAST or Harris corners. [16]. These features are detected in each frame and after that, they are matched with a sequence of frames. Numerous researchers have successfully addressed the mapping and localization tasks in controlled environments through such vision systems as Davison *et al.* [59] who use a single camera. Nonetheless, if the proposed process is not robust enough, the tasks are prone to fail. This problem is common in such unstructured and changing environments. Additionally, the environments studied in this work present also elements such as dust, sand, poor illumination or shadows and they provoke mapping and localization process extremely complex. Extreme unevenness of the ground is an additional issue to consider (see fig. 3.2). Because of this, the movement of the robot is not flat at all and can be considered a movement of 6 DoF (degrees of freedom). Given all these peculiarities

and challenges, both a laser and a vision system are considered to accurately construct a map of the environment and are installed on the robotic platform.

By using the visual and depth sensors, mapping and localization are addressed by means of the following process. First, from a specific pose of the environment, a scanning process is performed. During it, the sensors capture information on 360 degrees around the robot. The result is a local 3D map consisting of a point cloud that combines depth and color information. Once the local map has been built from a specific pose, the robot will move a relatively long distance to a new, unknown pose. To estimate this new pose, the environment will be scanned again and a new local map (point cloud) will be built. The translation and rotation from the first to the second position can be estimated by comparing these two local maps. After repeating this process from various positions, the set of local maps will compose a complete description of the environment (global map). This global map can be used not only to estimate the position of the robot as it moves, but also to determine the physical properties of the environment under the floor to control the spray gun and, after the insulation step, to validate if all areas are properly covered with foam. Also, in terms of the relatively high distance between consecutive positions, this is due to the fact that the data acquisition process takes quite some time [119]. Therefore, the pose estimation algorithm should work well considering this additional constraint.

Hence, obtaining a robust and accurate global map is very relevant. With this aim, having a precise knowledge of the pose where each local map was captured is crucial. This is why this work focuses on this problem: estimating the current pose of the robot with respect to the previous one. The problem will be solved either by using the point clouds obtained from both poses using a registration approach, or by using visual information to achieve robust matches with global-appearance and SURF features. Odometry information will not be used because the extreme unevenness of the ground introduces a severe error on it (by displacements, landslides and changes of orientation). In this way, the global map is expected to be robust to these phenomena.

On account of the issues exposed, in this chapter, an approach to solve the alignment between two consecutive locations is presented. It is capable of building a global map of the environment so that the robot can autonomously tackle the task of isolation in this type of environments. In this sense, the work is based on the robot architecture presented by Julia *et al.* [119]. In this previous work, a system was proposed for selecting the next best position for performing a 3D scan. Multiple scans were aligned using the Iterative Closest Point (ICP) algorithm and merged together into a global map model. However, this system depends on a correct functioning of the ICP algorithm which is prone to fail or converge to a local minimum due to the complexities of the underfloor environments. Consequently, we present an algorithm that solves the registration in such environments. Thus, the major contributions of this chapter are a novel method to enhance the results of the registration algorithm and an algorithm for making the global mapping process more robust against alignment failures.

The remainder of the chapter is structured as follows: Section 3.2 shows the acquisition system which was used in the experiments. Next, section 3.3 presents the

methods proposed to obtain the alignment between poses the algorithm to solve wrong alignment cases. Section 3.4 presents the experimental results and the discussions. Section 3.5 outlines the conclusions. Last, section 3.6 presents the publications related to the present work

## 3.2 Acquisition of the Data

The data acquisition system consists of a 2D laser sensor and a monocular camera. They are attached to a turret that rotates around its vertical axis, which is perpendicular to the base of the robot. The laser is mounted for scanning in a vertical plane, which changes as the turret rotates. This system acquires complete 360-degree information from the environment surrounding the robot. Hence, from a specific pose, the robot can capture a scan of the environment and a set of RGB images (see fig. 3.3 (a)). The complete system is described introduced in the present section. Additionally, subsection 3.2.1 describes the reference systems. After that, subsection 3.2.2 details the image acquisition process and the process to assembly the 3D point cloud. Last, subsection 3.2.3 explains how the color information is added to the depth data.

### 3.2.1 Reference Frames

This work uses three reference frames: the robot, the camera and the laser reference frames. Fig. 3.3 (a) shows the robot reference frame. $X_R$ is the vertical axis and $Y_R, Z_R$ are the axes which define the plane of movement of the robot. Fig. 3.3 (b) shows the camera reference system, whose axes are $X_C, Y_C, Z_C$. Last, fig. 3.4 shows the laser reference system $\{X_L, Y_L, Z_L\}$. The laser as well as the camera frames rotate around their $X$ axes. The laser provides a set of distance readings $\rho_i$ that are measured at different angles $\theta_i$. These readings are expressed as 3D points in the laser frame $q_i^{[l]} \in \mathbb{R}^3 = [\rho_i \cos\theta_i \quad \rho_i \sin\theta_i \quad 0]^T$ and they can be transformed to the robot frame by:

$$q_{i,j}^{[f]} = R_{\phi_j} T_L q_{i,j}^{[l]} \tag{3.1}$$

where $T_L \in \mathbb{SE}_3$ is the transformation that relates the calibrated position of the laser in the robot frame and $R_{\phi_j} \in \mathbb{SO}_3$ is the rotation matrix that expresses that the turret has rotated an angle $\phi_j$. Furthermore, the conversion between 3D vectors and the corresponding homogeneous 4D vectors is omitted with the aim of simplifying the notation.

### 3.2.2 Point Cloud and Image Acquisition

During a complete acquisition process, the robot remains stable at a specific position $\vec{p}_s$ and both the camera and the laser capture data as the turret rotates a complete revolution around the axis vertical. In this way, this data contains 360-degree environmental information around the robot. On the one hand, the camera acquires 37 RGB images as the turret rotates a complete revolution, with orientations evenly distributed around the $x$ axis of the robot frame. The number of 37 RGB images per

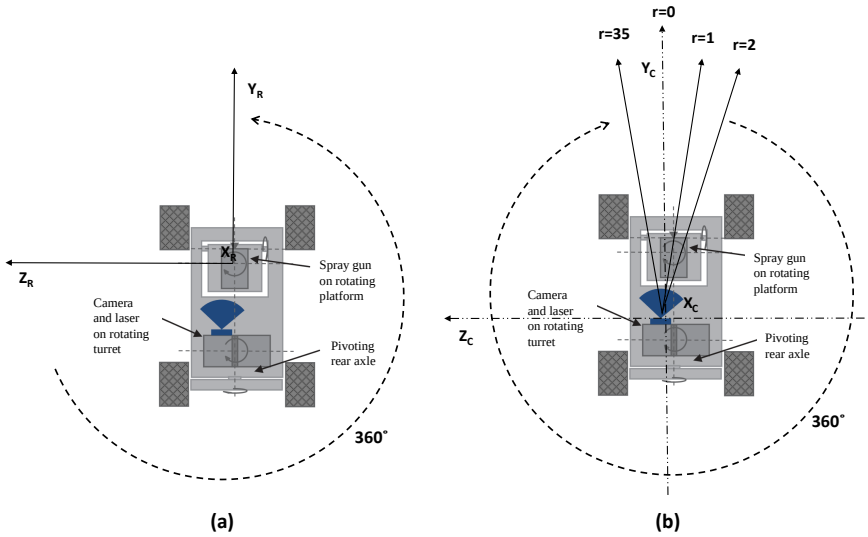



**Figure 3.3:** (a) Robot sensors and robot frame of reference. (b) Schematic description of the camera frame of reference and the image acquisition process.

pose is chosen to produce an overlay at low capture distances of at least two-thirds of the image between consecutive captures. Significant overlap is required to perform correct visual alignment between poses. Each image $K$ is acquired from the position corresponding to the pose $s$ of the robot and is defined in a set as $K_{s,r} \in \mathbb{R}^{N_x \times N_y}$, where $N_x \times N_y$ is the resolution of the image (in this case $1448 \times 1928$ pixels) and $r$ defines the orientation of the turret. The orientation of the first capture and the direction in which the turret spins may change between two different capture positions $s$ and $s-1$.

On the other hand, the laser system performs several scans during this process. Each scan covers a vertical plane (fig. 3.4(a)) in which the resolution is equal to 0.36 deg. (angle between two consecutive beams).This vertical scan covers 240 degrees, but only 120 degrees (the central ones) are used. In addition, the motor that spins the turret has 2400 steps. Therefore, the minimum angle between two consecutive exploration planes is equal to 0.15 degrees (fig. 3.4(b)). Hence, a point cloud formed for about 800,000 points is created from each $\vec{p}_s$ and is named by $P_{original,s}$.

### 3.2.3 Adding Color to the Point Cloud

According to the color information provided to each pixel in the related images, the association is as follows:

$$c_{\{i,j\},k} = I_k(\pi(H_{cal}T_{cam}R_{\phi_j}q_{i,j}^{[f]})) \tag{3.2}$$

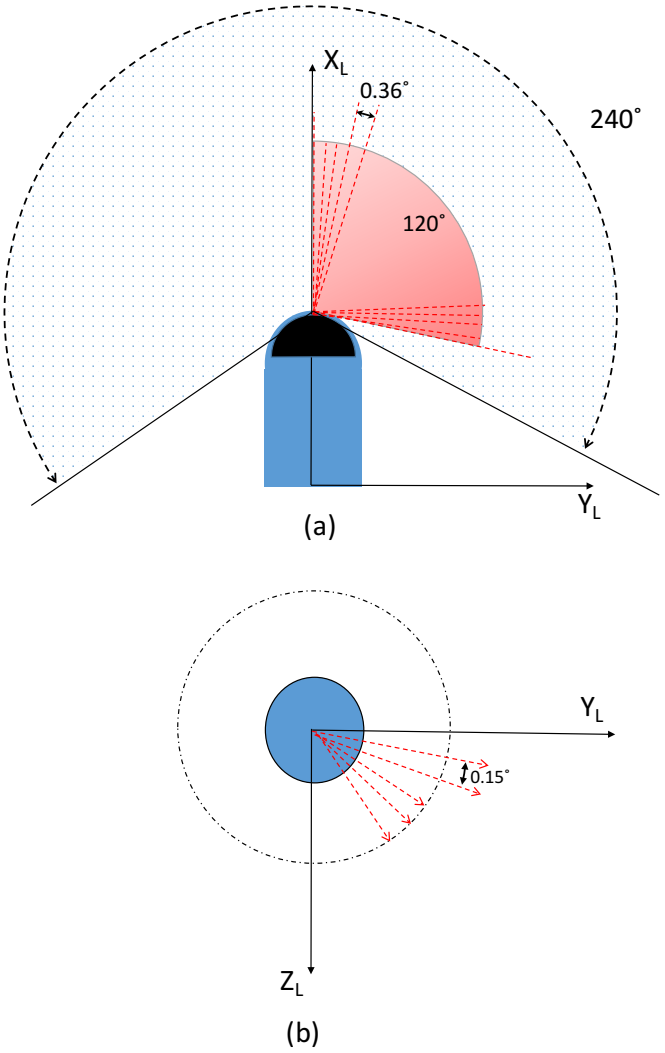



**Figure 3.4:** Schematic description of laser system. The laser frame of reference rotates around the $x_L$ axis. Figure (a) shows one laser scan and figure (b) is a bird eye's view of the scan planes during a whole acquisition process.

where $R_{\phi_j} \in \mathbb{SO}_3$ is the rotation matrix corresponding to the turret at the angle $\phi_k$ at which the image was acquired. $T_{cam} \in \mathbb{SE}_3$ is the transformation corresponding to the calibrated camera pose in the robot frame. $H_{cal}$ is the calibrated camera matrix and $u = \pi(x)$ is a function which performs the dehomogenization of $x \in \mathbb{R}^3 = (x, y, z)$ in order to obtain $u = (x/z, y/z)$. $I_k : \Omega \to \mathbb{N}^3$ is the subpixel mapping between the image space domain $\Omega \subset \mathbb{R}^2$ and the color values corresponding to the rectified image

$K$.

Despite color information is not used during the localization process, because it does not contain distinctive information in underground environments (due to its low color variety). This information is essential once the foam insulation has been sprayed, to check that the necessary areas have been covered correctly. Hence, this is why it is added to the cloud.

## 3.3 Techniques Proposed

In the present work, the mobile robot follows a trajectory that covers the environment to be modeled, initially unknown, and new methods are proposed to estimate this trajectory. The trajectory is defined as a set of adjacent positions traversed consecutively by the robot. The robot captures a set of visual and depth data from each pose that cover a 360-degree field of view. Notwithstanding, as it was mentioned in 5.1, due to the time consuming, the distance between consecutive poses is relatively high. A considerable amount of the works solve the registration with depth data by using the ICP (Iterative Closest Point) algorithm. Many of these algorithms present a good balance between accuracy and computing time. Nonetheless, their main problem is the need for a relatively accurate initial estimation to converge to the global optimum rather than the local minimum. The high similarity of the data acquired in the environments of this work aggravates this problem. In addition, the poses obtained through odometry are not reliable due to the characteristics of the terrain.

As it is presented in several previous works, a common method to solve the alignment between poses is through a coarse and a subsequent fine alignment. The idea is to obtain a rough but fast alignment estimation with a lower accurate data and after that, this information is used along with a more accurate data to carry out the alignment estimation more accurately. In this sense, two possibilities have been commonly proposed. The first option uses the visual information for the coarse alignment step and the point clouds in the refinement step (see fig. 3.5). First, this family of registration methods extracts keypoints from visual information. Second, a descriptor is calculated for each key point. This value is calculated using neighbourhood information and the result is a vector that characterizes the keypoint and allows it to be distinguished from other keypoints. Third, the keypoints in one image match the keypoints in another image captured from a different robot pose. Fourth, alignments are established between different images. Usually, there are some coincidences that are wrong, so a rejection step can be applied (a common method is RANSAC). Fifth, after establishing solid mappings between keypoints, the transformation matrix is calculated and used as the initial matrix for the refinement step (ICP algorithm), which provides a more accurate matrix.

In the second option, the depth information is used in the approximate alignment. Keypoints are extracted from scenes and their 3D positions are calculated using laser information. After that, the descriptors are calculated using in-depth information and a comparison is performed. Then, as in the first case, a rejection step is addressed

and an approximate transformation matrix is calculated through the correspondences between keypoints. Finally, a more accurate matrix is obtained using ICP.

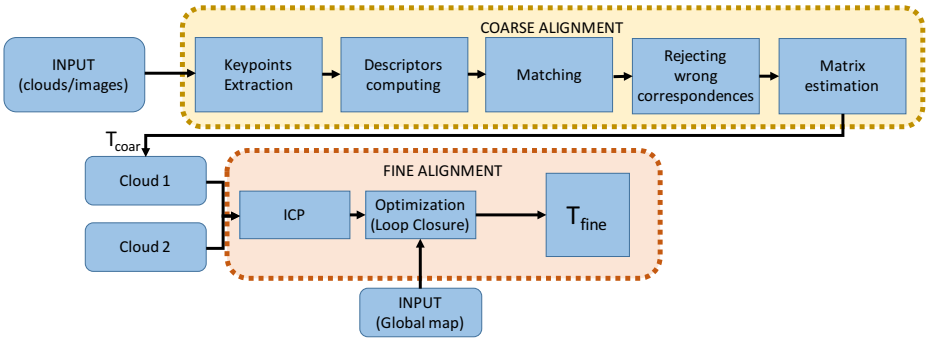


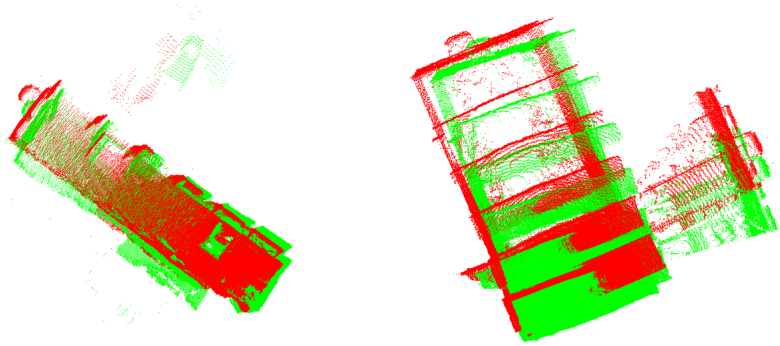

**Figure 3.5:** Schematic description of the usual localization method.

Preliminary experiments confirmed the negative impact to estimate the alignment by ICP due to the difficulties of building crawl spaces, as it was presented in section 5.1. Furthermore, the large distance between consecutive poses produces also considerable mistakes. An example of unsuccessful registration by ICP is shown in fig. 3.6. This example shows the typical results obtained when using ICP in underfloor environments with large distance between consecutive poses. Hence, the pose estimation algorithm must be robust enough. Throughout this section, novel methods based on depth and visual data are proposed to carry out the estimation between consecutive poses. These methods use registration approaches for this purpose.

### 3.3.1 Use of Depth Data

This subsection proposes a version of the second option by using the depth information in the coarse alignment, since the visual data obtained from the underfloor voids provide considerable disadvantages (see fig. 3.2). There is a lack of illumination, the walls are composed of bricks that do not contain characteristic information and the lower part of the floor is usually composed of identical beams and evenly distributed. Initial data are two point clouds captured from two poses *s* and *s-1* ($P_{original,s}$ and $P_{original,s-1}$). The goal is to obtain the transformation matrix (relative position and orientation) between these two poses, using a registration approach with the two point clouds. The proposed algorithm consists of three main steps: point selection, registration and validation.

Regarding the **points selection** step, this is crucial, since the original point clouds are composed of a large amount of points. This is due to the fact that the insulation task requires accurate information. If the ICP algorithm used all this information, the computation time would be excessively high. In addition, the information collected from the upper and lower planes, especially the wooden beams at the top



(a) (b)

**Figure 3.6:** Examples of the registration process, using two point clouds captured from different poses in a sample environment. The alignment is unsuccessful. (a) Lateral view and (b) Bird's eye view.

of the environment, would be detrimental as it would generate confusion. As noted above, these beams are almost equal and equidistant, so many points can be mistakenly considered to be correctly matched. Consequently, a misalignment could be accepted as successful because of the large number of matching points (see fig. 3.6). Such environments are very prone to present such erroneous results due to the geometry of the upper and lower planes. That is why we propose to delete this information before the registration process.

Hence, considering the previous premises, a process is tackled to select some of the points of the original cloud. This process consists of two steps. First, a homogeneous filtration is performed to considerably reduce the number of points. Second, segmentation is performed to select only the points located in the planes that are most useful for achieving a successful registration. With the aim of addressing these purposes, some functions of the Point Cloud Library (PCL) [236] have been used:

- The homogeneous filtering is addressed by following the scheme shown in fig. 3.7. Initially, a VoxelGrid filter is used. It consists of creating a grid of 3D voxels (in this case, a 1 x 1 x 1 cm cube) over the point clouds. Subsequently, the points inside each cube are approximated with their centroid. After that, a random sampling filter is applied, which randomly selects a number of points from the resulting cloud and discards the rest. The more points are eliminated, the faster this step and also the next ones will be. Nonetheless, removing too many points can be counterproductive because the subsequent ICP algorithm may not work well. Thus, despite random filtering, the structure must be maintained. After several tests, the conclusion is that maintaining 30% of the points is the optimal

value for balancing speed and reliability. From this step, a reduced point cloud is obtained for each position of the robot ($P_{downsampled,s}$ and $P_{downsampled,s-1}$).

- The aim of segmentation is to eliminate those points that belong to the upper and lower planes. To address this step, the point cloud obtained after step 1 is grouped into planes. The planes whose normal vector is parallel to the axis $x$ are considered as the upper and lower and the rest as walls. The algorithm removes the top and bottom planes, discards the points that belong to them, and keeps the rest. In order to consider the presence of beams in the upper planes and eliminate their information, once the planes that are placed on the robot are detected, the one of lower height is extracted and all the information in and on itself is eliminated. As a result, clearer clouds are obtained ($P_{filtered,s}$ and $P_{filtered,s-1}$).

To sum up, the process starts with the original point clouds ($P_{original}$), obtained from the data acquisition, which contain more than 800.000 points. Then, the downsampled point clouds ($P_{downsampled}$) are obtained after applying VoxelGrid and random sampling filters. Finally, the filtered point clouds ($P_{filtered}$) are obtained through segmentation and removal of the top and bottom planes. These clouds will be used for the registration step. Fig. 3.8 shows the whole process with an original point cloud. Fig. 3.8(a) is $P_{original,s}$, fig. 3.8(b) is $P_{downsampled,s}$ and fig. 3.8(c) is $P_{filtered,s}$.

The next step consists in carrying out a **registration** process between the two filtered clouds ($P_{filtered,s}$ and $P_{filtered,s-1}$). The results of this process are: (a) the transformation matrix ($T_{s,s-1}$) that relates both poses; (b) the number of matched points ($N_{s,s-1}$) and (c) the $EFS_{s,s-1}$ (Euclidean Fitness Score) defined in eq. 3.3.

$$EFS_{s,s-1} = \sum_{j=1}^{N_{s,s-1}} dist(P_s^{est}(j), P_s(j))^2 \tag{3.3}$$

where $dist(P_s^{est}(j), P_s(j))$ is the Euclidean distance between the j-th matched point of the clouds $P_s^{est}$ and $P_s$. $P_s^{est} = T_{s,s-1} \times P_{s-1}$.

Classical ICP algorithms may present erroneous results with respect to the detection of relative orientations between positions. This is due to the fact that the algorithm can converge to a local minimum when the target environment has some symmetry. The subsoil environments modeled in this work are very prone to present this problem as they have two main directions in the horizontal plane (those perpendicular to the walls) and the walls may erroneously coincide with their opposites. Since the odometry information is untrustworthy, it cannot be used to have an initial estimation to apply the ICP algorithm. To overcome this problem, the following four initial conditions are considered: no translation and three rotations (90, 180, and 270 degrees) around the vertical axis. After that, the classical ICP algorithm runs four times in parallel (one

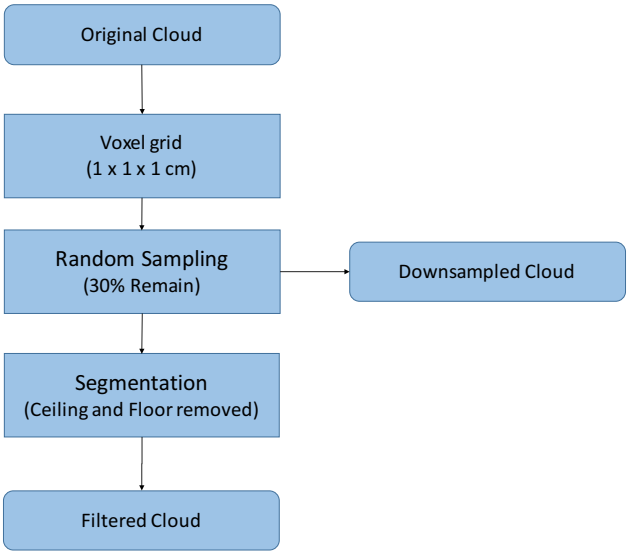



**Figure 3.7:** Filtering stages.

for each of the four initial estimates). The alignment between the four addressed that maximizes eq. 3.4 is considered the optimal one.

$$\max\{\alpha N_{s,s-1}^{k} + \beta \frac{1}{EFS_{s,s-1}^{k}}\} \tag{3.4}$$

Where $\alpha$ and $\beta$ are two weighting values that were empirically tuned according to the characteristics of the environment and $k$ is an index that defines the initial orientation $(k = 0, 90, 180, 270)$.

The process to select the best alignment between the two consecutive poses is shown in Fig. 3.9 shows the process to select the best alignment between continuing poses. To summarize, this step takes into consideration four initial versions of the cloud $P_{filtered,s}$ aligned with $P_{filtered,s-1}$ by using ICP. The transformation matrix $T_{s,s-1}$ of the resultant optimal alignment is retained.

The previous step provides the optimal alignment matrix $T_{s,s-1}$ once all four possibilities have been evaluated. Nevertheless, this does not guarantee that the alignment is correct. Hence, the algorithm should include a validation step.

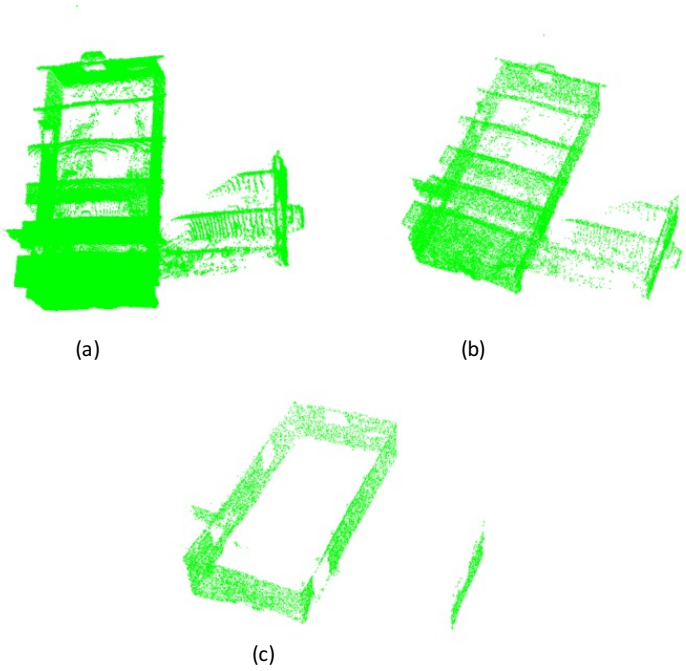



**Figure 3.8:** Points selection process. (a) Original cloud. (b) Reduced point cloud after VoxelGrid and random filtering. (c) Resulting cloud after removing top and bottom planes.

To validate this alignment matrix, the downsampled point clouds are considered ($P_{downsampled}$). In the previous subsection, $P_{filtered}$ has been used because the removal of the upper and lower planes would be beneficial for the registration process. Nonetheless, to verify whether this process was really successful, this information must be considered. Otherwise, failures could not be detected along the $x$ axis.

Three parameters are considered to address the validation:

- The Euclidean Fitness Score.
- The ratio of correspondences over the number of points in $P_{downsampled,s}$ (eq. 3.5).
- The ratio of correspondences over the number of points in $P_{downsampled,s-1}$ (eq. 3.6).

$$rat_s = \frac{N_{s,s-1}}{N_{points(s)}} \tag{3.5}$$

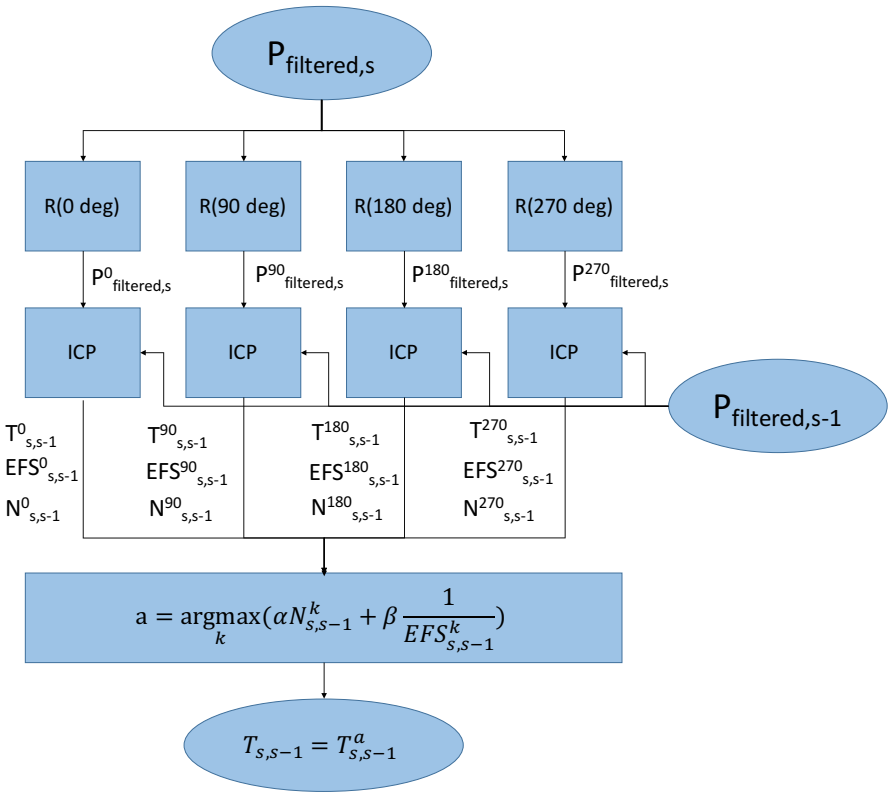



**Figure 3.9:** Algorithm diagram. First, the cloud $s$ is rotated 4 times with 0, 90, 180 and 270 degrees. Each resulting cloud is compared to the *s-1* using ICP. Through the values of EFS (Euclidean Fitness Score) and the number of matched points, the optimal transformation matrix is chosen.

$$rat_{s-1} = \frac{N_{s,s-1}}{N_{points(s-1)}} \tag{3.6}$$

where $N_{points(s)}$ and $N_{points(s-1)}$ are the number of points in the downsampled point clouds $P_{downsampled,s}$ and $P_{downsampled,s-1}$ respectively. $N_{s,s-1}$ is the number of correspondences between $P_{filtered,s}$ and $P_{filtered,s-1}$.

Using only one of these two proportions may seem like enough to validate the result. However, although one percentage might be high enough to meet the validation threshold, the other percentage may not reach that threshold, and consequently alignment should be rejected.

Therefore, given the equation:

$$\left[\eta\Delta rat_s + \lambda\Delta rat_{s-1} + \mu\Delta\frac{1}{EFS} > Validation_{th}\right] \tag{3.7}$$

where $\eta$, $\lambda$ and $\mu$ are weighting values and $Validation_{th}$ is the threshold. If the sum of the left side is higher than the established threshold ($Validation_{th}$), then the alignment between the two point clouds will be considered successful and accepted. Otherwise, the alignment will be considered failed and will be rejected. Furthermore, it should be emphasized that although the registration between $P_{filtered,s}$ and $P_{filtered,s-1}$ is not correct, the inverse registration (registration between $P_{filtered,s-1}$ and $P_{filtered,s}$) may be correct. Hence, in a case of unsatisfactory validation, the inverse registration will be performed and validated to see if it meets the eq. 3.7.

By this third step, the alignment matrix $T_{s,s-1}$ obtained after step 2 is validated. Two possibilities can be given:

1. Equation 3.7 is satisfied. Then, a successful alignment between the current pose and the previous one is co considered, i.e the matrix $T_{s,s-1}$ is considered correct.

2. Equation 3.7 is not satisfied. Then, the alignment matrix $T_{s,s-1}$ is not accepted as valid and, thus, the current pose $\vec{p}_s$ cannot be estimated with respect to the previous one $\vec{p}_{s-1}$. The matrix $T_{s,s-1}$ is considered incorrect.

### 3.3.2 Algorithm to Locate Lost Poses

Despite proposing a robust algorithm for alignment, sometimes there are unsuccessful cases. Typically, these cases tend to appear either when the distance between the two poses is relatively prominent or when there is a big difference between the captured environments (i.e. the robot has entered in a different room). For this reason, in the present work is presented a pose estimation algorithm when the registration carried out with the previous pose was not successful. This algorithm attempts to align poses that were not properly aligned and also to locate poses that did not meet the validation condition. After the alignment step, with the aim to obtain the alignment matrix between $P_{filtered,s}$ and $P_{filtered,s-1}$, three different cases are considered (see algorithm 1):

1. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is correct (alignment matrix $T_{s,s-1}$ from the alignment algorithm is considered valid) and the previous pose ($\vec{p}_{s-1}$) is correctly located. A pose is considered well located when its position is well known through a successful alignment between this pose and the previous one, the position is also known. In this case, the current pose $\vec{p}_s$ will be estimated through the matrix $T_{s,s-1}$. Thus, the pose $\vec{p}_s$ will be considered as correctly located.

2. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is not valid ( eq. 3.7 is not met). In this case, a near pose would be searched in order to obtain a good alignment (see algorithm 2). In this sense, despite the fact that $T_{s,s-1}$ is not correct, this matrix is used to make a rough estimation of the pose $\vec{p}_s$ with respect to $\vec{p}_{s-1}$. Afterwards, the distances between the pose $\vec{p}_s$ and the previous ones ($\vec{p}_n$ where $n = 0, ..., s-2$ are known) are calculated by using:

$$d_{s,n} = \sqrt{(x_s - x_n)^2 + (y_s - y_n)^2} \quad ; \quad n = 0, ..., s-2 \tag{3.8}$$

where $(x_s, y_s)$ and $(x_n, y_n)$ are the Cartesian coordinates of the poses $\vec{p}_s$ and $\vec{p}_n$ within the map. After calculating the distances, the previous poses are sorted by distance and then the registration is tried between the filtered $P_{cloud,s}$ and the nearest cloud. The registration step is repeated with the next nearest clouds while the registration is not done correctly.

Two situations can occur. If a good alignment with a pose $\vec{p}_n$ is obtained, then the alignment problem will be successfully solved for the pose $\vec{p}_s$. The transformation matrix will be stored and the pose $\vec{p}_s$ would be located in the map. If a successful alignment is not reached with any of the previous poses, $\vec{p}_s$ will be saved in a list of poses that are pending to be located. The poses that are in the pending list will not be considered in subsequent registration attempts because their location is not well known.

3. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is correct (i.e. the alignment matrix $T_{s,s-1}$ is considered valid), but the location of the previous pose ($\vec{p}_{s-1}$) is not considered valid. If the alignment between the two consecutive poses is considered valid but the previous pose ($\vec{p}_{s-1}$) is not correctly located (i.e. it is in the pending list), then, the registration of $P_{filtered,s}$ will be addressed with the closest clouds until a good alignment is obtained. If $\vec{p}_s$ is successfully aligned with any of the well located poses, the transformation matrix will be stored and the pose $\vec{p}_s$ will be included in the map. Additionally, $\vec{p}_{s-1}$ will be removed from the pending list and it will be considered correctly located too (see algorithm 3).

The algorithm includes a final step, which is always performed when a pose has been aligned correctly. After locating the new pose, the algorithm attempts to align that pose with each pose that is on the pending list. Although this step increases the computation time, it is very beneficial, since it can reduce the number of poses that are not located yet. Therefore, more information can be obtained about the map and the movement of the robot.

### 3.3.3 Use of Visual Data

In this subsection, the localization process is addressed as a problem to align the information captured from two consecutive poses *s* and *s-1*. The proposed method consists in selecting robust points from the point clouds $P_{original,s}$ and $P_{original,s-1}$ acquired from two consecutive poses as the robot moves through the underfloor environment. Previous research works have proposed the use of the point clouds with

**Algorithm 1** Online algorithm

**if** *Registration*($P_{filtered,s}, P_{filtered,s-1}$) OK **then**
  **if** $\vec{p}_{s-1}$ Located **then**
    Store($T_{s,s-1}$)
    Locate($\vec{p}_s$)
  **else**
    Case B (algorithm 3)
  **end if**
**else**
  Case A (algorithm 2)
**end if**

**Algorithm 2** Case A

$T_{s,s-1}$ is not accurate.
$p_{correct}$: Array of located poses.
$v$: Array of correct poses. Sorted by distance.
$d_k$: Array of distances between $\vec{p}_s$ and the rest of correct poses $\vec{p}_i$.
$P^{est}_{filtered,s} = T_{s,s-1} \times P_{filtered,s-1}$
**for** `i=0;` $(i < s-1)$`;` `i++` **do**
  $d_k(i) = \sqrt{(x^{est}_s - x_i)^2 + (y^{est}_s - y_i)^2}$
**end for**
$v = sort\{p_{correct}, d_k\}$
**for** `j=0; {`$j$ $<$ $(N_{poses} - 2)$ `and` *Registration*($P_{filtered,s}, P_{filtered,v(j)}$) `is WRONG} ; j++` **do**
  **if** *Registration*($P_{filtered,s}, P_{filtered,v(j)}$) OK **then**
    Store($T_{s,v(j)}$)
    Locate($\vec{p}_s$)
    END CASE A
  **end if**
**end for**
$List_{pending} \leftarrow \vec{p}_s$
END CASE A

**Algorithm 3** Case B

$\vec{p}_{s-1}$ is not accurate.
$T_{s,s-1}$ is accurate.
$p_{correct}$: Array of located poses.
$v$: Array of correct poses. Sorted by distance.
$d_k$: Array of distances between $\vec{p}_s$ and the rest of correct poses $\vec{p}_i$.
$P^{est}_{filtered,s} = T_{s,s-1} \times P_{,filtered,s-1}$
**for** `i=0; (`$i < s-1$`); i++` **do**
    $d_k(i) = \sqrt{(x^{est}_{filtered,s} - x_{filtered,i}-)^2 + (y^{est}_{filtered,s} - y_{filtered,i})^2}$
**end for**
$v = sort\{p_{correct}, d_k\}$
**for** `j=0; {`$j < (N_{poses} - 2)$ `and` $Registration(P_{filtered,s}, P_{filtered,v(j)})$ `is` *WRONG*`} ; j++` **do**
    **if** $Registration(P_{filtered,s}, P_{filtered,v(j)})$ OK **then**
        Store($T_{s,v(j)}$)
        Locate($\vec{p}_s$)
        Locate($\vec{p}_{s-1}$)
        $List_{pending} \not\leftarrow \vec{p}_s$
        END CASE B
    **end if**
**end for**
$List_{pending} \leftarrow \vec{p}_s$
END CASE B

the Iterative Closest Point (ICP) algorithm to calculate the registration [107] or similar methods that derive from ICP such as CPD (Coherent Drift Point) [194] and NDT (Normal-Distributions Transform) [24]. Nevertheless, this subsection proposes a novel alternative which estimates the alignment using visual information. Hence, the aim of this subsection is to estimate the transformation matrix $T_{s,s-1}$ between the poses $s$ and $s-1$. If the pose $s-1$ is known, then, once the transformation matrix has been calculated, the pose $s$ can be estimated and integrated into the model.

Considering the facts that the alignment algorithms based on 'pure' ICP registration do not work properly for the proposed task, the visual information acquired by the camera is used to obtain robust matches. The method proposed in this subsection to estimate the transformation matrix $T_{s,s-1}$ consists of the following steps:

1. Pairing up the images in the set $K_{s-1,l}$ with the images in the set $K_{s,m}$, $l, m = 0, ..., 36$. For each image in the first set, the most similar image in the second set is calculated and matched with the first image. It should be noted that the turret performs a uniform and constant rotation of 360 degrees while capturing images from each set.

2. Coincidence of visual characteristics. For each of the pairs of images resulting from the previous step, visual features are extracted and described and correspon-

dences are established between them. As a result, each pair of images provides a list of visual correspondences.

3. Performing the alignment by means of information of depth. The depth of each corresponding point is calculated and, as a result, two new point clouds are generated with a significantly reduced number of points. These point clouds are expected to provide solid alignment because they are constructed using only points that have been shown to have trustworthy matching. This process ends with the estimation of the transformation matrix $T_{s,s-1}$.

To summarize, the whole algorithm is represented in fig. 3.10 and it basically consists in three steps: (1) Matching images by global-appearance descriptors, (2) matching visual features and extraction, and (3) alignment of depth information.

As for **matching images by global-appearance descriptors**, the aim of this step consists in paring up the images captured from the position $s-1$ with the images captured from the following position $s$. The movement between poses regarding orientation can not be estimated with enough accuracy, since the odometry provided is not trustworthy. Hence, in order to obtain a reliable measurement, an algorithm is proposed with the objective to find out which image in the set $K_{s,m}$ is most similar to each image in the set $K_{s-1,l}$, $l, m = 0, ..., 36$. For this purpose, global-appearance descriptors are proposed to compare the images pairwise and to make the pairing process. In this case, the Fourier Signature (FS) descriptor [178] has been proposed.

As for this global-appearance description method, it consists basically in obtaining the one-dimensional Discrete Fourier Transform (1D-DFT) of each row from an image with $N_x$ rows and $N_y$ columns. This way, each row $x$ of the original image $r_x = \{r_{x,0}, r_{x,1}, ..., r_{x,N_y-1}\}$, $x = 0, ..., N_{x-1}$ is transformed into the sequence of complex numbers $F_x = \{F_{x,1}, F_{x,2}, ..., F_{x,N_y-1}\}$, $x = 0, ..., N_{x-1}$ by using eq. 3.3.3 [214]:

$$F_{x,k} = \sum_{n=0}^{N_y-1} r_x, n \cdot e^{-j(2\pi/N_y)k_n},$$
$$k = 0, ..., N_{y-1}, x = 0, ..., N_{x-1}$$

After transforming the whole image, the resultant matrix $F(v, y)$, where $v$ is the frequency variable , is expressed in $cycles/pixel$. The most relevant information is concentrated on the low frequency components, and the high frequency components tend to present more noise. This way, a compression is conducted based on discarding the last columns of the matrix and retaining only the first $N_k$ columns. Hence the resulting compressed matrix is named as Fourier Signature $F(v, y) \in R^{N_x \times N_y}$. This resultant matrix can be decomposed into a magnitude matrix $A(v, y)$ and an arguments matrix $\Phi(v, y)$. The magnitudes matrix contains non localized information on the global appearance of the scene, then it can be used as a global-appearance descriptor of the original image. Therefore, considering all these facts, a holistic descriptor is calculated

for each one of the images contained in the set of $K_{s-1,l}$ and $K_{s,m}$. This information is stored in the matrices $A_{s-1}$ and $A_s$ respectively.

After obtaining all the descriptors, the comparison between images can be done. That is, each image of the set $K_{s-1,l}$ is compared with one image from the set $K_{s,m}$ by using the Euclidean distance. The image that presents the minimum distance is paired up. According to this process, it is possible that the image in the set $K_{s,m}$ is assigned to several images in the set $K_{s-1,l}$. However, there must be only some offset in the order or acquisition of the images from both poses. Additionally, the direction in which the turret rotates can be different. Therefore, the correct association between two images depends on two variables $(r, t)$. The first one indicates the rotation direction (clockwise or counterclockwise) $r = [-1, 1]$ and $t$ represents the relative offset between the first image of each set. Estimating these values is necessary to carry out robustly the movement estimation $M_{s-1,s}$ between poses $s-1$ and $s$. Hence, once $M_{s-1,s}$ has been estimated, a comparison of the $2\times37$ possible paring matrices is done, considering $r = [-1, 1]$ and $t = [0, 1, ..., 36]$. The algorithm compares $M_{s-1,s}$ with all the possible solutions and selects the most similar one (using the Hadamard product as the criterion to obtain the degree of similitude between matrices), which is named $M'_{s-1,s}$. Once $r$ and $t$ are known, the images of both sets are paired up according to the final matrix of match-ups $M'_{s-1,s}$. Mathematically, once $t$ and $r$ are known, the pairings can be calculated through the next expression. The image $K_{s-1,l}$ is paired up with the image $K_{s,m}$, where:

$$m = \begin{cases} (l + r \cdot t) \mod 37 & \text{if } r = 1 \\ (37 - l - r \cdot t) \mod 37 & \text{if } r = -1 \end{cases} \tag{3.9}$$

After pairing up the set of images $K_{s-1,l}$ with $K_{s,m}$, the following step consists in **matching visual features**, that is, conducting the detection, description and matching of local features. First, the visual features of each image are detected and described by means of the Speeded Up Robust Features (SURF) algorithm [16]. The choice of this descriptor is due the robustness presented in preliminary experiments concerning the target environment. Once the visual features have been obtained and described, a matching step is tackled. Then, as result, a set of matched visual features is obtained, for each pair of images. An example of this process is depicted in the fig. 3.11. Taking into consideration that the images acquired by the robot in a specific pose present a high degree of overlapping, only 6 pairs of equally spaced images are used $(K_{s-1,l\cdot i}, K_{s,m\cdot i})$ with $i = 1, ..., 6$ in order to avoid adding redundant information (see fig. 3.10)

The third step of the registration process consists in **extracting and aligning the depth information**, that is, recovering the 3D information of each keypoint detected. In order to carry out the 3D information recovery, the image that presents the keypoint as well as the original point cloud are considered. After that, the 3D coordinates of the matched keypoints are available and, then, two new point clouds are created, one corresponding to the pose $s-1$ and the other to the pose $s$. Due to

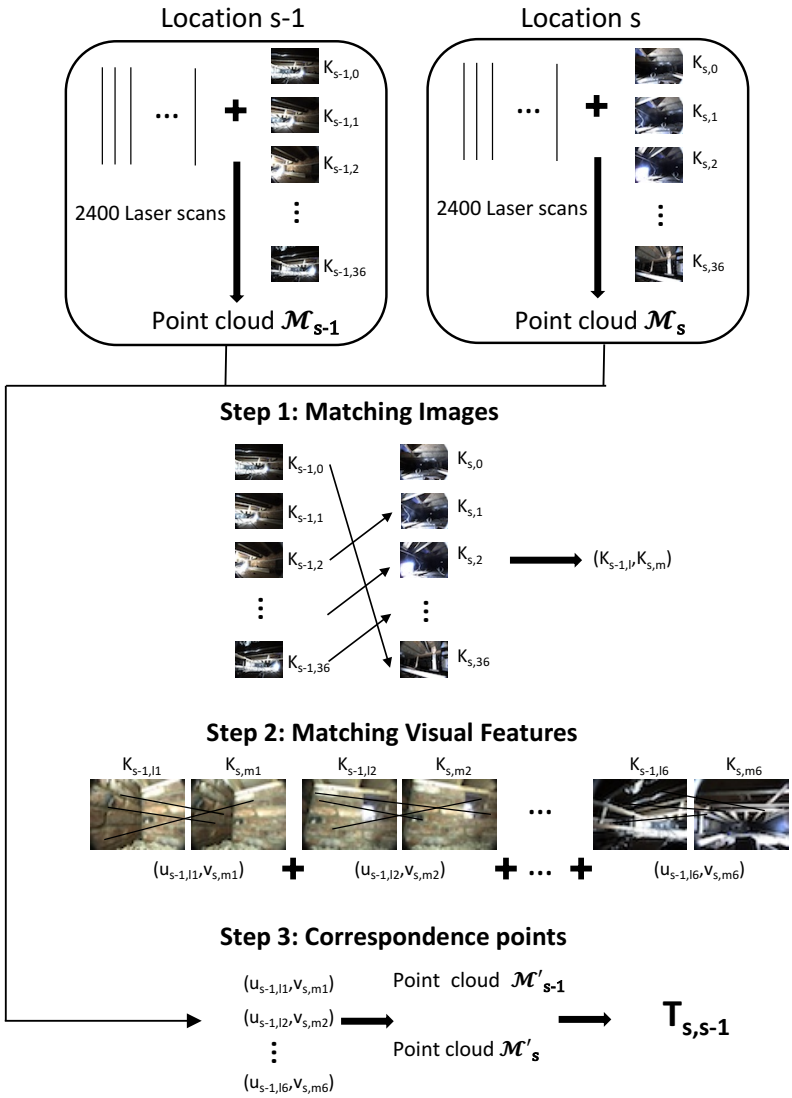



**Figure 3.10:** Schematic description of the visual-data-based alignment algorithm. The images in both sets are firstly paired. After that, the visual keypoints are extracted and combined considering each pair of images. Last, 3D points are generated from the matches and the transformation matrix $T_{s,s-1}$ is obtained.

the fact that the points contained in ceiling and floor are expected to contain inconsistencies, they are removed from the point cloud set. Hence, the final point clouds of

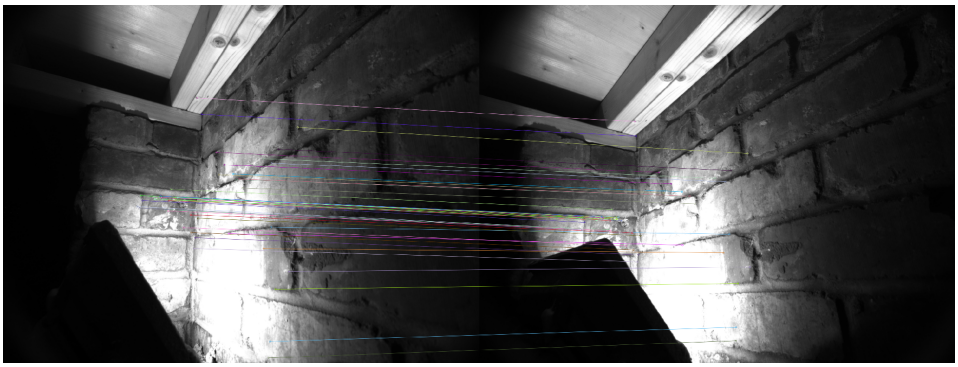


**Figure 3.11:** Example of the extraction of keypoints (in this case, SURF keypoints) and matching with a previous image.

the poses $s-1$ and $s$ are named $P'_{s-1}$ and $P'_s$ respectively and they have a significantly lower number of points compared to the original clouds. Afterwards, the transformation matrix $T_{s-1,s}$, which represents the relative rotation and translation of the pose $s$ with respect $s-1$, is obtained through using the alignment algorithm of the Point Cloud Library (PCL) [236] over the two new point clouds. As the number of points in each cloud is relatively small, the necessary time to reach a solution is reduced.

The benefit of the proposed method is twofold. First, the proposed process ensures that subsequent mappings are more robust and trustworthy, as keypoints have been previously selected from similar images. This feature is especially relevant in challenging environments such as target environments (building crawl spaces). Second, the number of matches used to estimate the alignment matrix is substantially less than the use of the full point cloud provided by the laser sensor, then this improves the computation time.

## 3.4 Experiments

This section presents the experiments and results carried out through applying the methods proposed in previous sections. The algorithms were run on a $2 \times 2.66$ GHz Dual-Core Intel Xeon CPU ® with 10 GB of memory. To develop the experiments, the robot took some sets of laser measurements and images from different poses from various environments and assembled a point cloud per pose. Therefore, there is a point cloud and 37 images associated with each pose of the robot within the environment ($P_s$ is the point cloud in the pose $\vec{p}_s$). As mentioned before, the characteristics of the terrain do not allow to obtain sufficiently precise odometry measures. Consequently, for experiments related to depth data, the quality of the calculated alignment will be measured through confidence values that indicate the quality of the estimated transformation matrix. In the case of experiments related to visual data, in-depth data information is used as ground truth. Therefore, an alignment matrix is considered correct or rejected according to the criteria presented in subsection 3.3.1.

First, the environment and datasets are described in subsection 3.4.1. After that three experiments have been carried out to validate the algorithms proposed with the depth data (3.4.2) and other two experiments have been carried out concerning the use of visual data (3.4.3). Regarding the experiments conducted with depth data, three three experiments are performed to verify the validity of the presented algorithms. The first experiment is to evaluate the method for aligning consecutive positions (algorithm presented in 3.3.1). The second experiment focuses on the alignment of poses that were not successfully registered with the previous one (explained in section 3.3.2). Last, an experiment is proposed to adjust the validation threshold of the eq. 3.7. As for the experiments tackled with visual data, the algorithm for matching images with using global appearance is first tested, and then the performance of the alignment algorithm is evaluated.

### 3.4.1 Dataset

A set of data captured by ourselves is considered to perform the experiments. Six different underfloor environments are used to collect the data. Within each environment, the robot followed a trajectory and captured data from various poses. For each pose, a point cloud is stored. The table 3.1 shows the number of poses considered for each environment.

| **Environment** | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| **Number of poses** | 4 | 9 | 10 | 10 | 21 | 21 |

**Table 3.1:** Number of poses in each environment

Fig. 3.12 shows the appearance of some of the environments. Environments 1 and 2 are relatively small (the average distance between captures is less than 10 cm) and are well structured. Environment 3 is larger and consists of several rooms. This way, consecutive point clouds can differ considerably, so it is expected to be more challenging. Environment 4 has a simpler structure compared to previous environments, but is the one that presents a greater distance between consecutive positions. Finally, 5 and 6 correspond to the same environment but before and after applying the foam insulation to the bottom of the floor.

Environment 2 Environment 4 Environment 6

**Figure 3.12:** Examples of images taken from different environments that have been used to develop the experiments.

### 3.4.2 Experiments Using Depth Data

Concerning the experiments carried out by using depth information, the six datasets were used to evaluate the proposed algorithms to solve the registration problem in environments that present a challenge for characterization. For these experiments, an alignment matrix ($T$) will be considered either correct or rejected through the criteria presented in subsection 3.3.1. Three experiments are addressed to verify the validity of the presented algorithms. The first experiment is to evaluate the method for aligning consecutive positions (algorithm presented in subsection 3.3.1). The second experiment focuses on the alignment of poses that were not successfully registered with the previous one (explained in subsection 3.3.2). Finally, an experiment is proposed to adjust the validation threshold of the eq. 3.7.

#### 3.4.2.1 Experiment 1. Alignment between consecutive poses through the proposed registration algorithm.

The first experiment consists of a registration between consecutive positions and is developed through the alignment algorithm presented by subsection 3.3.1. The percentage of correspondences is calculated as subsection 3.3.1 details (eq. 3.5 and 3.6), since the number of matches depends on the characteristics of the point cloud (number of points after downsampling, shape, etc.). In addition, computation time to complete the registration was also collected.

Fig. 3.13 and 3.14 show the results of each pair of consecutive positions $(s-1, s)$ in the six environments: the percentage of matches $r_s$ and $r_{s-1}$ (eq. 3.5 and 3.6) and the EFS value (eq. 3.3) divided by the number of correspondences ($N_{s,s-1}$). In addition, table 3.2 shows the average computation time and the average number of matches for each environment.

As explained in subsection 3.4.2, an alignment is considered valid when equation 3.7 is accomplished. According to it, a high number of correspondences and low values of EFS denote good alignments. Taking it into account, an analysis of figures 3.13 and 3.14 permits knowing which registrations are successful and which are not.

As it is shown, the alignments tackled in the environments 1 and 2 were all successful, since the parameters to measure the correspondences present similar and also high values. There is no alignment whose values $rat_{s-1}, rat_s$ and $EFS/N_{s,s-1}$ are substantially worse than the others, so none of the alignments should be rejected. Nevertheless, in the environment 3, the percentage of matched points between the poses $6$ and $7$ and also between $7$ and $8$ are considerably lower than those of the rest of alignments and, at the same time, the EFS values are substantially higher. This denotes that the alignment between these point clouds was unsuccessful and hence these alignments should be rejected. In the environment 4, the alignments between $P_5 - P_6$ and between $P_8 - P_9$ present a percentage of correspondences that is slightly lower than in the others. Since this difference is relatively low, these alignments should be considered valid. The two last registrations in the environment 5 ($P_{18} - P_{19}$ and $P_{19} - P_{20}$) have respectively an average of 16% and 25% of correspondences, which

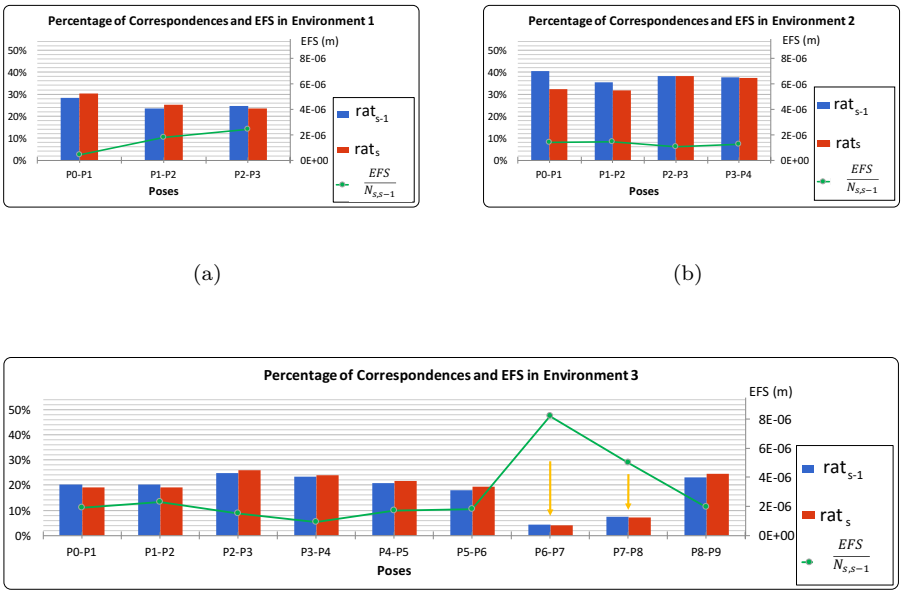
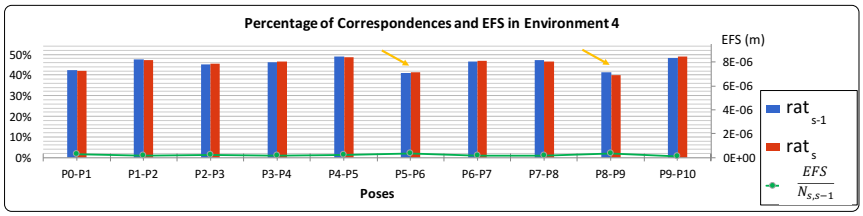



**Figure 3.13:** Results of the experiment 1. Percentage of matched points and EFS divided by the number of matches in environment (a) 1 , (b) 2 and (c) 3.

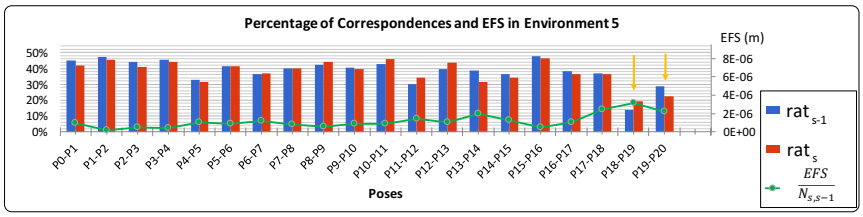
are considerably lower than the rest of registrations (around 40% of correspondences), while the EFS values for these alignments are also higher. This way, these two cases should be categorized as unsuccessful. In the environment 6, similar issues can be found in $P_{10} - P_{11}$ and $P_{19} - P_{20}$.

Hence, from the results collected through these experiments, the conclusion reached is that both parameters (number of correspondences between consecutive point clouds and the Euclidean Fitness Score) are capable of validating the alignment results between point clouds.

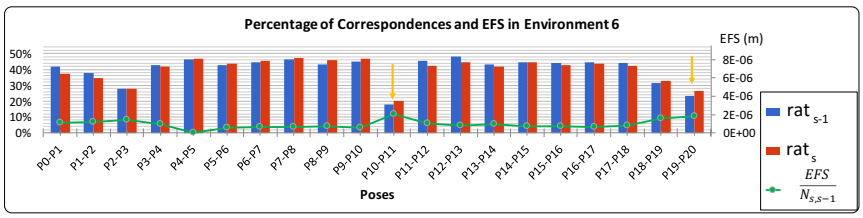
Finally, in this experiment, the calculation time of the alignment between point clouds was also measured. Table 3.2 shows the average computation time and the average number of correspondences for each environment. From this table, it is noticed that the problem can be solved in few seconds.




(a)




(b)




(c)

**Figure 3.14:** Results of experiment 1. Percentage of matched points and EFS divided by the number of matches in environment (a) 4 , (b) 5 and (c) 6.

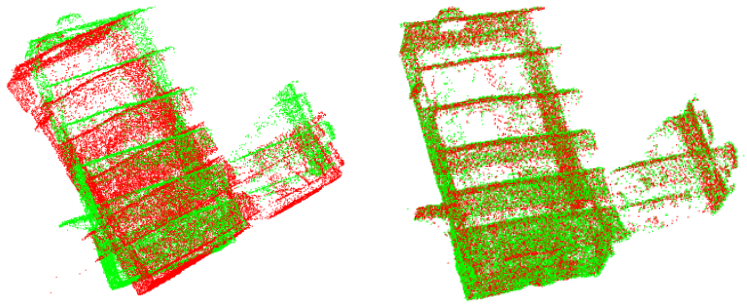
| Environment | Average Time (ms) | Average N. correspondences |
|---|---|---|
| 1 | 4224 | 10411 |
| 2 | 3639 | 13332 |
| 3 | 4184 | 10855 |
| 4 | 4934 | 16577 |
| 5 | 5955 | 14466 |
| 6 | 5284 | 14732 |

**Table 3.2:** Results of experiment 1. Average computing time and average number of correspondences of the registration process in the six studied environments.

#### 3.4.2.2 Experiment 2. Alignment between poses that have not been successfully aligned with the previous one.

This experiment assesses the algorithm presented in the subsection 3.3.2. As the results of the experiment 1 show, although the alignment achieve successful results for a majority of cases, some poses were not well aligned with the previous one. To solve this problem the algorithm that tries to align poses with other poses different from the previous one is executed. This experiment has been evaluated using environments 5 and 6.

Figures 3.15 and 3.16 show two alignments of sample point clouds through the transformation matrices calculated in experiment 1 and in experiment 2. Fig. 3.15 shows point clouds from the environment 5 and fig. 3.16 shows point clouds from the environment 6. First, concerning environment 5, the alignment calculated in experiment 1 between the poses $18$ and $19$ was not successful (fig. 3.15(a)). Nonetheless, the experiment 2 provides a successful alignment with the pose $11$ (fig. 3.15(b)). Second, in the case of environment 6, the experiment 1 was unable to calculate a correct alignment between the 10th and the 11th poses (fig 3.16(a)). In this way, experiment 2 was run and, as a result, the pose $11$ was successfully aligned with the pose $13$ (fig. 3.16(b)). In this case, the alignment was performed following this process. First, the point cloud was obtained at pose $11$ and the 1 algorithm was executed. Registration between poses $10$ and $11$ was unsuccessful. Hence, the 2 algorithm was executed. The registration between $10$ and $11$ was unsuccessful. Therefore, the algorithm 2 was run. The previous poses (from $\vec{p}_0$ to $\vec{p}_9$) were sorted according to the distance to the pose $11$ ($\vec{p}_{11}$ was calculated as $T_{11,10} \times \vec{p}_{10}$, where $T_{11,10}$ was a coarse alignment matrix). After trying to align unsuccessfully the pose $11$ with all the previous ones, this pose was stored on the pending list. Second, a new pose $(12)$ was obtained and successfully aligned with $11$ using the algorithm 1. Nevertheless, since $11$ was on the pending list, the Case B (algorithm 3) was executed. The pose $12$ was unsuccessfully registered with any of the previous poses $(0, 1...., 10)$. Therefore, this pose was also stored on the pending list. Again, a new pose $(13)$ was obtained and was correctly registered with $(12)$. After that, the algorithm 3 was executed and pose $13$ was successfully registered with the pose $10$. Thus, both the $13$ and $12$ poses were located correctly and $12$ was removed from the pending list. Last, as mentioned in subsection 3.3.2, after localizing a new pose, the algorithm executes a final step to try to localize poses that were still



(a) (b)

**Figure 3.15:** Examples of alignment between point clouds. (a) Incorrect alignment between poses 18 and 19 in environment 5. (b) Correct alignment of pose 19 with pose 11 in environment 5 after using case A algorithm 2 of the novel registration algorithm.

in the pending list. Registration between 13 and 11 was successfully addressed, thus, 11 was successfully located as $\vec{p}_{11} = T_{11,13} \times \vec{p}_{13}$.

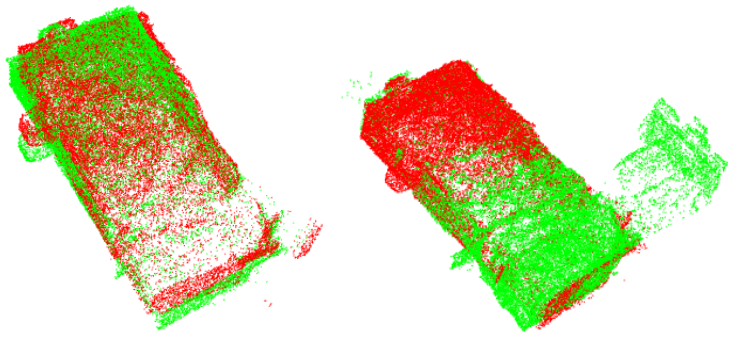
Fig. 3.17 shows the computing time of the alignment process. This time takes reasonably low values when the pose is successfully aligned with the previous one (experiment 1) and increases substantially when experiment 2 has to be processed to achieve a correct alignment.

Therefore, despite using the algorithm that aligns consecutive poses allows to get correct alignments, there is a low number of cases that still do not present valid alignments. Consequently, an algorithm to align non-consecutive poses has been developed and tested. This algorithm not only attempts to align incorrect poses with previous ones, but it also manages a pending list for the cases in which the alignment with previous poses is unsuccessful with the aim of aligning the pose with future ones.

#### 3.4.2.3 Experiment 3. Validation threshold adjustment

The results of experiment 1 showed that the percentage of correspondences ($rat_s$ and $rat_{s-1}$) and the EFS take different values depending on the environment (fig. 3.13 and 3.14). This is due to the fact that the characteristics of point clouds depend on some factors such as the size of the environment, the elements that make it up, and its structure. This leads us to the conclusion that the validation threshold (eq. 3.3.1) can not be global. If we tried to set the same threshold for all environments, most records would be categorized incorrectly; either because successful alignments would be considered unsuccessful, or even because incorrect alignments might be accepted as correct. The latter case is especially dangerous and should be categorically avoided.

To resolve this issue, the following process is followed. Initially, the first registration (between the poses $\vec{p}_0$ and $\vec{p}_1$) is addressed and the alignment results are



(a) (b)

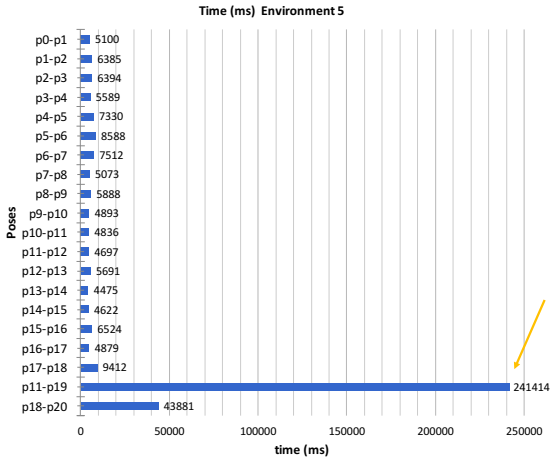
**Figure 3.16:** Examples of alignment between point clouds. (a) Incorrect alignment between poses 10 and 11 in the environment 6. (b) Correct alignment of the pose 11 with the pose 13 in the environment 6 after using the case B algorithm 3 of the novel registration algorithm.

obtained. The average value of the two percentages of correspondences ($rat_s$ and $rat_{s-1}$) is obtained and the result is multiplied by a coefficient $\gamma$. The result will be considered as the threshold (eq. 3.7) for this environment. Subsequently, the following records in the environment will be accepted as correct if eq. 3.7 is met.

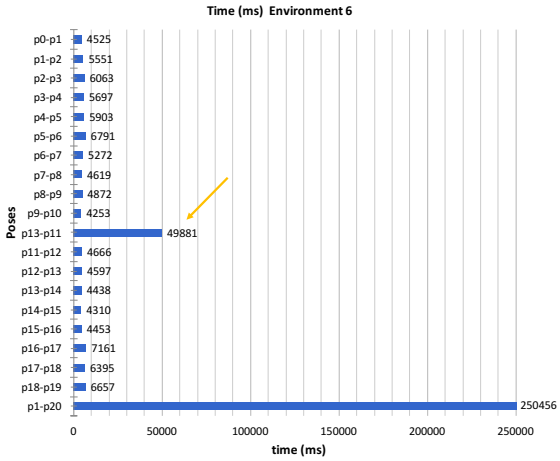
It should be noted that this threshold will only work well if the first registration was correct. Otherwise, the error will spread to the following registrations. To avoid this, the translation and rotation of the robot between the poses $\vec{p}_0$ and $\vec{p}_1$ should be small. This allows to tune the parameters by considering the diversity of the environments. In addition, the $\gamma$ value should be chosen by the user. In this experiment, different $\gamma$ values are considered to test the influence of this parameter on the validation results. Environments 3, 4, 5, and 6 are used to adjust this setting, as they are the most challenging. Additionally, it is considered a mix between environments 5 and 6 in order to add more complexity and taking advantage of the fact that environments 5 and 6 are the same except for the top plane.

Three values are tested for the $\gamma$ coefficient (0.6, 0.7 and 0.8). For each registration within an environment, four possible cases can occur. First, the registration between poses is detected as correct when in fact it is correct (**True positive**). Second, the record is detected as incorrect when in fact it is incorrect (**True negative**). Third, the record is detected as incorrect when in fact it is correct (**False negative**). Last, the record is detected as correct when in fact it is incorrect (**False positive**).

Therefore, for each $\gamma$ value, two graphs are plotted. The first shows in each environment how many registrations have been detected as *True positive, True negative, False negative or False positive*. The second graph shows how many decisions were correct (True Positive or True Negative) and how many decisions failed (False Negative or False Positive). The results obtained are shown in fig. 3.18, 3.19 and 3.18.

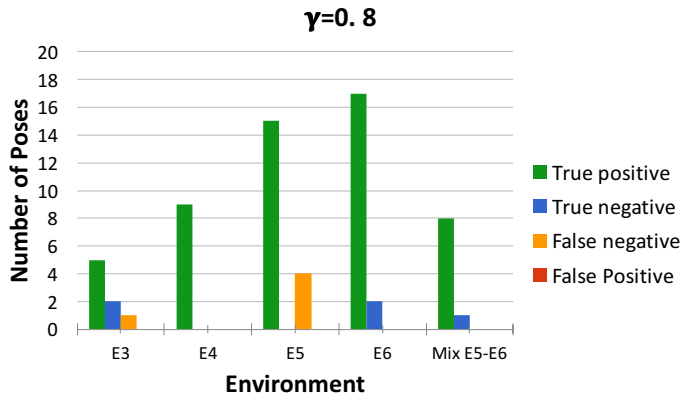



(a)

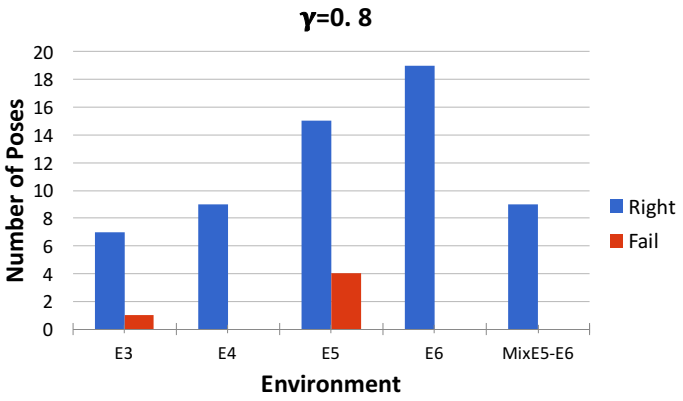



(b)

**Figure 3.17:** Computing time of experiment 2. (a) Environment 5. (b) Environment 6.

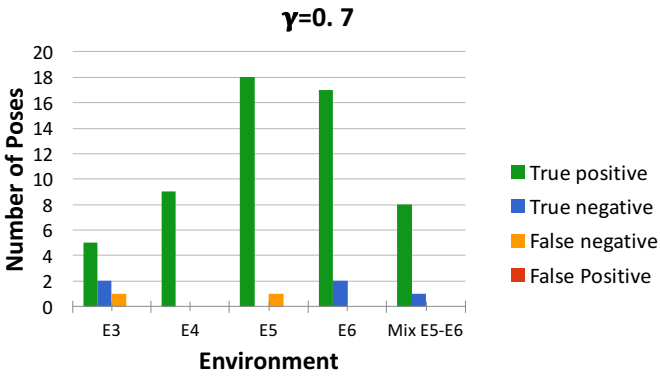



(a)

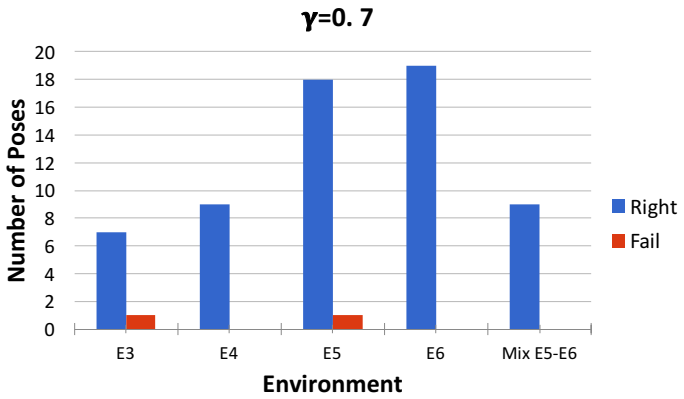



(b)

**Figure 3.18:** Experiment 3 performance when the coefficient $\gamma$ is 0.8. (a) Number of true positive, true negative, false negative and false positive registrations . (b) Number of times when the result of the algorithm is correct and the number of failures occurred.

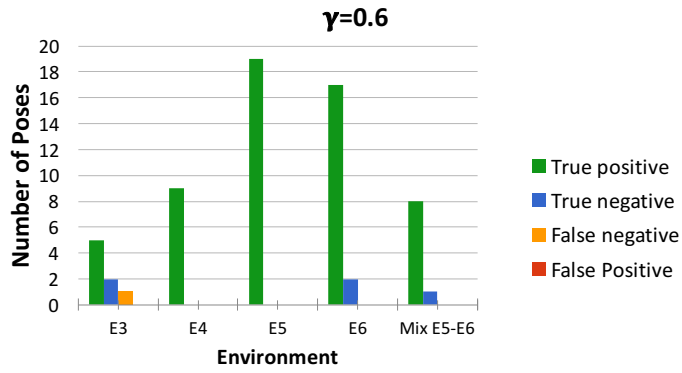



(a)

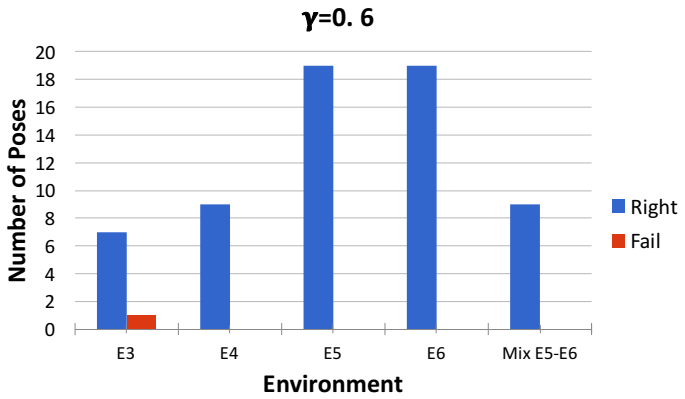



(b)

**Figure 3.19:** Experiment 3 performance when the coefficient $\gamma$ is 0.7. (a) Number of true positive, true negative, false negative and false positive registrations . (b) Number of times when the result of the algorithm is correct and the number of failures occurred.




(a)




(b)

**Figure 3.20:** Experiment 3 performance when the coefficient $\gamma$ is 0.6. (a) Number of true positive, true negative, false negative and false positive registrations . (b) Number of times when the result of the algorithm is correct and the number of failures occurred.

Using the transformation matrices calculated in experiment 1, and visually analyzing the results, the following registrations proved to be unsuccessful:

- Environment 3: $p_1 - p_2$ and $p_7 - p_8$.
- Environment 4: All registrations were correct.
- Environment 5: All registrations were correct.
- Environment 6: $p_{10} - p_{11}$ and $p_{19} - p_{20}$.
- Environment mix. 5-6: $p_2 - p_3$.

These results allow to make a deep analysis of fig. 3.18, 3.19 and 3.18.

- When $\gamma = 0.8$ (see fig. 3.18). In the environment 3, the two wrong registrations were detected as incorrect and one registration was rejected when it was actually aligned correctly. In environment 4, all poses were detected as correct. In environment 5, four poses were detected as misaligned when in fact they were correct. In environment 6, the two misaligned poses were detected as incorrect. Last, in the environment of the mixture, the incorrect registration was also detected. Consequently, the validation algorithm failed once in the environment 3 and four times in 5 (see fig. 3.18 (b)).
- When $\gamma = 0.7$ (see fig. 3.19). In environment 3, the two incorrectly aligned registrations were detected as incorrect and one registration was rejected when it was actually aligned correctly. In environment 4, all poses were detected as correct. In environment 5, it was detected that a pose was misaligned when it was actually aligned correctly. In environment 6, the two misaligned poses were detected as incorrect. Finally, in the mixing environment, the incorrect registration was detected. Consequently, the validation algorithm failed once in the environment 3 and once in 5 (see fig. 3.19(b)).
- When $\gamma = 0.6$ (see fig. 3.20). In environment 3, the two misaligned registrations were detected as incorrect and one registration was rejected when it was actually aligned correctly. In environment 4, all poses were detected as correct. In environment 5, all poses were detected as correct. In environment 6, the two incorrectly aligned poses were detected as incorrect. Finally, in the mixing environment, the incorrect registration was detected. Hence, the validation algorithm failed only once, in the environment when $\gamma = 0.6$ (see fig. 3.20(b)).

None of the three coefficients produce the false positive case (registration detected as successful when in fact it was not). This is very important because this situation must be firmly avoided. For a $\gamma$ coefficient of $0.8$,the threshold is too strict and therefore in many cases good alignments are rejected. When $\gamma = 0.6$, the calculated threshold appears to be correct more often, but $\gamma = 0.7$ ensures that incorrect alignments are not detected as correct. In conclusion, the range should be adjusted in the range of 0.6 to 0.7 for the users according to their needs.

### 3.4.3 Experiments Using Visual Data

Concerning the experiments carried out by using visual information, only three datasets were used. On the one hand, the environment 4 is especially challenging because of the poor lighting conditions and the lack of objects in the scenes, what makes it especially prone to visual aliasing and complicates the detection and correct matching of visual features. Within this environment, 9 locations are considered and data from these locations are captured. As a result, 9 point clouds and 333 RGB images are available to model the environment (37 images per position). On the other hand, environments 5 and 6 cover a wider area, whose size is around $2.5 \times 2.5$ m. Environment 5 contains information from 21 different poses but only 19 are used for this purpose as the two discarded are relatively close to other poses and can create visual confusion. Environment 3 also contains information from 21 different poses and in this case, all are used.

#### 3.4.3.1 Evaluation of the Algorithm to pair-up the images using global appearance

The visual alignment algorithm is run for every pair of consecutive locations $s-1$ and $s$. Table 3.3 shows the values of the variables $t$ and $r$, obtained after running the algorithm presented in subsection 3.3.3. Assuming that the image $K_{s-1,l_1}$ has been matched up with the image $K_{s,m_1}$, the results of the acquisition process are considered valid if the orientation of the turret is between $K_{s,m_1-1}$ and $K_{s,m_1+1}$ when $K_{s-1,l_1}$ was acquired.

Given this criterion, all the pairings provided by the algorithm turn out to be correct, despite the challenging properties of the three environments. An example of this pairing-up process can be seen in fig. 3.21 for the environment 5. The result of the image pairing process is $t = 36$, $r = 1$. This means that the image $K_{9,18}$ is matched-up with the image $K_{10,17}$ where $m_1 = (l_1 + r \cdot t) \mod 37$ (eq. 3.3.3). Fig. 3.21(a) shows the image $K_{9,18}$, and the images $K_{10,16}$ , $K_{10,17}$ and $K_{10,18}$ are shown on figures 3.21(b), 3.21(c) and 3.21(d) respectively. These figures show that the pairing provided by the algorithm is successful, since the orientation of the image $K_{9,18}$ is between the orientations of images $K_{10,16}$ and $K_{10,18}$. Additionally, the experiments have shown that the average necessary time to pair up the images of pose $s-1$ with the images of pose $s$ is equal to $90$ milliseconds.

#### 3.4.3.2 Evaluation of the alignment algorithm

This subsection assesses the performance of the alignment algorithm presented in subsection 3.3.3. For every pair of consecutive locations $s-1$, $s$, the second pose can be estimated with respect to the first one by using the transformation matrix calculated with the proposed alignment algorithm $T_{s-1,s}$. After considering each pair of consecutive locations and the three environments proposed for this aim, the results are shown in fig.3.22, 3.23 and 3.24. In these figures, both the position of the robot calculated with the algorithm and the ground truth (considered as the estimated position using

**Table 3.3:** Results of the image pairing-up process

| Position | | Environment 4 | | Environment 5 | | Environment 6 | |
|---|---|---|---|---|---|---|---|
| $s-1$ | $s$ | $t$ | $r$ | $t$ | $r$ | $t$ | $r$ |
| 0 | 1 | 36 | -1 | 1 | -1 | 2 | -1 |
| 1 | 2 | 0 | 1 | 36 | -1 | 35 | -1 |
| 2 | 3 | 36 | -1 | 0 | -1 | 1 | -1 |
| 3 | 4 | 36 | -1 | 0 | -1 | 36 | -1 |
| 4 | 5 | 0 | 1 | 0 | 1 | 36 | -1 |
| 5 | 6 | 0 | 1 | 36 | 1 | 35 | -1 |
| 6 | 7 | 36 | -1 | 0 | 1 | 0 | -1 |
| 7 | 8 | 36 | -1 | 1 | 1 | 0 | -1 |
| 8 | 9 | - | - | 36 | 1 | 0 | 1 |
| 9 | 10 | - | - | 36 | 1 | 36 | 1 |
| 10 | 11 | - | - | 0 | 1 | 36 | 1 |
| 11 | 12 | - | - | 24 | 1 | 36 | -1 |
| 12 | 13 | - | - | 4 | 1 | 36 | -1 |
| 13 | 14 | - | - | 35 | 1 | 36 | -1 |
| 14 | 15 | - | - | 33 | 1 | 0 | -1 |
| 15 | 16 | - | - | 36 | 1 | 36 | -1 |
| 16 | 17 | - | - | 35 | 1 | 36 | -1 |
| 17 | 18 | - | - | 35 | 1 | 1 | 1 |
| 18 | 19 | - | - | - | 1 | 31 | -1 |
| 19 | 20 | - | - | - | 1 | 10 | -1 |

depth data) are represented. For clarity, the position of the robot in the horizontal plane (yz) and in the vertical plane (xy) is shown separately.

Fig. 3.25 represents the localization error at each robot position (mm), considering each environment separately. Environment 4 shows relatively accurate results with error values below 4.5 mm, compared to ground truth (fig. 3.25(a)). With respect to environment 5, it presents slightly less accurate results, compared to environment 4, with a maximum global error around 25 mm in the pose 12 (fig. 3.25(b)). Last, regarding the results in environment 6, it is shown that the algorithm produces good results (global error lower than 20 mm), in the last two poses (19 and 20), where the method fails (figures 3.25(c) and 3.25(d)). In general, considering all the results, the method provides relatively accurate position estimations and is unsuccessful in only two cases. The results confirm that even in these particularly challenging underfloor environments, the proposed approach exhibits relatively accurate behaviour for estimating alignment between two consecutive poses and thus estimating the location of the robot from an initial position.

Last, table 3.4 shows some relevant information about each environment. The parameter #Features specifies the total number of visual keypoints matched between

(a)

(b) (c) (d)

**Figure 3.21:** Results of an image pairing-up process. Figure 3.21(a) is the image 18 acquired from location (pose) $s = 9$ (named image $K_{9,18}$). Figures 3.21(b), 3.21(c) and 3.21(d) are the images 16, 17 and 18 acquired from pose $s = 10$ (named, respectively, $K_{10,16}$, $K_{10,17}$ and $K_{10,18}$).

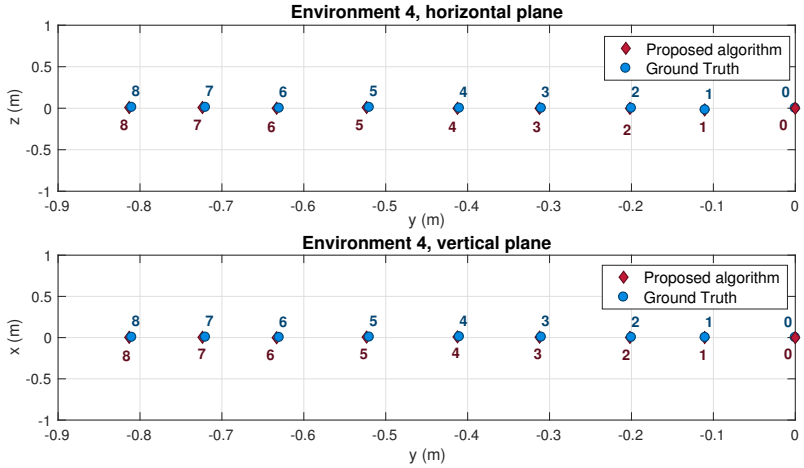



**Figure 3.22:** Results of the proposed alignment method in environment 4. Horizontal (yz) plane and vertical (xy) plane. The position of the robot obtained with the proposed algorithm and the ground truth, which was estimated from depth data. are shown, separately.

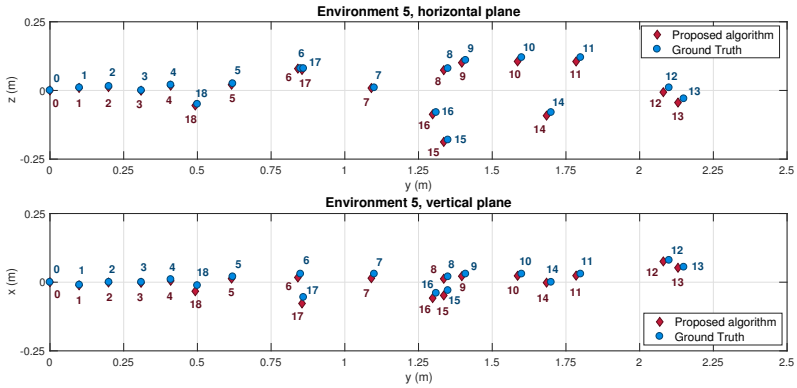



**Figure 3.23:** Results of the proposed alignment method in environment 5. Horizontal (yz) plane and vertical (xy) plane. The position of the robot obtained with the proposed algorithm and the ground truth, which was estimated from depth data. are shown, separately.

images. The second parameter, #Correspondences, indicates the number of points used by the system to do the alignment and estimate the transformation matrix once outliers have been removed. Finally, %Correspondences is the ratio between the two previous parameters. The results show that the parameter %Correspondences allows to know whether the alignment is correct or wrong. For instance, the estimation of pose 19 with respect to pose 18 in environment 6 presents $20\%$ the lowest number of correspondences, which is the lowest value of all the experiments. In this point, the algorithm starts to fail. Hence, a threshold to consider the alignment correct can be set around $40\%$, this way, subsequent poses are not aligned with respect to this unsuccessful ones to avoid spreading the error. Regarding computing time, this part of the process (finding the matches, getting the depth of these matches and aligning two point clouds) has an average calculation time of 760 ms. Considering this together with the time required to match the images, the average total time to obtain the transformation matrix between two consecutive poses is equal to 850 ms.

## 3.5 Conclusion

This chapter proposes novel methods to solve the localization problem of a mobile robot which moves through underfloor voids that present challenging features for traditional methods. To solve this problem, the robot is equipped with an RGB-D sensor that extracts images and depth data from the environment. The results presented in this work show the robustness and effectiveness of the proposed methods and their ability to cope with such challenging underfloor environments.

As for the contribution carried out with the depth data information, three main contributions were conducted. The first contribution is an algorithm that uses point cloud information to estimate the registration between poses. The proposed algorithm present noticeable improvements in comparison with previous proposed methods, due

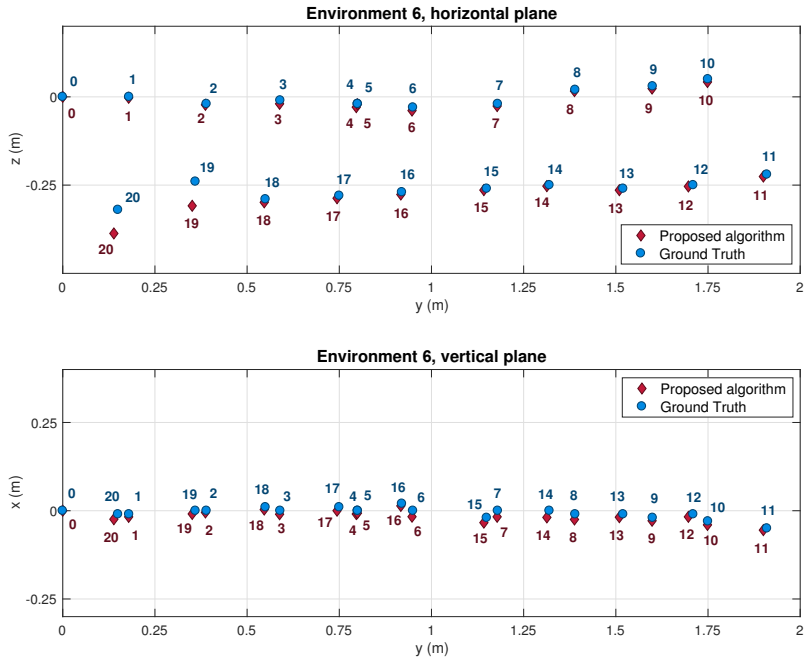
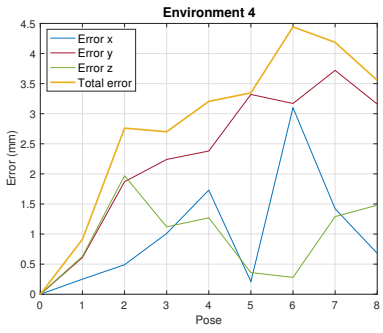


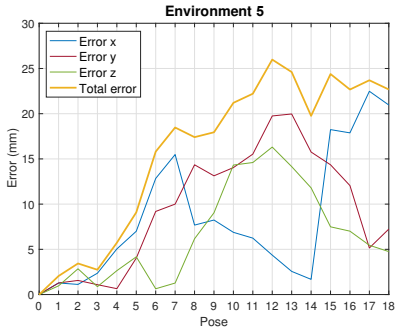

**Figure 3.24:** Results of the proposed alignment method in environment 6. Horizontal (yz) plane and vertical (xy) plane. The position of the robot obtained with the proposed algorithm and the ground truth, which was estimated from depth data. are shown, separately.

to the fact that they are not able to extract reliable features in the type of environments proposed. After obtaining the point clouds from the laser sensor, these are downsampled and the information in the top and bottom planes is removed with the aim to obtain clouds more light that lead to more robust results. Last, the transformation matrix is calculated through the registration and, thus, the alignment is estimated. The experiments showed that this algorithm works successfully in environments where the characterization using regular algorithms is difficult. Furthermore, although the registration algorithms are usually based on a coarse alignment (using visual information), fine alignment (using depth information), and an optional optimization, the method proposed here can reach sufficiently accurate results simply by using the depth information. The results show that this algorithm works successfully in most cases and also takes a few seconds to arrive at the solution.

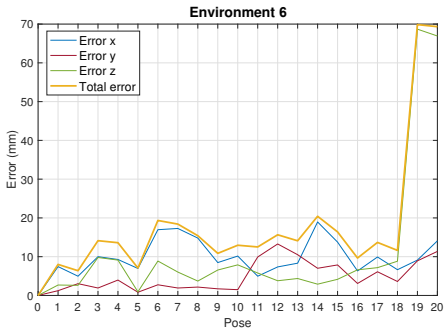
Moreover, as second contribution, a novel algorithm is proposed to solve the cases in which the alignment between consecutive poses was unsuccessful. This algorithm attempts to align the poses that were not well aligned with other poses within the environment. The obtained results show that this method successfully aligns the majority of cases. Therefore, this approach can manage a notorious utility for online



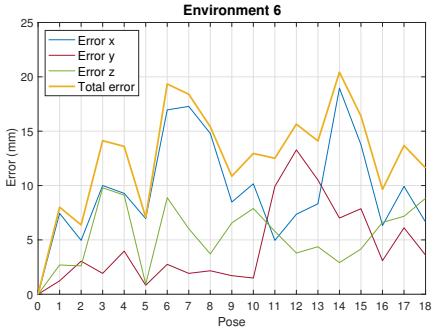

(a)




(b)




(c)




(d)

**Figure 3.25:** Error obtained along every axis for every pose in each environment and total error, expressed in mm. Fig. 3.25(a): environment 4; fig. 3.25(b): environment 5 and fig. 3.25(c): environment 6. Fig. 3.25(d) shows again the results obtained in environment 6 but removing the two last poses (which have proved to be unsuccessfully estimated).

**Table 3.4:** Results of the alignment using the previously selected 3D points.

| Poses | #Features | | | #Correspondences | | | %Correspondences | | |
|---|---|---|---|---|---|---|---|---|---|
| $(s-1)$ - $(s)$ | Env4 | Env5 | Env6 | Env4 | Env5 | Env6 | Env4 | Env5 | Env6 |
| 0-1 | 99 | 102 | 117 | 152 | 138 | 146 | 65 | 74 | 80 |
| 1-2 | 107 | 105 | 106 | 155 | 160 | 133 | 59 | 66 | 80 |
| 2-3 | 99 | 84 | 87 | 161 | 105 | 124 | 61 | 80 | 70 |
| 3-4 | 99 | 130 | 90 | 155 | 162 | 129 | 63 | 80 | 70 |
| 4-5 | 106 | 91 | 170 | 161 | 142 | 272 | 65 | 64 | 63 |
| 5-6 | 95 | 65 | 108 | 159 | 97 | 155 | 60 | 67 | 70 |
| 6-7 | 127 | 87 | 111 | 178 | 115 | 153 | 71 | 76 | 73 |
| 7-8 | 108 | 107 | 99 | 168 | 164 | 117 | 64 | 65 | 85 |
| 8-9 | - | 79 | 85 | - | 105 | 102 | - | 75 | 83 |
| 9-10 | - | 126 | 183 | - | 206 | 207 | - | 61 | 88 |
| 10-11 | - | 86 | 20 | - | 112 | 29 | - | 77 | 69 |
| 11-12 | - | 141 | 126 | - | 204 | 147 | - | 69 | 86 |
| 12-13 | - | 98 | 137 | - | 132 | 168 | - | 74 | 82 |
| 13-14 | - | 33 | 104 | - | 64 | 133 | - | 52 | 78 |
| 14-15 | - | 47 | 111 | - | 90 | 145 | - | 52 | 77 |
| 15-16 | - | 102 | 93 | - | 122 | 124 | - | 84 | 75 |
| 16-17 | - | 30 | 121 | - | 71 | 147 | - | 42 | 82 |
| 17-18 | - | 50 | 91 | | 85 | 118 | - | 59 | 77 |
| 18-19 | - | - | 12 | - | - | 59 | - | - | 20 |
| 19-20 | - | - | 87 | - | - | 147 | - | - | 59 |

localization processes where it is desirable to ensure that no information is lost while the robot obtains the exact route within the environment. Although additional computing time is required, barely no pose information is lost and an accurate global map can be created.

The percentage of correspondences together with the value of EFS allows to calculate a validation parameter. The third contribution is to adjust this validation threshold and test its influence on the accuracy of the validation process. Good results were obtained when $\gamma = 0.7$ and $\gamma = 0.6$. From these results, we can conclude that users (according to their needs) should choose a value within this range: when this coefficient approaches 0.6, the validation step tends to be correct in most cases; if it is closer to 0.7, the validation step reduces the chance of detecting failed alignments as correct.

Concerning the contributions related to the use of visual data, this approach extracts visual keypoints from sets of color images and matches them. The point clouds used to get the alignment are constructed using only these matches. Taking these point clouds as input, we use the PCL library to robustly estimate the transformations between two consecutive postures of the mobile robot. Finally, the approach is quantitatively evaluated using different RGB-D data sets acquired in real underfloor

environments. Experiments in these environments show that the framework presents successful results in position estimation.

## 3.6 Publications Related to this Chapter

The main results presented in this chapter are related to the following publications:

- C. Parra, S. Cebollada, L. Payá, M. Holloway, O. Reinoso. A Novel Method to Estimate the Position of a Mobile Robot in Underfloor Environments Using RGB-D Point Clouds. IEEE Access. Ed. IEEE. Vol. 8, pp. 9084-9101 (January 2020) [208] **JCR-SCI Impact Factor (2019): 3.745**, Quartile **Q1**.

  - This paper presents a the localization approach for a mobile robot, which is equipped with a laser scanner and an RGB camera. The data captured by both sensors is used to build point clouds that describe the appearance of the environment. To estimate the current position of the robot within the environment, a point cloud is captured and the aliment algorithm is tackled with point clouds previously captured in prior poses.

- S. Cebollada, L. Payá, M. Juliá, M. Holloway, O. Reinoso. Mapping and localization module in a mobile robot for insulating building crawl spaces. Automation in Construction. Ed. Elsevier. Vol 87, pp. 248-262 (March 2018) [41] **(SCI-JCR Impact Factor: 4.313, Q1)**.

  - This paper presents two depth-data based algorithms to robustly obtain the alignment between two point clouds captured in challenging underfloor environments.

- S. Cebollada, C. Parra, M. Juliá, M. Holloway, L.M. Jiménez, O. Reinoso. Alineamiento 3D desde posiciones no cercanas de un robot para trabajos en interiores a partir de imágenes RGB-D. XXXVII Jornadas de Automática (Madrid (Spain), 7-9 September 2016) .Ed. CEA-IFAC. ISBN:978-84-617-4298-1 - pp. 374-381 [238]

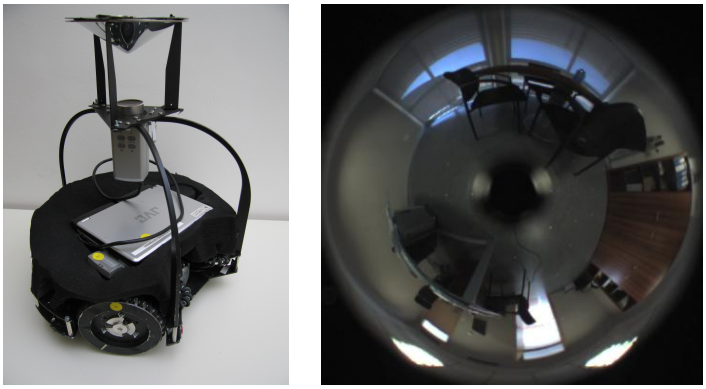
  - This paper presents methods based on visual and depth data to solve the alignment between poses within a challenging environment. Additionally, this paper also presents a method to manage the alignment of poses whose registration algorithm presented mistakes.

# 4 Compression and Subsequent Localization in Hierarchical Maps

## 4.1 Introduction

Vision sensors have been widely used for mapping and localization purposes. Among the different configurations frequently proposed by authors, as mentioned in chapter 1, omnidirectional cameras constitute a good alternative. This is due to the fact that they can provide a big amount of information with a field of view of 360 deg. around them and their cost is relatively low in comparison with other kinds of sensors. Concerning additional advantages through using this type of systems, the features in the images are more stable (because they stay longer as the robot moves) and permits both building rich maps and estimating the robot position. Omnidirectional cameras have been successfully used by different authors for mapping and localization. For example, Valiente *et al.* [283] used local features extracted from omnidirectional images to generate reliable visual odometry to improve the task of Simultaneous Localization And Mapping (SLAM). Marinho *et al.* [171] used feature extractions and machine learning techniques to tackle localization using omnidirectional images. Faessler *et al.* [72] present a vision-based quadrotor system for mapping a dense three-dimensional area in line for the purpose of eliminating the delay between the quadrotor and external systems. Payá *et al.* [211] conducted an extensive study, which presents a state of the art of the most relevant localization and mapping algorithms developed with omnidirectional visual information. An example of a mobile robot that has an omnidirectional camera mounted on it is shown in fig. 4.1(a) and an example of omnidirectional image is shown in fig. 4.1(b).

Subject to how to extract and represent the information most relevant, mapping and localization tasks based on visual information have been commonly solved



(a) (b)

**Figure 4.1:** (a) Example of a robot equipped with an omnidirectional vision system. Image licensed under Creative Commons Attribution-Share Alike 2.5 Generic license. (b) Example of an omnidirectional image that was captured from an office.

by using two frameworks. The first approach is based on detecting, describing and tracing relevant local features obtained from a set of scenes [282], [234]. The second approach consists in building a unique descriptor per image that contains information on its global appearance [137], [150], [279]. Concerning the methods that are based on using local features, they consist in extracting some outstanding pixel points from each scene and then creating a descriptor for each point, using the information around it (fig. 4.2(a)). These descriptors have been widely used for solving visual mapping and localization. A wide range of authors have proposed methods that use them, such as Angeli *et al.* who employ SIFT [6] or Murillo *et al.* who use SURF [191]. Nonetheless, these methods present some disadvantages such the environments must be rich in details in order to obtain reliable landmarks. Another drawback is that keypoints detection is not always robust against changes in the environments (e.g. changes of lighting conditions) and sometimes, description is not totally invariant to changes in the robot position. Moreover, these methods can be computationally complex, hence, in those cases, building models in real time would not be possible. As for the methods based on global-appearance or holistic descriptors, they consist in treating each image as a whole. That is, each image is represented by a unique descriptor that contains information about its global appearance (fig. 4.2(b)). These approaches lead to simpler mapping and localization algorithms, since each scene is described by only one descriptor. Thus, mapping and localization can be done simply by storing and comparing the descriptors in pairs. In addition, they could be more robust in dynamic, unstructured environments. As disadvantages, these methods present a lack of metric information. Visual aliasing can also create a negative impact on mapping and localization tasks, as indoor environments are prone to presenting repetitive visual structures. Additionally, modeling large environments would require a large number of

images, and this would introduce serious problems when this form of representation has to be used in real-time applications. Hence, holistic descriptor method is an intuitive alternative to solve the mapping and localization problem, but its robustness against several issues must be evaluated. Many authors have addressed mapping and localization using global-appearance descriptors (fig. 4.2(b)). For example, Liu *et al.* [159] propose a descriptor based on color features and geometric information. By using this descriptor, a topological map can be built. Payá *et al.* [210] develop a comparative analysis of some description methods to carry out the mapping and localization tasks by using omnidirectional vision sensors. Rituerto *et al.* [230] propose the use of the descriptor *gist* [150, 204] to create topological maps from omnidirectional images. More recently, Berenguer *et al.* [21] propose the Radon transform [224] as global-appearance descriptor of omnidirectional images as hierarchical localization method.

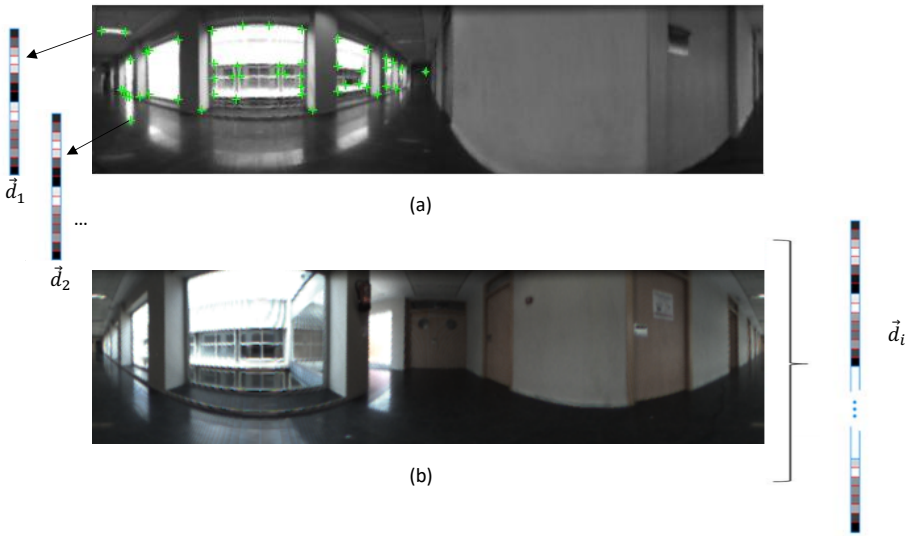



**Figure 4.2:** Two main approaches to obtain the most relevant information from images and use that information for visual mapping and localization purposes. (a) Detection, description and tracking of some relevant landmarks along a set of scenes. (b) Building a unique descriptor per image that contains information on its global appearance.

Concerning map building, in the related literature, two main frameworks have been proposed to address this task: metric maps, which represent the environment with geometric precision; and topological maps that describe the environment as a graph containing a set of locations with related links. In this sense, Garcia-Fidalgo and Ortiz [85] presented a review of the main approaches considered to perform topological mapping and localization through visual information in recent years. Regarding this second option, arranging the topological information hierarchically constitutes an efficient alternative. This framework consists of creating a map that consists of several layers with a hierarchical structure. High-level ones present a relatively compact

amount of information, allowing for a rough but quick localization. Low-level layers usually have more information and are used to refine the position. Hence, to address the problem of mapping and localization, hierarchical maps are an effective alternative. To cite one example, da Silva *et al.* [57] a localization and navigation approach for mobile robots using topological maps and using CNN to obtain omnidirectional image descriptors.

Some authors have proposed data compression strategies to build efficient high level maps using visual information.To create a visual model, initially, a set of images captured from various points of view of the environment is usually available for mapping. Among the compression methods, clustering algorithms can be used to compact the information in this model to create a high-level map, which would cluster this set of images into several groups containing visually similar scenes and represent each cluster with a representative instance. Some authors have used clustering algorithms to perform the compression task. For example, Valgren *et al.* [282] address online topological mapping using incremental spectral clustering. Štimec *et al.* [264] use an unsupervised clustering based on their own multiple-space algorithm to perform topological mapping hierarchically using omnidirectional images. More recently, Shi *et al.* [250] s have proposed the use of a differential clustering method to improve the compression of telemetry data. Ideally, using only visual information, a clustering algorithm should group images that have been captured at geometrically near points.

Therefore, in this chapter, we present an exhaustive comparative evaluation of several image descriptors to create compact models of an unknown environment and an approach to solve the localization problem using hierarchical models. Moreover, a comparative evaluation of some global descriptors is carried out to know which one behaves more robustly against illumination changes. The results of the present work can be useful in this field as they will allow to choose the best method of description and to tune correctly the main parameters in such a way that a compromise between computational cost and precision is reached. To construct and compress models only visual information is considered. This leads to purely topological models that must be able to deal with the phenomenon of visual aliasing. We focus on the use of holistic descriptors, as local descriptors have been considered in previous research. To carry out the batch of experiments, different sets of omnidirectional images captured in an indoor environment are used. They include regions with similar appearance and changes in lighting conditions. Additionally, this work also considers the use of approaches based on deep learning to describe the scenes globally. The aim consists in evaluating which method solves more efficiently the localization task under the conditions previously exposed.

The remainder of the chapter is structured as follows: Section 4.2 introduces the global-appearance descriptors that are tested along the chapter. After that, section 4.3 presents the clustering methods proposed to compress visual models. Section 4.4 shows the experiments tackled to test the validity of the proposed methods to solve the localization. Additionally, this chapter also presents the experiments concerning the evaluation of these methods under changing lighting conditions. Section 4.6 outlines the conclusions. Last, section 5.7 presents the publications related to the present work.

## 4.2 Holistic Descriptors

This section presents the holistic description methods proposed in this work to characterise the set of images. The use of this methods to carry out this study is justified, since global-appearance descriptors constitute an interesting alternative for mapping and localization. In this chapter, four methods are evaluated to compress visual models and also carry out visual localization: the Fourier Signature (FS), the Histogram of Oriented Gradients (HOG), the *gist* of the scenes and a holistic descriptor based on a Convolutional Neural Network (CNN).

Basically, the mobile robotics task that is proposed to study in the present work consists in the following. The robot moves along the floor plane and captures omnidirectional images using a hyperbolic mirror that is mounted over a camera along the vertical axis. After obtaining the omnidirectional images, a conversion to panoramic must be carried out ($im(x,y) \in \mathbb{R}^{N_x \times N_y}$), since panoramic images are the starting point of the propsoed method. Afterwards, a description methods (among the four proposed) is used to calculate the global-appearance descriptor vector $\vec{d} \in \mathbb{R}^{l \times 1}$. After using the description method, a descriptor for each image is calculated; thus, the visual model is composed of a set of descriptors, $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ where each descriptor is $\vec{d_j} \in \mathbb{C}^{l \times 1}$ and corresponds to the image $im_j$.

### 4.2.1 Fourier Signature Descriptor

This description method is based on the use of the Discrete Fourier Transform (DFT). The process to obtain the holistic descriptor is as follows. The Fourier signature (FS) of a panoramic image is calculated from the intensity matrix of the original image. After that, a new complex matrix is obtained ($IM(u,v) \in \mathbb{C}^{N_x \times N_y}$). Departing, the DFT of each row is calculated. Only the $k_1$ first columns of this matrix are retained, because the main information is in the low frequency components. Afterwards, the resultant matrix ($IM(u,v) \in \mathbb{C}^{N_x \times k_1}$) is decomposed into a matrix that contains the magnitudes and another that contains the arguments. The matrix of magnitudes ($A(u,y) \in \mathbb{R}^{N_x \times k_1}$) is invariant against changes of the robot orientation in the movement plane (if the original image is panoramic). Last, the holistic descriptor is obtained by arranging the $k_1$ columns of the magnitudes matrix in one single column ($\vec{d} \in \mathbb{R}^{N_x \cdot k_1 \times 1}$).

The Fourier Signature descriptor was firstly proposed by Menegatti *et al.* [178] to create an image-based memory to carry out a robot navigation task. Payá *et al.* [210] tackle a deep study about the computational cost and the error in localization by using this descriptor together with a Monte Carlo approach with the aim to solve localization tasks in indoor environments.

### 4.2.2 Histogram of Oriented Gradients Descriptor

The Histogram of Oriented Gradients (HOG) is a description method that has been extensively proposed to carry out object detection tasks. This descriptor is remarkable because it is easy to build, leads to successful results concerning detection tasks and

does not require a high computational cost. The building process is based on the orientation of the gradient in localized parts of the image. The development process carried out in this work consists basically in dividing the panoramic image into small regions ($k_2$ horizontal cells) and compiling a histogram with $b$ bins for the pixels that are included inside each cell by using their gradient orientation. The combination of this information provides the objective descriptor ($\vec{d} \in \mathbb{R}^{b \cdot k_2 \times 1}$). HOG description was originally used in mobile robotics by Dalal and Triggs [58] to solve people detection task. Zhu *et al.* [319] presented a version that improved the computational time as well as the efficiency to detect people. Dong *et al.* [66] propose an HOG-based multi-stage approach for object detection and pose recognition in the field of service robots. A detailed explanation of the description method proposed can be found in [209].

### 4.2.3 *Gist* Descriptor

The *gist* description was introduced by Oliva *et al.* [205] and it has been commonly used to recognize scenes. Since then, several versions have been developed, which are based on different features from the original images, such as color, orientation, texture, etc. [252]. Some authors have proposed the use of this holistic descriptor descriptor to solve visual mobile robotics tasks. For example, Chang *et al.* [47] used this global-appearance descriptor for localization and navigation. Murillo *et al.* [192] also used a reduced version (by means of a Principal Components Analysis) of the *gist* descriptor to solve the localization problem.

The version proposed throughout this thesis is described in [209] and works with the orientation information obtained by means of a set of Gabor filters. The process is basically as follows. Different resolution levels $m$ are obtained from the panoramic image. Then, $n_{masks}$ orientation filters are applied over each level. Last, the pixels of each image are grouped in $k_3$ horizontal blocks and the information obtained from each block is arranged in a vector ($\vec{d} \in \mathbb{R}^{n_{masks} \cdot m \cdot k_3 \times 1}$).

### 4.2.4 CNN-based Descriptor

This method comes from the use of deep learning for classification, as Krizhevsky *et al.* [136] do. The neural network training is addressed in two steps. First, it performs a learning process, that is, a set of images (which are already labelled) are collected and introduced into the network. Second, once trained, the network receives validation images (also labelled) and adjusts its internal parameters to optimize results. These steps are repeated until considering the resultant performance is accurate enough. Subsequently, the network is available to tackle the classification task: a new image is introduced and the CNN returns the most likely label. During the classification process, descriptors are obtained from the fully connected layers of the neural network. This information can be viewed as a holistic descriptor of the input image. Hence, they can also be used to perform the localization task in the same way as the previous proposed holistic descriptors. The neural network architecture we use in this chapter is *places* [318], that was trained with around 2.5 million images to categorize 205 possible types of scenes. Fig. 4.3 shows the architecture of this CNN.To obtain holistic

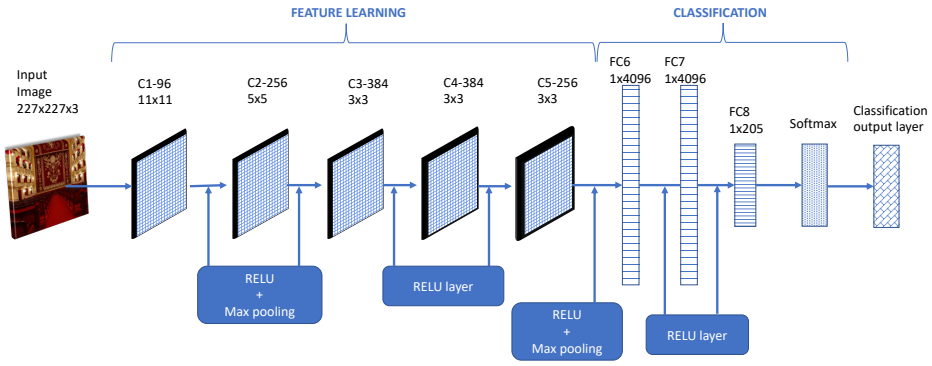



**Figure 4.3:** CNN *places* architecture, which is designed departing from the *'Caffe'* network. Layers $fc_7$ and $fc_8$ are used in this work to obtain global-appearance descriptors from the original input image.

descriptors of these layers, the networks are used directly with the pre-training network developed by the creators, so no retraining is addressed. CNN is used directly as shown in [318]. The descriptors extracted from this network correspond to those calculated in layers $fc_7$ and $fc_8$. These descriptors contain respectively 4096 ($\vec{d} \in \mathbb{R}^{4096\times 1}$) and 205 ($\vec{d} \in \mathbb{R}^{205\times 1}$) components. This type of descriptor has been used by other authors such as Mancini *et al.* [170], who use it to categorize sites with the Naïve Bayes classifier. As for mobile robot localization, Payá *et al.* [213]proposed descriptors based on CNN to create hierarchical visual models. Differently, Xu *et al.* [307] propose the use of a CNN that detects objects from the images and establishes relationships between the detected objects, then the established relationships are used to calculate the similarity between images.

### 4.2.5 Homomorphic Filter

In visual localization tasks, typical situations may happen such as lighting variations and changes in the position of some objects (chairs, tables, open doors, etc.). Therefore, the descriptors used to carry out the task must be robust against these circumstances. Based on studies like the showed by Fernandez *et al.* in [75], it is proved that some image pre-treatments can improve the localization accuracy in indoor environments with different lighting levels. Among the studied techniques, the use of homomorphic filter [89] is highlighted.

The homomorphic filter permits filtering the luminance and reflectance components from an image separately. The localization results have been improved by using the HOG descriptor with the homomorphic filter has showed. Nevertheless, in the FS and *gist* cases, the results were similar or worse than without using this pre-treatment filter. Therefore, in the present thesis, the following configurations are used throughout the experiments based on hand-crafted global-appearance descriptors: FS without filter, HOG with filter and *gist* without filter.

## 4.3 Clustering Methods to Compact the Visual Information

This section addresses the creation and compaction of topological models. In this case, the topological models proposed are based purely in visual information, then, only global-appearance descriptors are used. Moreover, the localization task is subsequently solved through these visual models. This way, the mobile robotics tasks are addressed through the next two steps.

1. Learning the visual model (mapping): Building a map of the environment and compacting it. A set of omnidirectional images are obtained from different positions within the environment and a global-appearance descriptor is calculated for each image. After capturing the images and calculating the holistic descriptors, a clustering method is used to determine the structure and compact the model.

2. Estimating the current position (Localization): After building and compressing the map, the robot is ready to estimate its position inside the model. That is, the robot obtains a new image from an unknown position, then it calculates a holistic descriptor and compares that descriptor with the set of descriptors obtained in the *learning* step. Using this comparison, the robot is able to estimate its position in the map.

Concerning the mapping step, the robot moves around the environment, captures omnidirectional images from different positions to cover the whole environment and convert those images to panoramic. This way, a set of images is collected $I = \{im_1, im_2, ..., im_N\}$ where $im_j \in \mathbb{R}^{N_x \times N_y}$. After that, a global-appearance descriptor is calculated for each image, thus, a set of descriptors is obtained $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ where $\vec{d_j} \in \mathbb{C}^{l \times 1}$.

The main problem of considering this mapping strategy appears when the environment of work has considerable dimensions. This is due to the fact that larger environments imply more images to capture, hence this also leads to require more memory space to store the data and also more computational time to process the information. This is the reason to propose compacting models as a possible solution in such a way that the compacted model retains most of the visual information and also permits solving the localization problem efficiently.

Therefore, through the present work, we propose a model compressing framework based on clustering methods, with the aim to create a two-layer hierarchical structure. The low-level layer is composed of a set of descriptors and the high-level layer is the set compacted via clustering. Each cluster is characterized by the common attributes of the instances that form its group. This way, the data set $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ is divided into $n_c$ clusters $C = \{C_1, C_2, ..., C_{n_c}\}$ under the

conditions:

$$\begin{aligned} C_i \neq \emptyset, i = 1, ..., n_c \\ \bigcup_{i=0}^{m} C_i = D \\ C_i \bigcap C_j = \emptyset, i \neq j, i, j = 1, ..., n_c. \end{aligned} \tag{4.1}$$

After clustering the information, each group of descriptors is reduced to a unique representative descriptor, which is obtained by calculating the average of all the descriptors that compose that cluster. Therefore, a set of representatives is obtained $R = \{\vec{r_1}, \vec{r_2}, ..., \vec{r_{n_c}}\}$, and this set of representatives compose the high-level layer (the model is compacted).

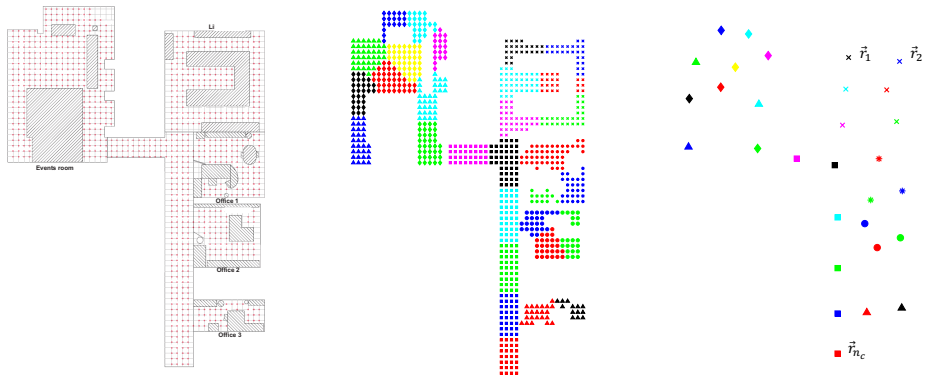
The compaction process can be visually explained by means of the pictures presented in the fig. 4.4. First, images are captured in different position within a whole map (fig. 4.4(a)). The result obtained from the clustering process is shown in fig 4.4(b). Last, the representatives descriptors are obtained (fig. 4.4(c)). The representative descriptors are calculated as the average descriptor among those grouped in the same cluster. Additionally, the position of these representative descriptors are also calculated as the average position of the capture points of the images included in the same cluster. These positions are used to represent visually the high-level layer and they are only used as a ground truth to test the performance obtained by using the compact map for localization. This positions are not used neither to build the map nor to localize the robot. Since in this chapter, different clustering methods are analysed and only visual information is used for this purpose.

Regarding the clustering process to compact the visual models, two methods are studied: spectral clustering and self-organizing maps. Concerning the visual data, three global-appearance descriptors are proposed: FS, HOG and *gist*. To evaluate the correctness of the approach, the geometrical compactness of the clusters and their utility to solve the localization task are tested.

### 4.3.1 Spectral Clustering Algorithm

Spectral clustering algorithms [167]have been proved to be suitable for processing large data. In the present work, a spectral normalized clustering algorithm is used as it was introduced by Ng *et al.* [200]. This algorithm has already been used for mapping along with local features extracted from the scenes [264, 281]. Spectral clustering may be more efficient than traditional methods such as k-means or hierarchical clustering in large environments due to the fact that spectral clustering considers mutual similarity between instances.

The algorithm proposed to compress the visual information departs from the set of global-appearance descriptors $D = \{\vec{d_1}, \vec{d_2}, ...\vec{d_N}\}$ obtained from the images collected in the environment and the desired number of clusters $n_c$. The similitude between descriptors is firstly calculated. This parameter is calculated for each pair of




(a) (b) (c)

**Figure 4.4:** Example of indoor map and information compression. (a) Positions where images were captured. (b) Result of the clustering process. (c) Each cluster is reduced to one representative.

descriptors contained in the dataset. Thus, a matrix of similitudes $S$ is obtained as $S_{ij} = e^{-\frac{|\vec{d_i} - \vec{d_j}|^2}{2\sigma^2}}$ where $\sigma$ is a parameter that controls the rapidity of reduction of the similitude when the distance between $\vec{d_i}$ and $\vec{d_j}$ increases. The process to carry out the clustering is as follows:

1. Calculation of the normalized Laplacian matrix:

$$L = I - D^{-1/2} S D^{1/2} \tag{4.2}$$

   where D is a diagonal matrix $D_i = \sum_{j=1}^{N} S_{ij}$.

2. Calculation of the $n_c$ main eigenvectors of $L$, $\{\vec{u_1}, \vec{u_2}, ..., \vec{u_{n_c}}\}$. Arranging these vectors by columns, the matrix $U \in \mathbb{R}^{N \times n_c}$ is obtained.

3. Normalization of the matrix $U$ to obtain the matrix $T \in \mathbb{R}^{N \times n_c}$.

4. Extraction of vector $\vec{y_i} \in \mathbb{R}^{n_c}$ from the i-th row of the matrix $T$. $i = 1, ..., N$.

5. Clustering of the $\vec{y_i}$ vectors by using a simple clustering algorithm (such as k-means or hierarchical clustering). Through this, the clusters $A_1, A_2, ..., A_{n_c}$ are obtained.

6. Obtaining the clusters with the original data as $C_1, C_2, ..., C_{n_c}$ where $C_i = \vec{d_j} \mid \vec{y_j} \in A_i$.

If the number of instances $N$ or the dimension $l$ is high, the computation of the $n_c$ eigenvectors will be computationally high. The solution proposed for this issue by

Luxburg [167] consists in cancelling some components of the similitude matrix. Then, only the components $S_{ij}$ so that $j$ is among the $t$ nearest neighbours of $i$ are retained in the matrix $S$. Afterwards, the $n_c$ first eigenvectors of the Laplacian matrix $L$ are calculated by using the Lanczos/Arnoldi factorization [262]. Last, for each cluster, a representative is obtained as the average visual descriptor of the set of descriptors that compose the cluster.

### 4.3.2 Cluster with a Self-Organizing Map Neural Network

As alternative, Self-Organizing Maps (SOM) is a clustering algorithm introduced by Kohonen [130] and is an effective option for making a mapping distribution when the data present a high dimensionality [286]. This algorithm has been commonly used to group or reduce data size. The map size of the neural network determines the number of clusters as $W_{SOM} \times H_{SOM} = n_c$. The data are then grouped into $n_c$ clusters.

SOM clustering method automatically learns to classify input vectors according to their similarity and topology in the input space. Rather than competitive layers, neighbouring neurons in the SOM learn to recognize neighbouring sections of the input space. Hence, SOM learn both the distribution (as the competitive layers do) and topology of the input vectors they are trained with. Neurons can be organized into a grid, hexagonal, or random topology. The SOM network identifies a wining neuron $i^*$ using the same procedure as the one used by the competitive layer but, instead of updating only the winning neuron, all neurons are updated within a given neighbourhood $N_{i^*}(d)$ of the winning neuron.

## 4.4 Solving the Localization Problem Using Compact Topological Maps

At this point, the robot holds a visual (compressed) model of the environment, which is a hierarchical model. Departing from this model, the robot firstly uses the high-level layer to carry out a rough localization and after that, a fine localization is tackled through the use of the low-level layer. The visual localization problem has been solved by many authors through local features by using probabilistic approaches such as particle filters or Monte Carlo localization ([276] and [216]). Nonetheless, the proposals developed with holistic descriptors are scarcer. Thus, the present chapter introduces a comparative of this type of information to estimate the position of the robot by means of a hierarchical map in a specific time instant. To measure the goodness of the methods studied, the coordinates provided by the ground truth are used. Nevertheless, this information is not used to solve the localization task, since the proposed method only considers visual information. This decision permits studying the feasibility of visual sensors as the only source of information to create a compact topological map and, more concisely, of holistic descriptors. Hence, not using the position information in the mapping and localization algorithms permits isolating the effect of the main parameters of these descriptors and knowing the performance of this kind of information. The accuracy of the method is evaluated through the following error equation $error = \sqrt{(x_{gt} - x_{est})^2 + (y_{gt} - y_{est})^2}$, where $(x_{gt}, y_{gt})$ is the pose provided by the

ground truth and $(x_{est}, y_{est})$ is the pose estimated by the algorithm for the captured image in the test dataset.

### 4.4.1 Distance Measures Between Descriptors

In order to know how similar two panoramic images are by means of their holistic descriptors, some distance measurements have been proposed. Previous research works ([210], [209]) have evaluated the relation between the geometrical distance between capture points and the distance between holistic descriptors. Those works show that even if the robot moves a short distance, the descriptor changes. Therefore, global-appearance descriptors can be used to detect even small movements.

In this manner, an evaluation among different distance methods can be carried out with the aim to obtain the method that best performs the distance between images. The lower the distance between those images is, the more similar they are. This kind of distances are used in the localization step. We consider two descriptors $\vec{a} \in \mathbb{R}^{l\times 1}$ and $\vec{b} \in \mathbb{R}^{l\times 1}$, where $a_i$ and $b_i$ are the $i-th$ components of $\vec{a}$ and $\vec{b}$ with $i = 1, ..., l$. The distances used in this work are:

- Euclidean distance. This a particular case of the weighted metric distance and is defined as:

$$dist_{euclidean}(\vec{a}, \vec{b}) = \sqrt{\sum_{i=1}^{l}(a_i - b_i)^2} \tag{4.3}$$

- Cosine distance. Departing from a similitude metric, which is defined as the scalar product between two vectors, the distance is defined as:

$$\begin{aligned} dist_{cosine}(\vec{a}, \vec{b}) &= 1 - sim_{cosine}(\vec{a}, \vec{b}) \\ sim_{cosine}(\vec{a}, \vec{b}) &= \frac{(\vec{a})^T \cdot \vec{b}}{|\vec{a}| \cdot |\vec{b}|} \end{aligned} \tag{4.4}$$

- Correlation distance. Again, departing from a similitude metric, which is defined as a normalized version of the scalar product between two vectors, the distance is defined as:

$$\begin{aligned} dist_{correlation}(\vec{a}, \vec{b}) &= 1 - sim_{correlation}(\vec{a}, \vec{b}) \\ sim_{correlation}(\vec{a}, \vec{b}) &= \frac{(\vec{a} - \overline{a})^T \cdot (\vec{b} - \overline{b})}{\sqrt{(\vec{a} - \overline{a})^T(\vec{a} - \overline{a})} \cdot \sqrt{(\vec{b} - \overline{b})^T(\vec{b} - \overline{b})}} \end{aligned} \tag{4.5}$$

where

$$\overline{a} = \frac{1}{l}\sum_{i=1}^{l} a_i; \qquad \overline{b} = \frac{1}{l}\sum_{i=1}^{l} b_i \tag{4.6}$$

### 4.4.2 Resolution of the Localization Problem Using Holistic Descriptors

Concerning the localization case that considers non-compacted models, the map is composed of a straightforward set of descriptors, that means that this set has not been treated to create a hierarchical map through any clustering process. Once this straightforward map is available, the localization process leads an image retrieval method:

1. The robot captures a new image in an instant $t$ from an unknown position ($im_t$).
2. It calculates the holistic descriptor of the captured image $\vec{d_t}$.
3. The distances between this new descriptor and the set of descriptors contained in the compact model are obtained. The comparison between descriptors is done through one of the distance metrics presented in the subsection 4.4.1.
4. A distance vector $l_t = \{l_{t1}, ..., l_{tN}\}$ is obtained where $l_{tj} = dist\{\vec{d_t}, \vec{d_j}\}$ according to any measure of distance.
5. The current position of the robot is considered to be the position of the nearest neighbor within the map (problem known as Image Retrieval [235]), i.e. the corresponding position of the robot is the position on the map that minimizes the distance $arg\min_j l_{tj}$. Thus, the position of the robot $(x, y)$ in the instant $t$ is estimated.

Concerning the image retrieval issue, this is an inefficient process due to the fact that the maps are usually composed by a huge number of images and the descriptors have a high size, so the computational cost and the space memory required could be a problem. Hence, localization methods based on compacted models may lead more efficient solutions.

As for the localization case that considers compacted models, in this case of study, clustering is used to compact the map. Additionally, indoor environments may present visual aliasing. As explained in section 4.3, clustering methods are proposed to carry out the compression of the visual model. Once the clustering is done, the map $\mathfrak{M}$ will be formed only by a set of clusters $C = \{C_1, ..., C_{n_c}\}$, where $n_c$ is the number of clusters. Thus, the compressed map consists of a set of cluster representatives $\{\vec{r_1}, ..., \vec{r_{n_c}}\}$ and the coordinates of each representative $\{(x, y)_{r_1}, ..., (x, y)_{r_{n_c}}\}$ .

The localization in this map is carried out as follows. (1) The robot captures a new image $im_t$ from an unknown position $(x_t, y_t)$, which must be estimated and (2) the descriptor corresponding to the new captured image is obtained ($\vec{d_t}$) by using any of the description algorithms explained in sec. 4.2 (FS, HOG, *gist* or CNN-based descriptors). (3) The distance vector is obtained $\vec{l_t} = \{l_{t1}, ..., l_{tn_c}\}$ where $l_{tj} = dist\{\vec{d_t}, \vec{r_j}\}$ is the distance (one of the three types explained in subsection 4.4.1) between the descriptor $\vec{d_t}$ and each representative $\vec{r_j}$. Finally, (4) the estimated position of the robot $(x_e, y_e)$ is the position associated to the nearest neighbour $d_t^{nn}|t = arg\min_j l_{tj}$. A block diagram about these steps is shown in fig. 4.5.

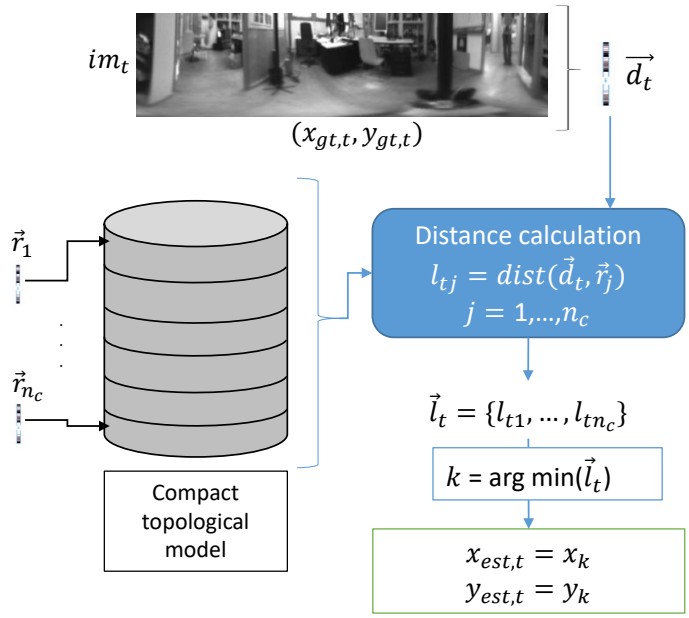



**Figure 4.5:** Block diagram on the steps to perform the localization task using compact models.

### 4.4.3 Hierarchical Localization

In the previous subsection, the localization has been proposed using only the compact map, i.e. only the high-level layer is used, and the result is an coarse localization. In this subsection, we go a step further and the localization is approached hierarchically. That is, first, a coarse localization is performed, as explained in subsection 4.4.2. Once the nearest cluster has been recovered, a second step is carried out to refine the estimation. Hence, the hierarchical localization task consists of the following processes: first, the robot describes the captured image in the instant of time $t$ (test image) $im_t \rightarrow \vec{d}_t$. After that, the distance vector is obtained again $\vec{l}_t = \{l_{t1}, ..., l_{tn_c}\}$. The most likely cluster is then selected as having the minimum value of $\vec{l}_t$. In this step, a new comparison is made between the descriptor of the test image $\vec{d}_t$ and the descriptors of the images that belong to the chosen cluster. From this step, a new distance vector is obtained $\vec{q}_t = \{q_{t1}, ..., q_{tm_i}\}$ where $m_i$ is the number of images within the selected cluster $i$. Last, the minimum value of $\vec{q}_t$ indicates the most similar image and therefore corresponds to the current position of the robot with a higher accuracy. Fig. 4.6 shows the block diagram about these steps. An alternative method has been also considered. This consists in selecting more than one representative as candidate in the rough localization step. This could reduce the error, but it also leads to more comparisons in the fine localization step.

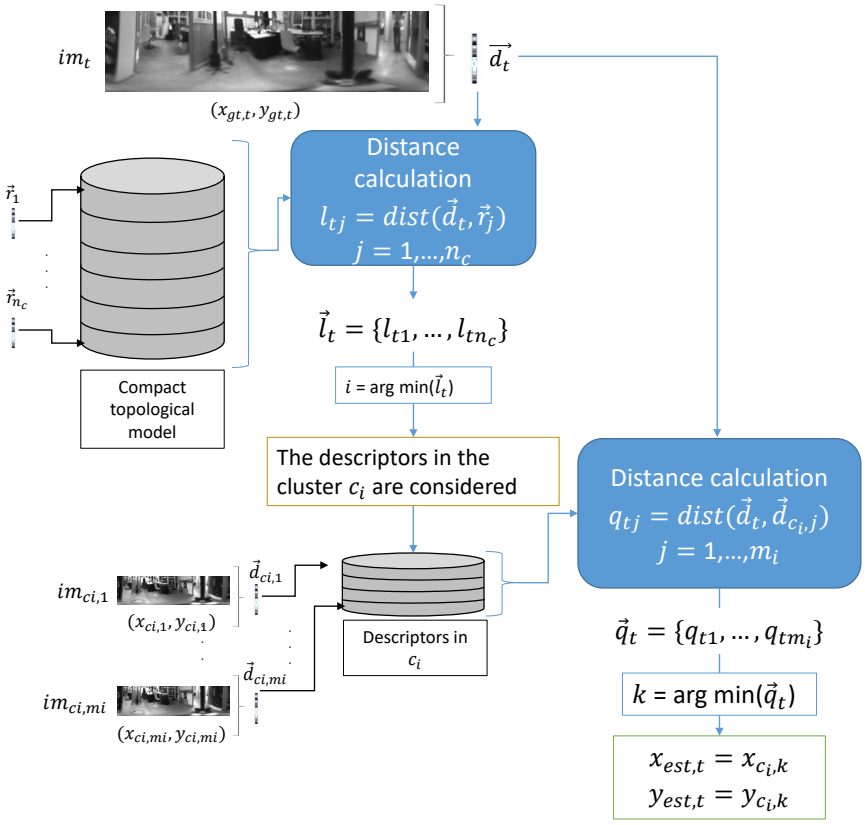



**Figure 4.6:** Block diagram on the steps to perform the task of hierarchical localization using compact models.

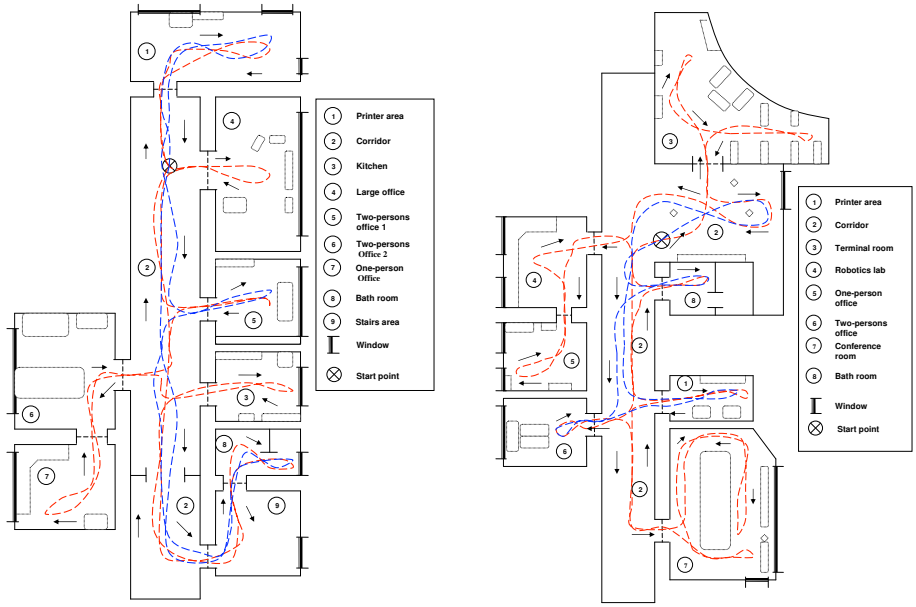
## 4.5 Experiments

This section presents the experiments and results carried out by applying the methods proposed in sections 4.3 and 4.4 to carry out the compaction of the models as well as the localization task by using the model proposed. The algorithms were run on a $2 \times 2.66$ GHz Dual-Core Intel Xeon CPU ® with 10 GB of memory.

### 4.5.1 Datasets

The experiments developed in the present chapter were developed by using two different types of databases; COLD DB, which contains images along a trajectory, and Quorum V DB, which contains grid-distributed visual information. Concerning the COLD (COsy Localization Database) database [222], it contains several sets of images captured in three different indoor environments that are located in three different cities: Freiburg, Saarbrücken (both in Germany) and Ljubliana (Slovenia). This database is

composed of omnidirectional images captured while the robot traversed several paths within the environments. The robot tackle the image capturing task under real operating conditions, that is, people that appear and dissapear from scenes, changes in the furniture, etc. The present work is tested with Freiburg and Saarbrücken. This selection is done because these environments are the two most challenging. Both datasets include several rooms such as corridors, personal offices, printer areas, kitchens, bathrooms, etc. Additionally, with the aim to represent the same distance between images as the distance presented in the *Quorum V* database, a downsampling is carried out to obtain an acquisition distance between images of 40 cm approximately. This distance is considered reasonable for indoor applications. Therefore, after downsampling, two training datasets are generated: $Freiburg_{training}$ and $Saarbrucken_{training}$ with 519 and 566 images respectively. Moreover, the remaining images are stored in a different dataset to use them to test the proposed methods. The fig. 4.7 shows the bird's eye view of the environments and the path that the robot traversed to obtain the images. To summarize, the table 4.1 shows the training and test datasets used in this work and the number of images and rooms that each of them contains.




(a) (b)

**Figure 4.7:** Bird's eye view of the COLD database. (a) Freiburg and (b) Saarbrücken environment. Extracted from https://www.cas.kth.se/COLD/

Regarding Quorum V, it is a publicly available database [11] consisting of a set of omnidirectional images captured in an interior building of the Miguel Hernández

University (Spain). The datasets include 3 offices, a library, a meeting room and a main corridor. This database consists of two sets of data; the first is a training dataset and consists of 872 images that were captured in a dense $40 \times 40$ cm grid of points. As for the second dataset, the test dataset, it consists of 77 images captured in different parts of the environment, in intermediate positions between the points of the training dataset and including changes in the environment (e.g. walking people, position of furniture, etc.). Fig. 4.8 shows the bird's eye view of the *Quorum V* database and grid points captured by the robot for the training dataset.

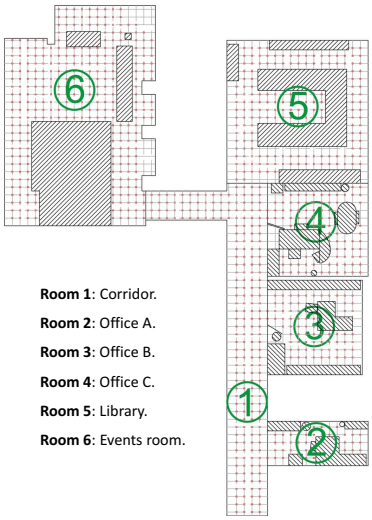



**Figure 4.8:** Bird's eye view of the *Quorum V* database.

**Table 4.1:** Datasets used to carry out the experiments.

| **Dataset name** | **Number of images** | **Number of rooms** |
|---|---|---|
| *QuorumV_training* | 872 | 6 |
| *QuorumV_test* | 77 | |
| *Freiburg_training* | 519 | 9 |
| *Freiburg_test* | 52 | |
| *Saarbrucken_training* | 566 | 8 |
| *Saarbrucken_test* | 57 | |

Additionally to the three proposed environments to evaluate the model compaction and the subsequent localization, the Freiburg and Saarbücken datasets were also used to evaluate the robustness of the localization methods under changes of illumination. The images of these datasets were collected under three different illumination conditions (cloudy days, sunny days and at nights). The images captured during cloudy weather are used to build a compact model through spectral clustering since they are the ones that are less affected by brightness, reflections, dark areas

and thus, they provide more information. The sunny weather images, which are only available for the Freiburg dataset, and also the images captured at night are used as test datasets to evaluate the localization task under lighting changes. These datasets contain also images that do not provide much information due to the acquisition position and blurry images. All these handicaps make these datasets suitable to carry out experiments under real operating conditions. The table 4.2 summarizes the sets created for the experiments related to test the localization methods under changes of illumination conditions. The fig. 4.9 shows some examples of omnidirectional images in both environments under the proposed illumination conditions.

**Table 4.2:** Datasets created from the COLD database to carry out the experiments.

| Dataset name | Illumination condition | Number of images | Path length (m) |
|---|---|---|---|
| Freiburg_training | Cloudy | 519 | 104.2 |
| Freiburg_test_night | Night | 58 | |
| Freiburg_test_sunny | Sunny | 45 | |
| Saarbrücken_training | Cloudy | 566 | 156.6 |
| Saarbrücken_test | Night | 57 | |

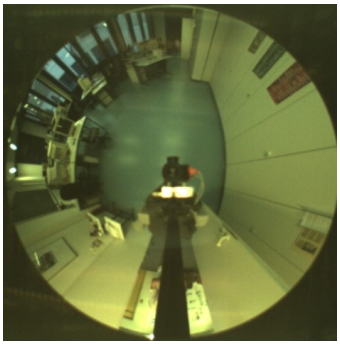
### 4.5.2 Creating Compact Maps through Clustering

This subsection evaluates the clustering methods to compact the information contained in a set of holistic descriptors. The present experiment consists in evaluating two clustering methods under the three environments proposed and considering three global-appearance descriptors. The first method (Method 1) consists on spectral clustering along with k-means as was explained in subsection 4.3.1. Other configurations were tested, such as to use of SOM instead of k-means to solve the step 5 of the spectral clustering, but the results were almost identical. For this reason, only the spectral clustering along with k-means to cluster the normalized matrix of the $n_c$ eigenvectors is shown. The second method (Method 2) consists on the use of SOM, which was explained in subsection 4.3.2.

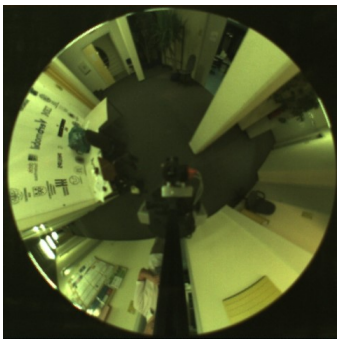
Therefore, for the two proposed methods, several experiments are tackled to study the influence of the parameters of the three global-appearance descriptors. Table 4.3 summarizes the experiments developed.

The values $k_1$, $k_2$ and $k_3$ define the length of each descriptor, but their meaning is not the same (equal values of $k_1$, $k_2$ and $k_3$ would not lead to the same size of descriptor). Hence, as the goal is to study the correct tuning of these values to use each descriptor as efficiently as possible, different values are applied for each descriptor throughout the experiments.

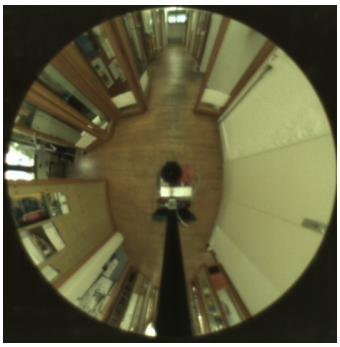
In order to determine whether the compression tackled brings compaction or not, we should establish certain measures that permit quantifying the compactness



(a)

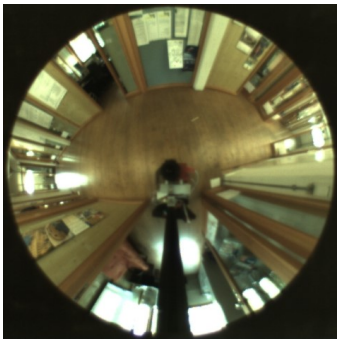


(b)



(c)

(d)



(e)

**Figure 4.9:** Some omnidirectional sample images belonging to the Saarbrücken environment under (a) cloudy and (b) night illumination conditions and also images that belong to the to the Freiburg environment under (a) cloudy, (b) night and (c) sunny illumination conditions.

**Table 4.3:** Summary of the parameters which have been varied to carry out the clustering experiments.

| **Parameter** | **Values** |
|---|---|
| Environment | Quorum V<br>Freiburg (COLD)<br>Saarbrücken(COLD) |
| Descriptor | FS<br>HOG<br>*gist* |
| Descriptor parameters | FS: $k_1$=4,8,16,32,64,128,256<br>HOG:$k_2$=2,4,16,32,64,128<br>*gist*: $k_3$=2,4,8,16,32,64<br>*gist*: $n_{masks}$=2,4,8,16,32,64 |
| Number of clusters | Quorum V: $n_c$=15 25 40 60 80 100<br>Freiburg: $n_c$=10 20 30 40 50 60 70<br>Saarbrücken $n_c$= 10 20 30 40 50 60 70 |

carried out in the map. In this context, the concept of silhouette is widely proposed. The quantification of the goodness of each method is carried out by three parameters:

- The average moment of inertia of the cluster.
- The average silhouette of the points.
- The average silhouette of the descriptors.

These values are collected after clustering the visual data contained in the models. As for the moment of inertia, it measures the compactness of the clusters (if the clusters group images captured from geometrically close points) and is calculated as:

$$M = \sum_{i=1}^{n_c} \frac{\sum_{j=1}^{n_i} dist((x,y)_{r_i},(x_j,y_j))^2}{n_i} \tag{4.7}$$

where $dist((x,y)_{r_i},(x_j,y_j))$ is the Euclidean distance between the coordinates of the representative $\vec{r_i}$ and the position of the j-th image that belongs to the cluster $C_i$ and $n_i$ is the number of images within this cluster.

Silhouette values indicate the degree of similarity between instances within the same cluster and at the same time dissimilarity with instances belonging to other clusters. The silhouette takes values in the range $]-1,1[$ and provides information on how compact the clusters are. Two types of silhouette are used, the average silhouette of points is defined as:

$$S_{points} = \frac{\sum_{w=1}^{N} s_w}{N} \tag{4.8}$$

$N$ is the number of instances (images) and $s_w$ is the silhouette of each instance and is calculated as:

$$s_w = \frac{b_w - a_w}{max(a_w, b_w)} \tag{4.9}$$

where $a_w$ is the average distance between the capture point of instance $\vec{d}_w$ and the capture points of the other instances in the same cluster, and $b_w$ is the minimum average distance between the point instance capture point $\vec{d}_w$ and the instance capture point on the other clusters.

In a different way, the average silhouette of the descriptors is commonly obtained as:

$$S_{descr} = \frac{\sum_{k=1}^{N} s_k}{N} \tag{4.10}$$

where $N$ is the total number of instances and $s_k$ is the silhouette of each instance. This value is calculated as:

$$s_k = \frac{b_k - a_k}{max(a_k, b_k)} \tag{4.11}$$

where $a_k$ is the average distance between the descriptor $\vec{d}_k$ and the descriptor of the rest of entities contained in the same cluster, $b_k$ is the minimum average distance between $\vec{d}_k$ and the instances contained in the other clusters.

The silhouette of the descriptors has traditionally been used to measure the compactness of the clusters. Nonetheless, it does not measure geometric compactness. That is why we introduce the silhouette of points, which can provide more appropriate information as we are interested to know if the clusters have grouped images nearby captured.

#### 4.5.2.1 Clustering in Quorum V Environment

Fig. 4.10 shows the results of the two clustering methods using FS as descriptor depending on the parameter $k_1$. Fig. 4.11 shows the results using HOG depending on the parameter $k_2$. Fig. 4.12 shows the results using *gist* depending on the parameter $k_3$ and with $n_{masks}$=16. These figures present the graphs that determine the goodness of each configuration to carry out the mapping task through clustering. The three figures show the moment of inertia and average silhouettes vs. the number of clusters. In all cases, for comparative purposes, the range of the vertical axis is the same. Furthermore, the fig. 4.13 shows the computing time necessary to cluster the environment through the two clustering methods.

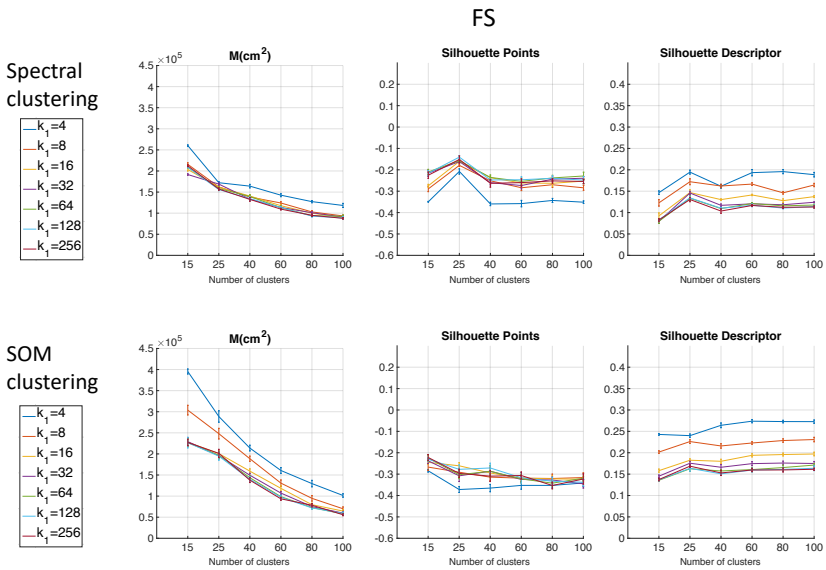



**Figure 4.10:** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when FS is used in the Quorum V environment.

As for the parameters used to measure the compactness of the maps, the smaller the moment of inertia and the higher the silhouettes, the more compact the map. Generally, method 1 (spectral clustering) produces the best results. Method 2 (SOM) does not improve these results. Regarding the use of the holistic descriptor with the spectral clustering method, FS is not able to create reliable clusters. As for HOG, the moment of inertia and silhouettes depend considerably on the value of $k_2$. When $k_2$ is low, the results are poor but when $k_2 > 8$, the moment of inertia as well as the silhouettes improve significantly. Last, concerning the *gist* descriptor, low values of $k_3$ produce low silhouettes and high moments of inertia, and high values of this parameter imply better results.

Concerning the computation time required to carry out the clustering through the two methods, the SOM method requires more time. The computing time for clustering using the FS descriptor is the highest whereas the time for clustering with either HOG or *gist* is lower. As expected, the computing time is directly proportional to the size of the descriptors.

Hence, in the case of HOG, a value of $k_2 = 32$ or $k_2 = 64$ might be a good choice to achieve a compromise between compactness and computation time and in the case of *gist*, an intermediate value of $k_3$ could be also a good choice for the same aim. In general, the FS descriptor performs the worst results: the moment of inertia is higher and the silhouettes are lower. Therefore, the best clustering results

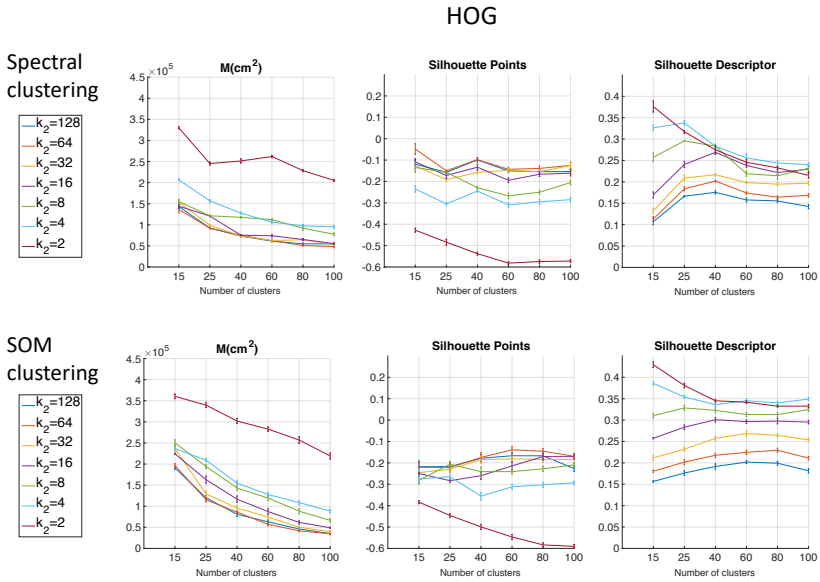



**Figure 4.11:** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when HOG is used in the Quorum V environment.

are obtained through the use of the Spectral clustering method and the use of HOG (for a configuration of $k_2 = [32, 64]$) or *gist* (for a configuration of $k_3 = [16, 32]$ and $n_{masks} = 16$) as holistic descriptor. Fig. 4.14 shows the results performed by using the clusters obtained with spectral clustering and *gist* with $k_3 = 32$ and $n_{masks} = 16$.

#### 4.5.2.2 Clustering in COLD Environments

The results obtained by using the Quorum V dataset have shown that the use of FS for clustering is less suitable. Considering this, only HOG and *gist* descriptors are analysed in the experiments with the COLD environments. Fig. 4.15 shows the results using HOG depending on the parameter $k_2$ in the Freiburg environment. Fig. 4.16 shows the results of the clustering methods using *gist* depending on the parameter $k_3$ and with $n_{masks}$=16 in the Freiburg environment. In the same way, for the Saarbrücken environment, fig. 4.17 shows the results using HOG and fig. 4.18 shows the results with *gist*. Regarding the use of the SOM clustering with HOG, it was not able to solve the clustering task for $k_2 = [4, 16]$ when $n_c > 60$.

Again, spectral clustering is the best method and, in this case, *gist* presents better clustering outcomes. Fig. 4.19 shows two clustering solutions obtained in the COLD environments by using the best configuration reached: spectral clustering + *gist* descriptor. Hence, through the experiments carried out in the environments of the COLD database, a confirmation of the results obtained in *Quorum V* is reached. Therefore, the proposed method is generalizable despite the type of map used (linear

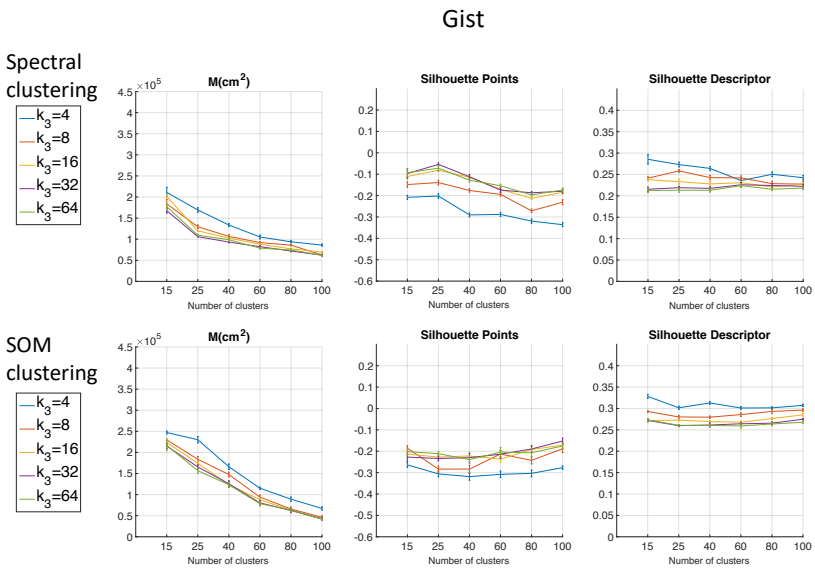



**Figure 4.12:** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when *gist* is used in the Quorum V environment.

or grid). As conclusion, the best option to carry out the compression of visual maps is reached when spectral clustering with *gist* is applied.

### 4.5.3 Localization using the Compact Models

This subsection evaluates the performance of the compact maps to solve the localization problem. The aim is to achieve a compactness that presents a balance between the computing time and the localization accuracy. To perform the evaluation, the Spectral Clustering algorithm is selected with the *gist* descriptor ($k_3 = 32$ and $n_{masks} = 16$). With this configuration, an environmentally compacted map is constructed, using the training images. After that, test images are used to assess the localization problem. The previous subsection showed that the best option for building the compressed map was by using the *gist* descriptor. However, the three proposed holistic descriptors are again proposed to solve the localization task, because mapping and localization are two independent processes and the performance of the descriptors could be different in a localization framework. The localization method proposed to perform this experiment is basically to obtain test images of unknown positions within the map, for each test image is calculated its descriptor (either by FS, HOG or *gist*) and then it is compared with the cluster representatives of the compact map. Thereafter, the most similar cluster is preserved. In this experiment, three distance measurements are considered: (1) the correlation distance, (2) the cosine distance, and (3) the Euclidean distance. As mentioned in previous sections, in order to make a realistic comparison, only visual

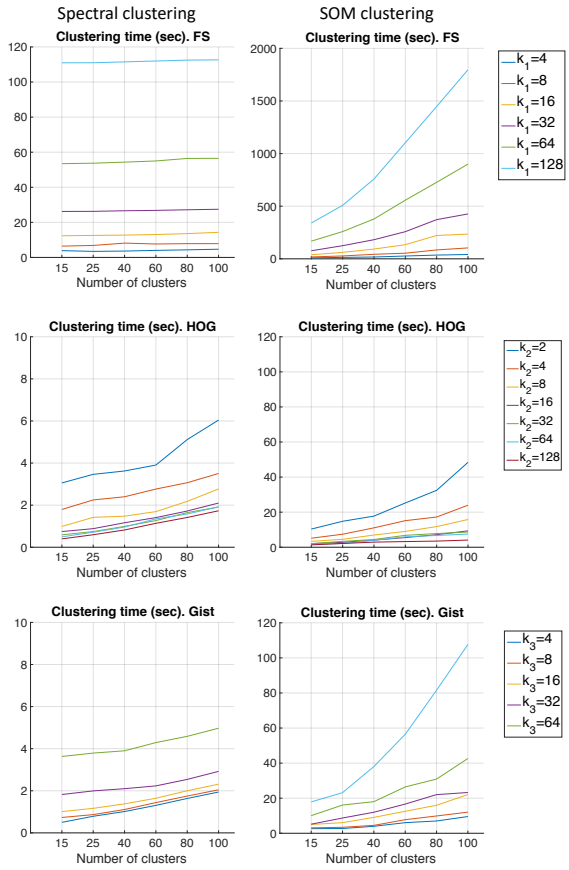



**Figure 4.13:** Results of the two clustering methods: computing time vs. number of clusters, when using FS, HOG and *gist* descriptors in the Quorum V environment.

information will be used to estimate the position of the robot. Metric information will be used only as ground truth, for comparative purposes. As in the clustering experiments, the datasets used to evaluate this experiment are the Quorum V, Freiburg and Saarbrücken, and no illumination variations are considered.

#### 4.5.3.1 Localization in the Quorum V environment

Regarding the localization within grid maps, fig. 4.20 and 4.21 shows the average localization error ($cm$) the computational time (in sec.) obtained respectively when FS (first row), HOG (second row) and *gist* (third row) are used as description method. As for HOG, the effect of homomorphic filtering adds a constant time of 0.02 sec per test image. Concerning the number of clusters, $n_c = 872$ is considered with the aim of providing localization results without compacting the map, i.e. using no compression

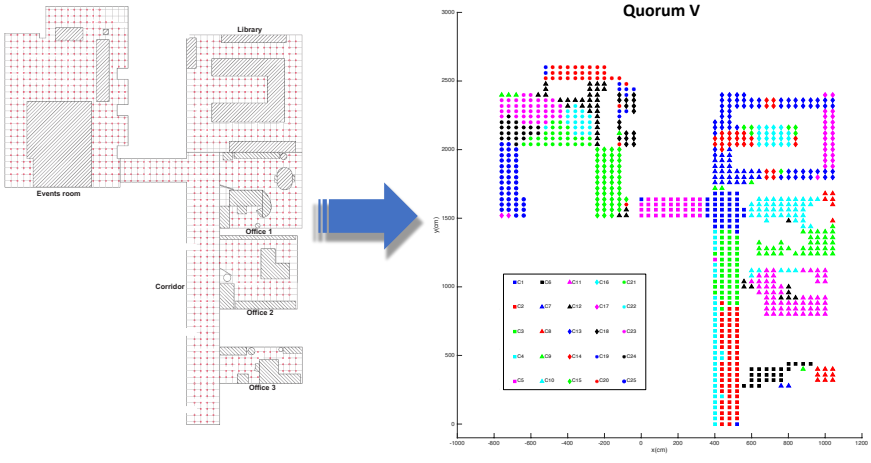



**Figure 4.14:** Quorum V environment. Cluster obtained with Spectral clustering with *gist* descriptor ($k_3 = 32, n_{masks} = 16$).

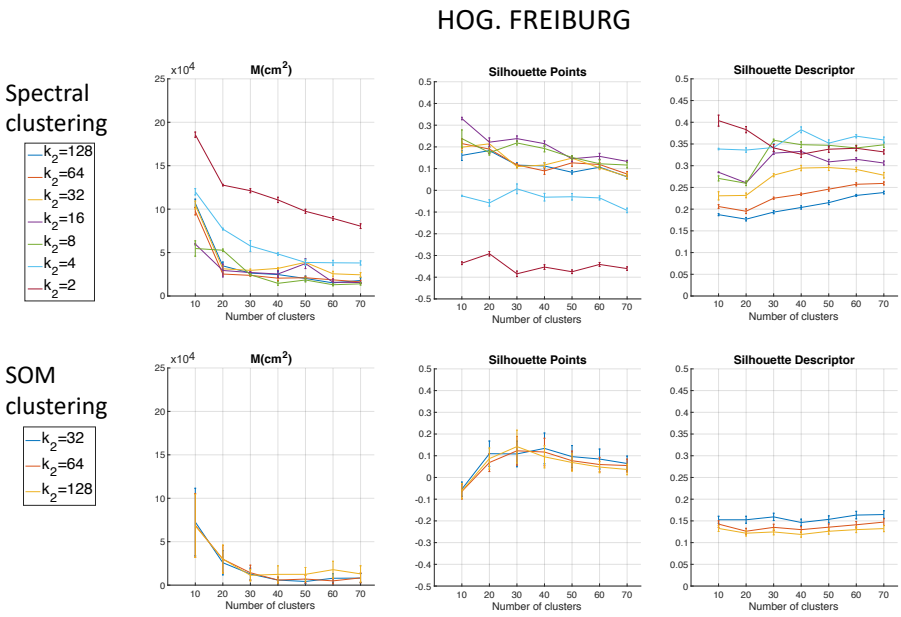



**Figure 4.15:** Results of the two clustering methods: average moment of inertia, average silhouette of points and average silhouette of descriptors vs. number of clusters, when using HOG in the Freiburg environment.

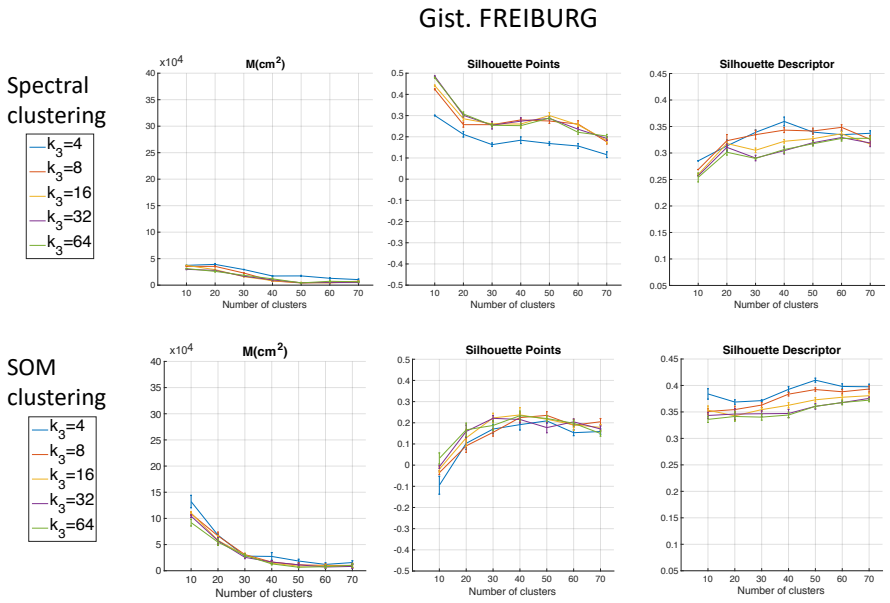



**Figure 4.16:** Results of the two clustering methods: average moment of inertia, average silhouette of points and average silhouette of descriptors vs. number of clusters, when using *gist* in the Freiburg environment.

and thus to know the relative utility of the compacted maps.

The FS descriptor is not good for localization as the best option (correlation distance) has errors between 650 $cm$ and 800 $cm$ depending on the number of clusters and the size of descriptor. HOG improves considerably the localization task with the exception of $k_2 = 2$. The average localization error decreases as the number of clusters increases and these values range from 500 $cm$ when $n_c$ is low and achieve values under 100 $cm$ (when $n_c$ is high). As for the *gist* descriptor, it also produces relatively good results, but not as good as those obtained by using HOG. The localization task achieves the best results when the correlation distance is used.

Concerning the computation time, with the FS descriptor, as the number of clusters increases, the computation time required for the localization task increases substantially. With HOG, the time is much lower than FS and remains constant regardless of the number of clusters. This means that the time to calculate the descriptor is greater than the time to compare it to the map. The calculation time required by *gist* is also worse than the required for HOG. The time required for *gist* is approximately twice the time using the HOG descriptor.

Overall, as the number of clusters increases, the calculation time required for the localization task also increases and the average localization error decreases. This

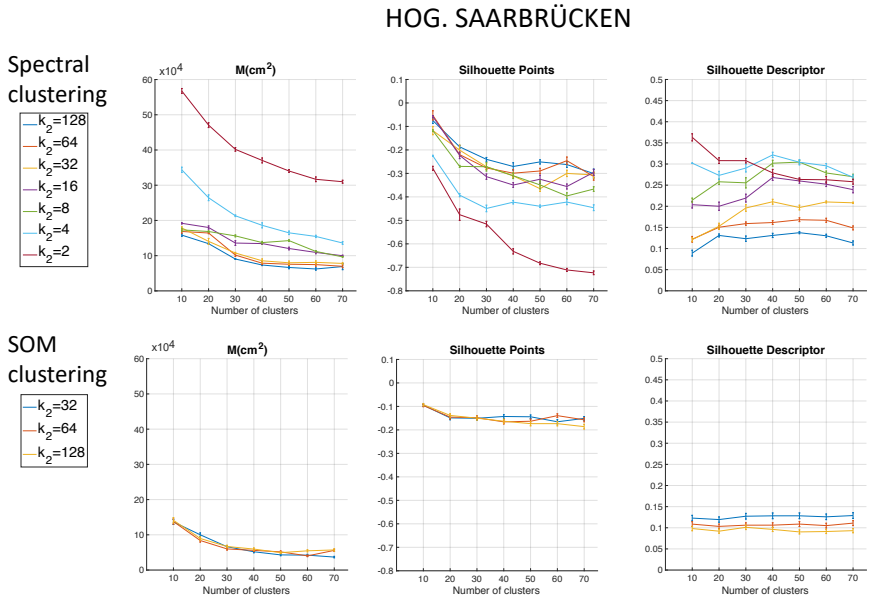



**Figure 4.17:** Results of the two clustering methods: average moment of inertia, average silhouette of points and average silhouette of descriptors vs. number of clusters, when using HOG in the Saarbrücken environment.

is an expected behaviour because a high number of clusters means that the map is less compact, thus the information stays in representatives of the clusters whose distance to the test image is lower. Therefore, the more clusters, the more comparisons with representatives have to be made. This leads to a longer computation time and a more accurate localization error. Hence, a balance must be achieved between these behaviours. Therefore, to address the localization in an environment whose properties are similar to the *Quorum V* environment (distributed grid data), the optimal values are obtained by using the HOG descriptor with $k_2 = [32, 64]$ and the correlation distance.

### 4.5.3.2 Localization in the Freiburg environment

With the aim of confirming the results obtained in Quorum V, the localization task is evaluated under the Freiburg environments. Freiburg is chosen among the three available databases in COLD because this environment presents more rooms and also is more challenging since the building presents many glass walls. In addition, FS descriptor is discarded in these localization experiments, since the FS descriptor has presented the worst results (see fig. 4.20). Euclidean distance results are omitted in this section because it presented the worst outcomes. Fig. 4.22 shows the average localization error ($cm$) obtained when HOG (first row) and *gist* (second row) are used respectively as descriptor. No compaction is also considered (in this case, $n_c = 519$).

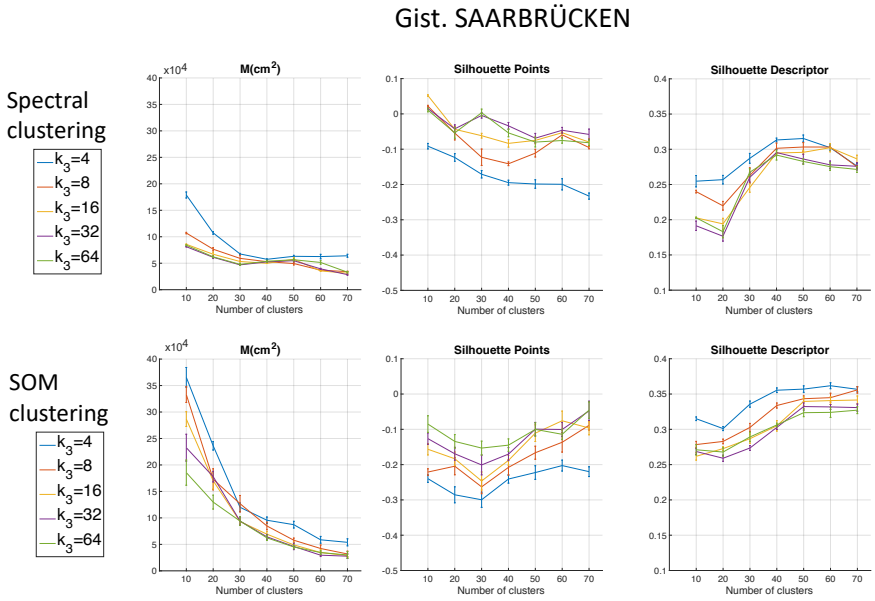


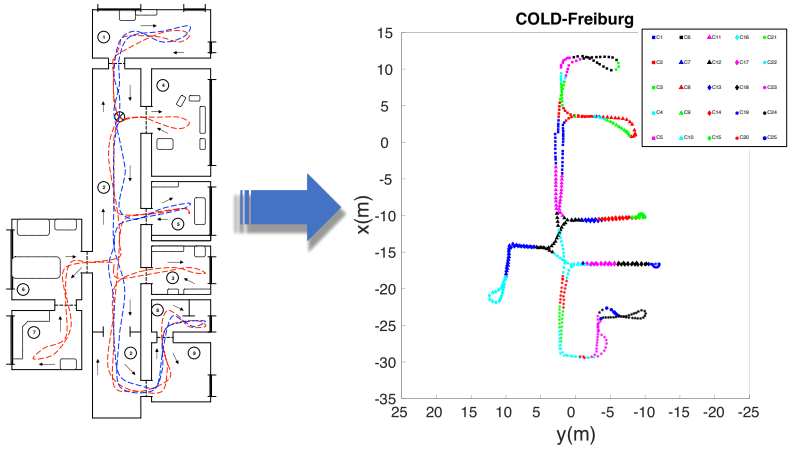

**Figure 4.18:** Results of the two clustering methods: average moment of inertia, average silhouette of points and average silhouette of descriptors vs. number of clusters, when using *gist* in the Saarbrücken environment.

In this experiment, some differences are noticed between the results collected in the *Quorum V* environment and the results in the Freiburg environment. When the number of clusters is low ($n_c = [15, 25, 40]$), the localization task presents a lower average localization error with *gist*. If this number is higher than 40, the localization error is very similar for HOG and *gist*. Comparing the results obtained with the two evaluated types of distances, no remarkable differences are found. Nonetheless, a slightly improvement can be noticed when the cosine distance is used. For example, the average error value when $n_c = 40$ in HOG is lower with cosine than with correlation.

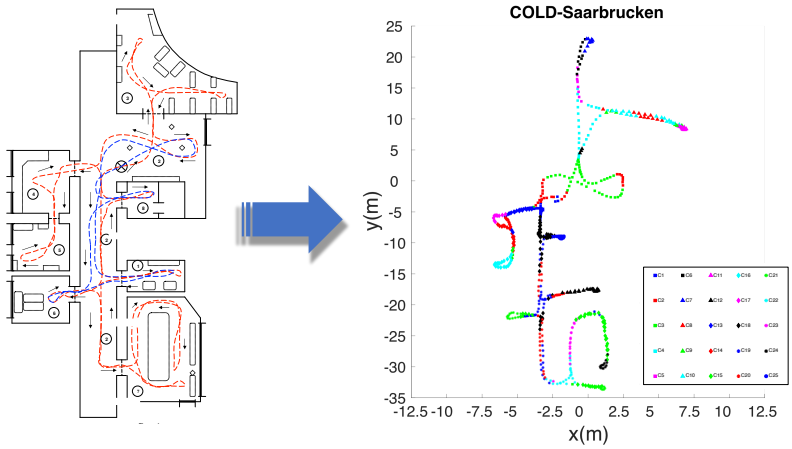
In addition, the value of $k_2$ in HOG is very important. The average error varies significantly according to it. Hence, to solve the localization in an environment whose properties are similar to Freiburg (information along a trajectory), the optimal values are obtained by using the HOG descriptor with $k_2 = [16, 32]$ and cosine distance.

#### 4.5.3.3 Localization when multiple maps are available

In some applications, several maps of some different environments are initially available. If the robot does not have information about the environment in which it is currently located, it must first use the visual information to select the correct environment. After that, the localization can be addressedd in the selected environment, as presented in section 4.4. With this in mind, this section studies the ability to select the appropriate




(a)




(b)

**Figure 4.19:** Clusters obtained in the COLD environments through the use of Spectral clustering and *gist* description. (a) Freiburg and (b) Saarbrücken environment.

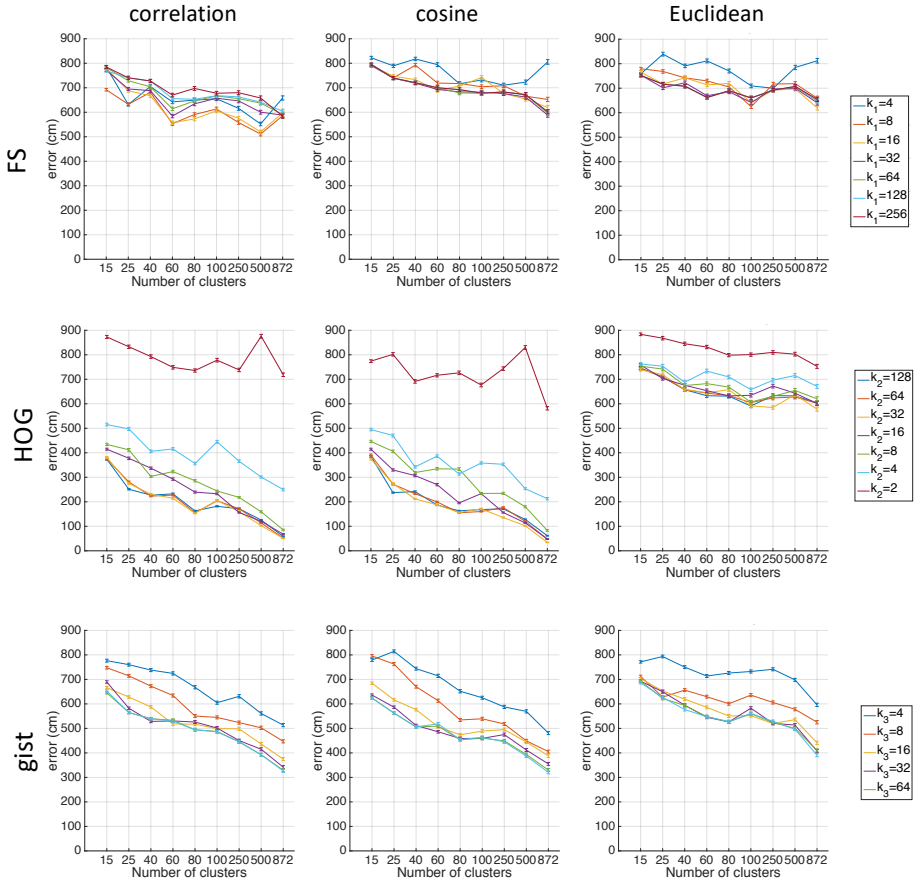



**Figure 4.20:** Results of the localization process in the Quorum V environment. FS, HOG and *gist* are used to describe the representatives of the clusters and the test images: average localization error (cm) vs. number of clusters.

environment. To test the goodness of the descriptors for this purpose, the two COLD maps built in section 4.5.2.2 are considered. In addition, a test dataset is created as a combination of images from the Freiburg and Saarbrücken environments. A total of 60 test images make up the test dataset (34 from Freiburg and 26 from Saarbrücken). In this experiment, only HOG and *gist* are tested again.In addition, since the cosine distance presents the best solutions for COLD, only this type of distance is proposed. Fig. 4.23 shows the success rate in selecting the appropriate environment for both descriptors.

In general, the correct selection of the environment is almost always made. There are many cases where 100% of success is achieved whereas the worst cases do not have a success rate lower than 75%. If the environment selection is performed

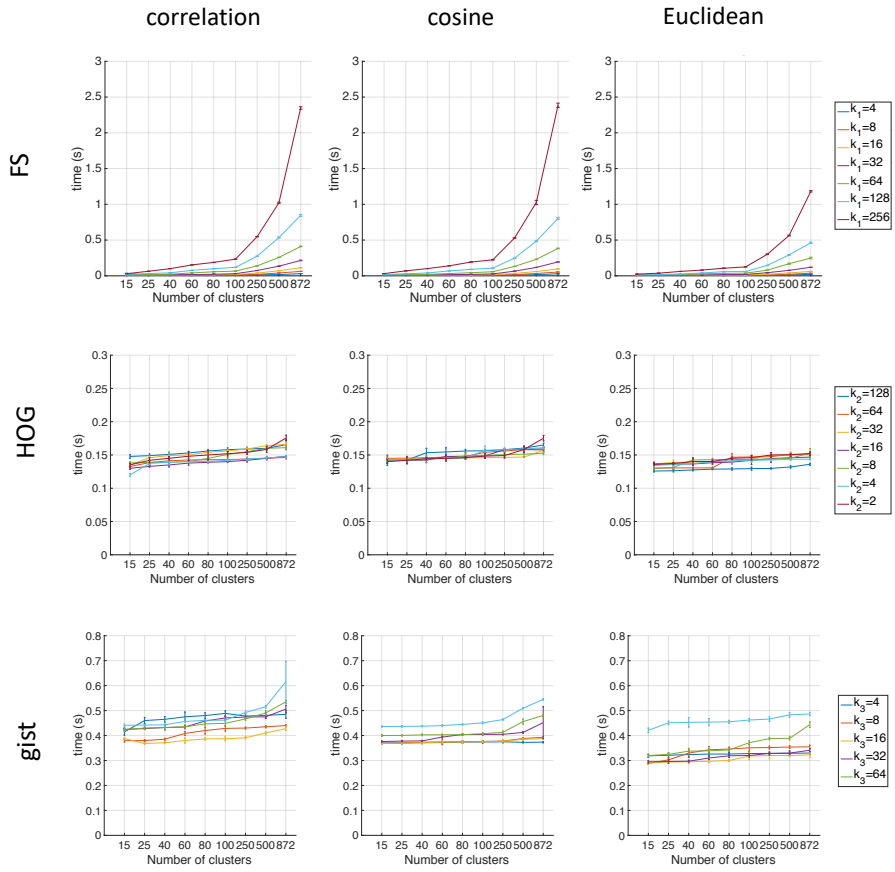



**Figure 4.21:** Results of the localization process in the Quorum V environment. FS, HOG and *gist* are used to describe the representatives of the clusters and the test images: average computing time vs. number of clusters.

with HOG, the results depend substantially on the chosen $k_2$ value. For example, the worst cases are presented for $k_2 = 2, 4$. Nonetheless, for $k_2 = 32 - 128$, a 100% of success is achieved. Using the descriptor *gist*, a 100% of success is given regardless of the number of clusters or the $k_3$ value.

#### 4.5.3.4 A comparative study of localization with straightforward and with compact maps

Compact maps obtained after clustering present an effective solution for performing the localization task on a high-level map as shown in the previous experiments. This process requires capturing a large number of images of the environment for mapping, prior to the clustering process. At this point, we could ask the following question:

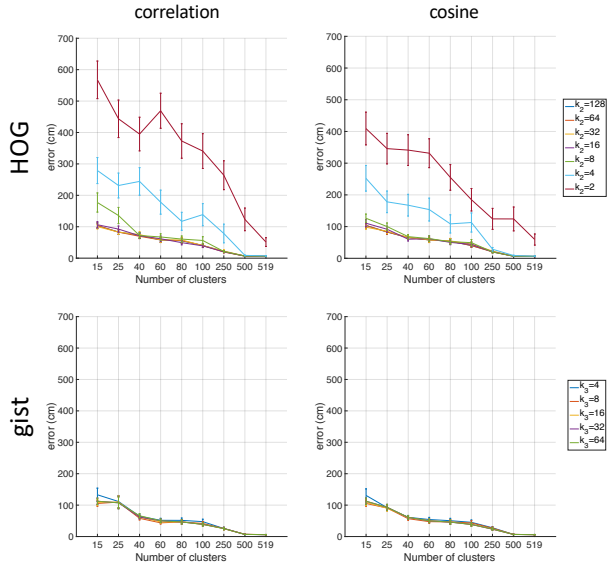



**Figure 4.22:** Results of the localization process in the Freiburg environment. HOG and *gist* are used to describe the representatives of the clusters and the test images: average localization error (cm) vs. number of clusters.

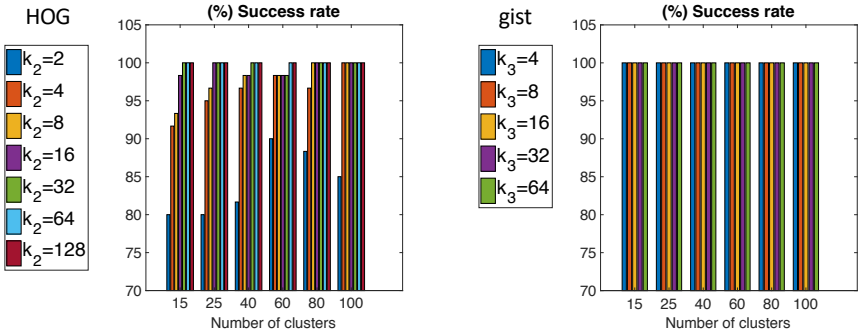



**Figure 4.23:** Percentage of success to detect the correct environment between Freiburg and Saarbrücken with FS, HOG and *gist* used to describe the representatives of the clusters and the test images: percentage of success vs. number of clusters.

should we capture this large number of images, or could we create a compact model directly, capturing only a limited number of images from the environment? This topic is studied in this section. Two types of models are considered: (a) a compact model obtained after clustering a large number of images and (b) a simple model, obtained simply by capturing a limited number of views of the environment. Both types of

models will be used to address the high-level localization task. The simple method we propose to retain representatives is tackled by downsampling the databases, that is, COLD databases are downsampled and only a certain number of images are retained. The utility of this simple method will be compared with the utility of the best compact model obtained in sec. 4.5.2.2 with spectral clustering.

Hence, two models are used as starting points to perform the localization task: (model 1) based on the representative instances obtained using the spectral clustering algorithm and (model 2) based on the instances obtained using database downsampling. After that, the localization task is studied in the Freiburg environment in the same way as it was done in subsection 4.5.3.2.

Fig. 4.24 shows a comparison of the usefulness of the two models in localization tasks. The cosine distance is selected to show these results, because this distance presented good results in previous localization experiments. The two best holistic descriptors for localization (HOG and *gist*) are shown. As you can see, the localization error gets worse when using the straightforward map. When the number of clusters is low, the model obtained using the spectral clustering presents the best localization results. For example, regardless of the descriptor, the average localization error is less than 100 cm when $n_c > 20$ for the model 1 and $n_c > 40$ for the model 2. The average localization error is smaller for model 2 only when the number of clusters is substantially high, $n_c > 80$ (HOG case) and $n_c > 70$ (*gist* case). This result means that the proposed alternative to spectral clustering can only be interesting when low compactness is required. Nontheless, if the number of clusters is low (high compactness), the spectral clustering provides better results. In conclusion, this experiment has shown that the use of simple methods to retain visual representatives is less efficient than the use of spectral clustering methods. Spectral clustering can create compact models that provide accurate localization results.

### 4.5.4 Localization under Changes of Illumination

This subsection evaluates the performance of the compact maps to solve the localization problem under changes of illumination. The objective is to test the robustness of the methods proposed when the illumination conditions during the localization task vary from the conditions given during the mapping task. As in previous localization experiments, the compressed map is obtained by means of spectral clustering with the holistic *gist* descriptor ($k_3 = 32$ and $n_{masks} = 16$). To carry out this experiment, the COLD environments (Freiburg and Saarbrücken) are used. As explained in 4.5.1, the images captured during cloudy illumination conditions (that are less affected by brightness, reflections and dark areas) are used to build a compact model through spectral clustering. The datasets with images captured during sunny days and at night are used as test images. Additionally, apart from the FS, HOG and *gist* descriptors, the two global-appearance descriptors based on deep learning (see subsec. 4.2.4) are also proposed. The same as previous experiments, the correlation distance, the cosine distance and the Euclidean distance are chosen to calculate the distance between descriptors.

Fig. 4.25 shows the average localization error vs. the number of clusters $n_c$ obtained in the Freiburg environment when the test dataset was night and fig. 4.26

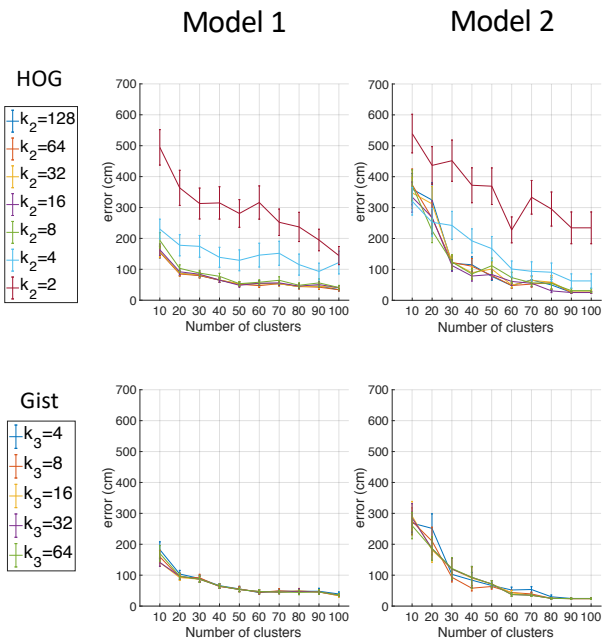



**Figure 4.24:** Results of the localization process in Freiburg by using two types of compact models. Average localization error (cm) versus number of clusters. Model 1 uses representatives obtained by spectral clustering, Model 2 obtains representatives by sampling the dataset.

.

when the test dataset was sunny; fig. 4.27 shows the average localization error obtained in the Saarbrücken environment when the test dataset was night. In all cases, the localization error is expressed in cm, and the colorbars that expresses this error has the same range, to facilitate a comparative evaluation between figures.

Focusing on the effects noticed by the changes of illumination conditions, if we compare the outputs obtained under night conditions and the ones obtained under sunny conditions (see fig. 4.25 and fig. 4.26 ), generally, sunny conditions have a more negative impact upon the localization. For example, when using the CNN descriptor layer *'fc7'*, if $n_c = 10$, the error under night conditions is over 200 cm whereas under sunny conditions, it is over 300 cm. If $n_c = 60$, the error under night conditions is under 100 cm and under sunny conditions, it is over 200 cm.

Among the four evaluated global-appearance descriptors, FS is the one which presents worst localization results generally. Concerning HOG, this descriptor presents relatively good localization error results. For instance, for night conditions in the Freiburg environment (see fig. 4.25), when a correct tuning of the $k_2$ parameter is tackled and for more than 50 clusters, the localization error values are under 100 cm. This outcome is acceptable considering the size of the environment (table 4.1) and the granularity of the compact map. In addition, the optimal results are presented

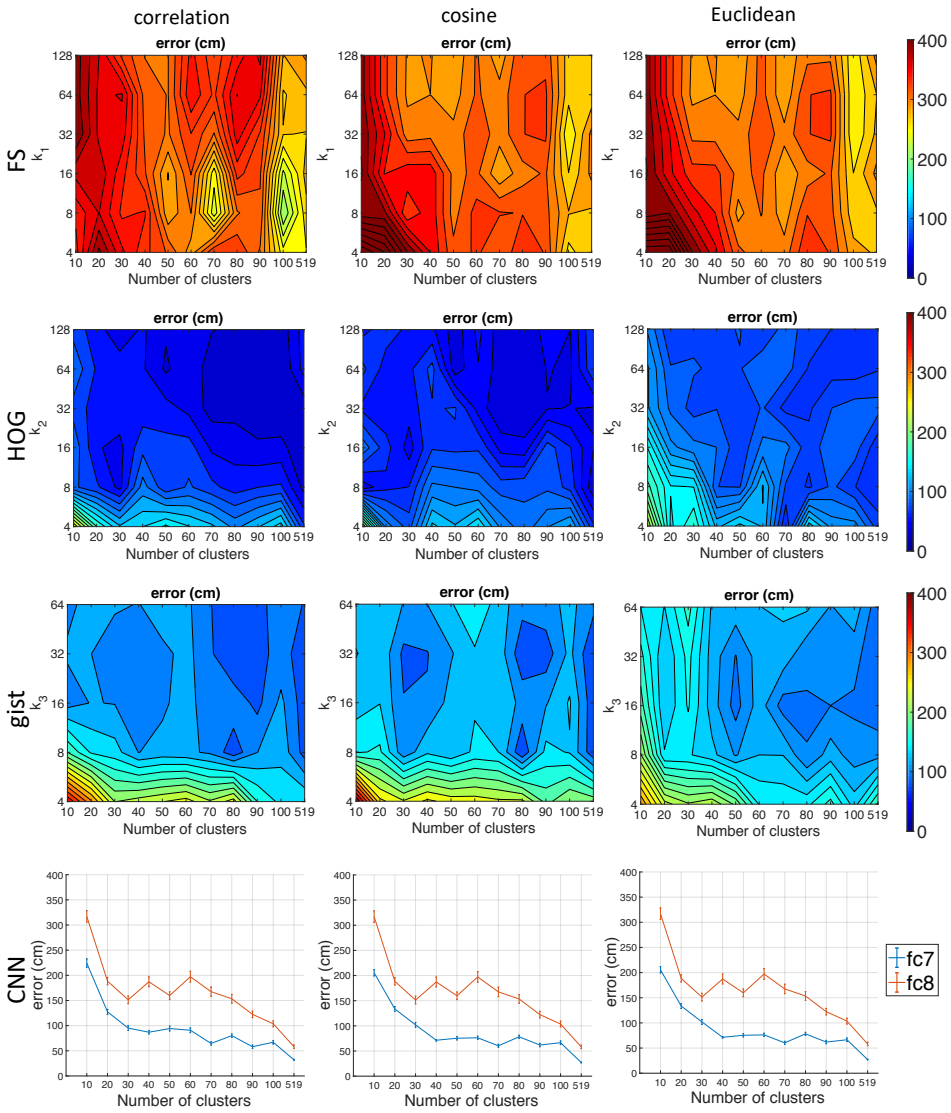



**Figure 4.25:** Results of the localization task when the night illumination conditions affect the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

by using the cosine distance to calculate the distance between descriptors. As for the Saarbrücken environment, HOG presents slightly worse results than in Freiburg (fig. 4.27). *Gist* presents also relatively good localization results and its influence by

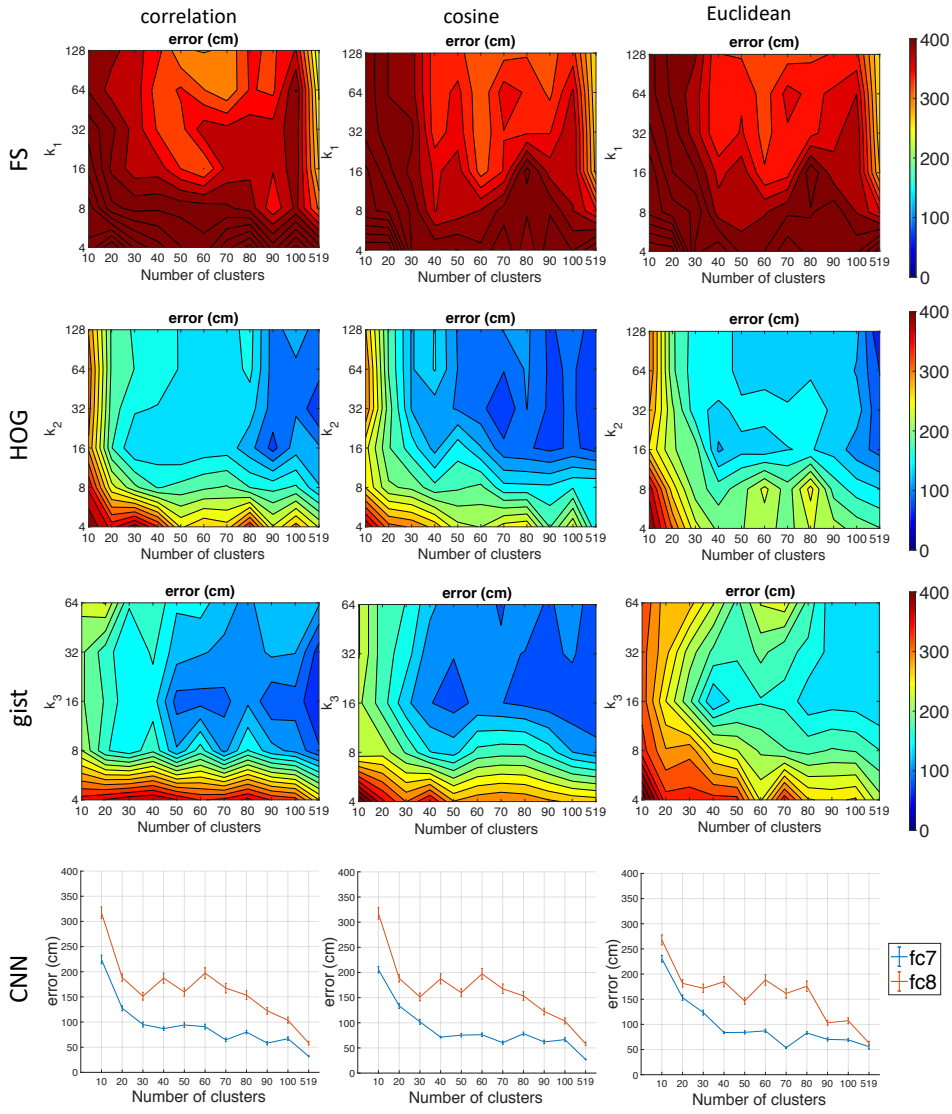



**Figure 4.26:** Results of the localization task when the sunny illumination conditions affect the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

the $k_3$ parameter (number of horizontal blocks) is lower than the HOG influence by $k_2$. For example, in the results presented using the *gist* descriptor (see fig. 4.25), the error decreases until the number of clusters is 40 and after that value, the average localization

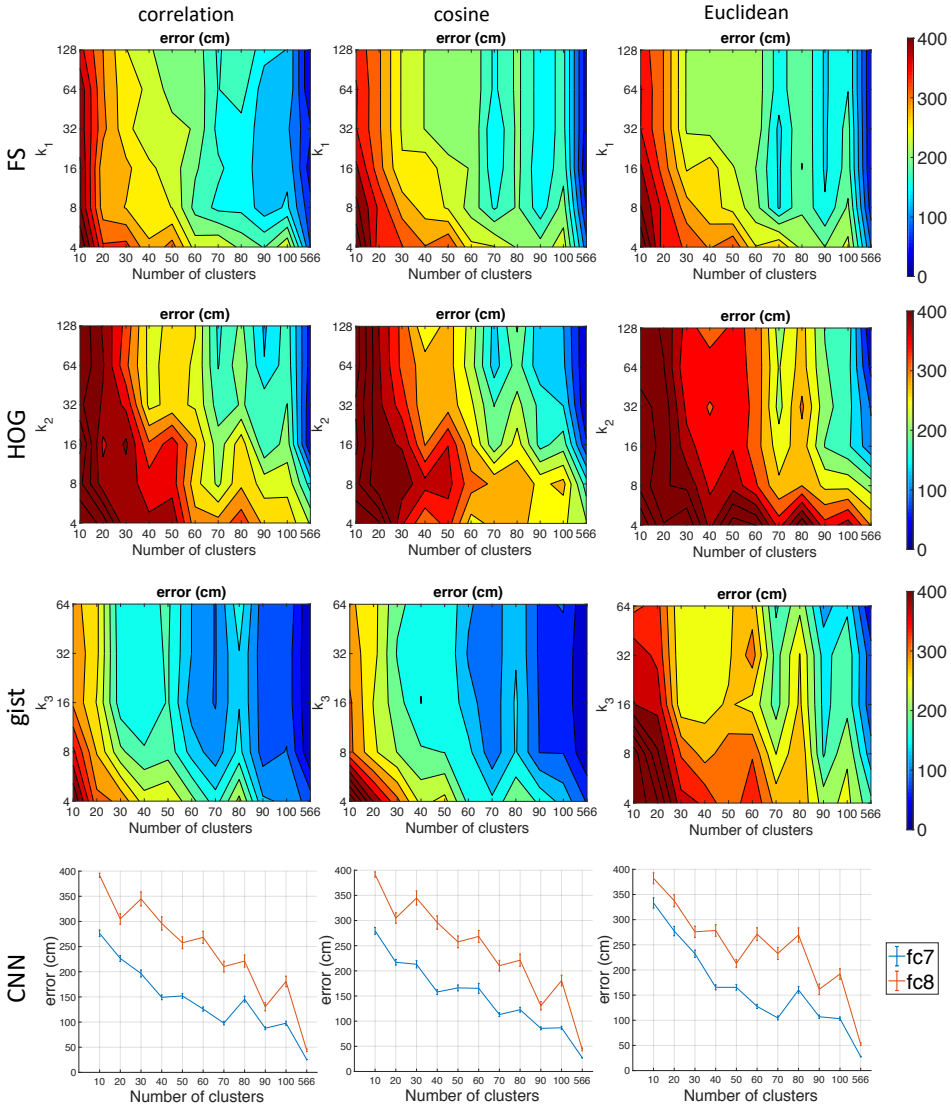



**Figure 4.27:** Results of the localization task when the night illumination conditions affect the Saarbrücken environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

error keeps almost constant. For the *gist* descriptor, the Euclidean distance presents the worst results whereas the cosine and correlation distances are quite similar. The CNN descriptor presents localization results as good as using HOG in Freiburg at night .Using the holistic descriptor obtained from the layer $fc_7$, the localization error is lower than 100 cm when $n_c > 30$ (using either correlation or cosine distance). Additionally,

the results obtained in the Saarbrücken environment with this descriptor are the best. However, CNN is more affected than HOG under sunny conditions (see fig. 4.26).

Between the two best descriptors with the best localization results (HOG and CNN), an evaluation of the computation time is obtained. To this end, the time required to calculate the global-appearance descriptors is performed. Table 4.4 shows the average computational time (sec) to compute the global-appearance descriptor for the *Freiburg_test_night* dataset. Regarding HOG, the values obtained remain almost constant regardless of the value of $k_2$. As for the use of the CNN descriptor, the related time values are higher (about 0.4 seconds).

**Table 4.4:** Computation time (sec) required to obtain the global-appearance descriptor (HOG and CNN) per each test image. Freiburg test dataset under night conditions.

| **Descriptor** | | **Time (ms)** |
|---|---|---|
| **HOG** | k2=2 | $131.0 \pm 0.58$ |
| | k2=4 | $145.8 \pm 0.34$ |
| | k2=8 | $148.3 \pm 0.12$ |
| | k2=16 | $158.1 \pm 0.19$ |
| | k2=32 | $177.8 \pm 0.93$ |
| | k2=64 | $192.3 \pm 0.41$ |
| **CNN** | 'fc7' | $444.7 \pm 5.62$ |
| | 'fc8' | $453.3 \pm 4.51$ |

In conclusion, FS and *gist* localization values are relatively worse. HOG presents better results in the Freiburg environment, but in the Saarbrücken environment, results for HOG are poor, whereas CNN keeps being also good. Despite the computing time is not as low as the HOG one, it is not substantially higher than the HOG results. Hence, among the different holistic descriptors studied to solve the localization task in environments under changes of illumination, CNN performs the optimal solution. As for the illumination changes, HOG is less affected by the sunny conditions than the rest of descriptors. Regarding which type of distance measure is better to calculate the distance between descriptors, both correlation and cosine present similar outputs.

### 4.5.5 Localization by Using Hierarchical Models

In this subsection, the hierarchical localization is evaluated in the Freiburg and Saarbrücken environments under two illumination conditions (night and sunny) calculating two types of distances (correlation and cosine) and using two kind of descriptors (HOG and CNN-based). The Euclidean distance is discarded since it presented the worst localization error results in the experiment 4.5.4. This experiment does not show FS nor *gist* descriptors because, as shown in subsection 4.5.4, these results are worse for localization purposes. This subsection also evaluates the pre-selection of more than one cluster in the rough localization step.

Fig. 4.28 shows the average localization error (cm) vs. the number of clusters $n_c$ obtained in the Freiburg environment when the test dataset was night and either

one or two clusters are selected to carry out the fine localization. Fig. 4.29 shows the average localization error (cm) obtained in the Freiburg environment results when the test dataset was sunny and one cluster is selected for fine localization; fig. 4.30 shows the average localization error (cm) vs. the number of clusters $n_c$ obtained in the Saarbrücken environment when the test dataset was night and one and two clusters are selected to carry out the fine localization.

This method produces an efficient process for refining the localization with the use of the low-level layer. However, this method only improves when the number of clusters is low and no substantial differences are observed when the number of clusters is relatively high. To select more than one cluster to perform fine localization is only interesting when a large compression is performed, otherwise selecting only one cluster produces more efficient results because its computation time is relatively low.

## 4.6 Conclusion

In this chapter, we present a deep study about the use of hierarchical models to carry out the localization by using omnidirectional visual information in indoor environments. With this aim, three datasets from indoor environments are used. These datasets are composed by either panoramic images or omnidirectional images which are transformed to panoramic. Our main contributions and the conclusions reached in this chapter are summarized as follows.

- The work proposes two different **methods for compacting topological maps**. Throughout the experiments, in order to compact the information, the number of instances has been reduced to a value in the range of 10 to 100. That means a reduction of instances up to between 1.1% and 11.5% of the original number. The proposed methods are (1) spectral clustering and (2) Self-Organized Maps. In addition, three holistic descriptors are used, since they present a good solution for environments where data dimensionality is high.

  The work shows that it is possible to reduce drastically the visual information from the original model. Regarding the use of methods to compress visual models, spectral clustering has proved to be, in general, more efficient than the SOM clustering. Also, the global-appearance descriptor which presents better behaviour to carry out the clustering task is *gist*. Concerning the localization task using the compact models, HOG presents generally the best outcomes independently on the type of map. The best results concerning mapping and localization are summarized in the fig. 4.31.

- After compressing the original model, the resulting map can be used to solve the. Therefore, an evaluation is tackled with the aim of measuring the goodness of the localization task by using compact models and holistic descriptors. In this case, three descriptors (FS, HOG and *gist*) are evaluated.

  As for the computing time, comparing the localization results obtained after compaction and by using raw, uncompacted models ($n_c = 872$, $n_c = 519$,

and $n_c = 566$ respectively for Quorum V, Freiburg and Saarbrücken), compact models have proven to be an effective tool for reducing computation time and maintaining localization accuracy (see sec. 4.4).

- As for the detection of the environment and a subsequent localization, a mixture between indoor environments is created in order to evaluate if it is possible, first, to **detect the right environment** and second, to estimate the position of the instance. From this experiment, HOG is the description method that presents the best localization performance. In addition, *gist* presents the most successful results for selecting the correct environment of a test instance from a combined dataset. Using this descriptor, $100\%$ of success is achieved regardless of the number of clusters and the value $k_3$.

- Concerning the **utility of clustering methods**, a comparison was carried out between clustering models and straightforward downsampling methods. The conclusion reached is that the use of clustering methods to tackle the compression step has proved to be more efficient than carrying out directly a downsampling of the images from the database, despite straightforward methods might be faster and easier. This is due to the fact that straightforward methods to compress the information are not capable of keeping more information about the environment than the proposed spectral clustering method.

- Regarding the use of holistic descriptors to solve **localization under changes of illumination**, an experiment was tackled to test different holistic descriptors against changes of illumination to solve the localization by using compact maps. The localization task is tackled once the compact model of the environment is created (using spectral clustering with *gist* descriptor). In this case, apart from the three previously holistic description methods proposed, we also use holistic descriptors obtained from intermediate layers of a CNN. In this regard, FS outputs again the worst results and HOG usually leads to the best solutions. The use of CNN-based descriptors has presented good results. Nevertheless, CNN is more affected by the sunny illumination conditions than HOG is and CNN also needs more computing time to calculate the descriptor than HOG. Furthermore, in general the sunny illumination conditions affect more negatively the performance of the methods than the night conditions.

- Last, a **hierarchical localization** method has been developed and evaluated. In this sense, the related experiments evaluated the efficiency of this method. Additionally, several configurations and input information (holistic descriptors) were also tested. As conclusion, this localization method may result interesting when high levels of compression are presented. Moreover, just selecting one cluster as candidate may be enough for the majority of cases. The table 4.5 summarizes the localization error obtained along the experiments proposed in this chapter without changes of illumination. Through this table, it is easy to conclude that the CNN-based descriptor provides the best results to carry out the localization task for both localization methods although HOG also presents good results when the localization is addressed hierarchically.

**Table 4.5:** Summary of the minimum localization error values obtained through the two localization methods and the four descriptors evaluated throughout this work.

| **Localization Method** | **Descriptor** | **Minimum localization error (cm)** | | | | |
|---|---|---|---|---|---|---|
| | | $n_c = 20$ | $n_c = 40$ | $n_c = 60$ | $n_c = 80$ | $n_c = 100$ |
| Using a compact model | FS | 403.25 | 331.88 | 350.88 | 351.81 | 267.60 |
| | HOG | 149.75 | 128.19 | 118.53 | 112.20 | 90.85 |
| | gist | 201.33 | 125.77 | 149.45 | 125.92 | 153.38 |
| | CNN | 133.54 | 71.39 | 76.23 | 78.27 | 66.51 |
| Hierarchical Localization | FS | 360.73 | 308.40 | 327.13 | 328.32 | 252.17 |
| | HOG | 61.41 | 81.27 | 54.89 | 33.59 | 42.52 |
| | gist | 136.27 | 99.10 | 136.58 | 85.02 | 125.31 |
| | CNN | 69.52 | 38.68 | 50.57 | 49.01 | 50.73 |

## 4.7 Publications Related to this Chapter

The main results presented in this chapter are related to the following publications:

- L. Payá, W. Mayol, S. Cebollada, O. Reinoso. Compression of topological models and localization using the global appearance of visual information. ICRA 2017. IEEE International Conference on Robotics and Automation (Singapore, May 29th - June 3rd 2017). Ed. IEEE ISBN:978-1-5090-4632-4 - pp. 5630-5637 [212]

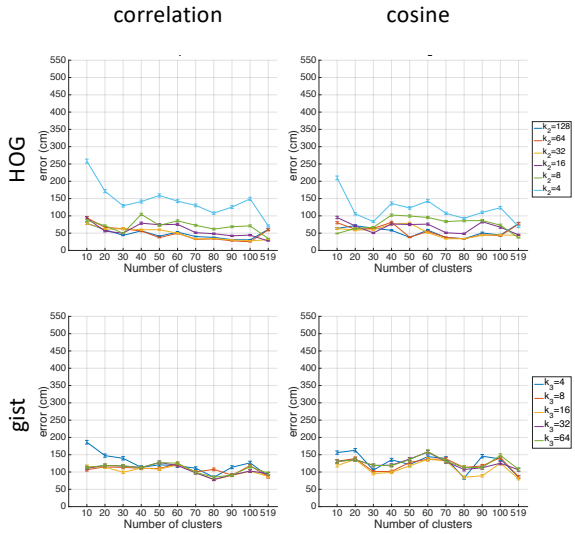
  - This paper presents the development and evaluation of the spectral clustering approach to obtain compact topological models in the Quorum V dataset (grid map). This model is subsequently tested by studying their utility to solve the robot localization problem. Omnidirectional visual information and global-appearance descriptors are used both to create and compress the models and to estimate the position of the robot.

- S. Cebollada, L. Payá, W. Mayol, O. Reinoso. Evaluation of Clustering Methods in Compression of Topological Models and Visual Place Recognition Using Global Appearance Descriptors. Applied Sciences. Ed. MDPI ISSN:2076-3417 Vol 9(3), 377 (2019) [44] **(JCR-SCI Impact Factor: 2.474, Q2)**.

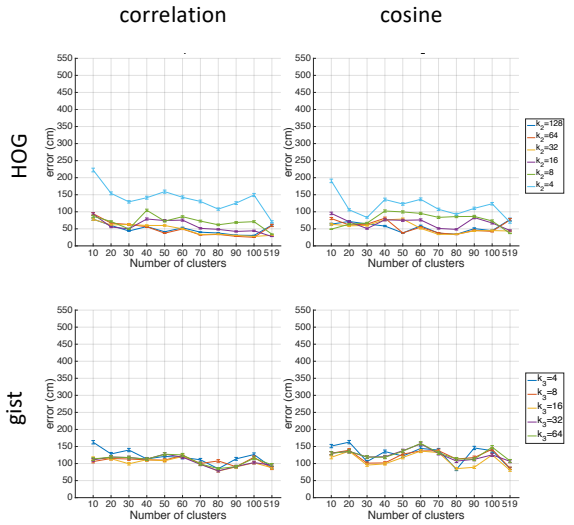
  - This paper presents a profound study about the compression of topological models of indoor environments. Two clustering methods are tested in order to know their utility as well as to build a model of the environment and to solve the localization task. Omnidirectional images are used both to create the compact model and to estimate the robot position within the environment. These images are characterized through global-appearance descriptors. To evaluate the goodness of the proposed clustering algorithms, Quorum V, Freiburg and Saarbrücken datasets are considered.

- Sergio Cebollada, Luis Payá, Vicente Román, Oscar Reinoso. Hierarchical Localization in Topological Models Under Varying Illumination Using Holistic Visual Descriptors. IEEE Access. Ed. IEEE ISSN:2169-3536 Vol 7(1), pp. 49580-49595 (2019) [45] **JCR-SCI Impact Factor: 3.745, Q1**.

  - This paper presents a hierarchical localization framework within indoor environments. To carry out this task severe variations of the illumination conditions are considered. The only source of information both to build a model of the environment and to solve the localization problem is a catadioptric vision system, whose images are processed globally to obtain holistic descriptors.




(a)




(b)

**Figure 4.28:** Results of the complete hierarchical localization task when the night lighting condition affects the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Pre-selection of either (a) one ($c = 1$) or (b) two ($c = 2$) clusters as the most likely options.

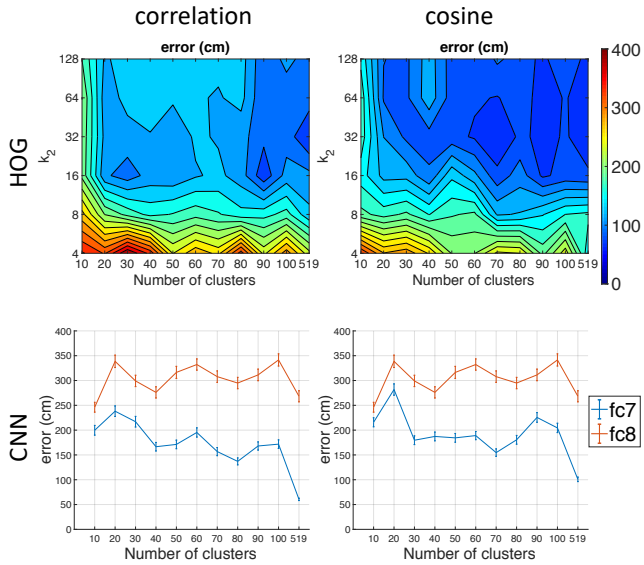
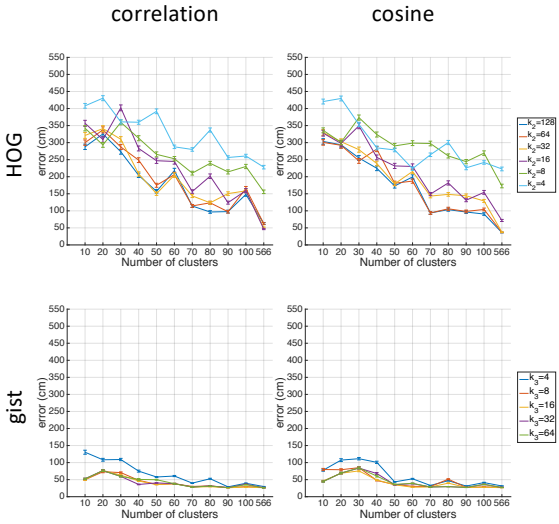
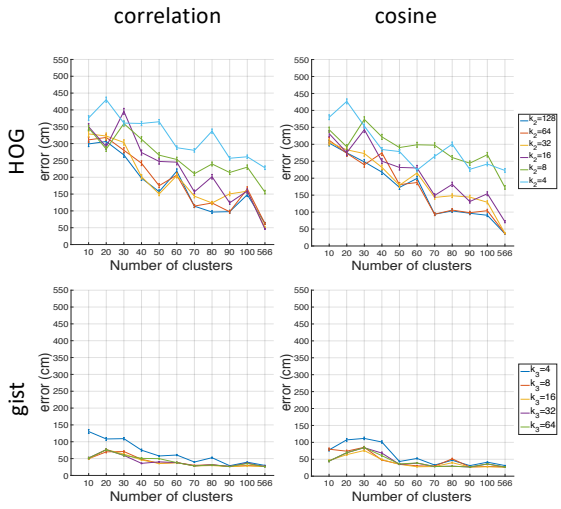



**Figure 4.29:** Results of the complete hierarchical localization task when the sunny lighting condition affects the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Pre-selection of one cluster as the most likely option.

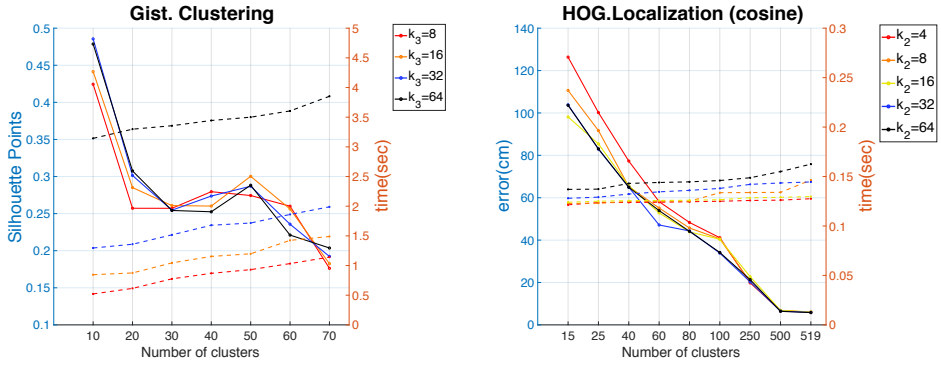



(a)




(b)

**Figure 4.30:** Results of the complete hierarchical localization task when the night lighting condition affects the Saarbrücken environment. Average localization error(cm) vs. number of clusters and descriptor size. Pre-selection of either (a) one ($c = 1$) or (b) two ($c = 2$) clusters as the most likely options.




(a) (b)

**Figure 4.31:** Best results of the clustering and localization processes. (a) Clustering with *gist* and spectral clustering: Silhouette of Points (left axis, solid lines) and computing time (right axis, dashed lines) vs. number of clusters. (b) Localization with HOG and cosine distance: average localization error (cm)(left axis, solid lines) and computing time (right axis, dashed lines) vs. number of clusters. Freiburg environment.

# 5 Use of Machine Learning Techniques for Localization

## 5.1 Introduction

Throughout the last years, many definitions about AI have been proposed. For instance, Minsky [184] defined AI as "the science of making machines do things that would require intelligence if done by men". For Haugeland [100], artificial intelligence is, "the exciting new effort to make computers think (...) in the full and literal sense". Charniak *et al.* [48] define AI as "the study of mental faculties through the use of computational model". More recently, according to Schalkoff [243], AI is "a field of study that seeks to explain and emulate intelligent behaviour in terms of computational process". If we focus on the use of AI to solve robotics tasks, we can express this science as a set of techniques that are applied in computer programming to solve problems whose difficulty requires a certain grade of intelligence.

Machine learning techniques have contributed to solve a variety of problems in mobile robotics during the last years. For example, Gonzalez *et al.* [88] use machine learning to detect different levels of slippage for robotic missions in Mars and Dymczyk *et al.* [70] propose the use of a boosted classifier to classify landmark observations and carry out the localization task in a more robust fashion. The present work proposes the performance of some machine learning techniques for addressing mapping and localization tasks. The efficiency of the proposed tools is tested regarding their ability to estimate the position of the robot within the environment using the information stored in the map. The proposed method uses images obtained by an omnidirectional vision system as a unique source of information. The approach consists in the use of a variety of classifiers, neural networks and clustering algorithms, used in combination with holistic visual descriptors, to solve the localization problem.

As it has been exposed in previous chapters, global-appearance descriptors have been proposed to extract characteristic information from images and use this information to carry out the mapping and localization tasks. The first approaches were commonly known as hand-crafted descriptors, that is, they were based on analytic methods. Some of these methods were presented in chapter 4 and they were successfully used to carry out compression of visual maps as well as its related localization. Nonetheless, during the last few years, some authors have proposed the use of deep learning techniques to create global-appearance descriptors. To cite one example, Xu *et al.* [306] used a CNN-based descriptor to obtain the most probable position within an indoor map through Monte Carlo Localization and also to solve the kidnapping problem. The use of intermediate layers from convolutional neural networks to obtain holistic descriptors has been commonly proposed. Normally, the last fully-connected layers have been used since they get profit from information learned in previous layers and thus they provide a reliable characterization of the input image.

Therefore, through the present chapter, we present novel localization methods with omnidirectional visual information in indoor environments. Holistic descriptors are obtained from the images and they are used to solve the objective tasks. Our main contributions in this chapter are summarized as follows.

1. We present a study about the use of classifiers to solve the hierarchical localization. Global-appearance descriptors are obtained from the visual dataset. Afterwards, descriptors are introduced into classifier models, which estimate the most probably area where the image was captured (rough localization step).

2. We present an alternative method to solve the fine localization step once the capturing area was estimated. This method is based on a fitting neural network that estimates the position where the image was captured within the area selected. The learning process of the present method is also carried out through the use of holistic descriptors.

3. We study the use of different deep learning tools with the aim to obtain holistic descriptors that perform robustness with the aim to solve the localization task. These novel methods are compared with hand-crafted holistic descriptors based on analytic methods. The goodness of these methods are measured according to the accuracy and computing time.

The remainder of the paper is structured as follows: Section 5.2 outlines the machine learning tools used throughout this work. After that, section 5.3 outlines the holistic descriptors based on deep learning tools that are proposed to solve a standard localization task. Section 5.4 explains the use of the machine learning techniques together with holistic descriptors to solve the localization in hierarchical maps. Then, section 5.5 presents the experimental results and the discussion about them. Last, section 5.6 outlines the conclusions reached.

## 5.2 Machine Learning Tools

Machine learning methods try to automate the construction of analytic models from data analysis. These methods belong to the AI (Artificial Intelligence) branch and they are based on the idea that the systems can learn to identify patterns departing from the data. Common machine learning techniques include decision trees, support vector machines and ensemble methods. Numerous works regarding the use of machine learning tools together with visual information can be found in the related literature. For example, Clark *et al.* [53] carry out a mapping and a posterior re-localization task by means of feeding a LSTM network with holistic descriptors obtained from a CNN; Neto [199] proposes a topological localization system based on monocular images, learning classifier systems and self-organizing maps. The use of these techniques is widely extended in the mobile robotics field. To cite one example, Triebel *et al.* [277] propose the Informative Vector Machine (IVM) classifier for semantic mapping in autonomous mobile robotics. Regarding the present work, the machine learning tools employed are the following: (a) clustering, (b) classification, and (c) data fitting neural networks. Their fundamentals are outlined in the following paragraphs.

In the present chapter, machine learning are proposed to carry out the hierarchical localization, since they are expected to provide advantages both in the rough and in the fine localization steps. The improvements reached by using of these techniques are shown along the chapter, and they can be outlined as follows:

- More robustness against severe changes of lighting conditions.
- They present an efficient solution when the sizes of the data and the map are large.
- Higher hit ratio in localization in the high-level map.
- More accurate resolution to estimate the position of the robot, since there will not be limitations due to the resolution of the map (distance between consecutive images within the dataset). Hence, more accuracy in the fine localization step.

As for **Clustering** algorithms, they consist in grouping information according to some given criteria. The most usual criterion is the distance or the similitude between the information, which is usually arranged in vectors. Clustering techniques have proved to be a good solution to group visual data and more concretely data provided as holistic descriptors [213]. Additionally, chapter 4 evaluated the use of certain clustering algorithms concerning their usefulness to compact visual models. The aim to compact those visual models is to create hierarchical structures based on layers and use those models to tackle the localization task more efficiently.

Considering the results obtained in the previous chapter, spectral clustering [167] is the method selected in the present work to carry out the clustering step. This algorithm is proposed to perform a non-supervised labelling of a visual dataset composed by

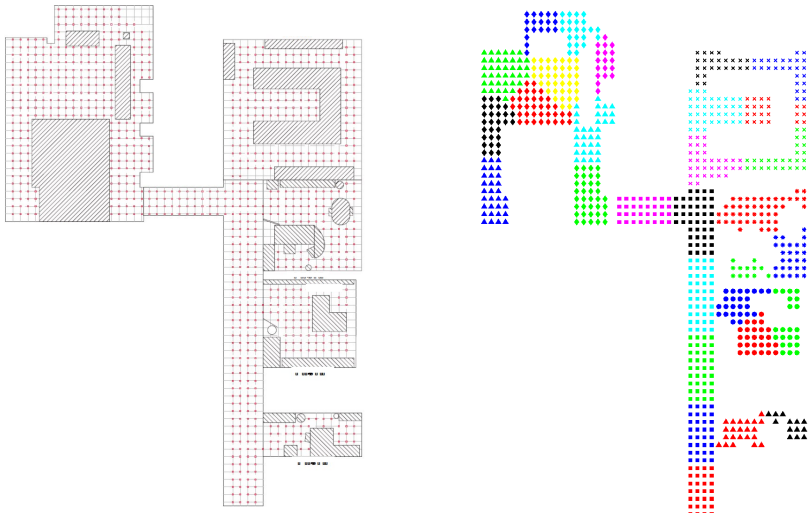



**Figure 5.1:** Example of an indoor map and a non-supervised labelling by means of a clustering method. (a) The red dots show the positions where the images were captured. (b) Result of the clustering process. Each image is grouped according to its visual similitude with the rest of them. This clustering process is tackled by using only visual information.

holistic descriptors obtained from omnidirectional images captured during the mapping task. Fig. 5.1 shows an example of the clustering carried out in an indoor environment.

Regarding the **classification** tool, this consists in predicting the class of given data instances. Classes are also called as labels and they represent categories. Before using this tool to predict categories, the model requires a learning step, where a huge variety of input data ($x_{train}$) together with the corresponding labels ($y_{train}$) are trained. These data used to train the model must be a good enough representation of the whole type of data likely to be present. Therefore, the model carries out a mapping function approximation from input variables to discrete output variables. Due to this, the model is expected to achieve a well tuning configuration and it is ready to receive new data ($x_{test}$) and estimate their categories ($y_{estimated}$). Along the present chapter, three types of classifications have been proposed, since they have already been used by other author to solve mobile robotics tasks:

- **Naïve Bayes (NB) classification**. It is based on Bayes' theorem with independence assumptions between the features. This classification was introduced by Maron [173] as a method for text categorization to solve the problem of judging documents (such as spam or legitimate, sports or politics, etc.) with word frequencies as features. To cite one example of use in mobile robotics, Posada *et*

*al.* [221] propose a naive Bayes classifier to predict the occupancy in a local environment of the robot by fusing the evidence provided by different segmentations and models and hence to carry out a visual navigation.

- **Shallow Neural Network Pattern Recognition classification**. The shallow neural network is a framework to solve different tasks through processing different data inputs and also through using many different machine learning algorithms [287]. Such systems "learn" to perform tasks by considering examples (training data), generally without being programmed with any task-specific rules. The network automatically generates identifying characteristics from the learning material that it processes. As an example of use, Barshan *et al.* [15] used this type of classifier in mobile robotics to differentiate with higher accuracy targets obtained by SONAR.

- **Support Vector Machine (SVM) classification**. Introduced by Cortes and Vapnik [55], SVM can be used for classification or regression purposes. The algorithm considers each data element as a point in an n-dimensional space (where n is the number of features) and the value of each feature is the value of a particular coordinate. Then, a classification is tackled by finding the hyperplane or hyperplanes that best differentiate the categories. As an example of use, Iagnemma and Ward [111] propose an approach based on signal recognition to detect the immobilization of autonomous mobile robots in outdoor terrain through the use of SVM classifiers, which are used to form class boundaries in a feature space composed of statistics related to wheel inertia and speed measurements.

Last, concerning the **function approximation network**, this is proposed to fit practical functions. This is, the neural network is not trained to predict the correct category of the input data but to estimate a value among a specific range ($x = [x_{min}, x_{max}]$, where $x_{min}, x_{max} \in \mathbb{R}$). The training of the network is carried out through a supervised learning, i.e., the learning process assumes the availability of a labelled set of training data made up of $N_{train}$ input—output examples. This work proposes the use of this type of neural networks to estimate the position $(x, y)$ of the robot within an specific area or room.

Figures 5.2, 5.3 and 5.4 show the way the proposed machine learning tools are used throughout this work to solve the mapping and subsequent localization tasks. In these diagrams, $im$ is the panoramic image; $(x, y)$ are the coordinates where a specific image was captured; $\vec{d}$ is the global-appearance descriptor calculated for a specific image; *k* is the number of categories; $c_i$ is the *i-th* category; $N_{train}$ is the total number of images used for training; $im_{test}$, $(x_{test}, y_{test})$ and $\vec{d}_{test}$ are the image, coordinates and descriptor calculated for a test image; $p(c_i)$ is the likelihood that the input $\vec{d}_{test}$ belongs to the category $c_i$; $(x_i,y_i)$ are the coordinates of the position within the environment which correspond to the *i-th* image; and $(x_{est},y_{est})$ are the coordinates which have been estimated for a test image.

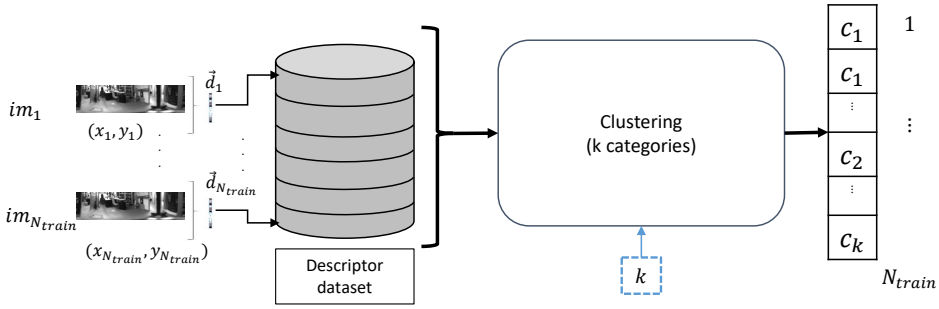



**Figure 5.2:** Diagram of the clustering tool used throughout this work to solve the mapping task. This tool groups the visual description data in $k$ clusters regarding their similitude. Every descriptor is then associated to a specific cluster ($C_1, C_2, ...,$ or $C_k$).

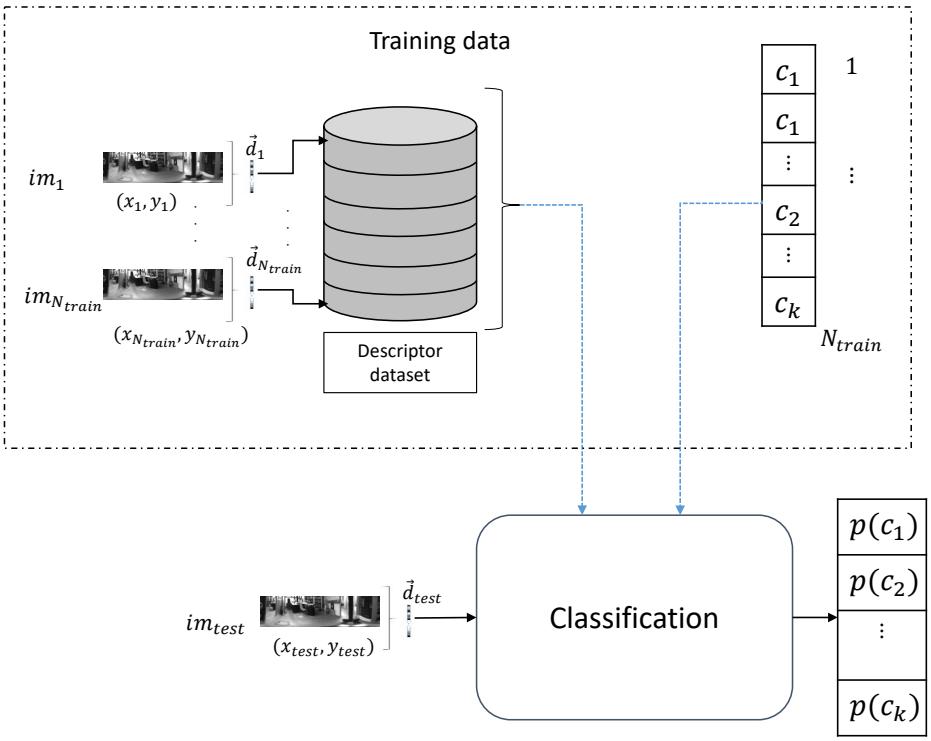



**Figure 5.3:** Diagram of the classification tool used throughout this work to solve the mapping task. This tool learns to classify global-appearance descriptors departing from the training data (descriptor dataset) and their labeling information. Once the classifier is trained, it is expected to correctly classify a new descriptor from a new test image captured.

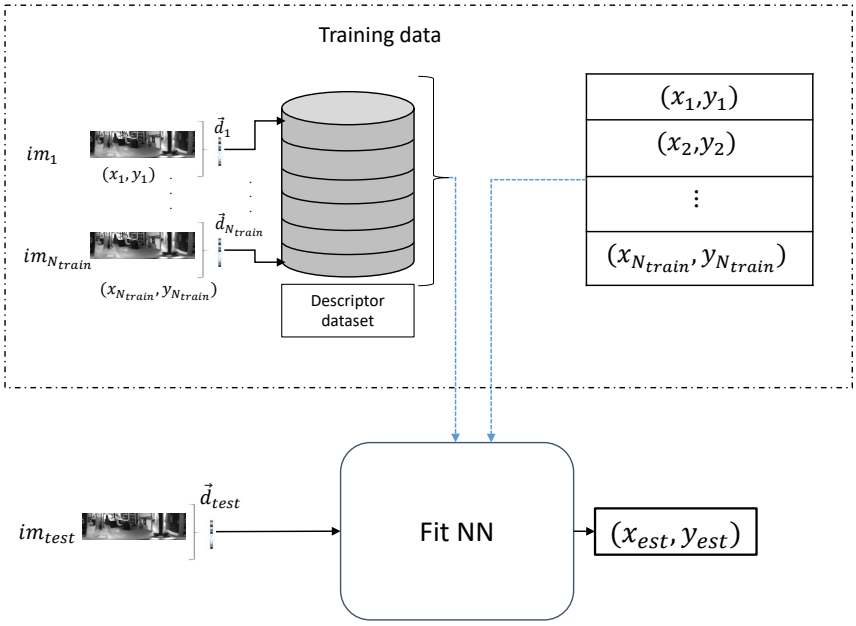



**Figure 5.4:** Diagram of the data fitting neural network tool used throughout this work to solve the mapping task. Similar to the classifiers, in this case, coordinates are introduced to the network instead of labels. Once the tool is trained, it is expected to estimate the coordinates which correspond to the test image.

The learning process of the machine learning tools is carried out by using a visual dataset, which consists in $N_{train}$ holistic descriptors that were calculated from $N_{train}$ panoramic images obtained along the environment. In clustering (fig. 5.2), visual description information is introduced and the algorithm groups the information in $k$ categories by considering the similitude between descriptors. As for the classifier tool (fig. 5.3), a training phase is previously performed, that is, a set of training data items (visual descriptors and their correct categories) are introduced and the configuration parameters are tuned according to those values. After that, the classification task can be performed when a new input descriptor is presented. Alike to the classifiers, the data fitting network (see fig. 5.4) needs a training step. Nevertheless, in this case the target data are not categories but they are real values within a range.

## 5.3 Holistic Descriptors Based on Deep Learning Tools

As showed by several authors and also in previous chapters of the present thesis, global-appearance description has been proposed to solve the localization task due to the advantages these methods present, such as they permit straightforward localization algorithms. The first global-appearance descriptors used in computer vision were descriptors based on hand-crafted methods. They are basically based on calculations

of gradients and orientation of the different pixels that compose the image. To cite one example of use in mobile robotics, Su *et al.* [266] used a holistic descriptor to reduce pose search space with the aim to solve the kidnapped robot problem in indoor environments under different conditions. Among the wide range of global-appearance description methods, the present thesis has studied the use of FS, HOG and *gist*. Additionally, based on previously showed results (see chapter 4), HOG and *gist* have proved to present interesting results for localization tasks.

Nonetheless, in recent years, the emergence of deep learning techniques has boosted the use of descriptors based on these methods. For instance, Naseer *et al.* [197] propose a localization method based on global appearance (by using histograms of oriented gradient descriptors and features of deep convolutional neural networks) to solve the localization problem and keep in parallel hypotheses of possible trajectories. This work studies the use of convolutional neural networks (CNN) and the use of automatic encoders to solve the localization task. The idea is to obtain vectors that characterize images using some deep learning techniques. On the one hand, these methods can be very interesting as their use can focus on specific types of images (such as indoor environments in our case) and therefore provide more efficient descriptors. On the other hand, these methods involve pre-training which usually involves a large data processing and a remarkable time.

Focusing on the use of **CNNs**, these networks have been widely proposed to carry out classification/categorization tasks. In this sense, (1) a set of images correctly labelled is collected and introduced into the network to tackle the learning process and after that, (2) the network is properly available to face the classification (test image as input and the CNN outputs the most likely label option). The CNNs are composed by several hidden layers whose parameters and weights are tuned through the training iterations. CNN is explained with more detail in chapter 6. In the present chapter, some hidden layers outputs are used to obtain holistic descriptors. This idea have already been proposed by some authors such as Mancini *et al.* [170], who use them to carry out place categorization with the Naïve Bayes classifier. Similar to that work, the work presented in the chapter 4 also proposes the use of CNN-based descriptors to create hierarchical visual models for mobile robot localization. The present work addresses an exhaustive evaluation of the layer of the same network with the aim to obtain more efficient holistic descriptors for localization.

The CNN architecture that was used is *places* [318], which was trained with around 2.5 million images to categorize 205 possible kinds of scenes. It should mention that neither the previous chapter nor the present work tackle a re-training, but the CNN is used with the parameters learned originally. The network, which is based on the caffe CNN, basically consists in (1) an input layer, (2) several intermediate hidden layers and (3) an output layer. Within the intermediate layers, the first phase consists in (2.1) layers for featuring learning (whose layers incorporate several filters and the output generated are used as input for the next layer) and (2.2) layers for classification (whose layers are fully-connected and they generate vectors that provide information for classification). The intermediate layers carry out operations over the input data with the aim to learn the specific characteristics of those data. The convolution layers

apply several filters over the input images and each one activates certain characteristics and the pooling layers simplify the outputs, since they tackle a non-lineal downsampling. These layers are repeated along the network and each layer learns different characteristics.

In the present work, we have evaluated the output information from three fully-connected layers ($fc_6, fc_7$ and $fc_8$) whose output size are $4096 \times 1$, $4096 \times 1$ and $205 \times 1$ respectively. In addition, we have obtained two descriptors from the output of 2D convolutional layers ($conv_4$ and $conv_5$). These layers apply several sliding convolutional filters to the input with the aim of activating certain characteristics of the image.Therefore, the output of these layers is a set of images that are the input image after being filtered. Last, a descriptor of these layers is basically obtained by selecting an image from the output dataset and arranging the matrix data into a single vector. Since the size of the output images is $13 \times 13$, the size of the descriptor is $169 \times 1$.

As for the use of **autoencoders**, the aim of these neural networks is to reconstruct the output through compressing the input into a latent-space representation [109]. They are designed to copy perfectly the input data. The model learns to prioritize which aspects of the input should be copied. Traditionally, autoencoders have been proposed for dimensionality reduction or for feature learning [90]. This kind of neural networks are composed by a hidden internal layer that produces a latent representation $h$. Basically, the network consists in two parts: in the first one, an encoding is tackled ($h = f(x)$) and in the second part, a decoding or reconstruction ($r = g(h)$) with the objective $g(f(x)) = x$ (see fig. 5.5).

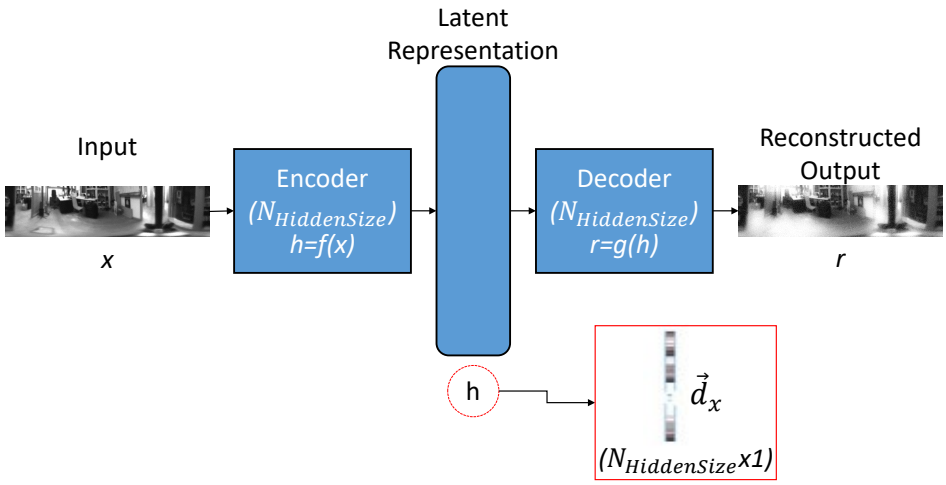



**Figure 5.5:** Autoencoder architecture design and extraction of features departing from the latent representation.

Regarding the use of this deep learning tool for dimensionality reduction, it has provided reconstructions with an error rate fewer than using other techniques such

as PCA [104]. Moreover, by means of dimensionality reduction, information retrieval has also obtained substantial benefits. With the use of autoencoders and its related dimensionality reduction, searching becomes more efficient. For example, Pfeiffer *et al.* [218] present a study about learning-to-rank and query refinement approaches for information retrieval in the pharmacogenomic domain. Zhu *et al.* [321] propose using autoencoder for feature learning on the 2D images with the aim to carry out 3D shape retrieval. Moreover, some works can be found in the state of the art concerning the use of autoencoders for mobile robotics. Sergeant *et al.* [247] address a navigation task by using a deep autoencoder to learn from a dataset sensor input how to navigate. Wang *et al.* [294] propose an autoencoder for fusion and extraction of multiple visual features from different sensors with the aim of carrying out motion planner based on deep reinforcement learning. Gao and Zhang in [84] use autoencoders to detect loops for visual Simultaneous Localization And Mapping (SLAM).

## 5.4 Hierarchical Localization Based on Machine Learning

The use of holistic descriptors to solve the mapping and localization tasks has been widely proposed during the last years. For instance, Faessler *et al.* [72] present a vision-based quadrotor system to map a dense three-dimensional area. Korrapati and Mezouar [132] propose the use of omnidirectional images through global-appearance descriptors to build topological maps and also a loop closure detection method.

In the previous chapter (see chapter 4), we proposed a method to build hierarchical maps through a combination of clustering methods and holistic descriptors. The results obtained proved that the proposed approach is a feasible alternative to build robust compact maps. In the present chapter, the map is structured into two layers. First, the low-level layer is composed of a set of descriptors, each one corresponding to an image in the original training dataset. Second, the set of descriptors $D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{N_{train}}\}$ is divided into $n_c$ groups by using a labelling method, where each group of descriptors is expected to contain information from zones which are visually distinctive. Once the labelling process has been carried out, each zone of the high-level map is represented by a representative descriptor. Hence, a set of representatives is obtained $R = \{\vec{r}_1, \vec{r}_2, ..., \vec{r}_{n_c}\}$, where $\vec{r}_i$ is the representative obtained for the *i-th* group, and this set is the high-level map. Therefore, the localization process is solved hierarchically after building the maps. This process basically consists in the following steps:

1. The robot captures a test image ($im_{test}$) from an unknown position.
2. It calculates the holistic descriptor of the image ($\vec{d}_{test}$).
3. This descriptor is compared with the representative descriptors from the high-level map ($\vec{r}_1, ..., \vec{r}_{n_c}$) and a zone is selected as the most likely one ($c_i$).
4. The most similar descriptor is selected ($\vec{d}_{c_i,k}$) and the position of $im_{test}$ is estimated as the position of the robot from which the image whose descriptor is $\vec{d}_{c_i,k}$ was captured.

### 5.4.1 Classifiers for Localization in the High-Level Map

The present work proposes an alternative hierarchical localization method based on the use of classifiers to estimate the position in the high-level layer (rough localization step). This process consists in introducing the holistic descriptor into a classifier that outputs the corresponding capturing zone. The use of this tool is expected to provide some advantages in comparison to the method presented previously, which basically consists in finding the nearest neighbour among the representatives in the high-level map. The proposed localization method is not only expected to be more robust against changes of illumination, but it is also expected to provide a more advantageous success selecting the correct area within the high-level map despite visual aliasing. Additionally, the use of classifiers is suitable for the cases when the data size is large. For instance, Rahimi and Recht [225] propose the use of local features and machine learning tools to face regression and classification tasks with large scale data and they have obtained results that are competitive with state-of-the-art algorithms in accuracy, training time, and evaluation time. The use of the classifier is included in the step (3) of the hierarchical localization process. In this step, the descriptor $\vec{d}_{test}$ is fed into a classifier, which retrieves the most likely area. The proposed process is depicted in fig. 5.6. and it consists in the following steps:

1. The robot captures a test image ($im_{test}$) from an unknown position.
2. It calculates the holistic descriptor of the image ($\vec{d}_{test}$).
3. The descriptor obtained is fed into a classifier, which retrieves the most likely area.
4. After the classification, the descriptor is compared with the descriptors of the low-level map that are included in the selected area and the position of the robot is estimated as the position that corresponds to the most similar descriptor from the selected area.

### 5.4.2 Solving the Fine Localization Problem Using Function Approximation through a Data Fitting Neural Network

This work also proposes the use of neural networks to solve the fine localization step. The proposed neural network is trained with the aim to overcome the issues carried by the use of visual data as unique source of information, which are mainly the visual aliasing and the resolution in localization due to the distance between consecutive acquisition points in the training dataset. This data fitting neural network is trained with the aim to approximate the coordinates (x,y) of the test image. The network is trained with a set of visual descriptors from a training dataset, along with a coordinates (x,y) in the floor plane from which each training image was captured.

Therefore, a data fitting Neural Network can be used to solve the fine localization as a function approximation problem. Before the localization task, the model

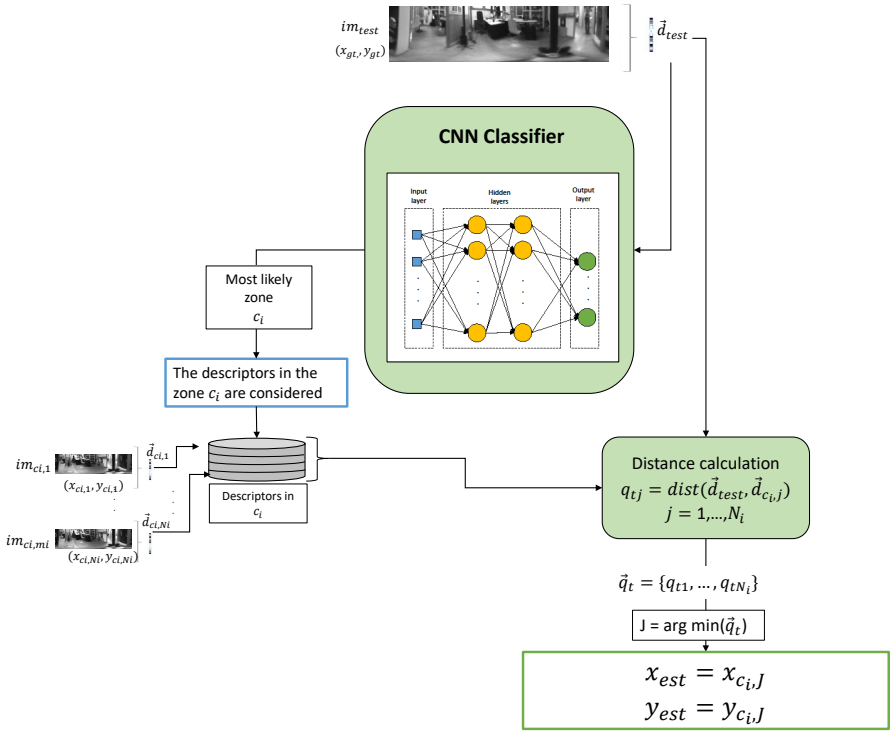



**Figure 5.6:** Hierarchical localization diagram. A classifier is previously trained. To start the localization process, a new image $im_{test}$ is captured and its holistic descriptor $\vec{d}_{test}$ is obtained. First, the rough localization is solved through a classifier to retrieve the most likely zone $c_i$ within the map. Second, the fine localization is solved by calculating the distances between $\vec{d}_{test}$ and the descriptors contained in the retrieved zone $c_i$ (stored in the vector $\vec{q}_t$). The most similar descriptor $\vec{d}_{c_i,k}$, which corresponds to the position $J$ of the vector $\vec{q}_t$, is retained. Finally, the position of $im_{test}$ is estimated as the coordinates where $im_{c_i,k}$ was captured. $(x_{est}, y_{est})$ are the retrieved coordinates.

construction is carried out. This process consists in training a net for each area by using the descriptors included in that area and the $(x, y)$ coordinates from which each image was captured. After training the $n_c$ networks, the complete localization process is addressed through the following steps:

1. The robot captures a test image ($im_{test}$) from an unknown position.
2. It calculates the holistic descriptor of the image ($\vec{d}_{test}$).
3. The descriptor obtained is fed into a classifier, which retrieves the most likely area.

4. After the classification, the descriptor $\vec{d}_{test}$ is fed into the previously trained neural network corresponding to the zone retrieved in the previous step.

The outputs of this network are the predicted coordinates of the test image (see fig. 5.7) which are an estimation of the position of the robot. Again (also explained in sec. 5.2), in order to retrieve the position by neural networks, a training must be carried out before using this tool. Additionally, data augmentation is applied over the visual dataset with the aim to improve the training process. Through this technique, the machine learning tools are supplied by $N_{effects}$ times the total number of training images ($N_{train\_augmented} = N_{train} \times N_{effects}$), where $N_{effects}$ is the number of effects applied over the original images dataset.

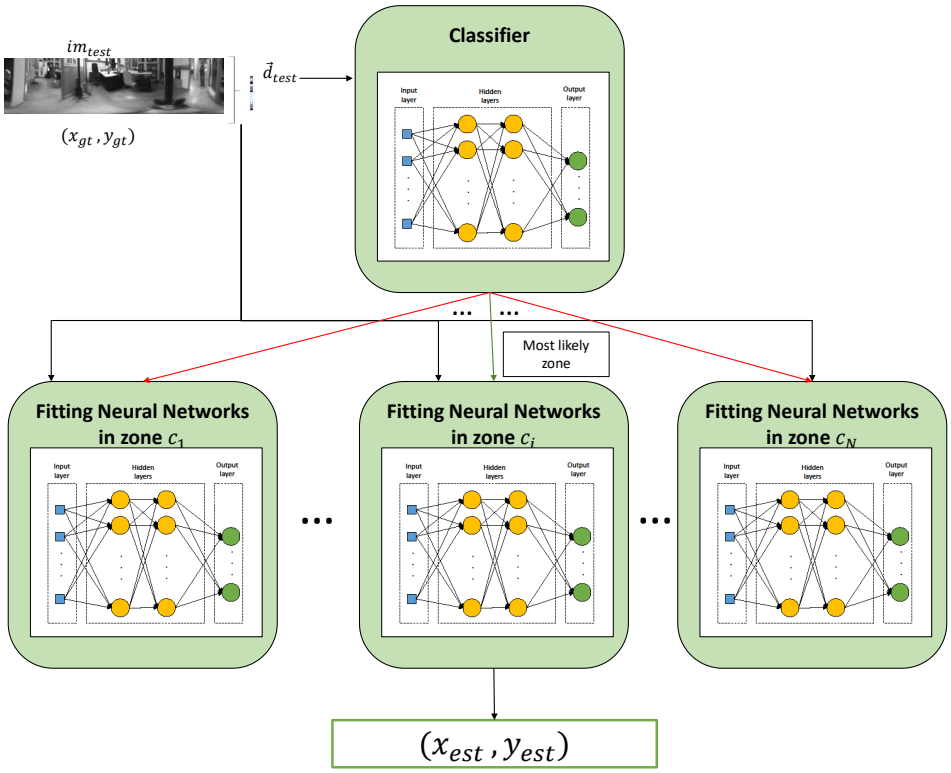



**Figure 5.7:** Hierarchical localization diagram. A classifier and data fitting neural networks have been previously trained. To start the localization process, a new image $im_{test}$ is captured and its holistic descriptor $\vec{d}_{test}$ is obtained. First, the rough localization is solved through a classifier to predict the most likely zone $c_i$ within the map. Second, the fine localization is solved through the data fitting neural network which was trained with the descriptors contained in the retrieved zone $c_i$. Finally, the position of $im_{test}$ is estimated as the most likely position within the previously selected zone. $(x_{gt}, y_{gt})$ and $(x_{est}, y_{est})$ are respectively the coordinates obtained from the ground truth and estimated.

## 5.5 Experiments

Continuing the experiments carried out in chapter 4, the experiments proposed in the present chapter also used the COLD DB [222]. This dataset was chosen because it includes several handicaps that make the experiments proper to analyse the performance of the algorithm under real-operation conditions. There are dynamic changes in the environment such as in the position of furniture and objects or people walking, there is blur effect over some images due to the movement of the camera and also, a high degree of visual aliasing can be perceived in the images. Furthermore, among the three available datasets, Freiburg present features more challenging regarding the visual localization task, since the environment presents a large amount of windows (which complicates the solution of the problem) and the route carried out is the longest. Again, three test datasets with different illumination conditions are used (cloudy, sunny and night datasets) with 2596, 2231 and 2876 images respectively. Last, the experiments have been tackled in a computer with a CPU Intel Core i7-7700® at 3,6 GHz and through Matlab® programming.

### 5.5.1 Hierarchical Localization with Machine Learning

#### 5.5.1.1 Experiment 1: Rough localization

This subsection shows the experiments related with the rough localization in a hierarchical localization model. The first experiment evaluates the performance of the three classifiers to carry out the selection of the corresponding area within the environment. The three classifiers are (a) Naïve Bayes, (b) a classifier based on a shallow neural network that is trained to recognize categories and (c) the SVM (Support-Vector Machine) classifier. These classifiers are trained to solve the category retrieval by means of global-appearance description information obtained from the training dataset. Those descriptors are put into the classifier along with the labels, which indicate the corresponding category for every descriptor.

Concerning the labelling of the training data, two alternatives have been considered. In the first part of the experiment, a manual labelling of the data is carried out, in which the labels correspond to the number of the room where each image was captured, so these labels are integers in the range $[1, 9]$, since the Freiburg dataset contains $9$ rooms. The second part of this experiment evaluates the use of an automatic labelling instead of using the labelling provided by the dataset. This is, using an AI tool to determine the corresponding group of each image contained in the training dataset. We propose to use spectral clustering as AI tool, because it has already provided good solutions among other clustering frameworks in previous works. Due to the fact that spectral clustering groups the data according to their visual similitude, we consider that this automatically labelling can be a more practical solution.

The three holistic descriptors used to train the classifiers are *gist*, HOG and a descriptor obtained from the layer '$fc_7$' of the CNN *places* (this descriptor is named CNN-fc7 throughout the present chapter). Therefore, as fig. 5.8 shows, every classifier is firstly trained with the set of training descriptors and the labels. After that, the

classifier is ready to receive new descriptors (from the test datasets) and calculate the correct label. Moreover, in order to evaluate the use of these tools under changes of illumination conditions, this experiment is also carried out by using test datasets with different illumination conditions.

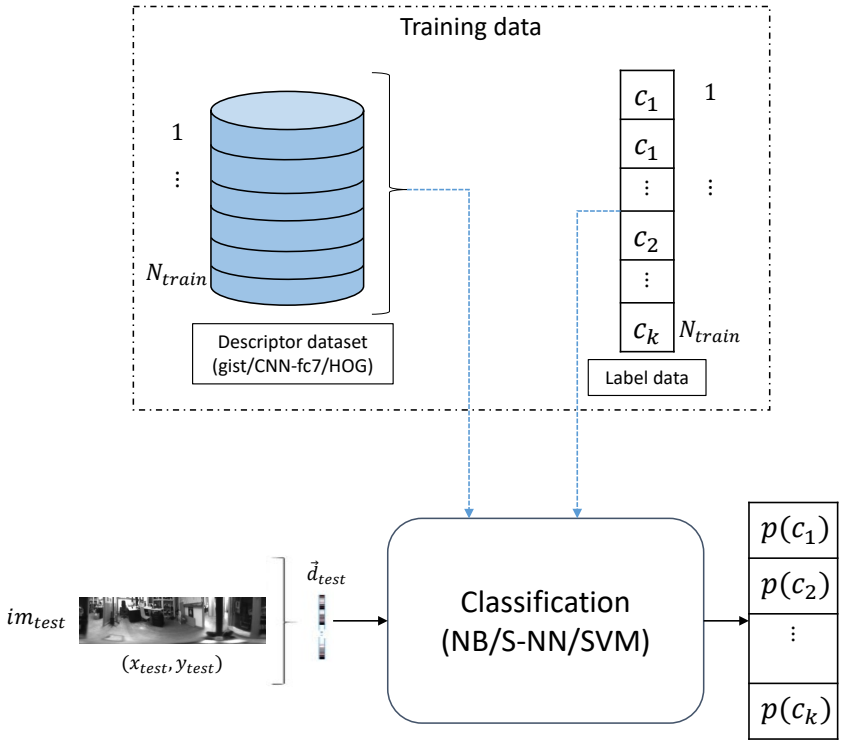



**Figure 5.8:** Experiment 1 diagram. First, the training is carried out: The holistic descriptors of the training data (with $N_{train}$ descriptors) and the corresponding labels (indicating the cluster or area $c_i$) are used to train the selected classifier. Once the training is done, the classifier is ready to solve the area estimation. The process starts with a new image ($im_{test}$), whose holistic descriptor ($\vec{d}_{test}$) is calculated and introduced in the classifier. Finally, it returns a likelihood vector where $p(c_i)$ is the probability that $\vec{d}_{test}$ belongs to the cluster/area $c_i$ ($i = 1, ..., k$, being $k$ the total number of clusters/areas). The most likely option is chosen as the most probably cluster/area where the test image $im_{test}$ was captured.

To evaluate the relative performance of each method, the hit ratio of every configuration (classifier + description method) is collected, i.e. the percentage of success of the classifier, using as input the test data. Table 5.1 shows the results obtained for the three illumination conditions. According to this table, the shallow neural network and the SVM classifier work successfully with the cloudy test images, since both provide hit ratios around 98% when *gist* or CNN-fc7 are used to describe the visual information. On the other hand, the Naïve Bayes Classifier returns the worst solutions and the HOG description proves not to be valid in combination with

**Table 5.1:** Experiment 1 results. Accuracy of the classifiers studied. Hit ratio and average computing time of the rough localization process under three illumination conditions.

| Classifier | Descriptor | Average time (s) | Hit ratio test (%) | | |
|---|---|---|---|---|---|
| | | | Cloudy | Night | Sunny |
| Naïve Bayes | gist | 0.17 | 86.74 | 75.66 | 70.10 |
| Naïve Bayes | CNN-fc7 | 0.51 | 86.13 | 77.78 | 62.39 |
| Naïve Bayes | HOG | 0.05 | 4.24 | 4.10 | 9.86 |
| Neural Network | gist | 0.05 | 98.57 | 83.55 | 82.61 |
| Neural Network | CNN-fc7 | 0.02 | 97.42 | 93.25 | 76.20 |
| Neural Network | HOG | 0.02 | 4.97 | 4.35 | 6.77 |
| SVM | gist | 0.09 | 98.61 | 84.63 | 85.03 |
| SVM | CNN-fc7 | 0.15 | 98.50 | 94.09 | 82.03 |
| SVM | HOG | 0.05 | 7.24 | 5.29 | 5.92 |
| Nearest Neighbour | gist | 0.06 | 80.89 | 70.58 | 70.10 |
| Nearest Neighbour | CNN-fc7 | 0.01 | 78.23 | 76.95 | 53.65 |
| Nearest Neighbour | HOG | 0.02 | 14.07 | 11.27 | 19.77 |

these tools. The optimal configuration is obtained with SVM+*gist*, which produces 98.61% accuracy. The confusion matrix obtained for this configuration with the cloudy test images is shown in fig. 5.9. This figure shows that the correct predictions are higher than 96% in all the areas of the environment. The worst case is produced in the 2-person office 2. In this case, there are 5 false positives, 4 with the corridor and 1 with the 1-person office, which are not critical since these false positives are produced with adjacent rooms and visually similar. Therefore, for future experiments, only the combinations shallow neural network or SVM along with *gist* or CNN-fc7 will be considered.

Concerning how the different illumination conditions affect the rough localization, we can observe that the hit ratio decreases for most of the areas. This effect was expected and was also noticed in the work presented previously (see chapter 4). Despite this deterioration, the results do not differ substantially from the obtained with the cloudy test images. Hence, we can conclude that the classifiers present robustness to carry out the rough localization task even when substantial changes of lighting conditions occur.

In order to compare different rough localization methods, the three last rows of table 5.1 also show the hit ratios obtained through using the nearest neighbour method search, since this was the method considered in the previous chapter. In this case, the high-level mapping consists in obtaining one representative descriptor per area (as the average descriptor of the training images contained in this area). The rough localization consists in calculating the distance between each test descriptor and the representatives and retrieving the room whose representative is the nearest neighbour. The method based on the nearest neighbour provides faster results than the classifiers. Notwithstanding that, the best option (using *gist* as holistic descriptor) provides hit ratios that are substantially lower than those provided by the shallow neural network and SVM classifiers.

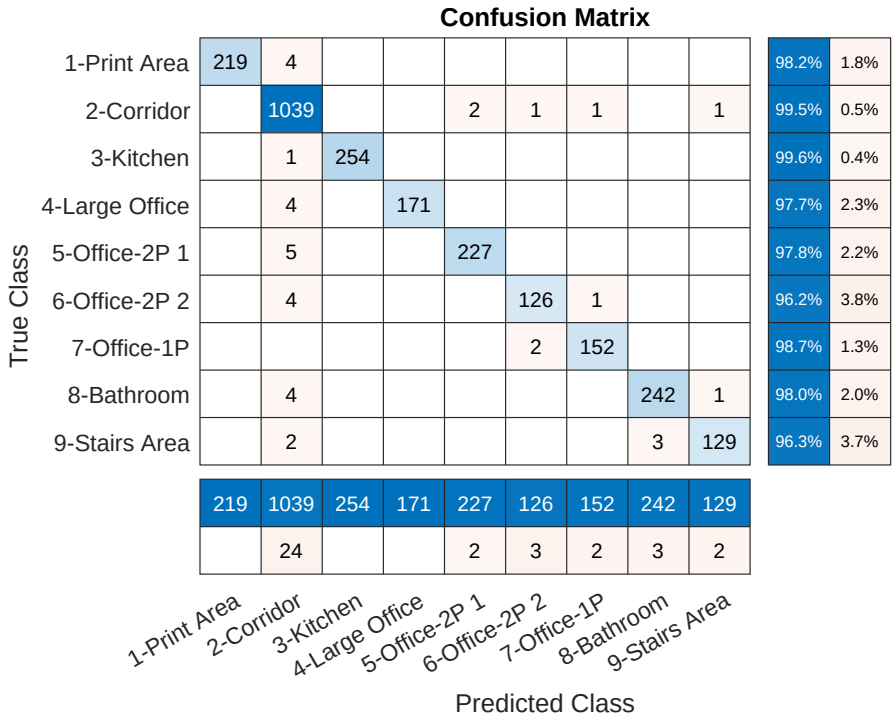



**Figure 5.9:** Confusion matrix of the classifier SVM along with the *gist* descriptor to estimate the area of the images in the cloudy test dataset.

Once proved the utility of the classifiers to tackle the rough localization task and robustness against illumination changes, the second part of this experiment evaluates the labelling of the training data. So far, the previous experiments have been developed considering a manual labelling of the dataset. This way, the training of the classifier has consisted basically in introducing the visual description of the images and its related label. Nevertheless, as explained before, using spectral clustering for automatic labelling can be a more feasible solution. Basically, the steps to label the information are the following. First, the holistic descriptors of the training images are calculated. Second, these descriptors are introduced into the spectral clustering algorithm and a number of clusters $n_c$ is manually selected. This tool outputs a vector of labels that specifies the cluster to which each descriptor belongs (more information regarding the spectral clustering process can be found in section 4.3). Once the vector of labels is available, the steps to train the classifier are the same than in the first part of the experiment (fig. 5.8). Therefore, the high-level mapping process is the following.

1. Global-appearance descriptors are obtained from the training images.

2. This visual information is firstly used to obtain automatically the labels, using spectral clustering ($n_c$ number of clusters).

3. The global-appearance descriptors together with the calculated labels are used to train the classifier.

For this purpose, spectral clustering with $n_c = 2, ..., 13$ clusters are considered, the holistic descriptor *gist* is used to obtain the visual information (to obtain the labels as well as to train the classifiers) and the classifiers SVM and shallow neural network are used. Furthermore, the nearest neighbour method is also used as a reference method to compare the performance of the classifiers with the results obtained in previous works. The results of this experiment are shown in fig. 5.10, 5.11 and 5.12.

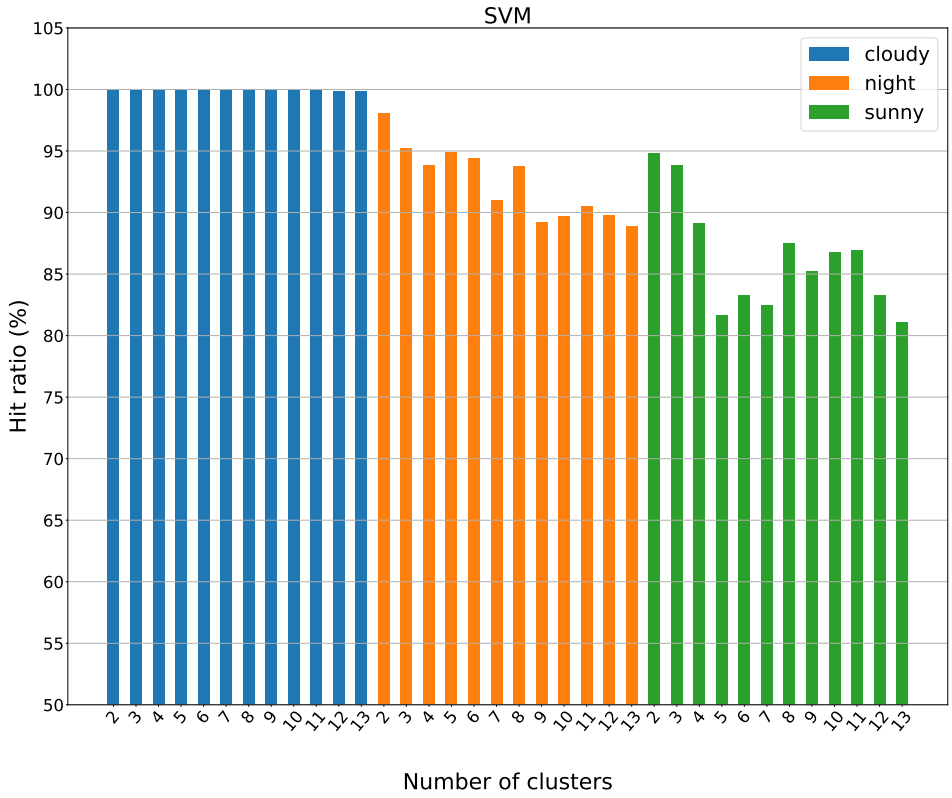



**Figure 5.10:** Experiment 1. Results for rough localization using automatic labeling. Hit ratio for the test images versus number of clusters. The localization step is carried out by means of the SVM classifier.

Both classifiers (SVM and shallow neural network) work better than the nearest neighbour method independently on the illumination conditions. The use of classifiers with cloudy test images provides hit ratios that are always around 100%. Among the

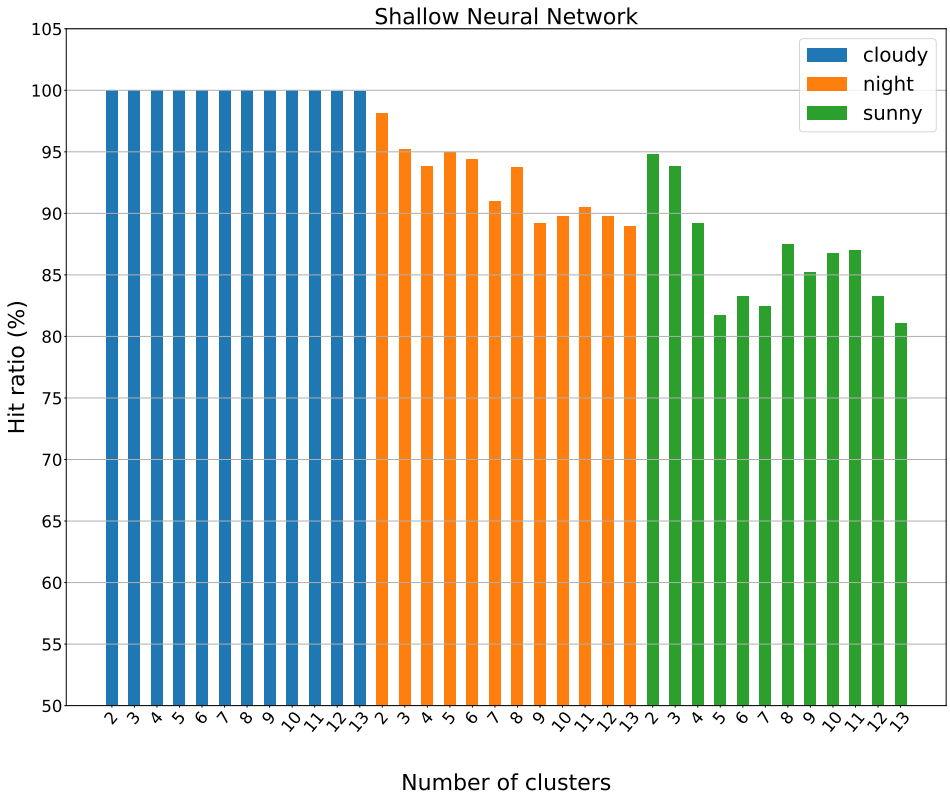



**Figure 5.11:** Experiment 1. Results for rough localization using automatic labeling. Hit ratio for the test images versus number of clusters. The localization step is carried out by means of a shallow neural network classifier.

three methods studied to evaluate the use of automatic labelling under dark illumination conditions (night dataset), the best results are achieved with the neural network classifier and the best results under sunny illumination conditions (sunny dataset) are achieved with the SVM classifier.

Concerning the comparison between labelling methods, the hit ratio is improved by using automatic labelling in comparison to the results obtained by manual labelling (see table 5.1). The hit ratios reached with neural network classifier and manual labelling (80.89, 70.58 and 70.10% for cloudy, night and sunny test datasets respectively) are lower than the obtained with the automatic labelling considering 9 clusters (99.65, 89.29 and 86.96% for cloudy, night and sunny test datasets respectively). Also, the same comparison using the SVM classifier outputs better results by using automatic labelling, for example, the hit ratio improves from 98.61, 84.63 and 85.03% to 100, 89.19 and 85.21% respectively with the three illumination conditions. Only when the number of clusters is higher than 10 and the test images are under sunny illumination conditions, the neural network classifier produces hit results lower than 80%. Therefore,

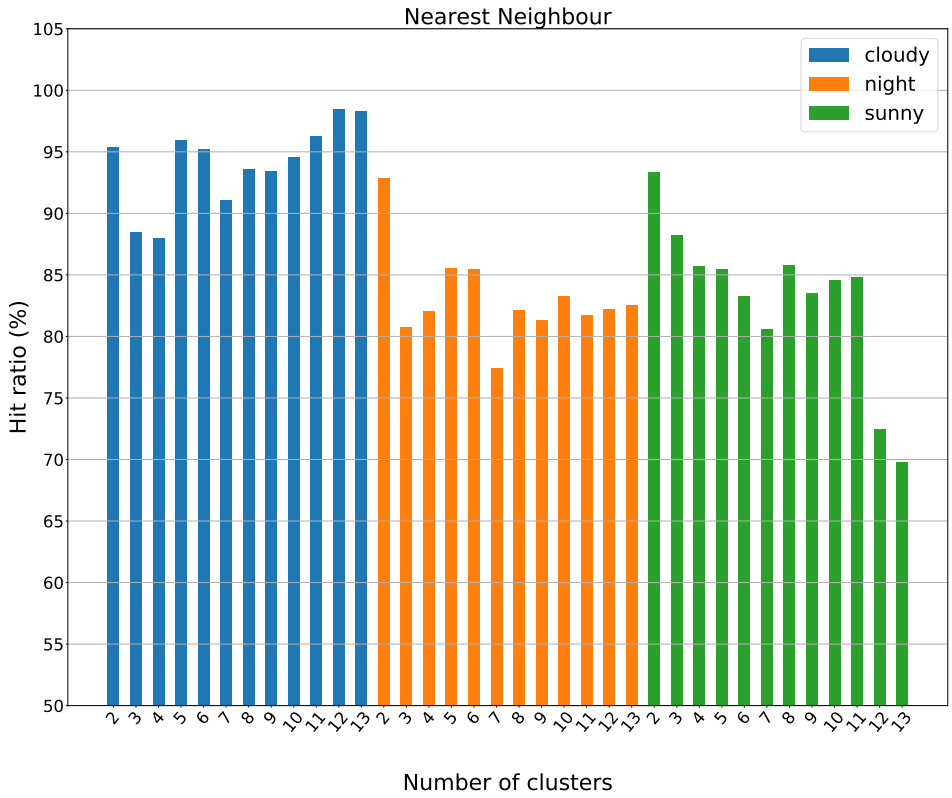



**Figure 5.12:** Experiment 1. Results for rough localization using automatic labeling. Hit ratio for the test images versus number of clusters. The localization step is carried out by means of the nearest neighbour method.

through the present experiment, we conclude that the combination of spectral clustering and a classifier improves considerably the rough localization step and performs more robustly against changes of illumination.

#### 5.5.1.2 Experiment 2: Fine localization

Subsection 5.5.1.1 has shown the utility of some classifiers to solve the rough localization step considering either manual or automatic data labelling. The present subsection goes one step ahead and tries to solve the fine localization. This fine localization consists basically in estimating the position within the selected area (predicted in the rough localization step). These experiments also consider changes of the illumination conditions and the labels to train the classifiers, as in the previous section, are obtained either manually or automatically. The experiments developed in this subsection are proposed departing from the assumption that the proper room/area is always correctly estimated in the previous (rough) localization step. Hence, we focus on the performance of the methods proposed for fine localization.

Two alternatives are considered to carry out the fine localization step: either (a) through an image retrieval approach, by calculating the nearest neighbour or (b) through the use of a data fitting neural network. Regarding the first option, the algorithm basically consists in comparing the descriptor of the test image ($\vec{d}_{test}$) with those descriptors in the training dataset that belong to the previously selected area. Afterwards, the most similar descriptor (nearest neighbour) is retrieved, and the point where the associated training image was captured is considered as the current position of the robot. This process is shown in the fig. 5.6. Regarding he use of the neural network, the process is shown through the fig. 5.7 and it consists of the following steps:

1. Before carrying out the steps related to the neural networks training, a data augmentation is performed with the aim to provide the neural network more information during the training. The data augmentation proposed consists in applying random rotation, darkness/brightness additions, blur effects, noise addition and/or reflection to the original images.
2. Two neural networks per area/room are fit. The first neural network of each room is trained to estimate the $x$ coordinate, given the descriptor of a test image. To do it, the training is carried out with the descriptors of the training images that belong to this specific area, using as labels the coordinate $x$ of their capture points.
3. The second network is trained equally, but using the coordinate $y$.
4. Once the data fitting neural networks are available, the fine localization can be solved: the descriptor of the test image ($\vec{d}_{test}$) is introduced into the neural networks of the area/room retrieved in the rough localization step, and the coordinates ($x$ and $y$ respectively) are estimated.

To evaluate the performance of the two methods proposed, the average localization error has been evaluated for each configuration (localization method + description method), this is, the average error obtained by using test datasets whose images were captured during different illumination conditions (sunny days and at night). The average computing time (from calculating the holistic descriptor until estimating the capturing position) is also collected. Table 5.2 shows the results obtained.

**Table 5.2:** Experiment 2 results. Accuracy of the methods studied to solve the fine localization. Average localization error and average computing time considering test images captured under three illumination conditions.

| **Method** | **Descriptor** | **Average time (ms)** | Average accuracy (m) | | |
|---|---|---|---|---|---|
| | | | Cloudy | Night | Sunny |
| Nearest Neighbour | gist | 89.64 | 1.82 | 1.83 | 2.18 |
| Nearest Neighbour | CNN-fc7 | 30.30 | 1.82 | 1.82 | 2.31 |
| Neural Network | gist | 57.34 | 1.86 | 1.91 | 1.98 |
| Neural Network | CNN-fc7 | 23.55 | 1.90 | 1.95 | 1.98 |

Concerning this table, the computing time for the data fitting neural network option is faster, with time values of 57.34 and 23.55 ms whereas, in the case of the nearest neighbour method, the average time is 86.64 and 30.30 ms. Regarding the localization error, the table shows that the fine localization method based on image retrieval outputs lightly better results than the alternative using data fitting neural networks with the exception of the results with sunny test images. Nonetheless, a more detailed analysis of the results provided by these methods leads to deeper conclusions. Table 5.3 shows the data collected for the fine localization by using nearest neighbour and data fitting neural networks, but in this case, it shows in detail the information per each area. Hence, this table shows that the average error with the neural network is low in areas 1, 3, 5, 6, 7 and 8. They present an error between 2.16 and 4.96 cm. Areas 2, 4 and 9 present worse results. The same drawbacks are presented when the nearest neighbour method is used. These results are due to the fact that the areas 2 (corridor) and 4 (kitchen) are the largest rooms, thus, their training and subsequent pose estimation can be more challenging. Despite these issues, this analysis permits finding out that the use of data fitting neural networks can be a profitable tool to estimate the position of the robot within a non-large and well delimited environment. In general, the results obtained are lightly more accurate than the obtained through the nearest neighbour process, because the estimation with data fitting neural network does not present the limitation of resolution given by the distance between consecutive images in the training set.

**Table 5.3:** Results for fine localization using nearest neighbour and data fitting neural network with gist descriptor. The average errors are collected separately for each area.

| Area | Error through nearest neighbour (cm) | Error through neural network (cm) |
|---|---|---|
| 1 | 5.22 | 4.96 |
| 2 | 417.17 | 430.08 |
| 3 | 4.19 | 2.16 |
| 4 | 167.54 | 168.40 |
| 5 | 4.11 | 3.61 |
| 6 | 5.23 | 3.72 |
| 7 | 5.11 | 3.15 |
| 8 | 4.09 | 3.18 |
| 9 | 6.20 | 4.67 |

Additionally, a comparison between manual and automatic labelling is also carried out. The figures 5.13 and 5.14 show the average error and time results of the fine localization by using nearest neighbour and data fitting neural network respectively. Both methods depart from automatic labelling and they also use the *gist* descriptor. From this graphs, it is shown that the nearest neighbour method performs better than the data fitting neural network. Regarding the computing time, both options provide similar values and they decrease as the number of clusters does, as expected. Finally, to compare the labelling options, the spectral clustering is configured with a number

of clusters $n_c = 9$, since the labelling provided by manual labelling defines 9 categories regarding the 9 rooms which compose the dataset. Through this comparison, automatic labelling proves to behave better than the manual option. For instance, comparing the fine localization with manual labelling and nearest neighbour method (see table 5.2; with *gist* descriptor) with the automatic labelling ($n_c = 9$ clusters) and nearest neighbour method (see fig.5.13), the average error results obtained for the second option (6.6; 32.5 and 62.5 cm respectively for the three illumination conditions) improves substantially the results obtained for the manual labelling option (182; 183 and 218 cm respectively).

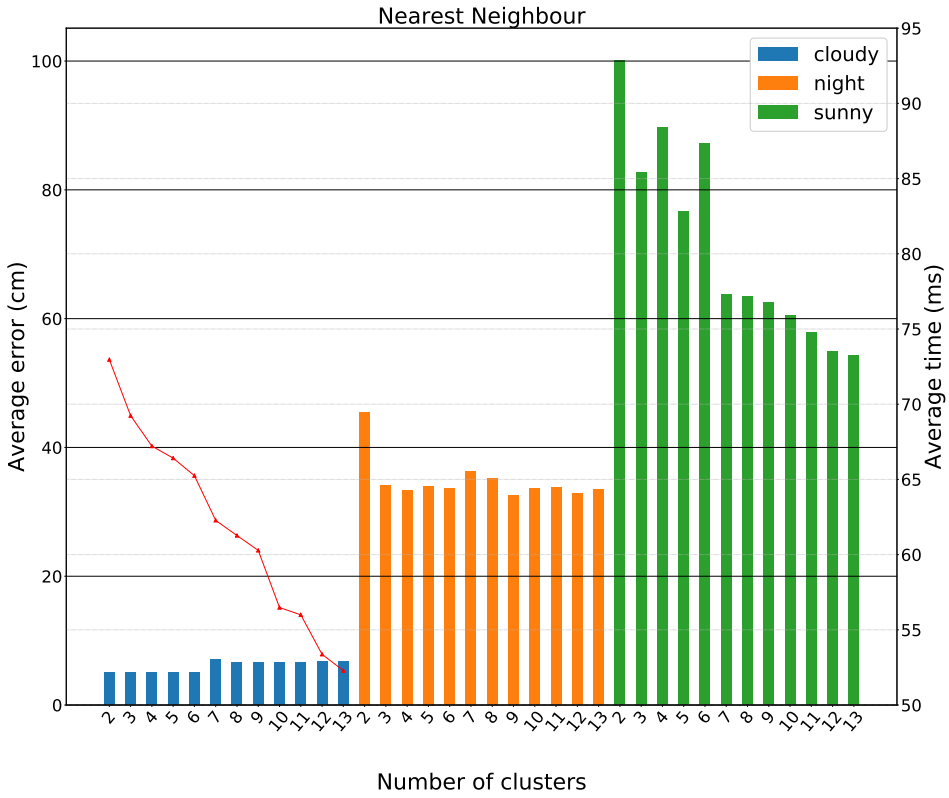



**Figure 5.13:** Experiment 2. Results for fine localization using automatic labeling. Average error (cm) and average computing time (ms) versus number of clusters. The localization is carried out by means of the nearest neighbour method.

#### 5.5.1.3 Experiment 3: Complete hierarchical localization

Subsubsection 5.5.1.1 and 5.5.1.2 have shown respectively the utility of some classifiers to solve the rough localization step and two suitable methods to carry out the fine localization step. The present experiment proposes the join of both items with the aim to solve the complete hierarchical localization (rough and fine localization), since this localization method has proved to be an efficient alternative. Hence, after selecting

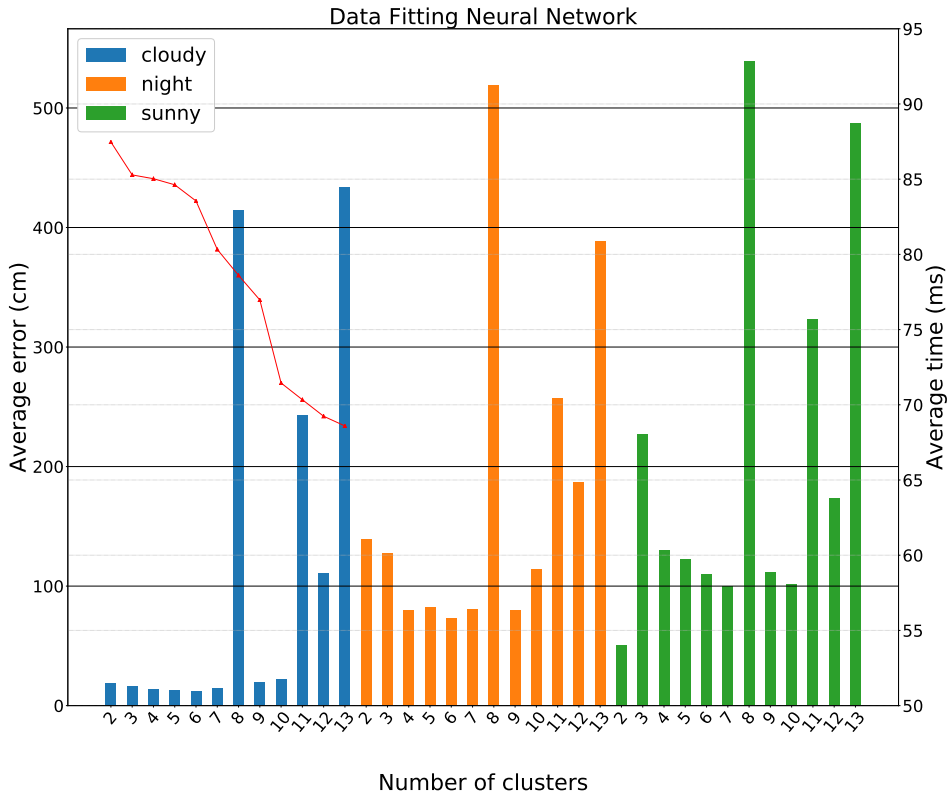



**Figure 5.14:** Experiment 2. Results for fine localization using automatic labeling. Average error (cm) and average computing time (ms) versus number of clusters. The localization is carried out by means of data fitting neural networks.

the area (by using classifiers), the position within the selected area is estimated. The experiment also evaluates changes of the lighting conditions and uses spectral clustering to obtain the labels of the training data (automatic labelling). Two alternatives are considered to carry out the fine localization step: either (a) through an image retrieval approach, by calculating the nearest neighbour or (b) through the use of a data fitting neural network.

To evaluate the performance of the proposed methods, the average localization error and the computing time have been calculated by using the three test datasets of Freiburg. The proposed hierarchical localization process has been evaluated by using the SVM classifier in the rough step and the *gist* descriptors, because the previous experiments have proved that this configuration works efficiently. The results obtained are showed in fig. 5.15 and 5.16. From them, we can analyse that the computing time increases as the number of cluster does, since the rough step (use of classifiers) requires more computing time than the fine step. Moreover, the nearest neighbour option is lightly faster. The average time values are between 115 and 748 ms whereas, in the

case of the neural network, the average time goes from 177 to 956 ms. Nonetheless, these results would be good enough to carry out a localization task online. Concerning the localization error, the figures show that the hierarchical localization method based on image retrieval for fine localization outputs better results than the alternative using data fitting neural network. However, the results obtained through data fitting neural network provide accurate values that are also valid enough to carry out the localization task. For instance, in the case of cloudy test data, the localization error takes values around 13 cm, which is quite low considering that the distance between images is around 20 cm. The results for $n_c = 8$ clusters provide high error values. This is due to the fact that the clustering algorithm was not able to find a compact representation of the environment. Regarding how the change of illumination conditions affect the issue, both test datasets introduce a localization error higher. This behaviour was noticed in previous works and thus, expected to appear in the present experiments. Nevertheless, this increment is smoother with the data fitting neural network option.

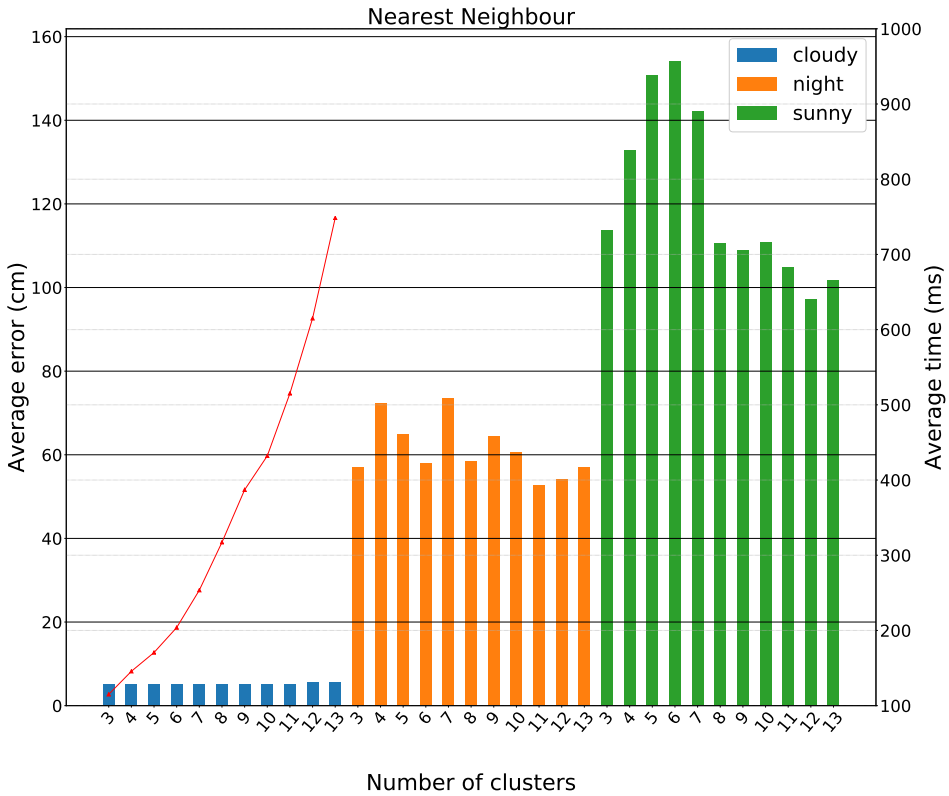



**Figure 5.15:** Experiment 3. Results for complete hierarchical localization using automatic labelling. Average error (cm) and average computing time (ms) versus number of clusters. The localization step is carried out by means of the nearest neighbour method.

Last, a comparison between hierarchical localization methods is established. On the one hand, the methods proposed in this work (rough step with classifiers and fine

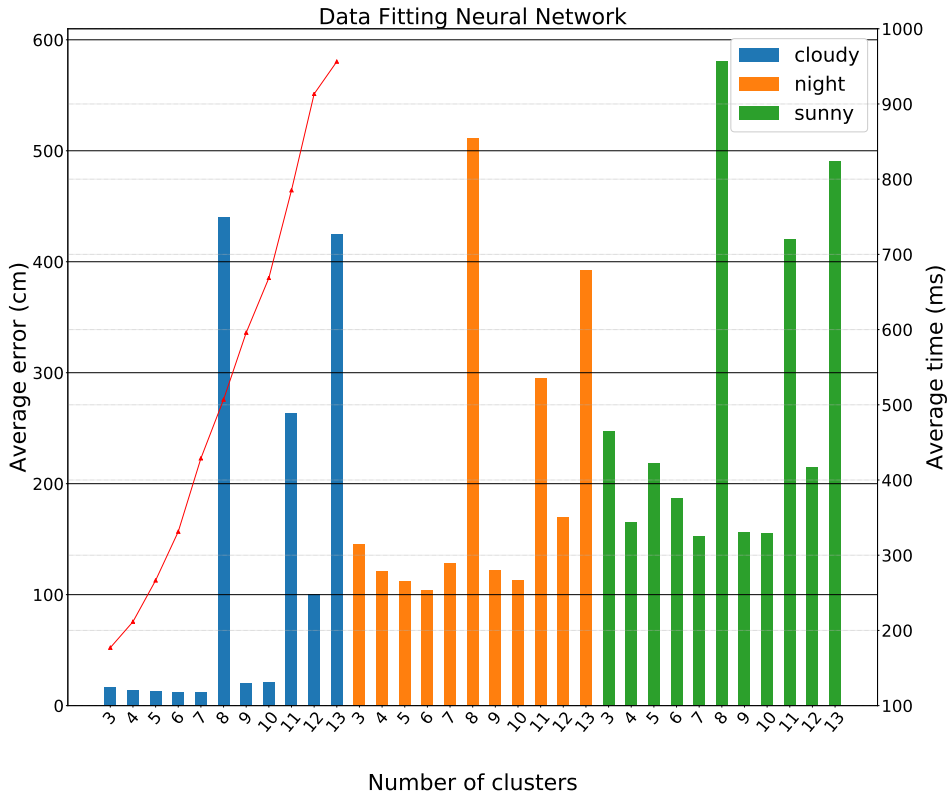



**Figure 5.16:** Experiment 3. Results for complete hierarchical localization using automatic labelling. Average error (cm) and average computing time (ms) versus number of clusters. The localization step is carried out by means of data fitting neural networks.

step with image retrieval). On the other hand, the method proposed in chapter 4 by using a representative per area obtained through spectral clustering in the rough step and image retrieval in the fine step. Fig. 5.17 shows the localization results (average localization error and computing time) versus the number of areas (clusters). This shows that the classifier introduces a more efficient alternative regarding the localization error at the expense of a higher computing time. Furthermore, fig. 5.18 shows the results obtained when the test dataset is composed of images captured at night (dark illumination conditions). Both methods are affected by this illumination condition, but the method based on classifiers presents a major robustness.

### 5.5.2 Localization Using Deep Learning based Holistic Descriptors

The localization task proposed along this experiment consists in an image retrieval problem. This is, obtaining the image from the training dataset that presents the highest similitude in relation to the new captured (test) image. Therefore, with the aim

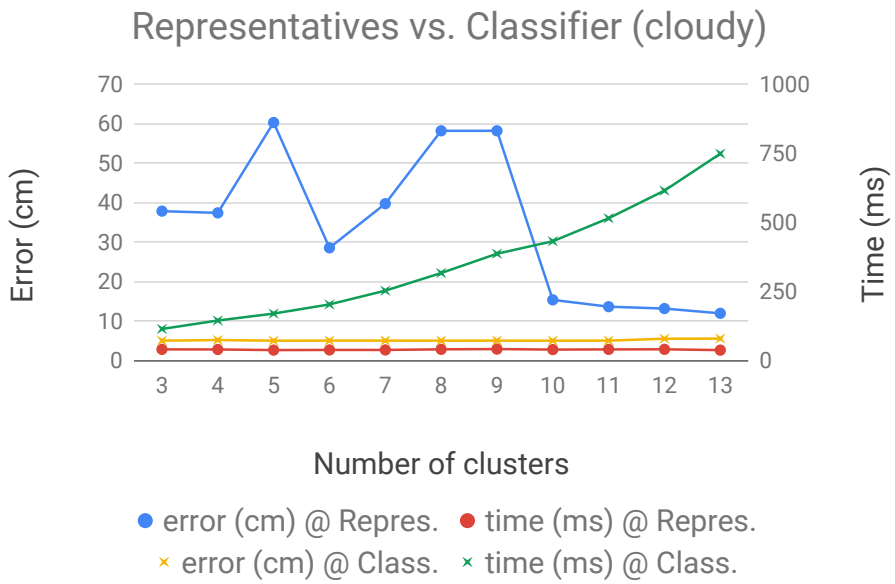



**Figure 5.17:** Comparison of hierarchical localization methods. Solving the rough localization by calculating the nearest neighbour to the representatives obtained by spectral clustering and through the SVM classifier. Average localization error and average computing time.

to carry out the proposed image retrieval task, a mapping process is previously carried out, i.e., visual data is captured from the environment. After obtaining the $N_{Train}$ images from different positions, holistic descriptors are calculated. The mapping task should be tackled before starting the localization. Last, the localization task is solved through the following steps:

- The robot captures a new image from an unknown position ($im_{test}$).
- For that test image, the corresponding descriptor is calculated ($\vec{d}_{test}$).
- Once the descriptor is available, the robot calculates the (cosine) distance (since this distance has presented the best solution in previous works to calculate similitude distance using global-appearance descriptors) between the test descriptor ($\vec{d}_{test}$) and each descriptor from the visual model ($\vec{d}_j$, where $j = 1, .., N_{Train}$).
- A vector of cosine distances is obtained as $\vec{h}_t = \{h_{t1}, ..., h_{tN_{Train}}\}$ where $h_{tj} = dist\{\vec{d}_{test}, \vec{d}_j\}$.
- The node that presents the minimum distance($d_t^{nn}|t = argmin_j h_{tj}$) corresponds to the estimated position of the robot.

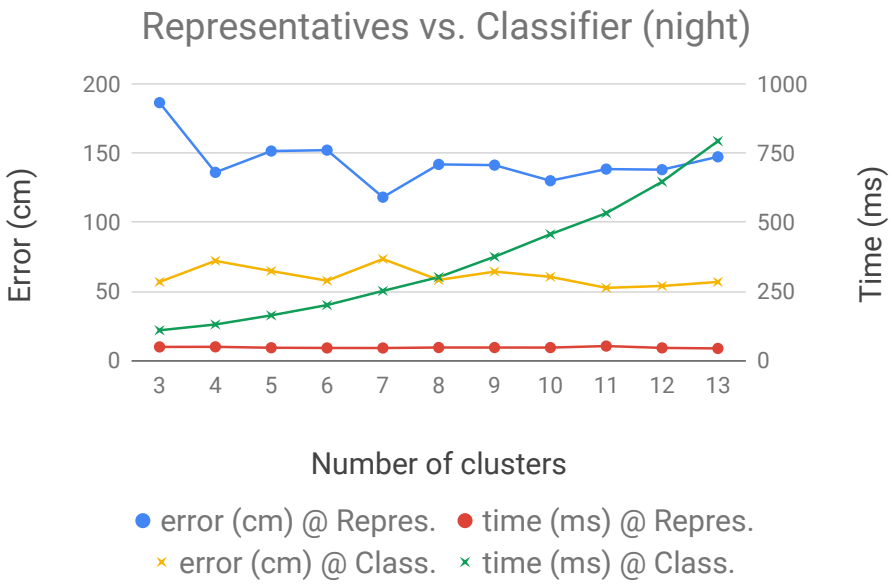



**Figure 5.18:** Comparison of hierarchical localization methods under dark illumination conditions. Solving the rough localization by calculating the nearest neighbour to the representatives obtained by spectral clustering and through the SVM classifier. Average localization error and average computing time.

The present experiment focuses on testing the goodness of the proposed holistic descriptor method based on deep learning techniques for localization. This HOG and *gist* descriptors are also used for this purpose with the aim to compare the results obtained by using deep-learning or hand-crafted descriptors. The experiments were carried out with the Freiburg training and $cloudy_{test}$ datasets. Nevertheless, no illumination changes have been considered. This lack of illumination changes is due to the fact that this work is focused on studying the goodness of the descriptors for localization task, however, as possible future work, we could tackle an extension to consider the illumination changes effects.

The evaluation of description methods is based on two parameters. First, the average localization error, which measures the Euclidean distance between the real position (provided by the ground truth dataset) where the test image was captured and the position estimated by the localization algorithm. Second, the average computing time, which is analysed by means of two values, (a) the computing time to calculate the descriptor and (b) the computing time to estimate the position of the test image.

The results obtained through the use of hand-crafted descriptors (HOG and *gist*) and the descriptors based on deep learning (autoencoders and CNNs) are shown respectively in the tables 5.4, 5.5 and 5.6. These tables show the size of the descriptor, the average localization error in cm, the average computing time to calculate the

descriptor and the average computing time to estimate the position of the test images (both in ms).

Concerning the results achieved by using descriptors based on analytic methods (see table 5.4), in the **HOG** case, the localization error does not decrease significantly as the size increases; the calculation time to calculate the descriptor is also barely constant, but the time to estimate the pose increases as the size of the descriptor increases. From these results we can conclude that the descriptor whose size is 64 is considered the best option, since this configuration has a relatively good accuracy and the minimum calculation time. In the case of the ***gist* descriptor**, the localization error decreases by millimeters as the descriptor size increases, but the time to calculate the descriptors and the time to to estimate the pose increases significantly as the size increases. We establish the descriptor with minimum size as the best option.

As for the descriptors obtained through the use of **autoencoders** (see table 5.5), the outputs obtained for both cases (autoenc-Frib and autoenc-SUN) whose size of hidden representation (number of neurons) is 10 show the worst localization error results. In the case of autoenc-Frib, the descriptors obtained from autoencoder with $N_{HiddenSize} = 50 - 500$ output relatively good results (localization error between 7,04 and 7,45 cm), but only for the autoenc-SUN with $N_{HiddenSize} = 500$ the results are similar. As for the computing times, the time to calculate the descriptor and the time to estimate the pose are directly proportional to the length of the descriptor. For example, in the case of $N_{HiddenSize} = 500$, with autoenc-Frib and autoenco-SUN, the average time are 1.166 s and 1.125 s respectively. Hence, we conclude that for autoenc-Frib, the best configuration is reached through the autoencoder based on 100 neurons, because the localization error is the minimum and the computing time is the third lowest. For autoenc-SUN, despite the configuration with $N_{HiddenSize} = 500$ presents the worst time values, it is selected as the best since the rest of configurations studied do not provide suitable solutions to solve the localization task.

Last, for the **CNN-based** descriptors (see table 5.6), all the different layers evaluated present good results. The convolutional 2D layers *'conv4'* and *'conv5'* (from the feature learning stage) achieve an accuracy of around 5 cm. This behaviour is reasonable since the aim of the first layers in a CNN is to obtain global characteristic information from the images and the further CNN layers (fully connected) are focused on optimizing the desired task. The use of the layers *'conv4'* and *'conv5'* to obtain holistic descriptors was scarce in mobile robotics based on visual information until the presented work and it is shown that they are able to provide very optimal solutions. Concerning the computation time to calculate the holistic descriptors, none of the studied layers need high values. Moreover, as it was expected, the further the corresponding layer is, the higher the time is. Again, the computing time to estimate the pose is directly proportional to the size of the descriptor. Among the convolutional 2D layers, the localization algorithm, based on the descriptor based on *'conv5'* needs less time than using *'conv4'*. Consequently, *'conv5'* is selected as the best layer to calculate holistic descriptors by means of using CNNs.

**Table 5.4:** Results obtained through the use of holistic descriptors based on hand-crafted methods (HOG and *gist*) to solve visual localization.

| Descriptor | Size | Error loc. (cm) | Time comp. descriptor (ms) | Time pose est. (ms) |
|---|---|---|---|---|
| HOG | **64** | $16.34 \pm 0.78$ | **44.64** | 0.38 |
| | 128 | $16.23 \pm 0.73$ | 45.27 | 0.51 |
| | 256 | $16.22 \pm 0.69$ | 45.33 | 2.48 |
| | 512 | $16.17 \pm 0.69$ | 46.52 | 4.75 |
| *gist* | **128** | $5.19 \pm 0.18$ | 10.30 | 0.45 |
| | 256 | $5.11 \pm 0.17$ | 11.98 | 2.19 |
| | 512 | $5.09 \pm 0.16$ | 21.21 | 4.17 |
| | 1024 | $5.08 \pm 0.16$ | 40.07 | 10.72 |

**Table 5.5:** Results obtained through the use of holistc descriptors based on autoencoders (autoenc-Frib and autoenc-SUN) to solve visual localization.

| Descriptor | Size | Error loc. (cm) | Time comp. descriptor (ms) | Time pose est. (ms) |
|---|---|---|---|---|
| autoenc-Frib | 10 | $599.83 \pm 3.83$ | 49.79 | 0.25 |
| | 50 | $8.61 \pm 2.29$ | 138.64 | 0.44 |
| | **100** | $7.04 \pm 0.85$ | 249.55 | 0.59 |
| | 200 | $7.45 \pm 0.23$ | 473.59 | 0.93 |
| | 500 | $7.22 \pm 0.19$ | 1166.49 | 4.54 |
| autoenc-SUN | 10 | $362.73 \pm 22.77$ | 54.99 | 0.28 |
| | 50 | $520.85 \pm 29.66$ | 138.61 | 0.43 |
| | 100 | $916.16 \pm 31.58$ | 252.39 | 0.59 |
| | 200 | $327.25 \pm 21.39$ | 477.48 | 0.90 |
| | **500** | $5.31 \pm 0.34$ | 1125.06 | 4.66 |

**Table 5.6:** Results obtained through the use of holistc descriptors based on *places* CNN (layers *'conv4'*, *'conv5'*, *'fc6'*, *'fc7'* and *'fc8'*) to solve visual localization.

| Layer | Size | Error loc. (cm) | Time comp. descriptor (ms) | Time pose est. (ms) |
|---|---|---|---|---|
| conv4 | 169 | $5.03 \pm 0.02$ | 6.64 | 1.62 |
| conv5 | 169 | $5.09 \pm 0.17$ | 6.66 | 0.63 |
| fc6 | 4096 | $5.14 \pm 0.18$ | 7.42 | 34.38 |
| fc7 | 4096 | $16.71 \pm 0.84$ | 8.58 | 33.22 |
| fc8 | 205 | $24.22 \pm 6.44$ | 8.88 | 0.72 |

### 5.5.3 Profound Study of Different CNN Layers to Obtain Robust Holistic Descriptors

Due to the results collected in the last experiment, the present subsection presents an extension regarding the use of different CNN layers to obtain holistic descriptors.

Those descriptors are evaluated to tackle the batch localization task and they are also evaluated against visual effects that may affect the accuracy of the proposed algorithm. Five experiments have been developed according to this topic. This batch of experiments were developed with the Saarbrücken dataset with the aim to confirm the results regardless the environment. Fig. 5.19 shows that the tendency of the descriptors obtained in Saarbrücken is similar than the obtained previously with the Freiburg dataset (see tables 5.4 and 5.6).

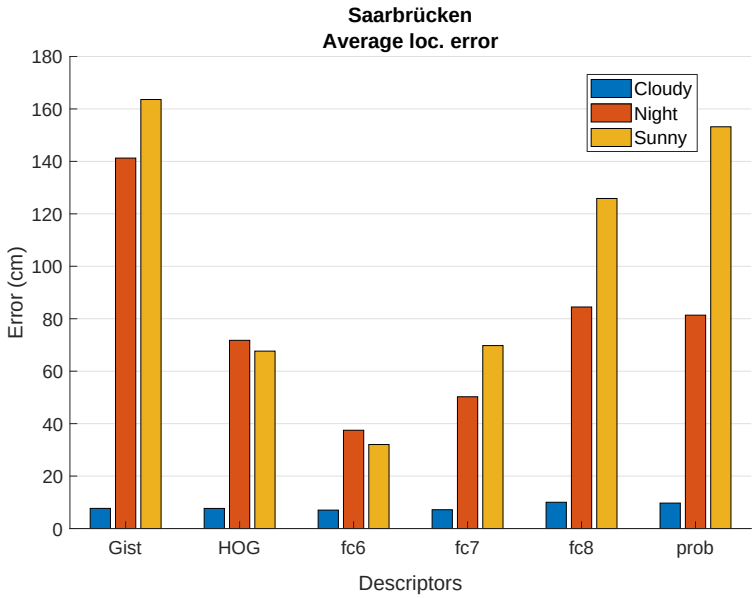



**Figure 5.19:** Average localization error through solving the batch localization in the Saarbrücken environment.

The first experiment extends the **evaluation of layers in the *places* CNN**. In the previous subsection, we evaluated the layers $conv_4$, $conv_5$, $fc_6$, $fc_7$ and $fc_8$. In this experiment we have also evaluated $conv_1$, $conv_2$ and $conv_3$. Additionally, we have also tested the descriptors with images obtained under different illumination conditions.

As we can observe in 5.20, all the convolutional layers provide competent results to carry out the localization task when no illumination changes are presented. Nevertheless, if we take the illumination changes into consideration, a worsen is noticed for most of the layers. For this case, $conv_2$, $conv_3$ and $fc_6$ are the most robust. On the other hand, considering the computing time to estimate the position, we can observe that the first layers provide faster solutions, since the information is extracted from these layers in early stages and the dimension of the descriptors is lower and this leads to a faster computation.

As for the second experiment, **different layers from different pre-trained CNNs** were evaluated. Apart from *places*, AlexNet and GoogleNet are also analyzed

**Table 5.7:** Layers studied for the pre-trained CNNs and size of the descriptors generated.

| CNN | Layer | Size of descriptor |
|---|---|---|
| AlexNet | conv1 | 3025 |
| | conv2 | 729 |
| | conv3 | 169 |
| | conv4 | 169 |
| | conv5 | 169 |
| GoogleNet | conv2-3x3 | 3136 |
| | inception_3a-1x1 | 784 |
| | inception_3b-1x1 | 784 |
| | inception_4a-1x1 | 196 |
| | inception_4b-1x1 | 196 |
| | inception_4c-1x1 | 196 |
| | inception_4d-1x1 | 196 |
| | inception_4e-1x1 | 196 |

to obtain holistic descriptors. The layers proposed for these networks and the size of the descriptor obtained are showed in the table 5.7. From each layer studied, a holistic descriptor was obtained and used to carry out the localization under three illumination conditions. The obtained results are showed in the table 5.8. To summarize, the best solutions obtained for each CNN are shown in the fig. 5.21.

From these results, we can conclude that AlexNet and GoogleNet are also an alternative to obtain holistic descriptors, despite they were initially trained to classify objects. The three networks are similarly affected by illumination changes and the fastest solution among the three networks is presented with *places* using the layer $conv_2$.

Concerning the robustness of the proposed holistic descriptors, experiments 3 and 4 carry out an evaluation about this problem. Experiment 3 evaluates the **noise, occlusion and blur effects** when the HOG, $conv_{2,places}$ and $conv_{4,AlexNet}$ descriptors are used to solve the localization task. On the other hand, experiment 4 evaluates **how the rotation affects the localization**. To tackle both experiments, each effect is applied over each test image from the cloudy test dataset. After that, the holistic descriptor is obtained and the batch localization algorithm is carried out.

The fig. 5.22 shows the results obtained by applying visual effects and random rotations. From fig. 5.22(a), we conclude that noise is the effect that more affects the localization independently the holistic descriptor used. As for the random rotation (see fig. 5.22(b)), the descriptors obtained from the fully connected layers are more robust than the descriptors from convolutional layers. HOG is the descriptor that is more affected by the noise effect but it behaves equally with and without rotation, this is due to the fact that the descriptor is calculated to be invariant against orientation.

Therefore, we conclude that the descriptors obtained from CNNs are not suitable when rotations should be considered, but they are more efficient when the system is prone to present noise.

Last, the fifth experiment carried out was about **varying the type of input image** to obtain holistic descriptors. Since the present experiment, we always have considered panoramic images with grayscale color because the proposed hand-crafted holistic descriptors work with this arrangement. Nevertheless, the use of other image arrangements can be also appropriate. For example, the process carried out to obtain a global-appearance descriptor from layers of AlexNet is the following. First, an RGB omnidirectional image is captured; then, this image is converted to grayscale and transformed to panoramic. After that, the resultant image is adapted to the input of the pre-trained network, that is, the image is replicated 3 times (from grayscale to RGB) and resized to the proper size of the input layer (in this case, $227 \times 227 \times 3$). This process is addressed since the objective is to compare the holistic descriptors obtained from HOG or gist with the layers of the CNNs, but it is also inefficient since abrupt resizing is tackled and color information is lost unnecessarily. Therefore, the aim of this experiment is to evaluate if other input images, which are less inefficient, can provide holistic descriptors suitable to solve the visual localization task.

Fig. 5.23 shows the average localization error obtained for each of the four arrangement evaluated (grayscale/color, panoramic/omnidirectional) by using the best layers and channels of the three studied CNNs. On the whole, we can conclude from this figure that color information provides a similar or lightly worsening. Furthermore, the descriptors obtained from omnidirectional images also perform efficient results. The localization error increases substantially for the *places* CNN, but this increment is lightly for AlexNet and GoogleNet.

From the point of view of the computing time, fig. 5.24 shows the results obtained by using the layer $conv_4$ from AlexNet. By and large, when the input image has color information, the time increases. On the other hand, this figure shows that the results concerned the use of grayscale panoramic images is slightly faster than using omnidirectional images. Nonetheless, this graph only shows the computing time measured since the time when the test image is ready to be used for obtaining the holistic descriptor. Hence, the time to carry out previous pre-treatments are not measured. For example, the time to convert an omnidirectional image to panoramic and the subsequent resizig (to present the same size than the required by the CNN) lasts an average around 2-3 seconds. From this experiment, the conclusion reached is that using omnidirectional images to obtain directly holistic descriptors can results suitable by using pre-trained CNNs such as AlexNet or GoogleNet. Due to the fact that the average localization error keeps the competent results and there is a considerable decrease of the computing time required to tackle the whole localization process.

## 5.6 Conclusion

In this chapter, we present novel localization methods based on the use of omnidirectional images with machine learning tools. This chapter has focused on two main

contributions: the use of machine learning tools to implement more efficient hierarchical localization approaches; and the study of deep learning based methods to extract holistic descriptors with the aim to improve the visual localization task.

Concerning the use of **machine learning tools** to enhance the hierarchical localization, this chapter presents a study about the use of classifiers to solve the rough localization and a regression fit neural network to solve the fine localization step. Both methods are based on using holistic description information. The experiments were carried out with an indoor dataset that contains omnidirectional images. These images present dynamic changes and blur effects and there are also different datasets within the same environment but captured under different illumination conditions (during cloudy days, during sunny days and at night). This work shows that most of the machine learning techniques proposed provide good localization results departing from a compact model. The classifiers have been validated as an efficient tool to perform the rough localization. SVM and shallow neural network classifiers together with holisitc descriptors (*gist* and CNN-fc7) provide high hit ratio to retrieve the corresponding room or area. Additionally, a data fitting neural network was proposed for the fine localization step. Although it does not improve significantly the results obtained by the image retrieval option, it actually works relatively well and robustly for most of the cases. The key to obtain better results would consist in either optimising the training step in the most challenging areas or finding a global-appearance descriptor that suits better the training of the network. Moreover, these techniques (classifiers and data fitting neural network) present robustness against changes of illumination. As for the labelling of the training data for the rough localization step, the localization results are improved in comparison to the ones with are based on the information provided by the ground truth (manual labelling). The last experiment regarding this topic is a comparison between hierarchical localization methods. Whereas the hierarchical localization based on classifiers provides more accurate localization results, the hierarchical localization based on representatives obtained from spectral clustering works faster.

Regarding the study about **deep learning based holistic descriptors**, the present chapter focuses on evaluating the goodness of the proposed methods to carry out the localization (non hierarchical without changes of illumination) task. These novel methods are compared between them and also with hand-crafted holistic descriptors based on analytic methods. Therefore, five global-appearance descriptors have been evaluated: two based on hand-crafted methods (HOG and *gist*), two based on autoencoders and one based on different CNN layers. The size of each descriptor is varied by adjusting some parameters (such as the number of bins in HOG or the size of the hidden representation of the autocoders) or selecting a different layer in the case of CNN. The localization error, the computing time to calculate the descriptor and the computing time to estimate the position of the robot have been used as parameters to measure the efficiency of these descriptors. Fig. 5.25 shows the results obtained for the best configuration of each evaluated descriptor. From the results, we conclude that the minimum localization error is achieved by the CNN-based descriptor option, but the *gist* descriptor and the autoenc-SUN descriptors show results similarly good. The CNN-based descriptor also presents the best option in terms of computing time

to calculate the descriptor. Nevertheless, in terms of time to estimate the pose of the robot, HOG is the fastest.

As for the use of autoencoders, using an autoencoder trained with images that belong to the environment of work allows a localizaiton algorithm with good-enough accuracy results. The general autoencoder trained with general images from different environments also works acceptably in the case of high size of hidden representation, but this leads to high computing times. Therefore, its use as tool to obtain holistic descriptors for panoramic images would be valid and the advantage of this method is that the autoencoder is trained just once, then the tool is suitable independently the environment.

Concerning the use of holistic descriptors based on CNN, the present chapter has proved that the convolutional 2D layers from the first part of the architecture perform very interesting descriptors despite they are not fully connected layers, which are the layers proposed typically in previous works. These descriptors have produced the optimal localization solutions among all the methods evaluated: size of descriptor relatively small, which leads to fast times to estimate the position; low computing time to calculate the descriptor; and very accurate localization (around 5 cm for a test dataset with visual information every 4 cm over a training dataset with visual information every 20/40 cm).

Furthermore, an extension of the experiments has showed that other pre-trained CNNs can be also used to obtain reliable global-appearance descriptors independently the original purpose of the notwork. This descriptors have proved to be more robust against visual effects (such as blur or occlusions) than the hand-crafted methods like HOG. Nevertheless, CNN-based descriptors are not robust against rotational changes. In this way, considerable problems appear when convolutional layers are used. In this case, if the localization task must take into consideration rotation, fully connected layers are more suitable, since the descriptors obtained from those layers are robuster and keep a competent enough average localization error. Last, this extension has also proved that omnidirectional images can be used directly to obtain holistic descriptors. In this way, the localization error is lighly affected, but the whole process is substantially faster.

## 5.7 Publications Related to this Chapter

The main results presented in this chapter are related to the following publications:

- S. Cebollada, L. Payá, A. Peidró, L.M. Jiménez, O. Reinoso. Estudio de descriptores holísticos basados en métodos analíticos y técnicas de deep learning para localización con robots móviles. In Libro de Actas de las Jornadas Nacionales de Robótica 2019. Ed. CEA ISBN:978-84-09-12133-5 - pp. 1-8 [42]

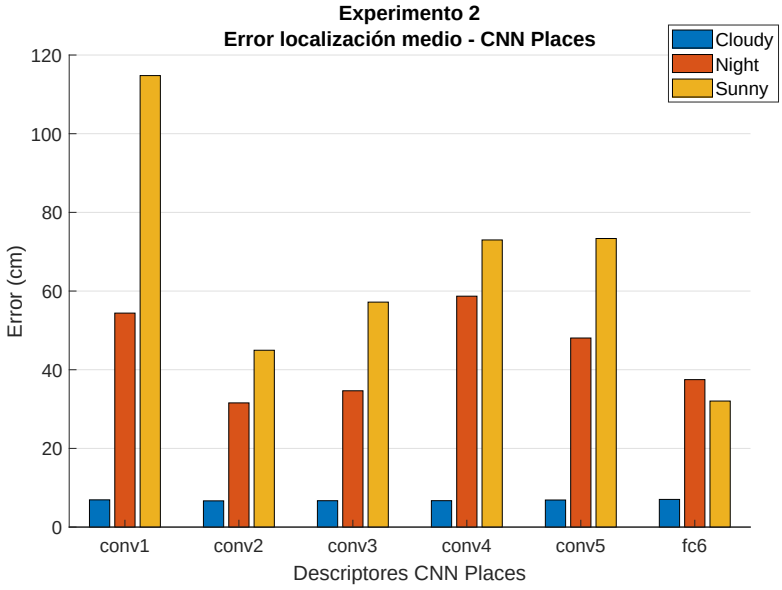
  - This paper proposes the use of different deep learning tools to obtain holistic descriptors. These techniques are evaluated under different configurations with the aim to obtain the most successful localization accuracy. This work proposes the use of convolutional 2D layer to obtain global-appearance descriptors. This layers are present in the first layer of the CNN architecture and its use has been scarcely proposed. On the other hand, autoencoders are presented as a possible tool to create global-appearance descriptors to optimize the information provided from panoramic visual images.

- S. Cebollada, L. Payá, D. Valiente, X. Jiang, O. Reinoso. An evaluation between global appearance descriptors based on analytic methods and Deep Learning techniques for localization in Autonomous Mobile Robots. In 2019 16th International Conference on Informatics in Control, Automation and Robotics, volume 2, pages 284-291, 2019. Ed. INSTICC ISBN:978-989-758-380-3 ISSN:2184-2809 [237]

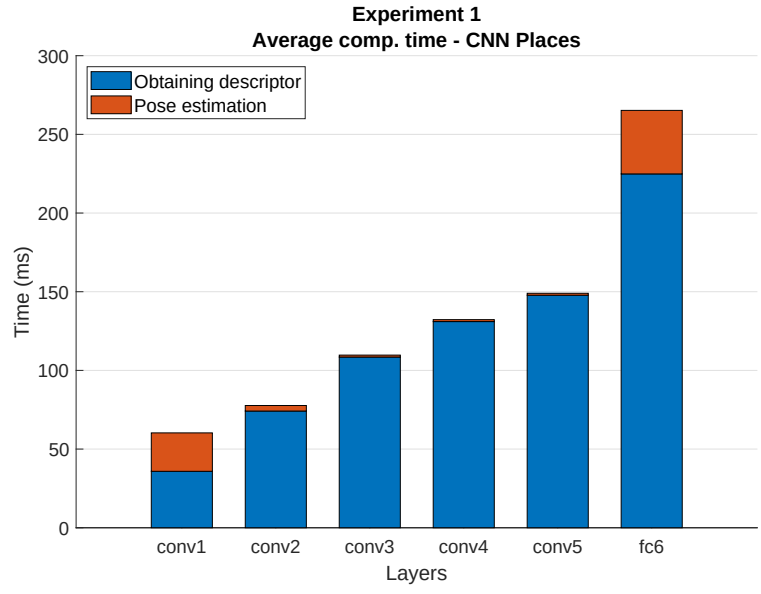
  - This paper presents an evaluation between different holistic descriptors to carry out the localization task. The unique information source used to solve this issue is an omnidirectional camera. The visual localization task is solved by means of a image retrieval problem. The global-appearance descriptors are obtained by using deep learning and they are compared between them and also with hand-crafted methods.

- S. Cebollada, V. Román, L. Payá, M. Flores, L.M. Jiménez, O. Reinoso. Uso de técnicas de machine learning para realizar mapping en robótica móvil. In Actas de las XL Jornadas de automática, pages 686-693. Ed. CEA-IFAC ISBN:978-84-9749-716-9 [46]

  - This paper presents a study about different classifiers based on machine learning to carry out the mapping and localization tasks. These classifiers are proposed to solve the rough localization task within a hierarchical visual model. After estimating the correct area, the fine localization is solved by means of an image retrieval problem. Holistic descriptors obtained from omnidirectional images is used in this work.




(a)

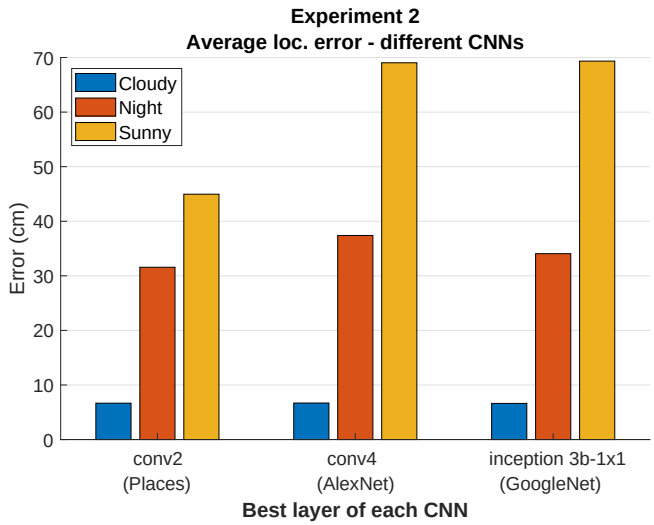



(b)

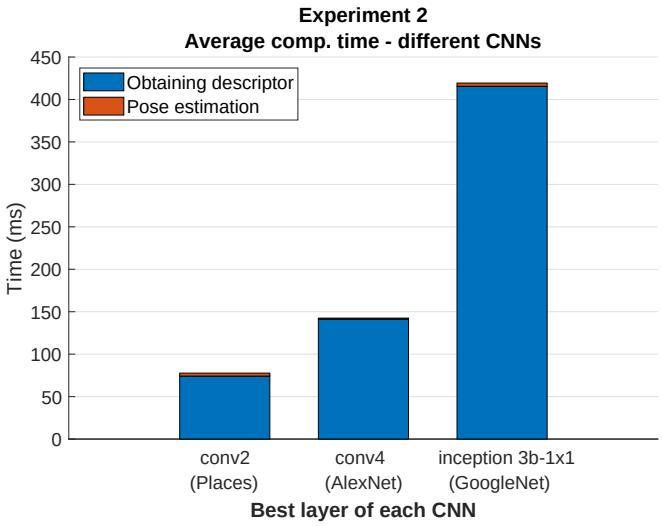
**Figure 5.20:** (a) Average localization error and (b) average computing time to carry out the batch localization through using holistic descriptors obtained from different layers of the *places* CNN.

**Table 5.8:** Localization results using AlexNet and GoogleNet to obtain holistic descriptors. Average localization error (cm), average computing time to calculate the descriptor and average computing time to estimate the pose.

| CNN | Layer | Illum. cond. | Avg. loc. error (cm) | Comp. time descriptor (ms) | Comp. time est. pose (ms) |
|---|---|---|---|---|---|
| AlexNet | conv1 | Cloudy | 6.87 + 0.55 | 37.06 | 24.81 |
| | | Night | 89.88 + 311.65 | 38.16 | 25.45 |
| | | Sunny | 133.79 + 482.02 | 44.42 | 31.18 |
| | conv2 | Cloudy | 6.70 + 0.52 | 80.53 | 3.80 |
| | | Night | 47.57 + 124.38 | 192.85 | 9.55 |
| | | Sunny | 84.85 + 382.29 | 101.63 | 5.07 |
| | conv3 | Cloudy | 6.72 + 0.52 | 117.92 | 1.30 |
| | | Night | 33.69 + 14.06 | 144.99 | 1.75 |
| | | Sunny | 135.15 + 568.18 | 134.08 | 1.65 |
| | conv4 | Cloudy | 6.69 + 0.52 | 141.15 | 1.29 |
| | | Night | 37.39 + 31.59 | 156.18 | 1.45 |
| | | Sunny | 69.03 + 278.04 | 146.37 | 1.34 |
| | conv5 | Cloudy | 6.73 + 0.53 | 155.23 | 1.25 |
| | | Night | 91.39 + 428.48 | 181.76 | 1.78 |
| | | Sunny | 70.24 + 273.79 | 239.46 | 2.34 |
| GoogleNet | conv2-3x3 | Cloudy | 6.72 + 0.52 | 364.70 | 25.69 |
| | | Night | 42.84 + 68.89 | 411.18 | 29.26 |
| | | Sunny | 94.21 + 330.58 | 412.52 | 30.90 |
| | inception_3a-1x1 | Cloudy | 6.68 + 0.52 | 379.34 | 3.94 |
| | | Night | 39.89 + 73.35 | 429.55 | 4.86 |
| | | Sunny | 62.11 + 176.05 | 424.88 | 4.76 |
| | inception_3b-1x1 | Cloudy | 6.62 + 0.50 | 415.54 | 3.78 |
| | | Night | 34.05 + 19.03 | 444.17 | 4.17 |
| | | Sunny | 69.33 + 380.54 | 494.77 | 4.94 |
| | inception_4a-1x1 | Cloudy | 6.75 + 0.52 | 489.85 | 1.38 |
| | | Night | 81.78 + 420.44 | 536.61 | 1.44 |
| | | Sunny | 64.24 + 270.90 | 504.90 | 1.44 |
| | inception_4b-1x1 | Cloudy | 6.70 + 0.52 | 502.94 | 1.36 |
| | | Night | 53.95 + 155.64 | 570.13 | 1.52 |
| | | Sunny | 91.93 + 430.71 | 503.60 | 1.35 |
| | inception_4c-1x1 | Cloudy | 6.79 + 0.53 | 528.95 | 1.34 |
| | | Night | 42.86 + 53.75 | 625.20 | 1.56 |
| | | Sunny | 65.39 + 253.96 | 529.26 | 1.34 |
| | inception_4d-1x1 | Cloudy | 6.71 + 0.53 | 570.26 | 1.38 |
| | | Night | 41.35 + 72.66 | 644.48 | 1.59 |
| | | Sunny | 51.99 + 144.04 | 559.64 | 1.36 |
| | inception_4e-1x1 | Cloudy | 6.80 + 0.54 | 608.35 | 1.36 |
| | | Night | 51.59 + 95.07 | 688.28 | 1.64 |
| | | Sunny | 112.96 + 561.73 | 598.00 | 1.34 |

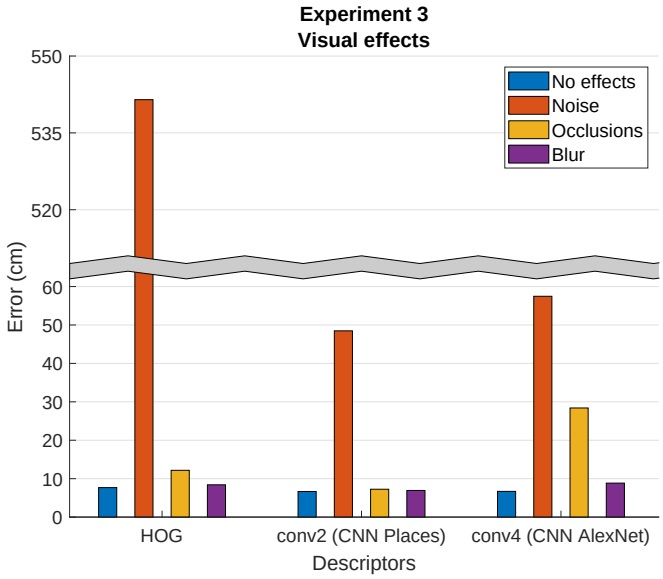



(a)

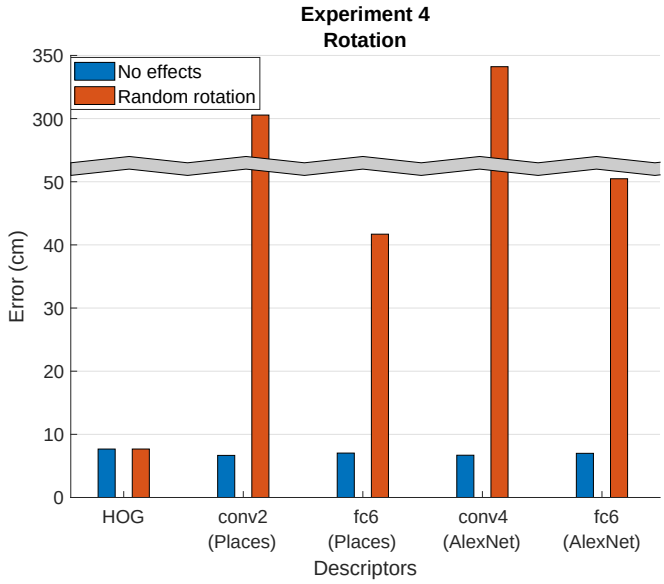



(b)

**Figure 5.21:** (a) Average localization error and (b) average computing time to carry out the batch localization through using holistic descriptors obtained from different layers of the pre-trained *places*, AlexNet and GoogleNet CNNs. This figure shows the best result obtained for each network.




(a)




(b)

**Figure 5.22:** Average localization error when the batch localization is carried out with (a) test images with noise, occlusions, blur effects and (b) with/without random rotations.

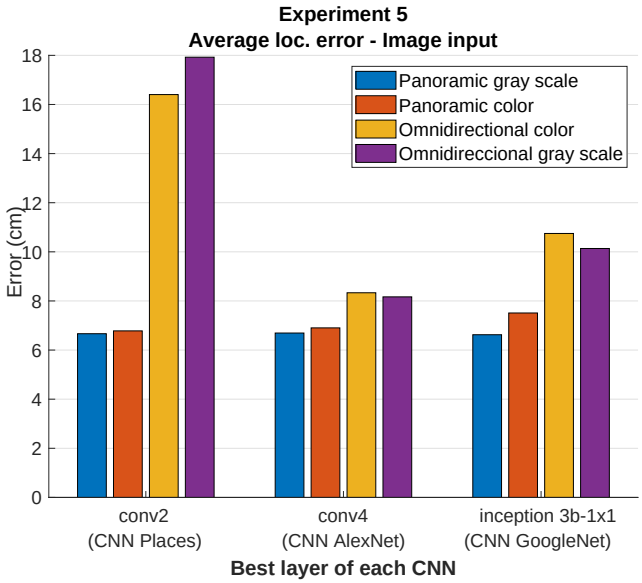



**Figure 5.23:** Experiment 5. Average localization error when the batch localization is carried out with grayscale panoramic, color panoramic, grayscale omnidirectional and color omnidirectional images.

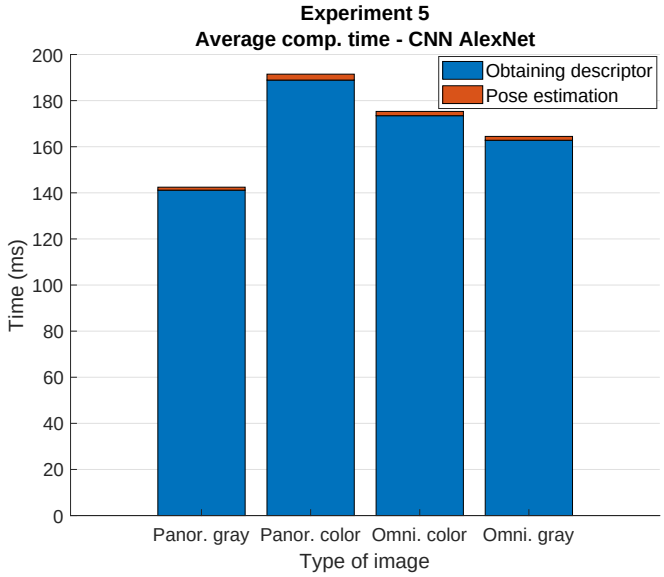



**Figure 5.24:** Experiment 5. Average compting time when the batch localization is carried out with descriptors obtained from AlexNet by using grayscale panoramic, color panoramic, grayscale omnidirectional and color omnidirectional images.

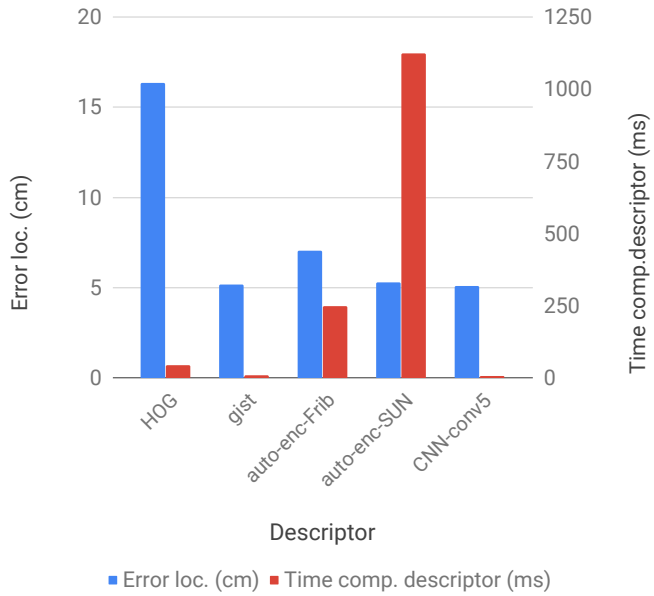



**Figure 5.25:** Summary of the best configuration for each descriptor studied.

# 6 Transfer Learning, Networks Creation and Training by Modifying Layers

## 6.1 Introduction

Continuing the present research line, this chapter focuses on the use of holistic descriptors to tackle the localization task. Regarding the map creation, we have demonstrated in previous chapters that arranging the information hierarchically is an efficient alternative to tackle the localization task. These chapters proved the effectiveness of hierarchical maps to solve the localization problem departing from global-appearance descriptors obtained from omnidirectional images. Furthermore, this localization task has also been tested successfully under changes of illumination.

As presented in chapter 5, during the past few years, many contributions have proposed the use of artificial intelligence techniques to address computer vision and robotics problems. For instance, regarding machine learning tools, Gonzalez *et al.* [88] use machine learning to detect different levels of slippage for robotic missions in Mars; Dymczyk *et al.* [70] present the use of a boosted classifier to classify landmark observations and carry out the localization task in a more robust fashion. Meattini *et al.* [176] propose a human-robot interface system based on electromyography sensors and through merging pattern recognition and factorization techniques, the robot learns the optimal hand configuration for grasping.

The present chapter focuses on the use of deep learning tools for mobile robotics applications. This way, a pre-trained convolutional neural network is developed to address the room retrieval task. This CNN is trained wit omnidirectional images obtained from the Freiburg (indoor) environment. This work also proposes using the trained CNN to carry out a novel hierarchical localization method. This localization method used

holistic descriptors extracted from intermediate layers with the aim of carrying out a fine localization, i.e., retrieving the position where the image was captured within the retrieved room. The robot estimates its position within the selected room/s through an image retrieval algorithm by comparing the obtained global-appearance descriptor with the visual model contained in the retrieved room.

### 6.1.1 Deep Learning Tools

Deep learning is a subfield of machine learning that has gained much interest recently, due to the improvements in processing systems. This technique basically consists in learning directly from a data set and their expected outputs (or correct labeling) by using layers of increasingly meaningful representations [52]. A number of recent works use such techniques in the field of robotics. For instance, Lenz *et al.* [149] propose a deep learning approach to solve the problem of detecting robotic grasps in a scene which contains objects; Levine *et al.* [151] trained a convolutional neural network for robotic grasping from monocular images through learning a hand-eye coordination; Shvets *et al.* [251] use deep learning segmentation to distinguish between different surgical instruments regarding Robot-Assisted Surgery. As for mobile robotics, Zhu *et al.* [320] propose deep reinforcement learning to address target-driven visual navigation.

These methods try to construct automatically high level data models through using matrix data and architectures that allow linear, non-linear, multiple and iterative transformations [20]. The idea is to train the architecture to reach a model that is capable of creating representations which best define the inputs. Many deep learning tools have been applied in a variety of fields such as computer vision [32], speech recognition [142] or audio-visual recognition [3] and they have proved to perform state-of-the-art results for many tasks.

Therefore, the present chapter is based on the use of deep learning to solve computer vision and robotics tasks. Specially, this work solves the visual localization problem by using convolutional neural networks (CNNs). This tool has been successfully used to solve computer vision applications such as face recognition [296], object detection [83] or and self-driving car to efficiently extract patterns and structural information from input images [51]. A wide review can be found in the work presented by Voulodimos *et al.* in [289].

As it is explained in [289], CNNs are inspired by the human visual system. They consist of local connections between neurons hierarchically organized that successively process and transform the image. Basically, CNNs are composed by three types of layers: convolutional layers, pooling layers and fully connected layers. Every layer transforms the input and generates an output according to the parameters established. This process is tackled throughout several layers until reaching the last layer, which is a fully connected layer that usually outputs a 1D feature vector. There are very well known CNNs whose architectures have been used as starting point to develop new computer vision tasks. For instance, AlexNet [136], which consists of eight layers (five convolutional layers and three fully connected layers) with a final 1000-way softmax and it is trained to classify into 1000 object categories, such as keyboard, pencil, and

a variety of animals. Another example is GoogLeNet [268], which presents 22 layers and it is also trained for object classification. Nevertheless, this network uses 12 times fewer parameters than AlexNet. A wide review of the more outstanding CNNs can be found in [206].

Regarding the use of CNNs to solve mobile robotics tasks, there are many works that have proved success by using this technique. For instance, Sinha *et al.* [257] propose a CNN to process data from a monocular camera and tackle an accurate robot re-localization in GPS-denied indoor and outdoor environments. Wozniak *et al.* [303] use a transfer learning technique to retrain a CNN to classify places among 16 rooms, in which the images are acquired by a humanoid robot. More recently, Chaves *et al.* [49] propose a CNN to build a semantic map. Concretely, they use the network to detect objects in images and after that, the results are placed within a geometric map of the environment.

The existing pre-trained networks were trained on images from many different classes, which may not include appropriate categories for specific application. There are other options that permit reusing existing CNNs for applications which are different from the application the CNN was trained for. The **transfer learning** technique basically consists in the process of retraining a pre-trained network to adapt it to a new task with a new set of images, that is, reusing the architecture, weights and parameters of a CNN which already works properly as starting point to obtain a new CNN with a different purpose. The main idea is to get profit of most of the intermediate layers, because their parameters have been tuned with a great number of images and contain useful information. The problem, then, is reduced to change the final layers (in order to re-adapt them to the new task proposed). Once the "new" network architecture is established, the training process starts through using the new input data. The components needed for transfer learning are: pre-trained network layers, training data, and algorithm options. This technique can save a huge amount of time for training and may lead to better results than creating a new network from scratch. This idea has been used by many authors. For example, Han *et al.* [97] use CNN transfer learning together with data augmentation and obtain good solutions despite the small size of the datasets used to address image classification. Also, as previously mentioned, Wozniak *et al.* [303] use the transfer learning technique to retrain the VGG-F network and retrain it to classify places among 16 rooms with images acquired by a humanoid robot. Mahendran *et al.* [169] propose a pose estimation problem with a CNN regression framework.

Furthermore, many authors have proposed the use of intermediate layers to **extract holistic descriptors**. In this way, once the network is properly available to face the desired task, the intermediate layers perform vector description that can be used to characterize the input data. This idea has already been used by some authors such as Arroyo *et al.* [8], who use a CNN that automatically learns to generate descriptors which are robust against changes of seasons in order to carry out a topological localization. Wozniak *et al.* [303] also use the feature extracted by the $fc_6$ layer to train a linear SVM (Support Vector Machine) classifier. Mancini *et al.* [170] use this visual information to carry out place categorization with a Naïve Bayes classifier. Payá *et*

*al.* [213] propose CNN-based descriptors to create hierarchical visual models for mobile robot localization. Moreover, in chapter 5, we tackle an exhaustive evaluation of holistic descriptors obtained from different layers of pre-trained CNNs to address the localization task in indoor environments and we also compare the obtained results with the results through using global-appearance descriptors based on analytic methods.

In summary, CNNs are a very popular tool to develop optimal solutions in mobile robotics since they significantly improve traditional machine learning tools along with computer vision. However, CNN present the downside of the training process, since these networks have to deal with a large number of parameters to learn properly and a high number of training samples is usually necessary. Additionally, over-fitting problems are common to happen, unless a varied and representative set of training data is available. To solve this issue, techniques such as stochastic pooling, dropout or data augmentation can be conducted.

Therefore, the aim of the present chapter is to create a CNN that is able to distinguish between different rooms from an indoor environment in order to estimate correctly in which room the robot currently is. Like in previous chapters, the unique source of information used to carry out mapping and localization is the set of images obtained by an omnidirectional vision sensor installed on the mobile robot within an indoor environment under real-operation conditions. Our main contributions in this work are summarized as follows.

1. We present a study about the use of a CNN classifier to retrieve the room where an input image was obtained.
2. We evaluate the use of different intermediate convolutional layers from the trained CNN to obtain holistic descriptors and use them to address the localization task.
3. We study the use of the proposed deep learning approach to solve the localization in different environments.
4. We propose an algorithm that considers the likelihood information provided by the CNN with the aim to strengthen the localization task.
5. We propose CNNs based on omnidirectional images for room retrieval and for position retrieval.

The remainder of the paper is structured as follows: Section 6.2 outlines the deep learning tool used along this chapter. After that, section 6.3 explains the use of the convolutional neural network to carry out the localization task and section 6.4 presents all the experiments which were tackled to test the validity of the proposed methods to solve the localization. Last, section 6.5 outlines the conclusions reached.

## 6.2 The Convolutional Neural Network

As mentioned in sec. 6.1, the aim of this work is to carry out the hierarchical localization task in indoor environments by using CNNs. To solve this task, only omnidirectional visual information is used. In this kind of environments, some adverse effects may occur which negatively affect the images captured, such as blur effect dynamic changes within the environments due to either furniture changes or occlusions caused by people walking. Additionally, the changes of illumination are very common in mobile robotics tasks, since the illumination conditions may vary depending on the moment of the day or the weather conditions. Hence, the method developed must be robust against these issues. Furthermore, the goodness of the localization method depends directly on the accuracy and the computing time, this is, the method must be capable of estimating the current position of the robot with the maximum possible accuracy and in the minimum amount of time.

To address these requirements, we propose using a CNN with the aim of solving the visual localization through hierarchical maps. This idea has been proposed in works such as [213] and also in chapter 5. Nevertheless, these previous works used pre-trained CNNs to calculate holistic descriptors. In this case, CNN are firstly re-trained with the images obtained from the environment and after that, the resultant CNN is used for room retrieval (classification task) as well as for desrcriptor extraction. Basically, it consists in developing a CNN that is able to solve the following tasks:

1. Estimating the room in which the robot captured the image.
2. Providing a global-appearance descriptor from an intermediate layer.
3. Using this information, to estimate the position of the robot within the estimated room more accurately.

This process will be explained in detail in section 6.3. Therefore, a CNN classification should be developed to estimate the room in the environment.This tool basically consists of classifying the given classes or categories of input data (in this case, images). Labels (also known as targets) represent the categories of the environment. Before using this tool for classification, the model requires training with a wide variety of input data ($x_{train}$) and their corresponding labels ($y_{train}$), thus the model carries out an approximation of a mapping function from input variables to discrete output variables until achieving a well tuning configuration. Then, the CNN is ready to receive new data ($x_{test}$) and estimate their categories ($y_{estimated}$).

Throughout this section, the CNN developed is presented in detail. The remainder of this section is structured as follows. Subsection 6.2.1 presents the dataset used to train the CNN, subsection 6.2.2 outlines the architecture and training method used to create the network. Last, subsection 6.2.3 details the data augmentation conducted to train the CNN.

### 6.2.1 The Dataset

Once again, the set of images used to carry out this work is the Freiburg dataset, obtained from the COLD dataset [222]. These images have been used both to train the CNN and to carry out the experiments. The majority of the experiments developed in this chapter were addressed with the panoramic version of the originally omnidirectional images, then, before using them, a conversion from omnidirectional to panoramic images is tackled, since the present work aims at establishing a continuation from the previous works. For example, one of the aims of this work is to compare the global-appearance descriptors obtained from the CNN with the hand-crafted description methods that depart from panoramic images. Furthermore, the design of a CNN based on panoramic images constitutes an interesting option, because this type of networks are commonly based on conventional images, hence, this CNN can be used for future similar works based also on panoramic images. Nonetheless, section 6.4 also presents experiments which are based on omnidirectional images with the aim of confirming the suitability of the CNNs to obtain characteristic information from omnidirectional images.

Like in previous chapters, the dataset obtained under cloudy illumination conditions is used as training dataset and it is downsampled to obtaining a dataset with an average distance of around 20 cm between consecutive images. The resultant dataset (training dataset) is used to train the CNN and it is also considered to establish the visual model for later localization. Additionally, three test datasets are again proposed: cloudy, sunny ad night datasets. Apart from using the Freiburg dataset, some extra evaluations are carried out with the Saarbrücken dataset, which is also contained in the COLD dataset. This dataset is used to evaluate the effectiveness of using the Freiburg CNN to obtain holistic descriptors in different environments. The training and test datasets are obtained in the same way: downsampling the cloudy dataset to obtain the training dataset and storing the discarded images to obtain the cloudy test dataset.

### 6.2.2 The Architecture and Training

Building and training a network from scratch can achieve reasonably good results, but it requires a lot of effort: experience with network architectures, a huge amount of training data and a considerable computing time. Using a pre-trained network such as AlexNet or GoogLeNet for transfer learning eases considerately the starting point. This is an interesting technique that can save a lot of training time and even get better results from creating a new network from scratch. It basically involves reusing the architecture and parameters of a CNN as a starting point to build a new CNN for a different purpose. Notwithstanding that, transfer learning works only if no early layers need to be modified, because the downstream architecture and parameters are no longer valid. Otherwise, transfer learning can not be used and training the parameters of the network from scratch is necessary. In this case, we use the AlexNet architecture (whose input size is $227 \times 227 \times 3$ ) and it must be resized to $120 \times 512 \times 3$, hence, the downstream and parameters are no longer valid.

Creating a complete network architecture is complex, so instead of trying to build an architecture from scratch, the present work proposes to use the AlexNet

architecture and follow a process similar to transfer learning (starting with pre-existing architectures) , but starting from scratch with the parameter setting.

We propose the use of AlexNet as CNN architecture to carry out the training task to develop our classification network because this architecture has been successfully used in previous works to develop new classification tasks (such as [97]) and its simplicity permits fast training. In this case, the last three layers need to be replaced to adapt the network to the desired classification task. These layers are: fully convolutional layer ($fc_8$), softmax layer and classification layer. These layers are replaced to carry out the desired classification (classification into one of the 9 rooms that belong to the Freiburg environment). Additionally, since the input layer of AlexNet was configured to receive $227 \times 227$ images and our panoramic datasets are composed by $128 \times 512$ images, the input layer is also replaced. Resizing the input panoramic images would avoid starting from scratch the training, but the resizing would be quite abrupt for this type of images and it would affect the resolution of the image. Additionally, a CNN based on panoramic images constitutes an interesting option.

After these changes of the original CNN, the network is ready to be trained with the training set of panoramic images. We trained the CNN off-line on NVIDIA GEFORCE GTX 1080TI ®GPU system. The training time was around 4 hours. After every 30 epochs, the performance of the partially trained network was evaluated by using the data for validation. The first training departs from the modified version of AlexNet. Once the first training is finished, the network obtained is used as departing network in the following training with a modification of the training parameters, i.e., for transfer learning. The idea is to conserve the architecture but to continue the tuning of the parameters involved in the intermediate layers. The fig. 6.1 shows the architecture used throughout this work.

### 6.2.3 Data Augmentation

A large training dataset is crucial for the performance of the deep learning model. Nevertheless, sometimes, the training dataset available is smaller than required and then, the deep model can not be properly trained to reach the desired solution. In order to solve this issue, the data augmentation technique has been proposed as a method to improve the performance of the model by augmenting the number of training instances and preventing overfitting. Data augmentation basically consists in creating new 'data' by applying different effects over the original images. Some authors have already used data augmentation to solve their deep learning tasks. For example, Guo and Gould [93] used data augmentation to improve a CNN training to solve a object detection task, Ding *et al.* [63] proposed three data augmentation methods to carry out a SAR target recognition in order to make the CNN robust against target translation, speckle variation in different observations, and pose missing. Salamon and Bello [241] propose audio data augmentation for overcoming the problem of environmental sound data scarcity and then create a CNN to classify these data. Moreover, Perez and Wang present in [217] a work about the effectiveness of the data augmentation to solve the classification by means of deep learning.

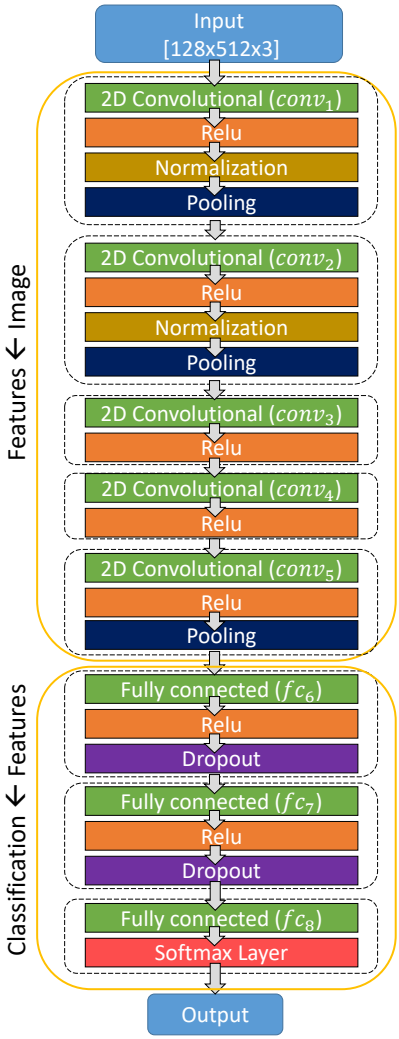



**Figure 6.1:** The CNN architecture created from the AlexNet architecture. The input layer is replaced to receive images with a size of $[128 \times 512 \times 3]$ and the last three layers ($fc_8$, softmax and the output classification layer) are also replaced to adapt the network to the proposed task, the room retrieval task.

Regarding the present work, the data augmentation proposed consists in applying visual effects over the original images from the training dataset. The effects applied are those that can actually occur when images are captured in real operating conditions:

- **Rotation**: A random rotation between 10 and 350 degrees is applied over the omnidirectional image, which implies a horizontal shift of the panoramic image.

- **Reflection**: The panoramic image is horizontally reflected.
- **Brightness**: The low intensity values are re-adjusted (increased) in order to create a new image brighter than the original one.
- **Darkness**: The high intensity values are re-adjusted (decreased) in order to create a new image darker than the original one. No image is applied brightness and darkness effects at the same time.
- **Gaussian noise**: White Gaussian noise is added to the image.
- **Occlusion**: This effect simulates the cases when some parts of the picture are hidden either by some parts of the sensor setup, or some event (such as a person who is in front of an object). This effect is applied by introducing geometrical gray objects over random parts of the image.
- **Blur effect**: This effect occurs when the image is captured while the camera is moving (the image is blurred).

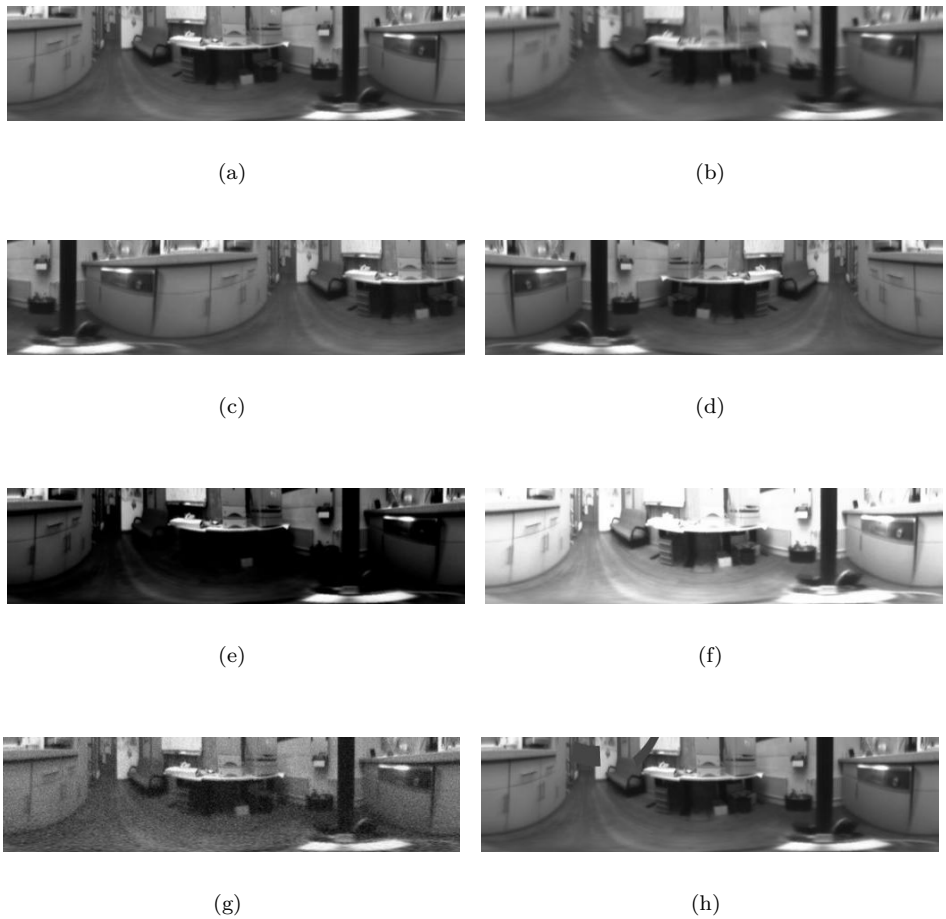
The fig. 6.2 shows some examples of the effects applied over a training image. The first image is the original one, obtained directly from the original training dataset, the rest of the images are the original but with a visual effect over it. Departing from the original training dataset, which contains 519 images, the data augmentation is applied and either none, one, or more than one effects are simultaneously applied (except for the bright and dark effects, which are never applied at the same time over an image). Hence, the total number of training images is enlarged to 49824 images.

## 6.3 Localization Using Deep Learning

### 6.3.1 Visual Description Methods. Batch Localization as an Image Retrieval Problem

As it was already said in previous chapters, due to the emergence of the deep learning techniques, some authors have proposed during the last few years CNNs to generate holistic descriptors with the activations of the intermediate layers. This method was proposed in detail in previous chapters (chapter 4 and chapter 5) to create hierarchical visual models. In those chapters, the original pre-trained CNN is only used with the purpose of obtaining global-appearance descriptors from the input images.

Concerning the way to address the localization task, the present work proposes a batch localization method by using the trained CNN to calculate global-appearance descriptors from the intermediate layers. The process is as follows: given a dataset of training images $(im_i(x,y) \in \mathbb{R}^{N_x \times N_y},\ i = 1, ..., N_{Train})$, a description method based on deep learning or hand-crafted methods is used to obtain the set of training descriptors, $D_{train} = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{N_{train}}\}$ where each descriptor is $\vec{d}_i \in \mathbb{R}^{l \times 1}$ and corresponds to the image $im_i$. These descriptors can be considered as a straightforward model of the environment. After that, given a test image $im_{test}$, the holistic descriptor



(a) (b) (c) (d) (e) (f) (g) (h)

**Figure 6.2:** Example of data augmentation. (a) Original image captured within the Freiburg environment. One effect is applied over each image: (b) blur effect, (c) random rotation, (d) reflection, (e) darkness, (f) brightness, (g) Gaussian noise, (h) occlusion.

is calculated ($\vec{d}_{test}$) and the batch localization is solved as an image retrieval problem. Fig. 6.3 shows the diagram regarding the batch localization process.

In previous chapters, the holistic description methods HOG and *gist* were used successfully to solve the mapping and localization tasks by means of panoramic images. Hence, in order to establish a comparison between analytic and deep-learning-based methods, before calculating the proposed descriptors, a conversion from omnidirectional to panoramic images is carried out. Also, a new strategy has been recently proposed by works such as [170] to obtain holistic descriptors from pre-trained CNNs. To sum up, this chapter proposes a comparison between holistic descriptors obtained from different

layers of the CNN developed and also presents a comparison with analytic methods (*gist* and HOG) and descriptors obtained from layers of a well-known CNN (AlexNet).

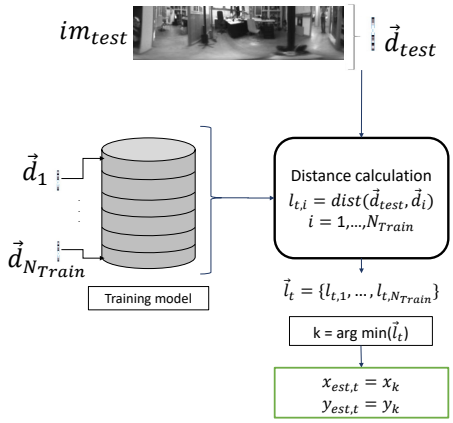



**Figure 6.3:** Batch localization diagram. The test image $im_{test}$ is compared with the descriptors obtained from the training model $D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{N_{Train}}\}$. The most similar descriptor $\vec{d}_k$ is retained and the position of the test image is estimated as the position where most similar image was captured.

### 6.3.2 Using a CNN to Learn a Model of the Environment, Room Retrieval and Image Description

The chapter 5 has shown the advantages of using a CNN to obtain holistic descriptors of the input image from intermediate layers. Their use can improve the results obtained for localization. Therefore, the present chapter goes one step ahead and proposes to use the CNN as a hierarchical model with the aim of:

1. Addressing the rough localization as a room retrieval problem (high-level layer) departing from the test image.
2. Using the likelihood information to optimize the rough step.
3. Obtaining holistic descriptors from the input images.

The descriptors of the training images will form the low-level layer, and they allow to solve a fine localization as an image retrieval problem, with the holistic descriptors of the test images (also obtained from the CNN).

As for the process to obtain the global-appearance descriptors by using the CNN, this is as follows.

1. The CNN is trained with the images from the training dataset (including data augmentation).

**Table 6.1:** Size of each descriptor obtained from the different layers of the CNN.

| Layer | Size of descriptor |
|---|---|
| $conv_4$ | 180 |
| $conv_5$ | 180 |
| $fc_6$ | 4096 |
| $fc_7$ | 4096 |
| $fc_8$ | 9 |

2. Once the CNN is trained, a test image $im_{test}$ is introduced into the CNN.

3. The holistic descriptors are obtained from different layers.

Fig. 6.4 shows the process to obtain the holistic descriptors from the CNN. On the one hand, concerning the descriptors extracted from the output of the $conv_4$, and $conv_5$ (2D convolutional) layers, they are calculated by selecting a channel from the layer and arranging the generated data (matrix) in a single column (vector). To establish the optimal channel per convolutional layer, previous experiments are carried out and afterwards, the same channel is used for all the experiments developed. On the other hand, in the case of $fc_6$, $fc_7$ and $f_8$ (fully connected) layers, the output is directly the vector used as descriptor. Table 6.1 summarizes the size of each descriptor calculated from the CNN.

Regarding the hierarchical localization, the method analyzed in chapter 4 consists basically in calculating the nearest neighbor method in two layers. Hence, for the high-level layer, the visual descriptors are grouped according to their similitude and a representative descriptor $R = \{\vec{r}_1, \vec{r}_2, ..., \vec{r}_{n_g}\}$ is obtained for each group, where $n_g$ is the number of groups. Afterwards, in order to solve the localization task, a new image is obtained $im_{test}$ and its holistic descriptor is calculated $\vec{d}_{test}$. This descriptor is compared with all the representatives $R$ and the most similar representative $\vec{r}_k$ is retained (rough localization step); after that, a new comparison is carried out between $\vec{d}_{test}$ and the descriptors contained in the group *k*, $D_k = \{\vec{d}_{k,1}, \vec{d}_{k,2}, ..., \vec{d}_{k,N_k}\}$ and, last, the position of the image $im_{test}$ is estimated as the position where the most similar image in the k-th group was captured (fine localization step).

Therefore, despite using the CNN to only extract global-appearance descriptors and addressing the batch localization, this tool can be also used as a hierarchical localization method. In this way, due to the fact that the CNN is trained as a classifier to retrieve the room within the whole environment, this can be proposed to address the rough localization, i.e., to retrieve the position in the high-level layer.

To sum up, the idea is to build a deep learning tool that, apart from retrieving the room where the image was captured, is also able to provide a global-appearance descriptor that characterizes the image better than the analytic methods proposed in the state of the art.

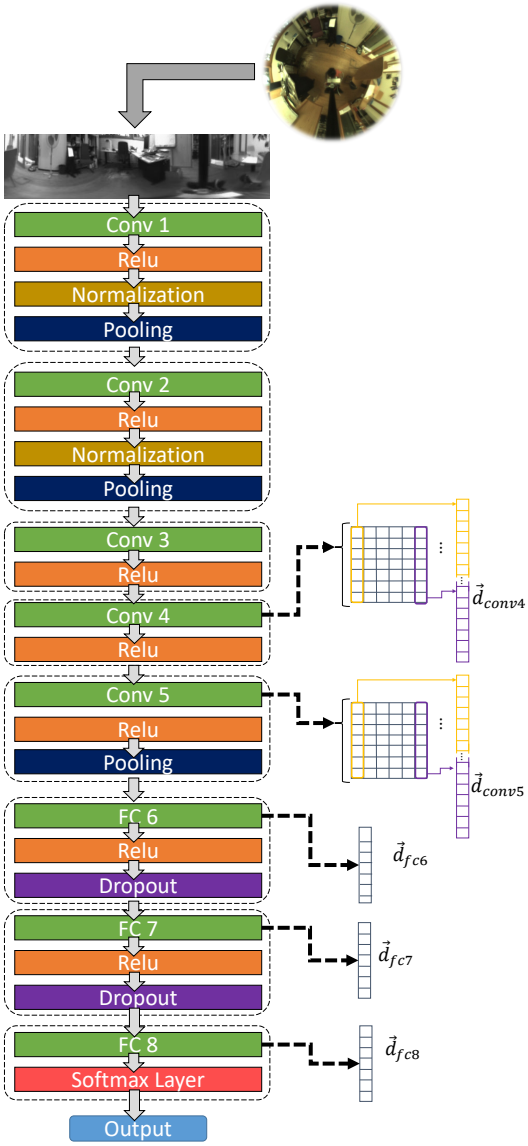



**Figure 6.4:** Diagram that shows the extraction of the global-appearance descriptor from the intermediate layers of the CNN. The architecture is inherited from AlexNet and the network was trained to retrieve the room within the Freiburg dataset.

Once the CNN is properly trained to classify, it will be capable of solving the rough localization step. Regarding the use of the CNN to solve the fine localization step (the room retrieval), this chapter proposes to use the intermediate layers $conv_4$,

$conv_5$, $fc_6$, $fc_7$ and $fc_8$ of the re-trained CNN to obtain global-appearance descriptors and to use them to estimate the position within a room where an image was captured.

Overall, the hierarchical localization is carried out as the diagram in fig. 6.5 shows. First (rough localization step), a test image $im_{test}$ is introduced into the CNN and the most likely room $c_i$ in which the image was captured is estimated from the information in the output layers. At the same time, the CNN is also capable of providing holistic descriptors ($\vec{d}_{test,conv_4}, \vec{d}_{test,conv_5}, \vec{d}_{test,fc_6}, \vec{d}_{test,fc_7}$ or $\vec{d}_{test,fc_8}$) from intermediate layers. Subsequently, after estimating the room, a more accurate localization is conducted (fine localization step). Is this stage, one of the descriptors $\vec{d}_{test}$ is compared with the descriptors $D_{c_i} = \{\vec{d}_{c_i,1}, \vec{d}_{c_i,2}, ..., \vec{d}_{c_i,N_i}\}$ from the training dataset which belong to the retrieved room $c_i$ and the most similar descriptor $\vec{d}_{c_i,k}$ is retained. Finally, the position where the test image was captured is estimated as the coordinates where $im_{c_i,k}$ was captured.

## 6.4 Experiments

This section presents in detail the experiments carried out in the present chapter concerning the training of the CNN for room retrieval and also its use to carry the localization task in indoor environments. The set of experiments were all tackled in Matlab ® with a PC which is equipped with a CPU Intel Core i7-7700 ® at 3.6 GHz. Furthermore, the training of the CNN was tackled with the help of a GPU NVIDIA GEFORCE GTX 1080TI ®. The remainder of this section is structured as follows. Subsection 6.4.1 introduces the development, training and performance of the CNN; subsection 6.4.2 shows the use of this tool to extract global-appearance descriptors to tackle the batch localization task; and subsection 6.4.3 presents the use of the CNN to carry out a hierarchical localization task.

### 6.4.1 Experiment 1. Development, Training and Evaluation of the CNN in a Room Retrieval Task

Developing a complete network architecture from scratch can be a complex process and the training could result unsuccessful. Therefore, we propose using a common architecture already developed by experts which has been tested in several previous works. The present chapter proposes the AlexNet architecture as starting point to develop a model of the environment. The choice of this network is due to the successful performance showed by other authors regarding its use for transfer learning such as [97]. As it was explained in subsection 6.2.2, in order to adapt the input and the output to the task desired (size of the panoramic images and classification among the 9 rooms of the Freiburg environment), a replacement of layers is tackled. Fig. 6.1 shows the final architecture. Consequently, the training process is as follows.

1. The CNN architecture is obtained from the AlexNet CNN and a layer replacement is tackled.

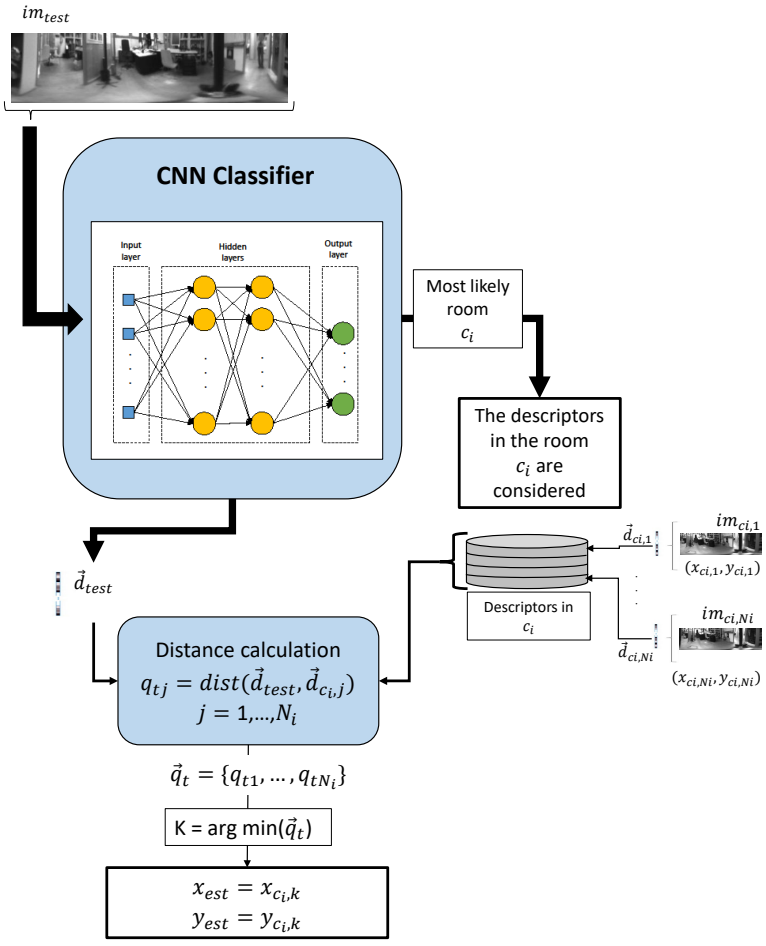



**Figure 6.5:** Hierarchical localization diagram. The test image $im_{test}$ is introduced into the CNN. The most likely room is retrieved $c_i$ and the holistic descriptor $\vec{d}_{test}$ is obtained from one of the layers. A nearest neighbor search is done with the descriptors from the training dataset included in the retrieved room and the most similar descriptor ($im_{c_i,k}$) is retained. The position of $im_{test}$ is estimated as the position where $im_{c_i,k}$ was captured.

2. The training data (set of images with labeling) is augmented by a data augmentation technique.

3. The training options are adjusted according to the training specifications.

4. Re-trainings of the network are conducted by adapting the training options to produce a more accurate CNN until the network is able to achieve a 97% of correct estimations by using validation data (data contained in the training dataset that are exclusively used during the process to check the amount of correct estimations with the current parameters established in each layer).

After training properly the CNN, its accuracy ($acc_{\%}$) is measured as $acc_{\%} = (N_{ok}/N_{test}) \times 100$, where $N_{ok}$ is the number of images that have been correctly retrieved and $N_{test}$ is the number of images that compose the test dataset evaluated. The test datasets (cloudy, night and sunny) are used to evaluate the accuracy of the CNN. Through this evaluation, the final accuracy values obtained were 98,71%, 96,52% and 92,87% respectively. Therefore, from the results obtained, the conclusion is that the CNN is properly trained to retrieve the input image into the room where it was captured. Figures 6.6, 6.7 and 6.8 show the confusion matrices obtained by introducing the cloudy, night and sunny test datasets into the network. From these figures, we can analyze that the few wrong classifications are produced with wrong rooms that are adjacent and visually similar to the correct one. For example, in cloudy, in the case of the images that belong to the 2-persons office 2 and were wrongly retrieved, the mistaken room was the contiguous and similar 1-person office room. Furthermore, more mistakes can be noticed when the evaluated images were captured under changes of illumination (night and sunny). For instance, under dark illumination conditions (night dataset), the stair area is wrongly predicted 47 times, 15 and 29 times are corridor and bathroom respectively, that are rooms adjacent and similar. Nonetheless, the printer area is wrongly retrieved 3 times. Regarding the results with the sunny illumination conditions, the wrong classifications between the 2-person office 2 and 1-person office room is increased.

Moreover, fig. 6.9 shows two bar charts concerning the likelihood behavior of the CNN when the estimations are correct or wrong. That is, they show the average likelihood of the evaluated images to belong to the room retrieved (the best option), the likelihood to belong to second best option and so forth. This information is calculated by the final layer of the CNN. As it is observed in fig. 6.9 (a), when the rooms of the images are correctly estimated, the correct option presents an average likelihood near to the 100% and the second best option presents an average likelihood of 1,09%. In contrast, the fig. 6.9 (b) shows these average percentages when the retrieved room is not correct. In this case, a considerably lower likelihood for the best option (74,24%) and a higher likelihood for the second best option (22,5%) is observed. Hence, from this information, the conclusion reached is that the likelihoods calculated by the CNN for a test image can be helpful to decide whether the classification was correct or wrong and also, which other rooms should be considered apart from the best option retrieved.

### 6.4.2 Experiment 2. Use of the CNN to Obtain Holistic Descriptors for Batch Localization

This experiment introduces an evaluation of the performance of the global-appearance descriptors obtained from different intermediate layers of the CNN to address the batch localization task. The idea, as explained in subsection 6.3.1 and shows the fig. 6.4, consists in introducing an image into the CNN and obtaining the holistic descriptor from the layers $conv_4$, $conv_5$, $fc_6$, $fc_7$ and $fc_8$. First, these description methods are used to build the visual model by calculating the holistic descriptor for each image contained in the training dataset ($D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_{N_{train}}\}$). After that, the localization is solved by using an image retrieval algorithm, that is, a test image is captured ($im_{test}$), its

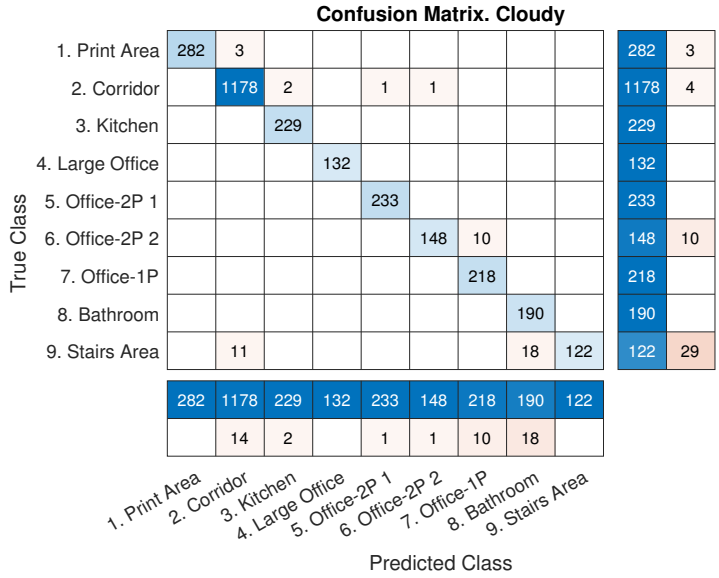



**Figure 6.6:** Confusion matrix obtained obtained after solving the first step of the hierarchical localization (room retrieval) with all the cloud test images.

global-appearance descriptor ($\vec{d}_{test}$) is obtained from a layer of the CNN; then, the descriptor is compared with all the descriptors contained in the training model $D$ and the most similar descriptor $\vec{d}_k$ is retained. Last, the position of the captured image $im_{test}$ is estimated as the position where $im_k$ was captured.

In this experiment, the cloudy test dataset is used to measure the effectiveness of the proposed description methods. Furthermore, the night and sunny datasets are used to evaluate the robustness of these descriptions against changes of illumination. Fig. 6.10 presents the results obtained by solving the batch localization with the test images. These results were obtained by using the holistic descriptors obtained from the CNN and also by classical description methods (HOG and *gist*). Fig. 6.10 shows the average localization error and also the average computing time. Regarding the localization error, this was calculated as the average Euclidean distance between the position estimated and the position provided by the ground truth of the dataset. As for the average computing time, this value measures the time required to carry out the whole process: from calculating the holistic descriptor of the test image until its position is estimated.

First, analyzing the localization without taking the changes of illumination into consideration (i.e. using the cloudy test dataset), the experiments show that the descriptor obtained from the layer $conv_4$ presents the minimum error (5,07 cm), followed by the descriptors from the layers $conv_5$ and $fc_6$ (5,09 cm for both cases). As for the computing time, the fastest option is also achieved with the $conv_4$ layer (6,7 ms), since the holistic descriptor obtained from this layer has a relatively small size (180 components) and the data obtained for this layer are calculated in an early stage of

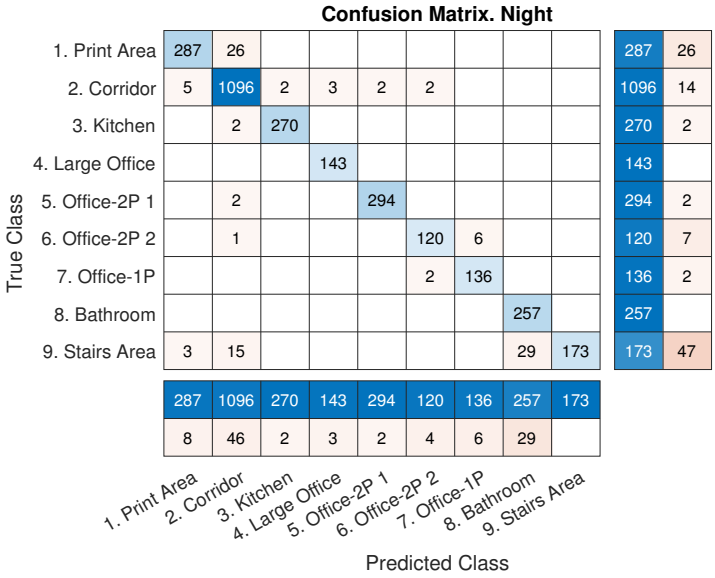



**Figure 6.7:** Confusion matrix obtained after solving the first step of the hierarchical localization (room retrieval) with all the night test images.

the CNN architecture. Comparing the holistic descriptors obtained from the CNN with the classic descriptors, the conclusion is that the descriptors obtained with the CNN improve the localization task both considering accuracy and computing time.

Regarding the results obtained with changes of illumination (using night and sunny datasets), as noticed in previous chapters, this effect worsens the localization task. In all the cases, the average localization error increases in comparison to the values obtained when no changes of illumination are considered. Overall, sunny illumination conditions affect more negatively the localization method proposed. Moreover, $conv_5$ and $fc_8$ are the layers of the CNN whose holistic descriptors are more affected. The most robust descriptors against changes of illumination are those generated by the layers $fc_6$ and $fc_7$. For instance, as for the holistic descriptor obtained from the $fc_6$ layer, the fig. 6.10 shows that the average localization error increases from 5,09 cm (without changes of illumination) to 28,80 and 38,94 cm (with night and sunny illumination conditions respectively). Notwithstanding that, the descriptor provided by the layers $fc_6$ and $fc_7$ perform substantially more accurately than the classical analytic methods under changes of lighting conditions.

Generally speaking, either layers $conv_4$, $conv_5$, $fc_6$ or $fc_7$ can result suitable to carry out the batch localization task task. If no changes of illumination are expected, the descriptors $\vec{d}_{conv_4}$ and $\vec{d}_{conv_5}$ are appropriate, since they work relatively fast (9.07 ms and 10.7 ms respectively). On the contrary, the descriptors $\vec{d}_{fc6}$ and $\vec{d}_{fc7}$ are suitable if there are changes of illumination at the expense of a lightly higher computing time. The descriptor obtained from the layer $fc_8$ works relatively fast (19.34 ms),

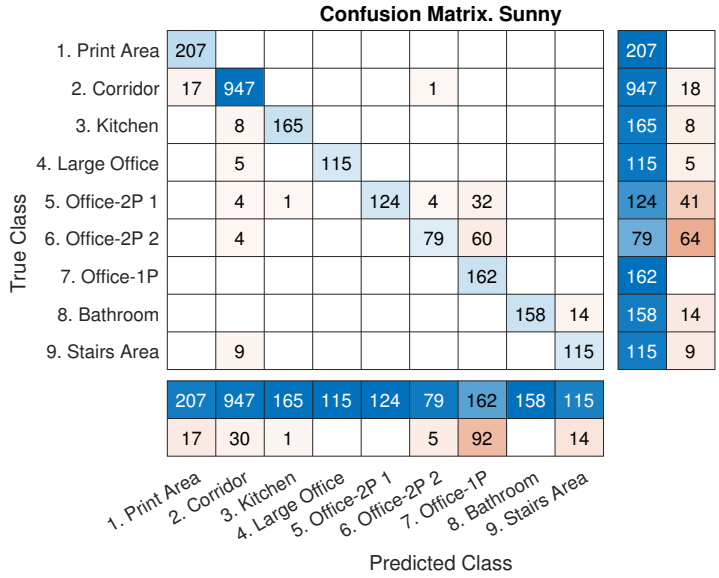
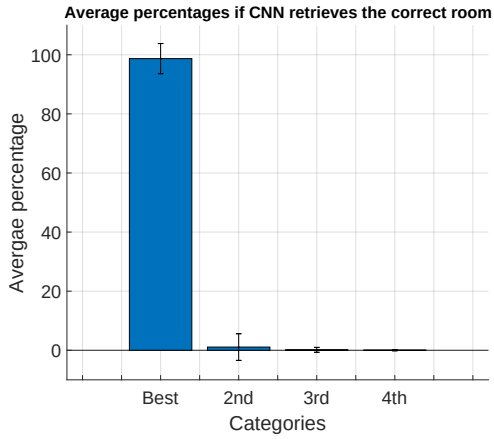


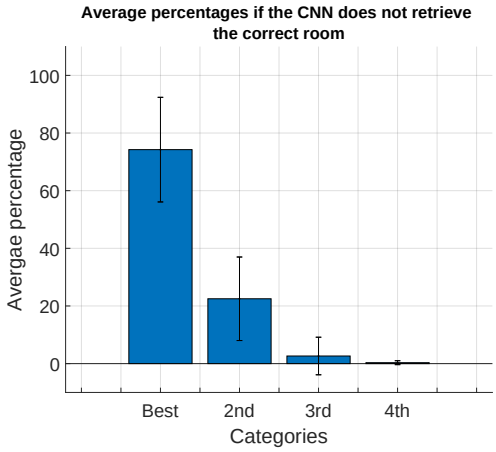

**Figure 6.8:** Confusion matrix obtained after solving the first step of the hierarchical localization (room retrieval) with all the sunny test images.

but the localization errors obtained are substantially worse comparing to the rest of descriptors evaluated.

After evaluating the use of the CNN to generate holistic descriptors, this work also aims to evaluate the use of this CNN to address the localization with images that were captured from a different environment. The idea is to check whether the CNN developed and trained with images from a specific environment can be generalized. That is, that the CNN generates robust holistic descriptors for images captured in other environments different from the one used for training. Hence, a short experiment is tackled, in this case, using the images from the Saarbrücken environment as test images. Again, average localization error and average computing time are collected for different description methods: four different layers of the Freiburg CNN proposed in this work, the *gist* descriptor and a descriptor based on the layers $conv_4$ and $fc_6$ of the original AlexNet network (without training nor replacing layers). The table 6.2 shows the results for localization with the Saarbrücken dataset by using the proposed holistic descriptors. As it is shown, most of the descriptors based on the Freiburg CNN are still relatively accurate. To illustrate one example, the performance of $\vec{d}_{conv_4}$ (Freib-CNN) is similar to $\vec{d}_{gist}$ and $\vec{d}_{fc_6}$ (AlexNet), but the calculation time is lower. Therefore, we conclude that obtaining holistic descriptors from the trained CNN is a relatively good method and it is generalizable to other environments different from the one which is used for training.




(a)




(b)

**Figure 6.9:** Average likelihood provided by the CNN. That is, average likelihood that the room retrieved is correct. This information is provided by the classification layer of the CNN. The graphs show the average likelihood when the classification is (a) correct or (b) wrong according to the ground truth of the test datasets.

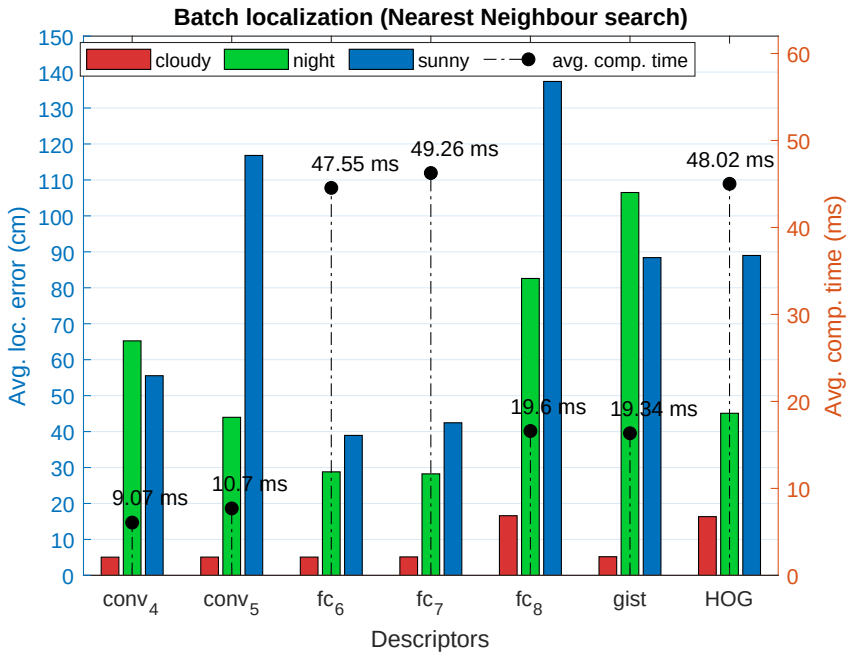



**Figure 6.10:** Visual localization solved by means of image retrieval considering different holistic descriptors: CNN based descriptors ($conv_4$, $conv_5$, $fc_6$, $fc_7$ and $fc_8$) and classic descriptors (*gist* and HOG). The efficiency is measured through the average localization error (cm) and also the average computing time (ms) required to calculate and estimate the position where the images were captured.

**Table 6.2:** Batch localization solved by means of image retrieval in Saarbrücken. The global-appearance descriptors used are obtained either from the Freiburg CNN, which was trained in this work ($conv_4$, $conv_5$, $fc_6$ and $fc_7$), from the AlexNet ($conv_4$ and $fc_6$), or by using the *gist* descriptor. The efficiency is measured through the average localization error (cm) and also the average computing time (ms) required to calculate and estimate the position where the images were captured.

| Descriptor | Avg. error (cm) | Avg. computing time (ms) |
|---|---|---|
| $conv_4$ (Freib-CNN) | $7.33 \pm 0.29$ | 11.07 |
| $conv_5$ (Freib-CNN) | $15.82 \pm 0.30$ | 12.14 |
| $fc_6$ (Freib-CNN) | $7.49 \pm 0.31$ | 58.61 |
| $fc_7$ (Freib-CNN) | $7.67 \pm 0.34$ | 61.15 |
| $conv_4$ (AlexNet) | $7.79 \pm 0.36$ | 11.28 |
| $fc_6$ (AlexNet) | $7.28 \pm 0.28$ | 56.87 |
| gist | $7.28 \pm 0.59$ | 460.6 |

### 6.4.3 Experiment 3. Use of the CNN to Tackle Hierarchical Localization

In the previous experiment, different holistic descriptors were evaluated to address the batch localization task. The present subsection focuses on evaluating the complete

use of the neural network to tackle the localization task hierarchically. In this way, the CNN is not only used to obtain holistic descriptors, but it is also used to retrieve the most probable room within the environment where the test image was captured. As it was explained in subsection 6.3.2, the hierarchical localization task proposed consists in: first (rough localization step), the test image is introduced to the CNN and it retrieves the most likely room where the image was captured by the robot. Second, a holistic descriptor is obtained from one of the layers of the CNN and this information is used to carry out the fine localization step by conducting an image retrieval algorithm by using the holistic descriptor of the test image and the holistic descriptors of the training images that belong to the retrieved room.

With the aim of comparing this localization method with the method proposed in the subsection 6.4.2, the evaluation is the same, i.e., the average localization error and the average computing time are obtained to tackle the hierarchical localization process. Moreover, this method is also compared with the hierarchical localization method proposed in chapter 4 with the descriptors *gist* and HOG and a clustering approach.

Fig. 6.11 presents the results obtained through the hierarchical localization proposed in the present chapter. In general, the descriptors based on CNN perform better than the descriptors based on hand-crafted methods. Comparing the descriptors based on CNN layers and the analytic ones, the localization error with CNN-based descriptors is considerably lower. This improvement is noticed independently of the illumination condition. Furthermore, the computing time required to solve the localization is also lower using the descriptors based on CNN.

If we compare the results obtained by applying batch localization and hierarchical localization, the second method introduces a lightly higher localization error. This is given in all the descriptors evaluated and is due to failures produced in the rough localization step (CNN does not retrieve the correct room the 100% of the times). However, if we focus on the results obtained by using the descriptors $\vec{d}_{fc_6}$ and $\vec{d}_{fc_7}$, they both present a robust behavior, since their results keep the localization error obtained through batch localization and at the same time, the computing time is substantially reduced. This behavior is kept for the three illumination conditions.

Concerning the localization error increment produced by the CNN wrong classifications, we checked that the CNN was properly trained, since it retrieves the room successfully the 98% of the cases. Additionally, observing the graphs from the fig. 6.9, extra information from the output layer of the CNN can be used to improve this method. These graphs show a considerably different behavior of the likelihoods when the CNN succeeds or fails. When the correct rooms is retrieved, the most probable room presents an average likelihood around 98% and the rest of options are under the 2%, whereas when the CNN retrieves a wrong room, the most probable room presents an average likelihood of 74,24% and the following two most likely options are substantially over 2%.

Therefore, departing from this analysis, the present chapter also proposes a novel hierarchical localization method based on the CNN to solve the rough localization step

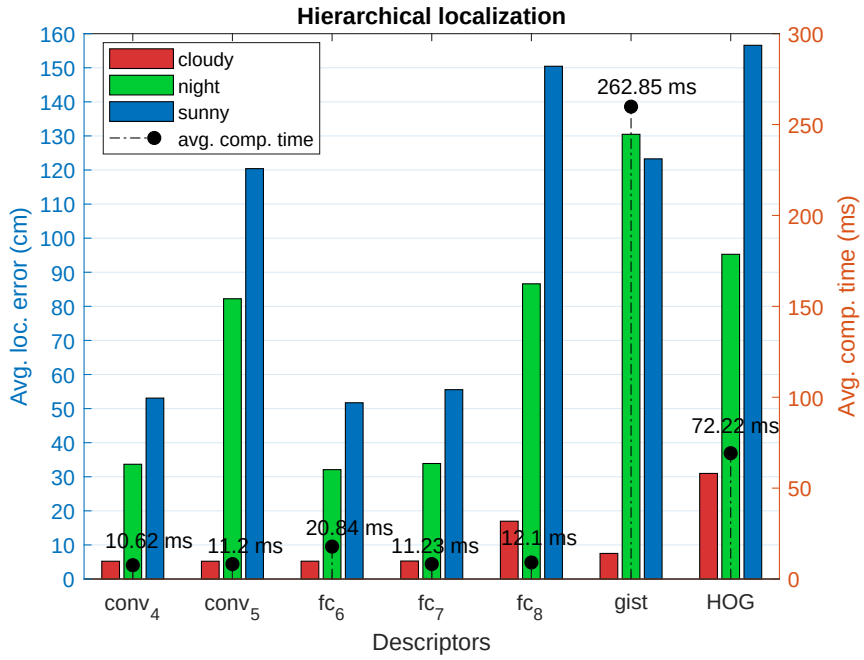



**Figure 6.11:** Hierarchical localization results based on CNN. The rough step is solved by retrieving the most likely room with the CNN. The fine step is solved by means of an image retrieval algorithm within the retrieved room by using holistic descriptors (horizontal axis). The efficiency is measured through the average localization error (cm) and also the average computing time required to calculate and estimate the position where the images were captured (ms).

but considering a threshold value to decide how many rooms are considered in the fine localization step. The whole method consists in the following steps. First, the test image is introduced into the CNN. The classification layer outputs 9 likelihoods. If the likelihood of the most probable room is higher than the threshold 1, $th_1$, this room is retrieved; otherwise, all the rooms whose likelihood is higher than the threshold 2, $th_2$ are retrieved. Afterwards, the fine localization is carried out again through image retrieval by comparing the holistic descriptor of the test image (obtained from a layer of the CNN) with the set of training descriptors contained in the retrieved rooms.

Therefore, through this new method, the hierarchical localization is carried out with all the test images and the results are presented in fig. 6.12. For this experiment, only $conv_4$, $conv_5$, $fc_6$ and $fc_7$ were evaluated, since $fc_8$ has proved in previous experiments not to be suitable to generate a holistic descriptor that characterizes the images. The thresholds values were tuned and the best configuration was $th_1 = 0.8$ and $th_2 = 0.1$. In this figure we can observe that for all the cases, the average computing time increases with respect to fig. 6.11. This increase was expected, since this method leads to consider more instances in the fine localization step. Regarding the descriptors generated from the layers $fc_6$ and $fc_7$, which were the cases that had a lower

computing time in hierarchical localization, their related computing time is increased from 20.84 and 11.23 to 27 and 27.1 ms respectively. Nevertheless, the localization process is still substantially faster than the obtained with the batch localization method based on image retrieval (fig. 6.10), because this method takes respectively an average computing time of 47.55 ms and 49.26 ms .

Regarding the localization error, most of the cases are improved. For example, in the case of the global-appearance descriptor $\vec{d}_{fc_6}$, the average localization error is reduced by using the hierarchical localization with thresholds: from 5.23; 32.09 and 51.71 cm to 5.13; 25.53 and 38.10 cm respectively for the cloudy, night and sunny conditions. Thus, we conclude that this novel method proposed to address the hierarchical localization task with thresholds is a competitive option regarding localization error and computing time.

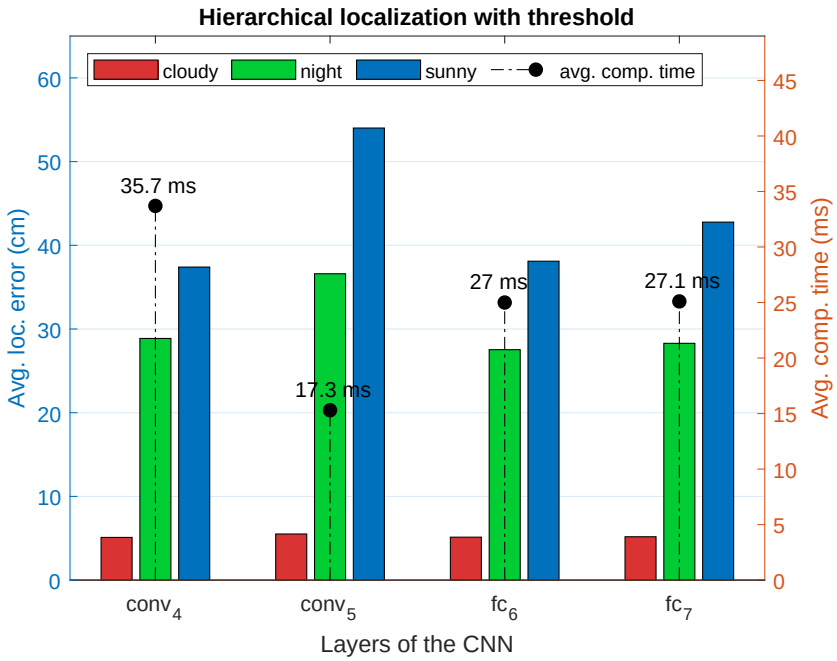



**Figure 6.12:** Hierarchical localization results. The rough step is solved by retrieving a number of rooms with the CNN. If the likelihood of the most probable room is not higher than an specified threshold $th_1$, all the rooms whose likelihood is higher than another specific threshold $th_2$ are considered for the fine localization step. As it was conducted in previous hierarchical localization methods, the fine step is solved by means of image retrieval through using CNN based global-appearance descriptors.

## 6.5 Conclusion

In this chapter, a novel localization methods based on the use of deep learning tool and omnidirectional images is presented. The tool developed is a convolutional neural

network trained for room retrieval. In this way, once the CNN has properly trained, the network receives a panoramic image as input and it retrieves the most likely room where the image was captured. Furthermore, this neural network is not only used to estimate rooms, but also to obtain global-appearance descriptors from its intermediate layers to characterize the information of the input image. Therefore, the present chapter evaluates the use of this tool to solve the localization task by means of three different methods: a batch localization (through image retrieval), a hierarchical localization based on different levels of accuracy and a hierarchical localization method with thresholds to decide which rooms are used in the fine localization stage.

A dataset of omnidirectional images captured in an indoor environment was used for training the CNN and also to tackle the localization experiments. These images were transformed to panoramic with the aim of comparing the results with previous methods based on classic holistic description methods. Moreover, this dataset also contain images captured under different illumination conditions. Hence, the robustness of the proposed against changes of illumination was also addressed.

Furthermore, a data augmentation technique is proposed to supply a larger visual dataset to train more robustly the CNN. This technique is also used to add adverse visual effects to the dataset used to test the accuracy of the CNN developed. Regarding the CNN design, the network inherits the architecture from AlexNet and changes the initial and the final set of layers. Then, it is re-trained with the panoramic images obtained from the dataset.

The studies tackled in the present chapter are the following:

- The CNN classifier developed have been validated as a tool to perform the rough step of a hierarchical localization process. Additionally, the behavior of the classification layer provides information that can be useful to detect wrong estimations.
- The global-appearance descriptors obtained from the intermediate layers $conv_4$, $conv_5$, $fc_6$ and $fc_7$ are more suitable to solve the localization task than the hand-crafted descriptors *gist* and HOG. Additionally, $fc_6$ and $fc_7$ produce holistic descriptors which prove to be quite robust against changes of illumination. Also, the descriptors obtained from the CNN are also suitable to solve visual localization in other different environments, but they do not improve substantially the results output by a descriptor obtained from other pre-trained CNNs such as AlexNet.
- The hierarchical localization based on the proposed CNN produces more efficient results regarding localization error and computing time than hierarchical methods based on classical descriptors and image retrieval. Additionally, considering the likelihood information provided by the classification layer of the CNN, the proposed method produces competent localization solutions.

Throughout fig. 6.13 we can obtain an overview of the localization performed overall along the Freiburg environment by using the three proposed methods based

on CNN and using the holistic descriptor generated by the layer $fc_6$. These figures show the bird's eye view of the ground truth of the (cloudy) test images and the estimated position. Fig. 6.13(a) shows the estimation by using batch localization, fig. 6.13(b) shows the estimation using hierarchical localization and fig. 6.13(c) shows the estimation when thresholds are applied to the hierarchical localization method. Furthermore, fig. 6.14 shows (a) the average localization error and (b) the average computing time required for each method to solve the localization task. The conclusion reached from these images is that hierarchical localization based on CNN keeps the precision of batch localization, but this method is substantially faster. The use of thresholds is useful to keep a good accuracy.

Future works will focus on developing a regression convolutional neural network that is able to estimate directly the position where the input images were captured.

## 6.6 Publications Related to this Chapter

The main results presented in this chapter are related to the following publications:

- S. Cebollada, L. Payá, M. Flores, V. Román, A. Peidró, O. Reinoso. A Deep Learning Tool to Solve Localization in Mobile Autonomous Robotics. In 2020 17th International Conference on Informatics in Control, Automation and Robotics. [43]

  - This paper proposes a deep learning tool to carry out the visual localization task for mobile autonomous robotics. A convolutional neural network (CNN) is trained with the aim of estimating the room where an image has been captured, within an indoor environment. This CNN is not only used as tool to solve a room estimation, but it is also used to obtain global-appearance descriptors of the input image from its intermediate layers. The localization task is addressed in two different ways: globally, as an image retrieval problem and hierarchically. Throughout this paper, the localization methods are evaluated with a visual set of omnidirectional images captured in indoor environments. The obtained results show that the proposed deep learning tool is an efficient solution to address visual localization tasks.

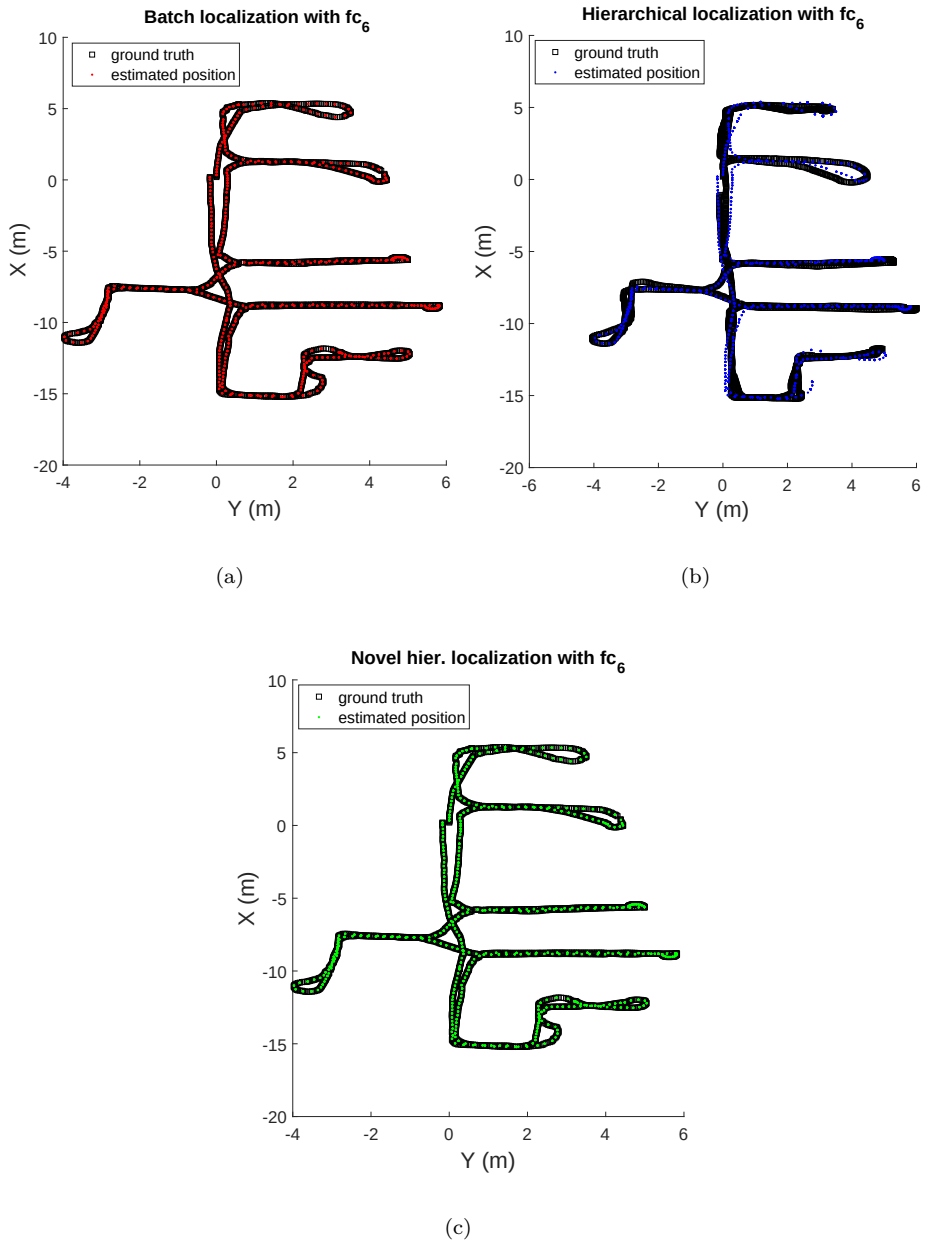
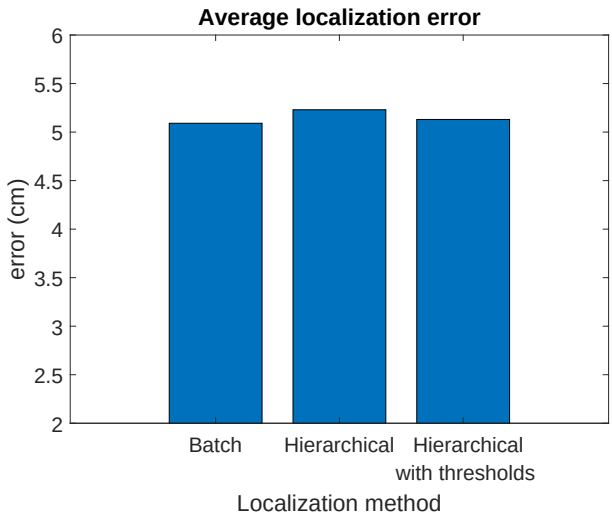
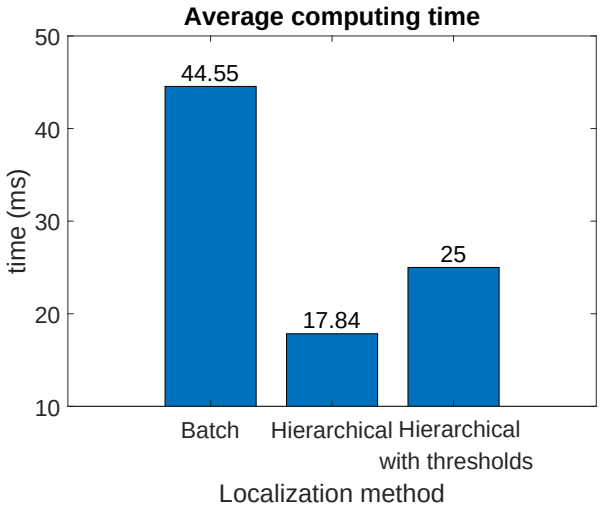



**Figure 6.13:** Comparison between poses estimation in the Freiburg cloudy test dataset by means of the different localization methods proposed in the present work: (a) Batch localization, (b) hierarchical localization and (c) hierarchical localization with thresholds. Both the ground truth and the estimated position of the test images are shown. These examples are based on the holistic descriptor generated by the layer $fc_6$ of the Freiburg CNN.




(a)




(b)

**Figure 6.14:** Comparison between localization methods: (a) average localization error and (b) average computing time are presented for the three methods proposed using the holistic descriptor generated by the layer $fc_6$ of the Freiburg CNN.

# 7 Conclusions and Future Work

After presenting in detail all the research work conducted under the framework of the present thesis, this final chapter summarizes the main contributions of this work. Furthermore, section 8.2 introduces possible extensions and future work which can be carried out from the research lines presented in this thesis.

## 7.1 Contributions

This thesis has presented a robust localization method to move through underfloor voids with depth data, a hierarchical model to carry out the localization in indoor environments by using omnidirectional visual information, a localization approach based on omnidirectional images and machine learning tools, and a novel convolutional neural network based on omnidirectional information which leads to competent localization methods. The analysis of the studies has consisted in evaluating the effectiveness of the proposed methods by measuring the average localization error and the average computing time. These analyses were all developed through the use of simulation tools: Matlab ® and C++. The main accomplishments and contributions of this thesis are summarized as follows.

**Chapter 3**

- It has been demonstrated that the proposed algorithm that uses point cloud information to estimate the registration between poses present noticeable improvements in comparison with previous proposed methods. The experiments showed that this algorithm works successfully in environments where the characterization using regular algorithms is difficult. Additionally, the method proposed

is capable of finding accurate enough results without using any kind of visual information provided by the camera. The results show that this algorithm performs successfully for the majority of the cases considering localization error and computing time.

- It has been presented an algorithm to solve the cases in which the alignment between consecutive poses within the environment is not successful. The results showed that this approach is able to align correctly in the majority of the cases. Hence, the use of this approach introduce a notorious utility to address the localization task when keeping all the visited poses (knowing the exact path) plays a crucial role. Although this method leads to extra computing time, scarcely no pose information is lost throughout the whole localization process.
- The proposed approaches are based on a validation parameter. This parameter determines whether the pose estimation was successful or not. Hence, this work presented a tuning of this parameter with the aim of establishing a trustworthy threshold. From the obtained results, the conclusion reached is that the algorithm works properly in a given range of values. Nevertheless, the exact value should be determined according to the requirements of the user.
- It has been also proposed a visual approach to carry out the localization task despite the environments present a difficult visual characterization. The proposed approach is not robust enough to carry out the localization with the accuracy presented with depth data. Notwithstanding that, this approach is beneficial to improve the depth-data-based method.

**Chapter 4**

- It has been presented two methods to compact topological maps. The visual models are built with holistic descriptors based on omnidirectional images, which were captured from different positions within the environment. The approaches proposed are based on clustering algorithms: spectral clustering and self-organizing maps. It has demonstrated that these methods are suitable to reduce drastically the visual information from the original model.
- It has been also shown that the compression method proposed performs more efficiently than a direct downsampling of the database. A straightforward downsampling can produce faster models, but the resultant models are not able to keep such amount of visual information as the proposed clustering methods do.
- It has been demonstrated that the compressed models can be used to address the localization task by using global-appearance descriptors. The results showed that the accuracy obtained as well as the computing time required to address the task are inversely proportional to the compaction rate. That is, the more is the model compressed, the less accurate is the localization and the less computing time is necessary (the faster is the estimation pose solved). Additionally, the approach proposed is validated for topological maps as well as for grid-based maps. Beside

this, the localization method is also capable of addressing the localization task when more than one environment is taken into consideration.

- A hierarchical method was proposed to carry out the localization departing from visual compressed models based on holistic descriptors. This proposed approach improves the results obtained regarding accuracy as well as computing time. On the whole, the proposed method is able to reach results as accurate as the batch localization method but saving considerable amount of time.
- The change of illumination conditions was evaluated and the results showed that despite the accuracy of the localization methods proposed is negatively affected, they are able to keep accurate localization results. On the other hand, the sunny illumination condition affected more negatively the performance of the approaches than the dark illumination (images captured at night).

**Chapter 5**

- It has been demonstrated that classifiers and the fit neural networks are useful machine learning tools to solve respectively the rough and fine localization tasks in a hierarchical framework. Both tools are fed with holistic descriptors that were obtained from the omnidiretional images by classic global-appearance methods and by deep learning tools.
- Regarding the classifiers, the room retrieval was improved by using labeling obtained from clustering methods rather than using the labeling provided by the ground truth.
- Data fitting networks were not able to improve considerably the fine localization step in some rooms within the indoor environment. Nevertheless, this study concluded that a more suitable training optimization would result crucial to solve the localization in challenging areas. Besides this, the fine localization method based on neural networks would not present the accuracy limitation presented by the image retrieval method.
- It has been presented two deep learning tools to obtain holistic descriptors from the omnidirectional visual data. Through a batch localization approach, the two methods, along with classic global-appearance description methods, were evaluated. It has been demonstrated that the CNN-based description methods provide the minimum localization error. Additionally, autoencoders also provide an efficient solution. Concerning this second tool, they are not only suitable to describe information related to a specific environment, but they also are capable of training a generic network that is able to obtain trustworthy descriptors regardless the environment. Despite this methods are relatively fast, they do not improve the speed presented by HOG description.
- It has been demonstrated that competent holistic descriptors can be obtained from early layers of the CNNs, since these layers are focused on extracting the

features of the images. Therefore, the obtained descriptors lead to produce accurate localization performance and at the same time, the computing time is the lowest among the description methods studied.

**Chapter 6**

- It has been presented a localization method based on the creation and training of a CNN that is capable of carrying out a competent localization task.
- It has demonstrated that the CNN developed is able to perform successfully the rough step of a hierarchical localization process. Furthermore, the classification layer provide likelihood information which can be used to reinforce the rough localization decision.
- It has been proved that data augmentation is a suitable technique to expand the original dataset with the aim of carrying out the training of a CNN from scratch with the aim of carrying out a room retrieval task.
- It has been demonstrated the robustness of the holistic descriptors obtained from the intermediate layers of the trained CNN. These descriptors perform more suitably the localization task than hand-crafted description methods.
- The hierarchical localization method based on CNN has performed successful localization results. Likelihood information has provided robustness in the room retrieval process (rough localization step), which has led to provide a localization method faster than the batch localization approach and more accurate than the previously proposed hierarchical localization methods.

## 7.2 Future Work

The following list proposes some future research works that can be stemmed from the research lines and results conducted throughout the present thesis.

- **Developing a more reliable and quicker characterization of the depth information.** It seems that developing a characterization method would play an important role in the localization task under challenging visual environments. This method should be able to provide relevant information from the depth data which allows trustworthy and fast algorithms. By means of deep learning techniques, global-appearance description methods could be applied to the environment with the aim of obtaining relevant characterization from the visual data, the depth data or a mixture of both. This characterization could perform faster algorithms with similar or even better accuracy than the currently proposed methods.

- **Developing a visual SLAM algorithm based on the hierarchical methods proposed.** The proposed methods to carry out the localization with omnidirectional images have been validated. Therefore, a SLAM approach can be developed based on the studied methods. The aim of this approach is to develop a system as autonomous as possible. We propose a incremental clustering approach which is able to tackle the compression task of the visual data as the robot is moving throughout the environment.

- **Optimization of the data fitting neural network.** Concerning the neural network presented in chapter 5 to estimate the position of the robot within a retrieved room, a future research line would be optimizing the training process. This neural network has provided competent results in the majority of the rooms, but it has presented unsuccessful results in some areas such as the corridor. However, in the future, an exhaustive study of the hyperparameters will be carried out with the aim of optimizing the training process. Additionally, a clustering process can be addressed for those rooms whose size make the position estimation more challenging.

- **Deep study of the autoencoder tools.** The present thesis presented a short study related to the use of an autoencoder neural network to calculate global-appearance descriptors. Nevertheless, in the future, a deep study of this neural network can be addressed. The following studies can be proposed.

  - Use of omnidirectional images to train the autoencoder.
  - Use RGB images instead of gray-scale.
  - Study of key hyperparameters which can optimize the network such as the number and type of hidden layers and filters or the dimension of the latent representation.
  - Use of different types of autoencoders such as denoising autoencoders or variational autoencoders. Concerning the denoising autoencoder, this is desiged to be capable of transforming noisy data into clean ones. Hence, several visual effects can be tested to perform the autoencoder. For example, images with different illumination or with rotation.

- **Evaluation of newer network architectures.** The CNN developed in chapter 6 departed from the AlexNet. Nevertheless, this network was created in 2012. Hence, in future works, newer architectures will be considered as starting point to build our neural network. This way, better results may be acquired in lower time.

- **Developing a CNN that outputs the position of the robot.** Carrying on the work presented in chapter 6, the next step will be to use the CNN developed to carry out a transfer learning with the aim of solving a regression task. In this case, the regression task will consist in estimating directly the position $(x, y)$ of the robot within the environment. This way, several issues should be conducted. For example, training one CNN with two outputs ($x$ and $y$) or training one CNN to

output the $x$ coordinate and another for the $y$. Also, using networks for different rooms or a one single network to estimate the position in the whole environment.

- **Deep study of CNNs based on omnidirectional images.** Short studies have been already conducted concerning this topic. A CNN was re-trained by using omnidirectional images with the aim of solving a room retrieval task. Hence, in the future, an evaluation of the holistic descriptors obtained from intermediate layers can be tackled. This description method can perform successful localization results, since they can provide accurate and fast localization approaches. Furthermore, the step to transform omnidirectional to panoramic will lead to save computing time, since the CNN will be trained directly with omnidirectional images. Moreover, this study will also attempt to build a regression CNN which can estimate the capturing position within the environment.

- **Use of Siamese neural networks.** In this thesis, the main idea consisted in obtaining an appropriate holistic descriptor and also an appropriate measure of distance which performed successfully the similarity between images. In this way, deep learning tools such as CNN were proved to perform successful holistic descriptors. In this line, Siamese networks are a similar version of the CNN. However, they are fed with two images and the output provides a value of similarity between them. Hence, by using Siamese neural network, the model will directly output the similitude between images. Therefore, this machine learning tool is proposed to be used in two ways.

  - By training the Siamese network with pair of images, it will learn to output an appearance similitude value which can be indirectly proportional to the image capturing distance or a similitude value which can be indirectly proportional to the rotational distance.
  - This network can be trained with pair of images which were captured at the same position but with different illumination conditions. Once the network is trained, a holistic descriptor can be obtained from intermediate layers. Thus, the obtained descriptor will be desired to be robust against changes of illumination.

# 8 Conclusiones y Trabajos Futuros

Tras presentar en detalle todo el trabajo de investigación realizado en el marco de la presente tesis, este capítulo final resume las principales aportaciones de este trabajo. Además, el apartado 8.2 introduce posibles ampliaciones y trabajos futuros que se pueden realizar a partir de las líneas de investigación presentadas en esta tesis.

## 8.1 Contribuciones

Esta tesis ha presentado un método de localización robusto para moverse a través de los subsuelos de los edificios utilizando información de profundidad, un modelo jerárquico para llevar a cabo la localización en entornos de interior mediante información visual omnidireccional, un método de localización basado en imágenes omnidireccionales y herramientas de aprendizaje máquina, y una red neuronal convolucional basada en información omnidireccional la cual es propuesta como solución eficiente para realizar métodos de localización. El análisis de los estudios ha consistido en evaluar la eficiencia de los métodos propuestos mediante la medición del error de localización medio y el tiempo de computo medio. Estos análisis fueron desarrollados mediante herramientas de simulación: Matlab ® y C++. Los principales logros y aportes de esta tesis se resumen a continuación.

**Capítulo 3**

- Se ha demostrado que el algoritmo propuesto que utiliza la información de la nube de puntos para estimar el registro entre poses presenta mejoras notables en

comparación con los métodos propuestos anteriormente. Los experimentos demostraron que este algoritmo funciona con éxito en entornos donde la caracterización utilizando algoritmos estándares es difícil. Además, el método propuesto es capaz de encontrar resultados suficientemente precisos sin utilizar ningún tipo de información visual proporcionada por la cámara. Los resultados muestran que este algoritmo funciona con éxito en la mayoría de los casos considerando el error de localización y el tiempo de cálculo.

- Se ha presentado un algoritmo para resolver los casos en los que la alineación entre poses consecutivas dentro del entorno no es exitosa. Los resultados mostraron que este enfoque puede resolver correctamente la mayoría de los casos. Por lo tanto, el uso de este enfoque introduce una utilidad notoria para abordar la tarea de localización cuando es crucial mantener información de todas las poses visitadas (conocer la ruta exacta). Aunque este método conlleva un tiempo de cálculo extra, apenas se pierde información de pose durante todo el proceso de localización.

- Los métodos propuestos se basan en un parámetro de validación. Este parámetro determina si la estimación de pose fue exitosa o no. Por ello, este trabajo presentó una puesta a punto de este parámetro con el objetivo de establecer un umbral confiable. De los resultados obtenidos se llega a la conclusión de que el algoritmo funciona correctamente en un rango de valores determinado. Sin embargo, el valor exacto debe determinarse de acuerdo con los requisitos del usuario.

- También se ha propuesto un enfoque visual para realizar la tarea de localización a pesar de que los entornos presentan una caracterización visual difícil. El enfoque propuesto no es lo suficientemente robusto para llevar a cabo la localización con la precisión presentada con los datos de profundidad. Sin embargo, este enfoque es beneficioso para mejorar el método basado en datos de profundidad.

**Capítulo 4**

- Se han presentado dos métodos para compactar mapas topológicos. Los modelos visuales se construyen con descriptores holísticos basados en imágenes omnidireccionales, que fueron capturadas desde diferentes posiciones dentro del entorno. Los enfoques propuestos se basan en algoritmos de agrupamiento: agrupamiento espectral y mapas auto-organizados. Se ha demostrado que estos métodos son adecuados para reducir drásticamente la información visual del modelo original.

- También se ha demostrado que el método de compresión propuesto funciona de manera más eficiente que una reducción directa de la base de datos. Una reducción de resolución sencilla puede producir modelos de manera más rápida, pero los modelos resultantes no pueden mantener tanta información visual como lo hacen los métodos de agrupación propuestos.

- Se ha demostrado que los modelos comprimidos se pueden utilizar para abordar la tarea de localización mediante el uso de descriptores de apariencia global. Los resultados mostraron que la precisión obtenida, así como el tiempo de cálculo requerido para abordar la tarea, son inversamente proporcionales a la tasa de compactación. Es decir, cuanto más comprimido está el modelo, menos precisa es la localización y menos tiempo de cálculo es necesario (más rápido se resuelve la pose de estimación). Además, el enfoque propuesto está validado tanto para mapas topológicos como para mapas basados en rejilla. Además de esto, el método de localización también es capaz de abordar la tarea de localización cuando se tiene en cuenta más de un entorno.

- Se propuso un método jerárquico para realizar la localización a partir de modelos visuales comprimidos basados en descriptores holísticos. Este enfoque propuesto mejora los resultados obtenidos en cuanto a precisión y tiempo de cálculo. En general, el método propuesto puede alcanzar resultados tan precisos como el método de localización estándar, pero ahorrando una cantidad considerable de tiempo.

- Se evaluó el cambio de condiciones de iluminación y los resultados mostraron que a pesar de que la precisión de los métodos de localización propuestos se ve afectada negativamente, son capaces de mantener resultados de localización precisos. Por otro lado, la condición de iluminación soleada afectó más negativamente al rendimiento de los métodos propuestos que la iluminación oscura (imágenes capturadas de noche).

**Capítulo 5**

- Se ha demostrado que los clasificadores y las redes neuronales de ajuste son herramientas de aprendizaje máquina útiles para resolver las tareas de localización grosera y fina, respectivamente, en un marco jerárquico. Ambas herramientas se alimentan con descriptores holísticos que se obtuvieron de las imágenes omnidireccionales mediante métodos clásicos de apariencia global y herramientas de aprendizaje profundo.

- Con respecto a los clasificadores, la recuperación de la sala se mejoró utilizando el etiquetado obtenido de los métodos de agrupamiento en lugar de utilizar el etiquetado proporcionado por el 'ground truth'.

- Las redes neuronales de ajuste de datos no pudieron mejorar considerablemente la localización fina en algunas habitaciones dentro del entorno interior. Sin embargo, este estudio concluyó que una optimización del entrenamiento más adecuada resultaría crucial para resolver la localización en áreas más difíciles. Además de esto, el método de localización fina basado en redes neuronales no presentaría la limitación de precisión que presenta el método de recuperación de imágenes.

- Se han presentado dos herramientas de aprendizaje profundo para obtener descriptores holísticos a partir de los datos visuales omnidireccionales. Mediante un enfoque de localización estándar se evaluaron los dos métodos, junto con los métodos clásicos de descripción de apariencia global. Se ha demostrado que los métodos de descripción basados en CNN proporcionan el mínimo error de localización. Además, los 'autoencoders' también proporcionan una solución eficiente. En cuanto a esta segunda herramienta, no solo son aptos para describir información relacionada con un entorno específico, sino que también son capaces de entrenar una red genérica que sea capaz de obtener descriptores fiables independientemente del entorno. A pesar de que estos métodos son relativamente rápidos, no mejoran la velocidad presentada por el descriptor HOG.

- Se ha demostrado que se pueden obtener descriptores holísticos competentes a partir de las primeras capas de las CNN, ya que estas capas se centran en extraer las características de las imágenes. Por lo tanto, los descriptores obtenidos conducen a producir un rendimiento de localización preciso y, al mismo tiempo, el tiempo de cálculo es el más bajo entre los métodos de descripción estudiados.

**Capítulo 6**

- Se ha presentado un método de localización basado en la creación y entrenamiento de una CNN que sea capaz de realizar una tarea de localización competente.

- Se ha demostrado que la CNN desarrollada es capaz de realizar con éxito la localización grosera dentro de un proceso de localización jerárquica. Además, la capa de clasificación proporciona información de probabilidad que se puede utilizar para reforzar la decisión de localización grosera.

- Se ha demostrado que el aumento de datos es una técnica adecuada para ampliar el conjunto de datos original con el objetivo de llevar a cabo el entrenamiento de una CNN desde cero con el objetivo establecido de recuperación de habitaciones.

- Se ha demostrado la robustez de los descriptores holísticos obtenidos de las capas intermedias de la CNN entrenada. Estos descriptores realizan mejor la tarea de localización que los métodos de descripción analíticos.

- El método de localización jerárquica basado en CNN ha obtenido resultados de localización satisfactorios. La información de probabilidad ha proporcionado solidez en el proceso de recuperación de la sala (localización grosera), lo que ha llevado a proporcionar un método de localización más rápido que el enfoque de localización estándar y más preciso que los métodos de localización jerárquica previamente propuestos.

## 8.2 Trabajos Futuros

La siguiente lista propone algunos trabajos de investigación futuros que se pueden derivar de las líneas de investigación y resultados realizados a lo largo de la presente tesis.

- **Desarrollar una caracterización más robusta y rápida de la información de profundidad.** Parece que desarrollar un método de caracterización jugaría un papel importante en la tarea de localización en entornos visuales desafiantes. Este método debería poder proporcionar información relevante a partir de los datos de profundidad que permitan algoritmos fiables y rápidos. Mediante técnicas de aprendizaje profundo, se podrían aplicar métodos de descripción de apariencia global al entorno con el objetivo de obtener una caracterización relevante a partir de los datos visuales, los datos de profundidad o una mezcla de ambos. Esta caracterización podría realizar algoritmos más rápidos con una precisión similar o incluso mejor que los métodos actualmente propuestos.
- **Desarrollo de un algoritmo SLAM visual basado en los métodos jerárquicos propuestos.** En la presente tesis se han validado los métodos propuestos para realizar la localización con imágenes omnidireccionales. Por lo tanto, se puede desarrollar un enfoque SLAM basado en los métodos estudiados. El objetivo de este enfoque es desarrollar un sistema lo más autónomo posible. Proponemos un enfoque de agrupamiento incremental que puede abordar la tarea de compresión de los datos visuales a medida que el robot se mueve por el entorno.
- **Optimización de la red neuronal de ajuste de datos.** Con respecto a la red neuronal presentada en la presente tesis para estimar la posición del robot dentro de una sala recuperada, una línea de investigación futura sería optimizar el proceso de entrenamiento. Esta red neuronal ha proporcionado resultados competentes en la mayoría de las salas, pero ha presentado resultados infructuosos en algunas áreas como el pasillo. No obstante, en el futuro se realizará un estudio exhaustivo de los hiperparámetros con el objetivo de optimizar el proceso de aprendizaje. Además, se puede abordar un proceso de agrupación en clústeres para aquellas salas cuyo tamaño produzca una estimación de la posición más difícil.
- **Estudio profundo de las herramientas de autoencoder.** La presente tesis presentó un breve estudio relacionado con el uso de una red neuronal de autoencoder para calcular descriptores de apariencia global. No obstante, en el futuro, se puede abordar un estudio en profundidad de esta red neuronal. Se pueden proponer los siguientes estudios.
  - Uso de imágenes omnidireccionales para entrenar el autoencoder.
  - Utilizar imágenes RGB en lugar de escala de grises.
  - Estudio de hiperparámetros clave que pueden optimizar la red como el número y tipo de capas y filtros ocultos o la dimensión de la representación latente.

  - Uso de diferentes tipos de autoencoders, como autoencoders de eliminación de ruido o autoencoders de variación. En cuanto al autoencoders de eliminación de ruido, está diseñado para que sea capaz de transformar datos ruidosos en limpios. Por lo tanto, se pueden probar varios efectos visuales para realizar el autoencoders. Por ejemplo, imágenes con diferente iluminación o con rotación.

- **Evaluación de arquitecturas de red más novedosas.** La CNN desarrollada en la presente tesis se generaron a partir de AlexNet. Sin embargo, esta red fue creada en 2012. Por lo tanto, en trabajos futuros, se considerarán arquitecturas más nuevas como punto de partida para construir nuestra red neuronal. De esta forma, se pueden obtener mejores resultados en menos tiempo.

- **Desarrollar una CNN que obtenga la posición del robot.** Continuando con el trabajo presentado en la presente tesis, el siguiente paso será utilizar la CNN desarrollada para llevar a cabo un 'transfer learning' con el objetivo de resolver una tarea de regresión. En este caso, la tarea de regresión consistirá en estimar directamente la posición ($x$, $y$) del robot dentro del entorno. De esta manera, se deben realizar varias cuestiones. Por ejemplo, entrenando una CNN con dos salidas o entrenando una CNN para generar la coordenada $x$ y otra para $y$. Además, utilizar redes para diferentes habitaciones o una única red para estimar la posición en todo el entorno.

- **Estudio profundo de las CNN basado en imágenes omnidireccionales.** Ya se han realizado estudios breves sobre este tema. Una CNN fue reentrenada mediante el uso de imágenes omnidireccionales con el objetivo de resolver una tarea de recuperación de habitaciones. Por lo tanto, en el futuro, se puede abordar una evaluación de los descriptores holísticos obtenidos de las capas intermedias. Este método de descripción puede producir resultados de localización exitosos, ya que pueden proporcionar enfoques de localización precisos y rápidos. Además, el paso de transformar la imagen omnidireccional a panorámica se podría ahorrar, ya que la CNN sería entrenada directamente con imágenes omnidireccionales. Lo cual implicaría un ahorro en el tiempo de computación. Además, este estudio también intentará construir una CNN de regresión que pueda estimar la posición de captura dentro del entorno.

- **Uso de redes neuronales siamesas.** En esta tesis, la idea principal consistió en obtener un descriptor holístico apropiado y también una medida apropiada de distancia que realizó con éxito la similitud entre imágenes. De esta manera, se demostró que las herramientas de aprendizaje profundo como CNN realizan descriptores holísticos exitosos. En esta línea, las redes siamesas son una versión similar de la CNN. Sin embargo, se alimentan con dos imágenes y la salida aporta un valor de similitud entre ellas. Por lo tanto, al usar la red neuronal siamesa, el modelo generará directamente la similitud entre imágenes. Por tanto, esta herramienta de aprendizaje automático se propone para cumplir dos objetivos.

  - Mediante el entrenamiento de la red siamesa con un par de imágenes, aprenderá a generar un valor de similitud de apariencia que puede ser indirec-

tamente proporcional a la distancia de captura de la imagen o un valor de similitud que puede ser indirectamente proporcional a la distancia de rotación.

– Esta red se puede entrenar con un par de imágenes que fueron capturadas en la misma posición, pero con diferentes condiciones de iluminación. Una vez que se entrena la red, se puede obtener un descriptor holístico de las capas intermedias. Por tanto, se deseará que el descriptor obtenido sea robusto frente a los cambios de iluminación.

# A Appendix: Set of Publications

The major contributions made in the present thesis are supported by four papers published in journals ranked in JCR (Science Edition). The metadata of these journal papers are presented next:

Journal Paper 1

Mapping and localization module in a mobile robot for insulating building crawl spaces. [41]
S. Cebollada, L. Payá, M. Juliá, M. Holloway, O. Reinoso
**Automation in Construction.** Vol 87, pp. 248-262 (March 2018)
ISSN:0926-5805. Ed. Elsevier
**JCR-SCI Impact Factor: 4.313**, Quartile **Q1**
Web: https://doi.org/10.1016/j.autcon.2017.11.007
DOI: 10.1016/j.autcon.2017.11.007

Journal Paper 2

Evaluation of Clustering Methods in Compression of Topological Models and Visual Place Recognition Using Global-Appearance Descriptors. [44]
S. Cebollada, L. Payá, W. Mayol, O. Reinoso
**Applied Sciences.** Vol 9(3), 377 (2019)
ISSN:2076-3417. Ed. MDPI
**JCR-SCI Impact Factor: 2.474**, Quartile **Q2**
Web: https://doi.org/10.3390/app9030377
DOI: 10.3390/app9030377

Journal Paper 3

Hierarchical Localization in Topological Models Under Varying Illumination Using Holistic Visual Descriptors. [45]
Sergio Cebollada, Luis Payá, Vicente Román, Oscar Reinoso
**IEEE Access.** Vol 7(1), pp. 49580-49595 (2019)
ISSN:2169-3536. Ed. IEEE
**JCR-SCI Impact Factor: 3.745**, Quartile **Q1**
Web: https://doi.org/10.1109/ACCESS.2019.2910581
DOI: 10.1109/ACCESS.2019.2910581

Journal Paper 4

A State-Of-The-Art Review on Mobile Robotics Tasks Using Artificial Intelligence and Visual Data. [40]
Sergio Cebollada, Luis Payá, Maria Flores, Adrián Peidró and Oscar Reinoso
**Expert Systems with Applications.** pp. 114195. (2020)
ISSN:0957-4174. Ed. Elsevier
**JCR-SCI Impact Factor: 11.0 (2019)**, Quartile **Q1**
Web: http://www.sciencedirect.com/science/article/pii/S095741742030926X
DOI: 10.1016/j.eswa.2020.114195

Preprints of these publications are appended next.

# Mapping and localization module in a mobile robot for insulating building crawl spaces

Sergio Cebollada [1], Luis Payá [1], Miguel Juliá [2], Mathew Holloway [3] and Oscar Reinoso [1]

[1] *Departamento de Ingeniería de Sistemas y Automática. Miguel Hernández University. Avda. de la Universidad s/n. 03202, Elche (Alicante), Spain ; E-Mails: { sergio.cebollada, lpaya, o.reinoso }@umh.es*

[2] *Q-Bot Limited, Block G, Riverside Business Centre, Bendon Valley, SW18 4UQ, London, UK; E-Mails: miguel.julia@q-bot.co*

[3] *Dyson School of Design Engineering, ImperialQuad-Core College London, 10 Princes Gardens, South Kensington, London, SW7 1NA. E-Mails: mathew.holloway@imperial.ac.uk*



**Abstract**

This paper presents a novel robotic system that applies spray foam insulation in underfloor voids in order to improve the energy efficiency of buildings. The work focuses on solving the mapping and localization problems in such environments, since they are a key factor in the autonomy of the robot. Solving these tasks in underfloor voids is especially challenging because the terrain is extremely uneven due to the presence of stones, bricks and sand. Within these environments, the robot should be able to localize itself and apply the insulation foam to the underside of the floor. The robot is equipped with a 2D laser sensor which permits building point clouds from several positions of the underfloor environment. The localization process is solved by estimating the position of the robot with respect to previously known positions. For this purpose, the alignment between point clouds is calculated. This paper describes two algorithms to robustly obtain the alignment between two positions. The proposed algorithms are tested with a set of point clouds captured with a laser scan in several environments under real working conditions. The results show that the localization problem can be solved in such challenging underfloor voids by using depth information.

*Keywords:* Autonomous Mobile Robot, Localization, Point Cloud Alignment

## 1. Introduction

There are many buildings in Europe and around the world which have voids between floor and foundations due to building methods. This kind of uninsulated suspended timber floors can be a key factor in heat loss as some studies show [1] [2]. This includes conductive heat loss to the ground and also infiltration of cold air through the underfloor environment and wooden flooring. Taking this fact into account, the energy efficiency of such existing buildings could be improved by means of under-floor insulation. The process to insulate under the floor usually consists in removing the carpet and floorboards, applying rigid panels or rolls of insulation and finally, putting everything back together. It causes a large amount of disturbance to the building occupants since often, they must vacate the premises during the installation. To make this process less disruptive and faster, a robotic vehicle can be used. This robotic vehicle should be able to access to voids, to move autonomously and to apply foam insulation. An autonomous mobile robot that could manoeuvre around the void and apply insulation where needed should be chosen. Within this group, we can distinguish three main types: wheeled, legged and aerial robots. When choosing one of these three types, two main requirements must be considered. First, an umbilical hose must be attached to the robot to transmit power and supply the robot with the foam insulation. To meet these requirements, the hose would weigh around 3.5 kg per linear meter [3]. Second, a spray nozzle must be mounted on the robot to eject the foam onto the underside of the floor. During this process, it will exert a pressure on the robot. Therefore, the robot must remain stable. For these reasons neither the legged nor the aerial robots (that also create too much dust and disturbance) would operate correctly. Thus, a wheeled or tracked platform is used in this work.

Q-bot has developed a novel robot to carry out the insulation task. The robot, shown in fig. 1, is composed of 4 small wheels (each one is individually driven by a motor and a gearbox with a peak power of 65 W), a front horizontal laser and an actuated camera-laser system (3D scanner). Furthermore, it has a spray nozzle which ejects insulation foam. Further information about the robot specifications can be found in [3] and also on the vendor website (`http://www.q-bot.co`). Initially, the insulation task

has been developed through teleoperated assistance, what means that the robot vehicle is driven by an expert human operator [4].

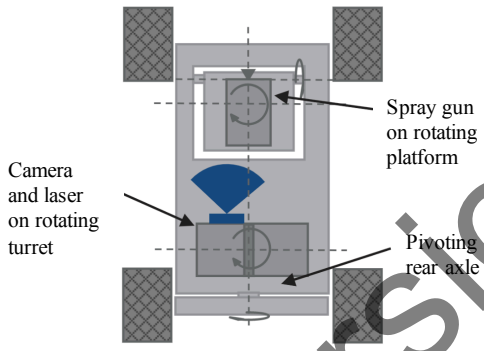



Figure 1: Bird eye's view of the robot and main components.

When a mobile robot is used to tackle this task, it is necessary that the robot move through the environment in order to apply the foam insulation on the required areas. When teleoperated assistance is used, the human operator recognizes and interprets the environment and takes the decisions about the movement of the robot and the task. However, despite the successful use of teleoperated robots, the use of autonomous robots would improve performance and speed while reducing cost. They would be able to complete the task without the continuous supervision of specialized workers.

If an autonomous development is desired, many issues appear and they must be addressed accurately. First, accessing to the environment can be carried out by making an access hatch and putting the robot into the underfloor environment. Once the robot is inside, a number of challenges must be overcome. The main problems come up because the terrain tends to be extremely uneven, as stones, bricks fragments or sand are often present. This way, the robot has to move through irregular 3D paths, considering also the presence of unknown obstacles. Two sample images of such typical underfloor environments are presented in figure 2. The robot must tackle the insulation task in this type of challenging and previously unknown environments.

Figure 2: Images of typical underfloor environments where the registration is difficult due to their characteristics.



To address autonomously this task, firstly, the robot must be capable of mapping the environment with enough accuracy and reliability in order to recognize the zones where insulation is required and, at the same time, be able to estimate accurately its position within the map. This process is known as *Simultaneous Localization And Mapping* (SLAM). In order to tackle the mapping and localization, it is necessary to use one or more sensors to obtain some information from the environments. On the one hand, the SLAM task has been traditionally addressed using range sensors, such as laser, which measure the distance to the surroundings and usually lead to models that show occupied and non-occupied zones [5] [6]. On the other hand, vision systems can also be used for this purpose and much research is being carried out on mapping and localization using cameras [7] [8]. Visual approaches try to build a map of the environment using sets of features obtained from different points of view. Many researchers have addressed successfully the SLAM task in controlled environments through these vision systems such as Davison *et al.* [9] who use

a single camera. Nevertheless, it is necessary to highlight the difficulty of the underfloor environments. The presence of elements such as dust, sand, poor illumination or shadows makes the mapping and localization process extremely complex.

The extreme unevenness of the floor is an additional issue to be taken into account (see fig. 2). Owing to it, the movement of the robot is not plane at all and it can be considered a 6 DoF (Degrees of Freedom) movement. Considering all these features and challenges, both a laser and a vision system are chosen to accurately build a map of the environment and installed on the robotic platform.

Using these sensors, the mapping and localization are carried out following the next process. First, from a specific pose (position and orientation) of the environment, a scanning process is performed. During it, the sensors capture information on 360 deg. around the robot. The result is a 3D local map which consists of a point cloud that combines both depth and color information. Once the local map has been built from a specific pose, the robot will move a relatively long distance to a new, unknown pose. To estimate this new pose, the environment will be scanned again and a new local map (point cloud) will be built. The translation and rotation from the first to the second pose can be estimated by comparing these two local maps. After repeating this process from several poses, the set of local maps will compose a complete description of the environment (global map). This global map can be used not only to estimate the position of the robot while it moves, but also to determine the physical properties of the underfloor environment in order to control the spray gun and, after the insulation step, in order to validate if all the areas are correctly covered with foam.

Taking these facts into account, obtaining a robust and exact global map is very important. With this objective, having an accurate knowledge of the pose where each local map was captured is crucial. This is why this work focuses on this problem: estimating the current pose of the robot with respect to the previous one. The problem will be solved by comparing the point clouds obtained from both poses using a registration approach, whose result is a transformation matrix that contains the rotation and traslation experienced by the robot from the first to the second pose. No information on odometry will be used because the extreme unevenness of the floor introduces a severe error on it (owing to shifting, slipping and orientation changes). This way, the global map is expected to

be robust against these phenomena.

Therefore, in this paper we present a procedure designed to solve the alignment between two consecutive locations and to build a global map of the environment so that the robot can autonomously develop the insulation task in this kind of environments. In this regard, the work is based on the autonomous surveying robot architecture introduced in [4]. In this previous work, a system was proposed for selecting the next best position for performing a 3D scan. Multiple scans were aligned using the Iterative Closest Point (ICP) algorithm and merged together into a global map model. However, this system depends on a correct functioning of the ICP algorithm which is prone to fail or converge to a local minimum due to the complexities of the underfloor environments. Consequently, we present an algorithm that solves the registration in such environments. Therefore, the two main contributions of this paper are a novel method to improve the results of the ICP algorithm and a system for making the global mapping process more robust to alignment failures.

The remainder of the paper is structured as follows: Section 2 outlines some previous related works. After that, Section 3 shows the acquisition system which was used in the experiments. Next, section 4 presents the method proposed to obtain the alignment between poses, while section 5 introduces the algorithm to solve wrong alignment cases. Section 6 presents the experimental results and the discussions about the results. Finally, section 7 outlines the conclusions and future research lines.

## 2. State of the art

Some authors have made use of mobile robots in tasks related to construction, such as inspection and maintenance. As an example, Yu *et al.* [10] present a semiautomated inspection system to detect concrete cracks in tunnels, based on image processing. Also, Tseng *et al.* [11] propose some strategies to inspect pavement for maintenance and rehabilitation activities, using a mobile robot. They develop some algorithms to control the motion of the surveying robot, including the use of computer vision, and use a virtual environment to test them. However, none of these proposals require a map of the environment.

Tan *et al.* [12] study the relationship between the mobile robot and the environment, analyzing how inclusive the environment is for the robot.

They also establish a taxonomy to classify the robot-environment interaction. In our application, the environment cannot be changed to favor the robot inclusiveness and the environment imposes a negative impact on the robot (which is unintentional but very robot-unfriendly), since it is full of dirt and construction debris. This way, the floor is very uneven and therefore the movement of the robot is not contained in a plane. Also, some parts of the environment will change substantially as the robot sprays the foam insulation because it will cover some areas. Robust algorithms must be implemented to overcome such difficult situations.

Hamledari *et al.* [13] also use computer vision in construction, with the goal of detecting components in under-construction indoor partitions, such as studs, insulation and electrical outlets. This task requires image processing algorithms capable of coping with changing viewpoints, highly cluttered scenes, occlusions, diverse lighting conditions and the achromatic characteristics of some objects. In the case presented here, not only should elements be detected, but also the robot should recognize features and localize itself in a transitory environment due to the fact that the underside of the floor is being insulated. Hence, the task is much more complex.

With regard to the mapping and localization process, many authors have used only visual features to solve the alignment problem. This method consists in detecting keypoints in images which have been obtained from different poses. Next, each keypoint is associated a set of values (descriptor) that characterizes it and distinguishes it from the others keypoints. Through the descriptors, a set of matches between keypoints from different poses are obtained and therefore, a matrix transformation can be calculated. SIFT [14], SURF [15] and BRISK [16] are three popular extraction and description algorithms. However, these approaches based on visual keypoints present many drawbacks in underfloor environments, such as their sensitivity to changes of lighting conditions. These environments may be completely dark and the light is only provided by light sources installed either on the robot and/or in a specific position of the environment. Hence, the shadows will generate inaccuracies both in the keypoints detected and in their descriptors.

Considering the disadvantages of a purely visual approach, another possibility consists in using the depth data obtained from laser scanners. These devices can capture point clouds from specific poses of the environment. Iterative Closest Point (ICP) [17] is the most used method to solve

the registration between two point clouds. Many variations of this method have been proposed, such as the one presented in [18], where plane-plane matches are considered instead of point-point, or [19], where SVD (*Singular Value Decomposition*) is used to calculate the transformation matrix. Jost and Hügli use a heuristic approach to find closest points and hence reduce the complexity and accelerate the process [20].

Additionally, there are some authors who use both visual and depth data to estimate the position of the robot and follow the process shown in fig. 3. Initially, the following information is available: one image and one point cloud obtained from the pose 1 and one image and point cloud obtained from pose 2. The process starts working with the visual information. The keypoints are extracted from both images, described and matched. Then, inconsistent correspondences are rejected (normally by RANSAC [21]) and an initial transformation matrix is obtained. This matrix gives a coarse alignment and is used subsequently as initial matrix for the ICP algorithm. This algorithm is run with the two point clouds and, as a result, a fine alignment and a more accurate transformation matrix is obtained. This approach has been used by a number of authors. Endres *et al.* [22] used either SURF, SIFT or ORB for Pairwise Feature Matching. Henry *et al.* [23] also used SIFT. However, instead of ICP point-to-point, they used ICP point-to-plane. More recently, dos Santos *et al.* [24] use visual information (SIFT) for the coarse alignment. However, instead of ICP they use SLIC (Simple Linear Iterative Clustering) in the fine alignment.

Low [25] presented an alternative method that estimates the translation and rotation by accumulating the point-to-plane constraints of all the correspondences in a matrix. Similar to it, a modified version of ICP based on weighting the points has arisen. Once the salient points are detected, some extra information is used to apply a weighting value. This way, each point would have more or less importance according to its weight. The use of salient points and additional information in order to assist the registration process gives more reliability and robustness even in situations where the geometry information is not enough. Cappelletto *et al.* [26] use the color information to weight through computing the distances between the matched points in the ICP algorithm. Also, Xie *et al.* [27] use the spatial distances of the SIFT descriptor to balance the errors during the error minimization process. For more information, an extensive comparison about the presented methods can be found in [28].

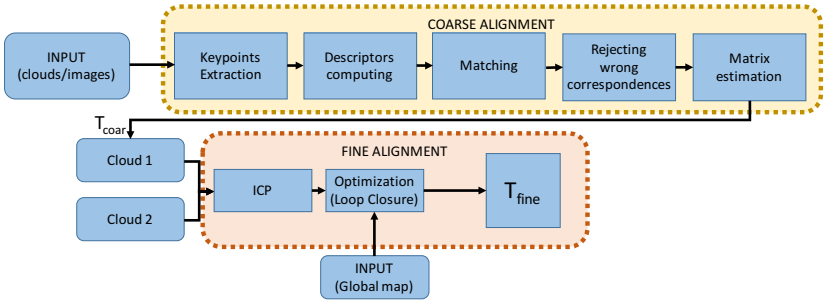



Figure 3: Schematic overview of the usual localization approach.

The up to now developed algorithms work quite well under many standard circumstances and some of them can provide accurate results in real time. Nevertheless, they are prone to fail in environments like the target one due to the lack of light, little color variety and the unstructured geometry. On the one hand, the high uncertainty of the robot position due to the irregularities of the terrain, as well as the irregular and unstructured appearance of the environment make the obtained keypoints untrustworthy for the estimation of the transformation matrix between consecutive poses. On the other hand, the characteristics of these environments (with elements on the floor whose position can change as the robot moves) and the upper planes information (whose shape will change after insulation) make that the classical depth registration methods do not present good results. Therefore, this paper introduces a robust procedure which permits building a global map of the environment for the insulation task and its subsequent inspection.

## 3. Data acquisition system

The data acquisition system is composed of a 2D laser sensor and a monocular camera. They are attached to a turret that rotates around its vertical axis, which is perpendicular to the basis of the robot. The laser is mounted to scan in a vertical plane, which changes as the turret rotates. This system acquires complete 360 degrees information from the environment around the robot. Therefore, from a specific pose, the robot

can capture a complete scan of the environment and a set of RGB images (see figure 4 (a)). The complete system is described in this section. First, subsection 3.1 describes the reference systems. After that, subsection 3.2 describes the image acquisition process and the process to assembly the 3D point cloud. At last, subsection 3.3 explains how color information is added to the point cloud.

### 3.1. Reference frames

Three reference frames are used in this work: the robot, laser and camera reference frames. First, fig. 4 (a) shows the robot reference frame. $X_R$ is the vertical axis and $Y_R, Z_R$ are the axes which define the plane of movement of the robot. Second, fig. 4 (b) shows the camera reference system, whose axes are $X_C, Y_C, Z_C$. Finally, fig. 5 shows the laser reference system $\{X_L, Y_L, Z_L\}$. Both the laser and camera frames rotate around their $X$ axes. The laser provides a set of distance readings $\rho_i$ measured at different angles $\theta_i$. These readings can be expressed as 3D points in the laser frame $q_i^{[l]} \in \mathbb{R}^3 = [\rho_i \cos\theta_i \quad \rho_i \sin\theta_i \quad 0]^T$ and they can be transformed to the robot frame by:

$$q_{i,j}^{[f]} = R_{\phi_j} T_L q_{i,j}^{[l]} \tag{1}$$

where $T_L \in \mathbb{SE}_3$ is the transformation that relates the calibrated position of the laser in the robot frame and $R_{\phi_j} \in \mathbb{SO}_3$ is the rotation matrix that expresses that the turret has rotated an angle $\phi_j$. We should note that, for simplicity of notation, the conversion between 3D-vectors and the corresponding homogeneous 4D-vectors has been omitted.

### 3.2. Data acquisition

During a complete acquisition process, the robot is steady at a specific pose $\vec{p}_s$ and both camera and laser capture data while the turret spins a complete revolution around the vertical axis. This way, these data contain environment information on 360 degrees around the robot. On the one hand, about the camera, each image $K$ is acquired from the position corresponding to the pose $s$ of the robot and is defined in a set as $K_{s,r} \in \mathbb{R}^{N_x \times N_y}$, where $N_x \times N_y$ is the resolution of the image (in this case $1448 \times 1928$ pixels) and $r$ defines the orientation of the turret. In this work, 36 images are captured from each pose, r=0,1,2,...,35, with a change of orientation equal to 10 degrees between consecutive values of $r$ (fig. 4 (b)) .

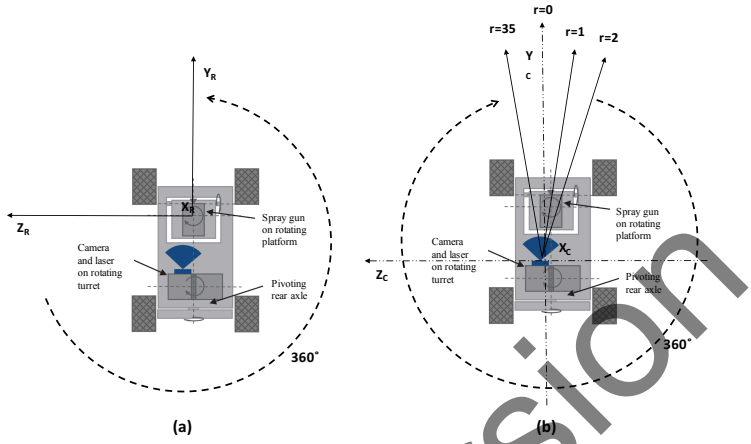



Figure 4: (a) Sensors of the robot and robot reference frame. (b) Schematic overview of the camera reference frame and the image acquisition process.



On the other hand, the laser system performs several scans during this process. Each scan covers a vertical plane (fig. 5(a)) in which the resolution is equal to 0.36 deg. (angle between two consecutive beams). This vertical scan covers 240 deg., but only 120 deg. (the central ones) are used. Also, the motor which spins the turret has 2400 steps. Hence, the minimum angle between two consecutive scan planes is equal to 0.15 deg. (fig. 5(b)). Therefore, a point cloud formed by around 800.000 points is created from each pose $\vec{p}_s$. We name this cloud $P_{original,s}$.

### 3.3. *Adding color to the point cloud*

The color of the corresponding pixels in the images is associated to each point by:

$$c_{\{i,j\},k} = I_k(\pi(H_{cal} T_{cam} R_{\phi_j} q_{i,j}^{[f]})) \tag{2}$$

where $R_{\phi_j} \in \mathbb{SO}_3$ is the rotation matrix corresponding to the turret at the angle $\phi_k$ at which the image was acquired. $T_{cam} \in \mathbb{SE}_3$ is the transformation corresponding to the calibrated camera pose in the robot frame. $H_{cal}$ is the calibrated camera matrix and $u = \pi(x)$ is a function which performs the dehomogenization of $x \in \mathbb{R}^3 = (x, y, z)$ in order to obtain $u = (x/z, y/z)$.

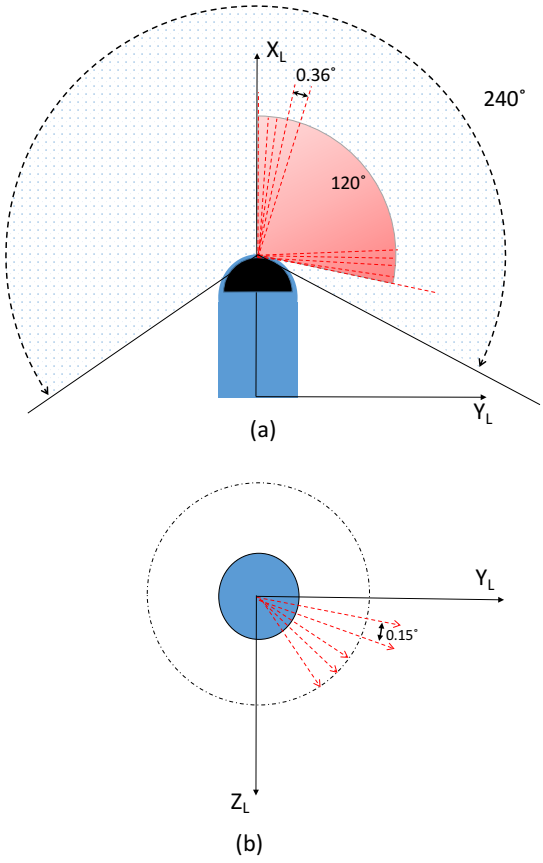



Figure 5: Schematic overview of the laser system. The laser reference frame rotates around the $x_L$ axis. (a) shows one laser scan and (b) is a bird eye's view of the scan planes during a complete acquisition process.

$I_k : \Omega \rightarrow \mathbb{N}^3$ is the subpixel mapping between the image space domain $\Omega \subset \mathbb{R}^2$ and the color values corresponding to the rectified image $K$.

The color information is essential once the foam insulation has been sprayed, to check that the necessary areas have been correctly covered. This is why it is added to the cloud. However, the color information is

not used during the localization process as it does not contain distinctive information in underfloor environments, because of their low variety of colors.

## 4. Alignment between consecutive poses

Many of the algorithms which use ICP for registration present a good balance between accuracy and computing time. However, their main problem is the necessity of a relatively accurate initial estimation to converge to the global optimum instead to a local minimum. The high similarity of the data acquired in the environments of this work exacerbates this problem. In addition, the poses obtained through the odometry are not reliable due to the characteristics of the terrain. Therefore, this information could not be used neither as initial estimation nor as ground truth information.

In order to solve the registration task, two possibilities have been commonly used. The first option uses the visual information for the coarse alignment step and the point clouds in the refinement step (see fig. 3). First, this family of registration methods extracts keypoints from the visual information. Second, a descriptor is computed for each keypoint. This value is computed through neighborhood information and the result is a vector that characterizes the keypoint and makes it possible to distinguish it from other keypoints. Third, the keypoints from one image are matched with the keypoints from another image captured from a different robot pose. Fourth, alignments between different images are established. Normally, there are some matches which are erroneous, hence, a rejector step can be applied (a common method is RANSAC). Fifth, after establishing robust correspondences between keypoints, the transformation matrix is computed and it is used as initial matrix for the refinement step (ICP algorithm), which provides a more accurate matrix.

In the second choice, the depth information is used in the coarse alignment. The keypoints are extracted from the scenes and their 3D positions are calculated using the laser information. After that, the descriptors are computed using depth information and a matching is carried out. Then, like in the first case, a rejector step is tackled and a coarse transformation matrix is computed through the correspondences between keypoints. Finally, through ICP, a more accurate matrix is obtained.

Despite all the possibilities that the visual information can provide, this is not very useful in the proposed environment. The underfloor voids

are small spaces where there is not much light. Walls are composed by bricks which do not contain characteristic information. The underside of the floor presents the same problem because it is usually composed of some beams which are identical and evenly distributed. Therefore, the visual information is not reliable and should be avoided during the alignment process. Consequently, a version of the second option is developed and it is fully explained in the next subsections. The initial data are two point clouds captured from two poses *s* and *s-1* ($P_{original,s}$ and $P_{original,s-1}$). The objective is obtaining the transformation matrix (relative position and orientation) between these two poses, using a registration approach with the two point clouds. The proposed algorithm consists of three main steps:

- Points selection
- Registration
- Validation

In the following subsections, the three parts will be explained.

### *4.1. Points selection*

The original point clouds are composed of a large amount of points. This is due to the fact that the insulation task requires accurate information. If the ICP algorithm used all this information, the computing time would be exceedingly high. Also, the information collected from the top and bottom planes, especially the wooden beams in the upper part of the environment, would be harmful as it would lead to confusion. As pointed out before, these beams are almost equal and equidistant, hence, many points might be erroneously considered as correctly matched. Consequently, an erroneous alignment could be accepted as successful because of the large amount of matched points (section 4.3 will formalize the criteria to consider an alignment as correct or incorrect). Fig. 6 shows the result of the alignment process of two complete point clouds captured in a sample environment (fig. 2). This case is clearly unsuccessful and this kind of environments is very prone to present such wrong results due to the geometry of the top and bottom planes. This is why we propose removing this information before the registration process in this work.

Therefore, taking the previous premises into account, a process is carried out to select some of the points of the original cloud. This process consists of two steps. First, a homogeneous filtering is carried out to reduce considerably the number of points. Second, a segmentation is performed

to select only the points situated in the planes which are the most helpful to reach a successful registration. Some functions of the Point Cloud Library (PCL) [29] are used for these purposes. More details on both steps are given in the next two paragraphs.

First, the homogeneous filtering is carried out following the diagram shown in fig. 7. Initially, a VoxelGrid filter is used. It consists in creating a 3D voxel grid (in this case, a 1 x 1 x 1 cm cube) over the point clouds. Afterwards, the points within each cube are approximated with their centroid. Finally, a random sampling filter is applied, which randomly selects a number of points of the resulting cloud and discards the rest. The more points are removed, the faster is this step and also the following ones. However, removing too many points can be counter-productive because the subsequent ICP algorithm might not work well. This way, despite the random filtering, the structure should be kept up. After several tests, the reached conclusion is that maintaining the 30% of the points is the optimal value in order to balance speed and reliability. From this step, a downsampled cloud is obtained for each pose of the robot ($P_{downsampled,s}$ and $P_{downsampled,s-1}$).

Second, the objective of the segmentation consists in removing those points that belong to the the top and bottom planes. To tackle this step, the point cloud obtained after step 1 is clustered into planes. The planes whose normal vector is parallel to the $X$ axis are considered as the top and the bottom ones and the rest are considered as walls. The algorithm removes the top and bottom planes, discards the points that belong to them and keeps the rest. To consider the presence of beams in the top planes and remove their information, once the planes that are situated over the robot have been detected, the one with the lowest height is extracted and all the information in and above it is removed. As a result, lighter clouds are obtained ($P_{filtered,s}$ and $P_{filtered,s-1}$). These are the clouds which will be used in the subsequent registration process. Fig. 8 shows the whole process with an original point cloud. Fig. 8(a) is $P_{original,s}$, fig. 8(b) is $P_{downsampled,s}$ and fig. 8(c) is $P_{filtered,s}$.

To sum up, the process starts with the original point clouds ($P_{original}$), obtained from the data acquisition, which contain more than 800.000 points. Then, the downsampled point clouds ($P_{downsampled}$) are obtained after applying VoxelGrid and random sampling filters. Finally, the filtered point clouds ($P_{filtered}$) are obtained through segmentation and removal of the top and bottom planes. These clouds will be used for the registration step

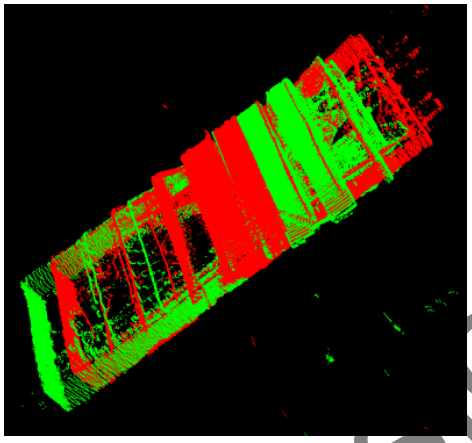


Figure 6: Results of a sample registration process, using two point clouds captured from different poses in a sample environment. The alignment is unsuccessful.

(subsection 4.2).

### *4.2. Registration*

The next step consists in carrying out a registration process between the two filtered clouds ($P_{filtered,s}$ and $P_{filtered,s-1}$). The results of this process are: (a) the transformation matrix ($T_{s,s-1}$) that relates both poses; (b) the number of matched points ($N_{s,s-1}$) and (c) the $EFS_{s,s-1}$ (Euclidean Fitness Score) defined in eq. 3.

$$EFS_{s,s-1} = \sum_{j=1}^{N_{s,s-1}} dist(P_s^{est}(j), P_s(j))^2 \tag{3}$$

where $dist(P_s^{est}(j), P_s(j))$ is the Euclidean distance between the j-th matched point of the clouds $P_s^{est}$ and $P_s$. $P_s^{est} = T_{s,s-1} \times P_{s-1}$.

Classical ICP algorithms may present wrong results as far as the detection of the relative orientations between poses is concerned. This is due to the fact that the algorithm may converge to a local minimum when the target environment presents some symmetry. The sub-soil environments modeled in this work are very prone to present this problem as they have

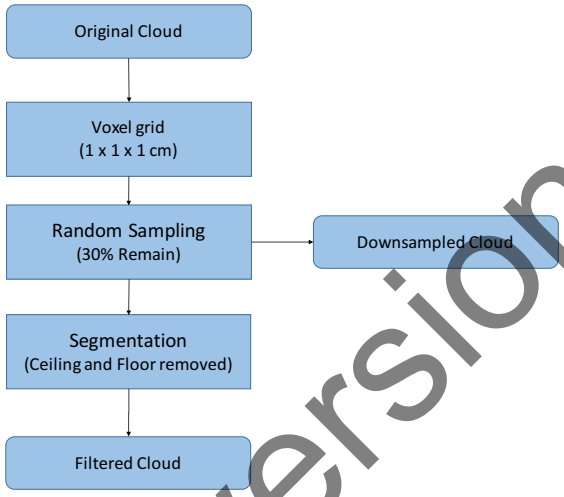



Figure 7: Filtering stages.

two main directions in the horizontal plane (those perpendicular to the walls) and the walls may be erroneously matched with their opposite ones.

Since the odometry information is not trustworthy, it can not be used to have an initial estimation to apply the ICP algorithm. To overcome this issue, the next four initial conditions are considered: no traslation and four rotations (0, 90, 180 and 270 degrees) around the vertical axis. After that, the classical ICP algorithm is run four times in parallel (one for each of the four initial estimations).

Among the four alignments carried out, the one that maximizes eq. 4 is considered the optimal one.

$$\max\{\alpha N^k_{s,s-1} + \beta \frac{1}{EFS^k_{s,s-1}}\} \tag{4}$$

Where $\alpha$ and $\beta$ are two weighting values which were tuned empirically

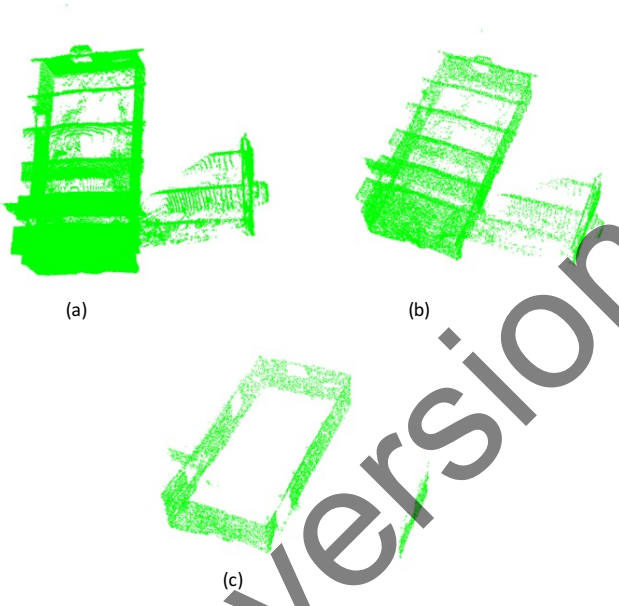



Figure 8: Points selection process. (a) Original cloud. (b) Downsampled cloud after VoxelGrid and random filtering. (c) Resulting cloud after removing top and bottom planes.



according to the characteristics of the environment and $k$ is an index that defines the initial orientation ($k = 0, 90, 180, 270$).

The process to select the best alignment between the two consecutive poses is shown in fig. 9. In conclusion, in this step, four initial versions of the cloud $P_{filtered,s}$ are considered and aligned with $P_{filtered,s-1}$ using ICP, and the transformation matrix $T_{s,s-1}$ of the optimal alignment is retained.

### *4.3. Validation*

The previous step provides the optimal alignment matrix $T_{s,s-1}$ once the four possibilities have been evaluated. However, this does not ensure that the alignment is correct. Therefore, the algorithm should include a validation step.

In order to validate this alignment matrix, the downsampled point clouds ($P_{downsampled}$) are considered (subsection 4.1). In the previous subsection, $P_{filtered}$ have been used due to the fact that removing the top and

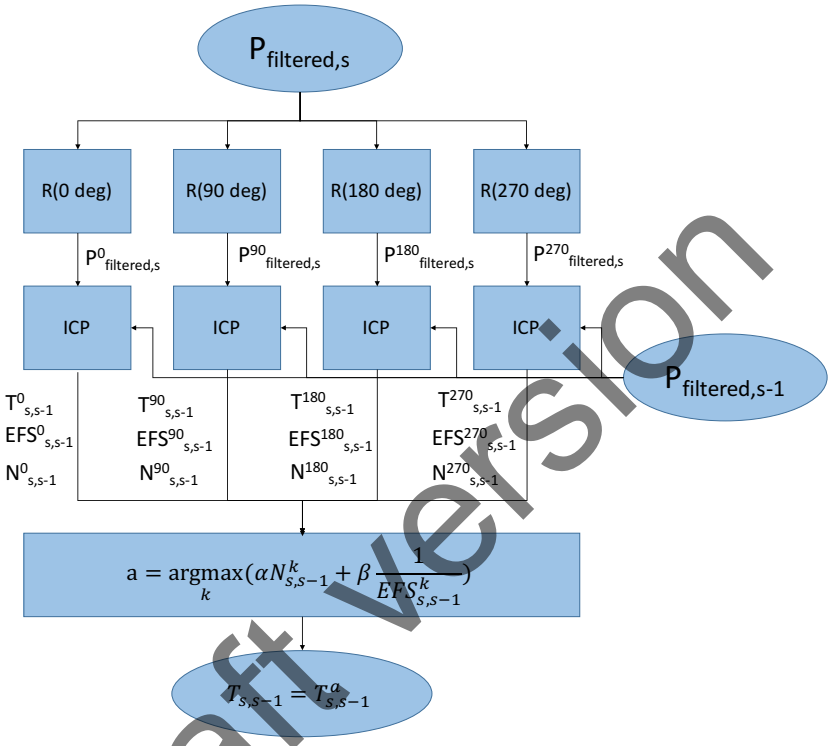



Figure 9: Algorithm diagram. Firstly, the cloud $s$ is rotated 4 times with 0, 90, 180 and 270 degrees. Each resulting cloud is compared with the cloud *s-1* using ICP. Through the EFS (Euclidean Fitness Score) values and the number of matched points, the optimal transformation matrix is chosen.



the bottom planes was beneficial for the registration process. However, in order to check if this process was really successful, this information must be taken into account. Otherwise, failures along the $x$ axis could not be detected.

For the validation purpose, three parameters are considered:

- the EFS.
- the ratio of correspondences over the number of points in $P_{downsampled,s}$

(eq. 5).

- the ratio of correspondences over the number of points in $P_{downsampled,s-1}$ (eq. 6).

$$r_s = \frac{N_{s,s-1}}{N_{points(s)}} \tag{5}$$

$$r_{s-1} = \frac{N_{s,s-1}}{N_{points(s-1)}} \tag{6}$$

where $N_{points(s)}$ and $N_{points(s-1)}$ are the number of points in the downsampled point clouds $P_{downsampled,s}$ and $P_{downsampled,s-1}$ respectively. $N_{s,s-1}$ is the number of correspondences between $P_{filtered,s}$ and $P_{filtered,s-1}$.

Using only one of these two ratios may seem enough to validate the result. However, although one percentage might be high enough to satisfy the validation threshold, the other percentage might not reach that threshold, and consequently, the alignment should be rejected.

Therefore, given the equation:

$$\left[\eta r_s + \lambda r_{s-1} + \mu \frac{1}{EFS} > Validation_{th}\right] \tag{7}$$

where $\eta$, $\lambda$ and $\mu$ are weighting values and $Validation_{th}$ is the established threshold. If the sum of the left side is higher than the established threshold ($Validation_{th}$), then the alignment between the two point clouds will be considered successful and accepted. Otherwise, the alignment will be considered unsuccessful and rejected.

It should be emphasized that even if the registration between $P_{filtered,s}$ and $P_{filtered,s-1}$ is not correct, the inverse registration (registration between $P_{filtered,s-1}$ and $P_{filtered,s}$) may be correct. Therefore, in an unsuccessful validation case, the inverse registration will be carried out and validated in order to know if it satisfies eq. 7.

In conclusion, through this third step, the alignment matrix $T_{s,s-1}$ obtained after the step 2 is validated. Two possibilities may happen:

1. Equation 7 is satisfied. In this case we can consider that a successful alignment has been carried out between the current pose and the previous one. The matrix $T_{s,s-1}$ will be considered correct.

2. Equation 7 is not satisfied. In this case, the alignment matrix $T_{s,s-1}$ should not be accepted as valid. This situation must be taken into account because the current pose $\vec{p}_s$ cannot be estimated with respect to the previous one $\vec{p}_{s-1}$. The matrix $T_{s,s-1}$ will be considered incorrect.

## 5. Algorithm to Locate Lost Poses

Despite the robustness of the proposed alignment algorithm, the preliminary experiments showed that a number of unsuccessful cases could appear when trying to estimate the current pose with respect to the previous one. Typically, these cases tend to appear either when the distance between the two poses is relatively high or when there is a big difference between the captured environments (i.e. the robot has entered in a different room). Therefore, it is necessary to develop an algorithm to estimate the pose of the robot when the registration with the previous pose was not successful. Not only must this algorithm try to align poses which were not well aligned, but it also has to locate poses which did not meet the validation condition.

After the alignment step (which was explained in the previous section), with the aim of obtaining the alignment matrix between $P_{filtered,s}$ and $P_{filtered,s-1}$, three different cases are taken into account (see algorithm 1):

1. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is correct (the alignment matrix $T_{s,s-1}$ obtained from the alignment algorithm is considered valid) and the previous pose ($\vec{p}_{s-1}$) is correctly located. A pose is considered well located when its position is well known through a successful alignment between that pose and the previous one, which position is also known.

2. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is not valid (the alignment matrix $T_{s,s-1}$ obtained from the alignment algorithm is not correct because eq. 7 is not met.).

3. The alignment between $P_{filtered,s}$ and $P_{filtered,s-1}$ is correct (i.e. the alignment matrix $T_{s,s-1}$ is considered valid), but the location of the previous pose ($\vec{p}_{s-1}$) is not considered valid.

About the first possibility, if the alignment between the two consecutive poses is considered valid and the previous pose ($\vec{p}_{s-1}$) is correctly located, then the current pose $\vec{p}_s$ will be estimated through the matrix $T_{s,s-1}$. Hence, the pose $\vec{p}_s$ will be considered as correctly located.

As for the second possibility, the alignment between the two consecutive poses is considered wrong. In this case, a near pose would be searched in order to obtain a good alignment (see algorithm 2). For this purpose, despite the fact that $T_{s,s-1}$ is not correct, this matrix is used to make a rough estimation of the pose $\vec{p}_s$ with respect to $\vec{p}_{s-1}$. Afterwards, the distances between the pose $\vec{p}_s$ and the previous ones ($\vec{p}_n$ where $n = 0,...,s-2$ are known) are calculated by using:

$$d_{s,n} = \sqrt{(x_s - x_n)^2 + (y_s - y_n)^2} \quad ; \quad n = 0,...,s-2 \tag{8}$$

where $(x_s, y_s)$ and $(x_n, y_n)$ are the Cartesian coordinates of the poses $\vec{p}_s$ and $\vec{p}_n$ within the map. Once the distances are calculated, the previous poses are sorted according to the distance and then, the registration between the cloud $P_{filtered,s}$ and the closest pose's cloud is attempted. The registration step is repeated with the following closest poses' clouds while the registration is unsuccessful.

At this time, two situations can occur. The first one happens when a good alignment with a pose $\vec{p}_n$ is obtained. In this case, the alignment problem would be successfully solved for the pose $\vec{p}_s$. The transformation matrix would be stored and the pose $\vec{p}_s$ would be located in the map. The second situation occurs when a successful alignment is reached with any previous pose. In this case, $\vec{p}_s$ would be saved in a list of poses which are pending to be located. The poses which are in the pending list will not be considered in subsequent registration attempts because their location is not well known.

Finally, as far as the third possibility is concerned, if the alignment between the two consecutive poses is considered valid but the previous pose ($\vec{p}_{s-1}$) is not correctly located (i.e. it is in the pending list), then, the registration of $P_{filtered,s}$ will be tackled with the closest clouds until a good alignment is obtained. If $\vec{p}_s$ is successfully aligned with any of the well located poses, the transformation matrix will be stored and the pose $\vec{p}_s$ will be included in the map. Moreover, $\vec{p}_{s-1}$ will be removed from the pending list and it will be considered correctly located too (see algorithm 3).

The algorithm includes a final step, which is always carried out when a pose has been successfully aligned. After locating the new pose, the algorithm tries to align it with each pose which is on the pending list. Although this step increases the computing time, it is very interesting because the amount of poses which are not located yet could be reduced. Hence, more information about the map and the robot movement could be obtained.

**Algorithm 1** Online algorithm

```
if Registration(P_filtered,s, P_filtered,s−1) OK then
    if p⃗_s−1 Located then
        Store(T_s,s−1)
        Locate(p⃗_s)
    else
        Case B
    end if
else
    Case A
end if
```

## 6. Experiments and Results

This section presents the experiments that have been carried out in order to show that the proposed algorithms solve the registration problem in environments that present a challenge for characterization.

The algorithms were run in a PC with two CPU Quad-Core Intel Xeon (®) at 2,8 GHz. In order to develop the experiments, some sets of laser measurements were taken by the robot from different poses of several environments and one point cloud per pose was assembled. So, there is a point cloud associated to each pose of the robot within the environment ($P_s$ is the point cloud in the pose $\vec{p}_s$). As mentioned above, the characteristics of the terrain do not permit obtaining accurate enough odometry measurements. Consequently, the quality of the computed alignment will be measured through confidence values which indicate the quality of the estimated transformation matrix. Therefore, an alignment matrix (T) will be considered either correct or rejected through the criteria presented in subsection 4.3.

**Algorithm 2** Case A

1: $T_{s,s-1}$ is not accurate.
2: $p_{correct}$: Array of located poses.
3: $v$: Array of correct poses. Sorted by distance.
4: $d_k$: Array of distances between $\vec{p}_s$ and the rest of correct poses $\vec{p}_i$.
5: $P^{est}_{filtered,s} = T_{s,s-1} \times P_{filtered,s-1}$
6: **for** i=0; $(i < s-1)$; i++ **do**
7: $\quad d_k(i) = \sqrt{(x_s^{est} - x_i)^2 + (y_s^{est} - y_i)^2}$
8: **end for**
9: $v = sort\{p_{correct}, d_k\}$
10: **for** j=0; $\{j < (N_{poses} - 2)$ and $Registration(P_{filtered,s}, P_{filtered,v(j)})$ is *WRONG*$\}$; j++ **do**
11: $\quad$ **if** $Registration(P_{filtered,s}, P_{filtered,v(j)})$ OK **then**
12: $\quad\quad$ Store$(T_{s,v(j)})$
13: $\quad\quad$ Locate$(\vec{p}_s)$
14: $\quad\quad$ END CASE A
15: $\quad$ **end if**
16: **end for**
17: $List_{pending} \leftarrow \vec{p}_s$
18: END CASE A

**Algorithm 3** Case B

$\vec{p}_{s-1}$ is not accurate.
$T_{s,s-1}$ is accurate.
$p_{correct}$: Array of located poses.
$v$: Array of correct poses. Sorted by distance.
$d_k$: Array of distances between $\vec{p}_s$ and the rest of correct poses $\vec{p}_i$.
$P^{est}_{filtered,s} = T_{s,s-1} \times P_{,filtered,s-1}$
**for** i=0; $(i < s-1)$; i++ **do**
    $d_k(i) = \sqrt{(x^{est}_{filtered,s} - x_{filtered,i-})^2 + (y^{est}_{filtered,s} - y_{filtered,i})^2}$
**end for**
$v = sort\{p_{correct}, d_k\}$
**for** j=0; $\{j < (N_{poses} - 2)$ and $Registration(P_{filtered,s}, P_{filtered,v(j)})$ is *WRONG*$\}$ ; j++ **do**
    **if** $Registration(P_{filtered,s}, P_{filtered,v(j)})$ OK **then**
        Store($T_{s,v(j)}$)
        Locate($\vec{p}_s$)
        Locate($\vec{p}_{s-1}$)
        $List_{pending} \nleftarrow \vec{p}_s$
        END CASE B
    **end if**
**end for**
$List_{pending} \leftarrow \vec{p}_s$
END CASE B

First, the environment and datasets are described in subsection 6.1. After that 3 different experiments have been carried out to validate the algorithms. Sections 6.2, 6.3 and 6.4 describe them and present the results.

### *6.1. Dataset*

A dataset captured by ourselves is considered to carry out the experiments. Six different underfloor environments are used to collect the data. Within each environment, the robot followed a trajectory and captured data from a number of poses. Table 1 shows the number of poses considered in each environment. A point cloud is stored for each pose. Fig. 10 shows the appearance of some of the environments. Environment 1 and 2 are relatively small (the average distance between captures is less than 10 cm) and well structured. Environment 3 is larger and composed of several rooms. This way, consecutive point clouds may differ considerably thus it is expected to be more challenging. Environment 4 has a simpler structure comparing to the previous environments, but it is the one that presents a higher distance between consecutive poses. At last, 5 and 6 correspond to the same environment but prior to and after applying foam insulation to the underside of the floor.

| **Environment** | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| **Number of poses** | 4 | 9 | 10 | 10 | 21 | 21 |

Table 1: Number of poses in each environment

Environment 2 Environment 4 Environment 6

Figure 10: Some examples of images extracted from different environments which have been used to develop the experiments.

Three experiments are carried out in order to check the validity of the presented algorithms. The first experiment consists in evaluating the

method to align consecutive poses (algorithm presented in section 4). The second experiment focuses on the alignment of poses which were not successfully registered with the previous one (explained in section 5). Finally, an experiment is proposed to tune the validation threshold of the eq. 7.

### 6.2. *Experiment 1. Alignment between consecutive poses through the proposed registration algorithm.*

The first experiment, which consists in a registration between consecutive poses, is developed through the alignment algorithm presented in the section 4. Due to the fact that the number of correspondences depends on the characteristics of the point cloud (shape, number of points after downsampling, etc.), the percentage of correspondences is calculated as is detailed in section 4 (eq. 5 and 6). After each experiment, the computing time to complete the registration was also collected.

Fig. 11 and 12 show the next results for each pair of consecutive poses $(\vec{p}_{s-1}, \vec{p}_s)$ in the six environments: the percentage of correspondences $r_s$ and $r_{s-1}$ (eq. 5 and 6) and the EFS value (eq. 3) divided over the number of correspondences ($N_{s,s-1}$). Also, table 2 shows the average computing time and the average number of correspondences for each environment.

As explained in subsection 4.3, an alignment is considered valid when equation 7 is accomplished. According to it, a high number of correspondences and low values of EFS denote good alignments. Taking it into account, an analysis of figures 11 and 12 permits knowing which registrations are successful and which are not.

These figures show that all the alignments were successful in environment 1 and 2, since all the correspondences are around the same values and they are relatively high. There is no alignment whose values $r_{s-1}, r_s$ and $EFS/N_{s,s-1}$ are substantially worse than the others, so none of the alignments should be rejected. In the environment 3, the percentage of matched points between the poses 6 and 7 and also between 7 and 8 are much lower than those of the rest of alignments and, at the same time, the EFS values are substantially higher. This would indicate that the alignment between these point clouds was not successful and hence these alignments should be rejected. In the environment 4, the alignments between $P_5 - P_6$ and between $P_8 - P_9$ present a percentage of correspondences which is slightly lower than in the others. Since this difference is relatively low, these alignments should be considered valid. The two last registrations in the environment 5 ($P_{18} - P_{19}$ and $P_{19} - P_{20}$) have respectively an average of

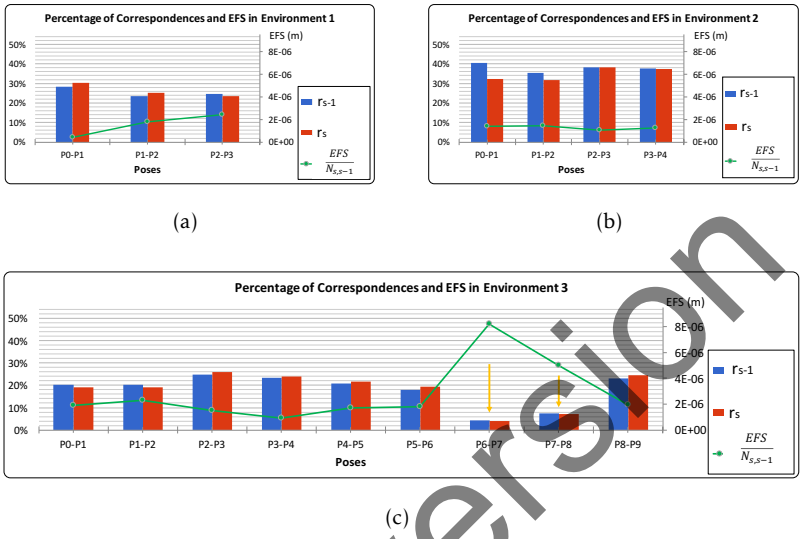
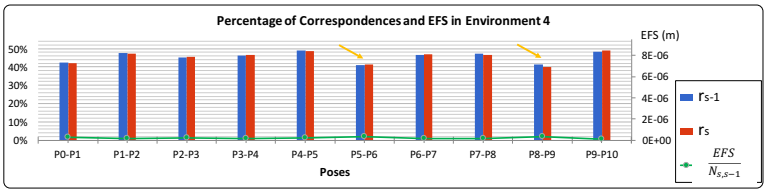



Figure 11: Results of experiment 1. Percentage of matched points and EFS divided by number of correspondences in the environment (a) 1 , (b) 2 and (c) 3.

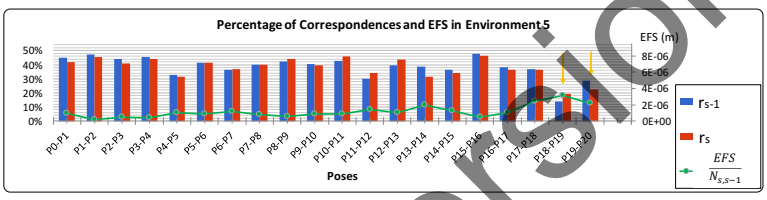
16% and 25% of correspondences, which are considerably lower than the rest of registrations (around 40% of correspondences), while the EFS values for these alignments are also higher. This way, these two cases should be categorized as unsuccessful. In the environment 6, similar issues can be found in $P_{10} - P_{11}$ and $P_{19} - P_{20}$.

Therefore, through the results shown above, we conclude that the number of correspondences between consecutive point clouds along with the Euclidean Fitness Score are capable of validating the results.

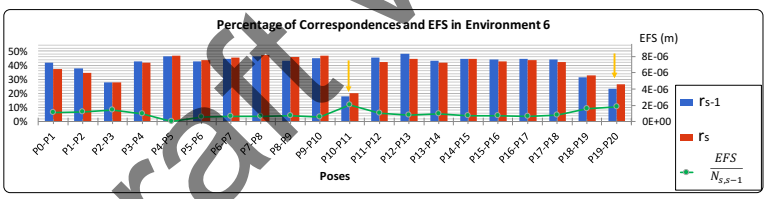
Finally, in this experiment, the computing time of the alignment between point clouds was also measured. Table 2 shows the average computation time and the average number of correspondences for each environment. The algorithm is able to solve the problem in few seconds.




(a)




(b)




(c)



Figure 12: Results of experiment 1. Percentage of matched points and EFS divided by the number of correspondences in the environment (a) 4 , (b) 5 and (c) 6.

### 6.3. *Experiment 2. Alignment between poses that have not been successfully aligned with the previous one.*

The second experiment assesses the algorithm presented in the section 5. The results of the experiment 1 show that, although the alignment works quite well in the majority of the cases, there are some poses which

| Environment | Average Time (ms) | Average N. correspondences |
|---|---|---|
| 1 | 4224 | 10411 |
| 2 | 3639 | 13332 |
| 3 | 4184 | 10855 |
| 4 | 4934 | 16577 |
| 5 | 5955 | 14466 |
| 6 | 5284 | 14732 |

Table 2: Experiment 1. Average computing time of the registration process in the six environments.

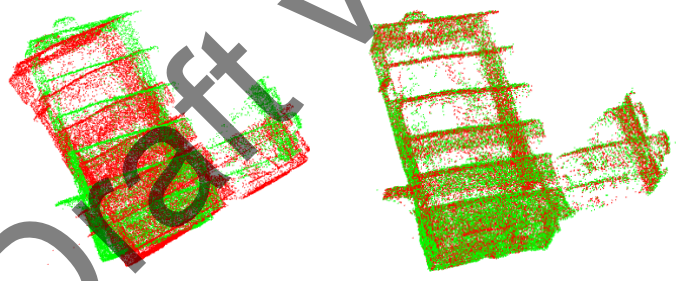
were not well aligned. To solve this problem, the algorithm which tries to align poses with other poses apart from the previous one is run. This experiment has been tested in the environments 5 and 6.

The figures 13 and 14 show two sample point cloud alignments through the transformation matrices computed in experiment 1 and in experiment 2. The figure 13 shows point clouds from the environment 5 and the figure 14 shows point clouds from the environment 6. First, about environment 5, the alignment calculated in experiment 1 between the poses 18 and 19 was unsuccessful (fig. 13(a)). However, the experiment 2 provides a successful alignment with the pose 11 (fig. 13(b)). Second, in the case of environment 6, the experiment 1 was not able to calculate a correct alignment between the poses 10 and 11 (fig 14(a)). This way, the experiment 2 was run and, as a result, the pose 11 was successfully aligned with the pose 13 (fig. 14(b)). In this case, the alignment was carried out following this process. First, the point cloud in the pose 11 was obtained and the algorithm 1 was run. The registration between 10 and 11 was unsuccessful. Therefore, the algorithm 2 was run. The previous poses (from $\vec{p}_0$ to $\vec{p}_9$) were sorted according to the distance to the pose 11 ($\vec{p}_{11}$ was calculated as $T_{11,10} \times \vec{p}_{10}$, where $T_{11,10}$ was a coarse alignment matrix). After trying to align unsuccessfully the pose 11 with all the previous ones, this pose was stored on the pending list. Second, a new pose (12) was obtained and successfully aligned with 11 using the algorithm 1. However, as 11 was on the pending list, the Case B (algorithm 3) was run. The pose 12 was not successfully registered with any of the previous poses $(0, 1...., 10)$. Hence, it was also stored on the pending list. Again, a new pose (13) was obtained and was correctly registered with (12). After that, the algorithm 3 was run. The pose 13 was successfully registered with the pose 10. Therefore,

both the pose 13 and 12 were successfully located and 12 was removed from the pending list. Finally, as mentioned in section 5, after localizing a new pose, the algorithm run a final step in order to try to localize poses which still were in the pending list. The registration between 13 and 11 was successfully carried out, hence, 11 was successfully located as $\vec{p}_{11} = T_{11,13} \times \vec{p}_{13}$.

Fig. 15 shows the computing time of the alignment process. This time takes reasonably low values in case that the pose is successfully aligned with the previous one (experiment 1) and it increases substantially when the experiment 2 has to be run to achieve a correct alignment.

Therefore, the algorithm that aligns consecutive poses permits getting correct alignment results in many cases. However, despite using this algorithm, there is a low number of cases that still do not present valid alignments. For this reason, an algorithm to align non-consecutive poses has been developed and tested. This algorithm not only tries to align incorrect poses with previous ones, but also, in cases in which the alignment with previous poses is unsuccessful, a pending list is managed in order to align the pose with future ones.

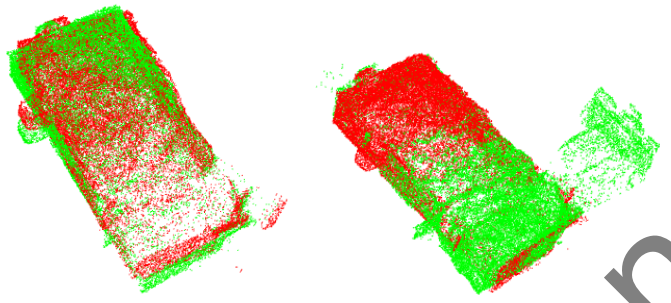


(a) (b)

Figure 13: (**a**) Wrong alignment between poses 18 and 19 in the environment 5. (**b**) Correct alignment of the pose 19 with the pose 11 in the environment 5 after using the case A (algorithm 2 of the novel registration algorithm.)

### *6.4. Experiment 3. Tuning the validation threshold*

The results of the experiment 1 showed that the percentage of correspondences ($r_s$ and $r_{s-1}$) and the EFS take different values depending on



(a) (b)

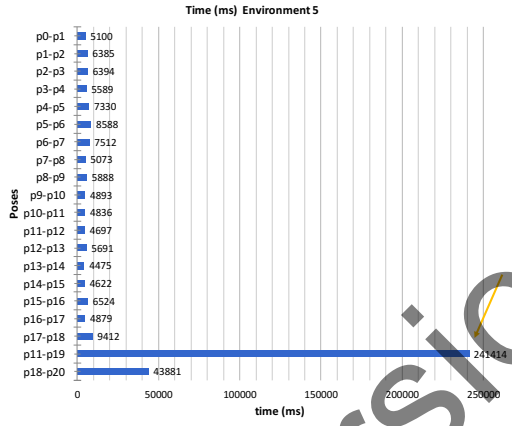
Figure 14: (**a**) Wrong alignment between poses 10 and 11 in the environment 6. (**b**) Correct alignment of the pose 11 with the pose 13 in the environment 6 after using the case B (algorithm 3 of the novel registration algorithm.)



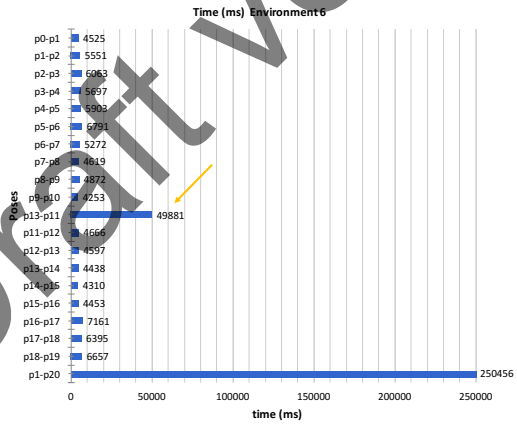
the environment (fig. 11 and 12). This is because the characteristics of the point clouds depend on some factors such as the size of the environment, the elements that compose it and how structured it is. It leads us to the conclusion that the validation threshold (eq. 7) cannot be global. If we tried to establish the same threshold for all the environments, most of the registrations would be categorized wrongly; either because successful alignments would be considered as unsuccessful, or even because incorrect alignments could be accepted as correct. This last case is specially dangerous and it should be categorically avoided.

In order to solve this issue, the next process is followed. Initially, the first registration (between the poses $\vec{p}_0$ and $\vec{p}_1$) is carried out and the alignment results are obtained. The average value of the two percentages of correspondences ($r_s$ and $r_{s-1}$) is obtained and the result is multiplied by a coefficient $\gamma$. The result will be considered as the threshold (eq. 7) for this environment. Afterwards, the following registrations in the environment will be accepted as correct if eq. 7 is met.

It should be pointed out that this threshold will only work well if the first registration was correct. Otherwise, the error will be spread in the following registrations. In order to avoid that, the translation and rotation of the robot between the poses $\vec{p}_0$ and $\vec{p}_1$ should be small. This permits tuning the parameters, taking the diversity of the environments into account. Additionally, the value of $\gamma$ should be chosen by the user. In this




(a)




(b)

Figure 15: Computing time of experiment 2. (**a**) Environment 5. (**b**) Environment 6.

experiment, different values of $\gamma$ are considered to test the influence of this parameter on the results of validation. Environments 3, 4, 5 and 6 are used to tune this parameter, as they are the most challenging ones. Additionally, a mixture between environments 5 and 6 is considered in order to add more complexity and taking advantage of the fact that the environments 5 and 6 are the same but the top plane.

Three values for the coefficient $\gamma$ are tested (0.6, 0.7 and 0.8). For each registration within an environment, four possible cases can occur. First, the registration between poses is detected as correct when in fact it is correct (**True positive**). Second, the registration is detected as incorrect when it is in fact incorrect (**True negative**). Third, the registration is detected as incorrect when it is actually correct (**False negative**). Last, the registration is detected as correct when it is actually incorrect (**False positive**).

Therefore, for each value of $\gamma$, two graphs are depicted. The first one shows in each environment how many registrations have been detected as *True positive*, *True negative*, *False negative* or *False positive*. The second graph shows how many decisions were right (True positive or True negative) and how many decisions failed (False negative or False positive). The results obtained are showed in fig. 16.

Using the transformation matrices calculated in experiment 1, and visually analyzing the results, the following registrations proved to be unsuccessful:

- Environment 3: $p_1 - p_2$ and $p_7 - p_8$.
- Environment 4: All registrations were correct.
- Environment 5: All registrations were correct.
- Environment 6: $p_{10} - p_{11}$ and $p_{19} - p_{20}$.
- Environment mix. 5-6: $p_2 - p_3$.

These results permit making a deep analysis of fig. 16. First, when $\gamma = 0.8$ (see figure 16 (a) and (b)), in the environment 3, the two wrong registrations were detected as incorrect and one registration was rejected when actually it was successfully aligned. In the environment 4, all the poses were detected as correct. In environment 5, four poses were detected as wrongly aligned when actually they were correct. In 6, the two wrongly aligned poses were detected as incorrect. Last, in the mixture environment, the incorrect registration was also detected. As a result (see fig. 16 (b)), the validation algorithm failed once in the environment 3 and four times in 5. Second, when $\gamma = 0.7$ (see figure 16 (c) and (d)), in

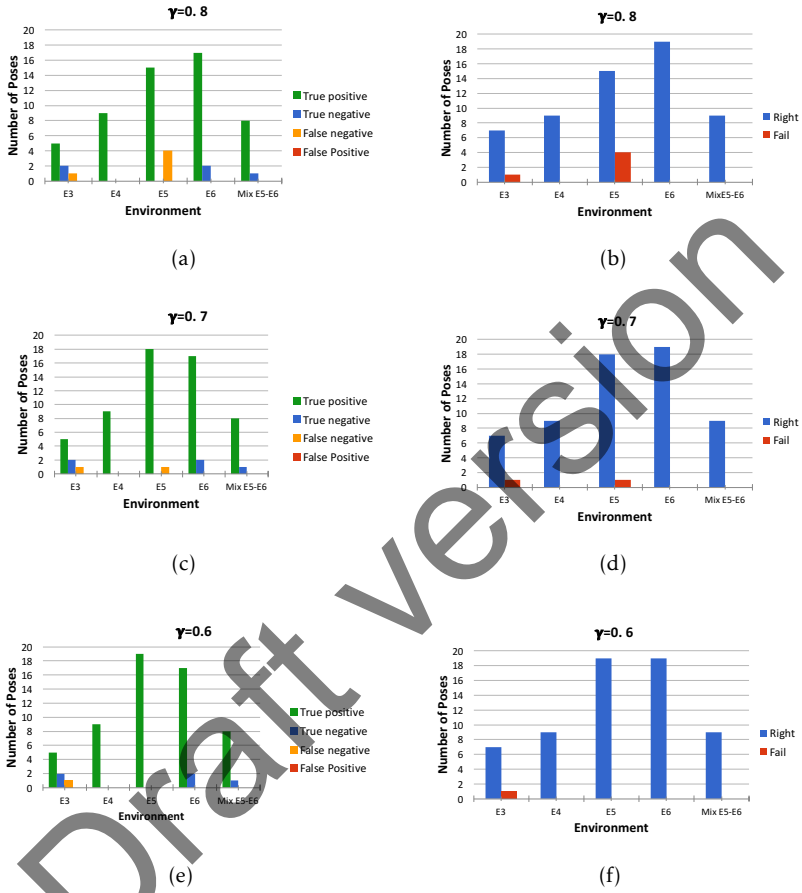



Figure 16: Experiment 3. True positive, true negative, false negative and false positive registrations when the coefficient $\gamma$ is (**a**) 0.8, (**c**) 0.7 and (**e**) 0.6. Number of times when the result of the algorithm is right and the number of failures when $\gamma$ is (**b**) 0.8, (**d**) 0.7 and (**f**) 0.6 .

the environment 3, the two wrongly aligned registrations were detected as incorrect and one registration was rejected when actually it was successfully aligned. In the environment 4, all the poses were detected as

correct. In environment 5, one pose was detected as wrong aligned when actually it was successfully aligned. In 6, the two poses wrongly aligned were detected as incorrect. Last, in the mixture environment, the incorrect registration was detected. Consequently, (see fig. 16(d)), the validation algorithm failed once in the environment 3 and once in 5. Finally, when $\gamma = 0.6$ (see figure 16 (e) and (f)), in the environment 3, the two registrations wrong aligned were detected as incorrect and one registration was rejected when actually it was successfully aligned. In the environment 4, all the poses where detected as correct. In environment 5, all the poses where detected as correct. In 6, the two wrong aligned poses were detected as incorrect. Last, in the mixture environment, the incorrect registration was detected. Thus, when $\gamma = 0.6$ (see fig. 16(f)), the validation algorithm failed only once, in the environment 3.

None of the three coefficients produced the False positive case (registration detected as successful when actually it was unsuccessful). This is very important because that situation must be strongly avoided. For a coefficient $\gamma = 0.8$, the threshold is too tough and thus, in many cases, good alignments are rejected. When $\gamma = 0.6$, the calculated threshold seems to be correct more often, but 0.7 ensures not to detect wrong alignments as correct. In conclusion, $\gamma$ must be tuned in the range 0.6 to 0.7 by the users according to their needs.

## 7. Conclusion

This paper proposes two novel methods to solve the registration task between poses in underfloor environments which present challenging features for traditional methods. The first contribution is an algorithm that uses point cloud information captured by the robot from different poses. After obtaining the point clouds, they are downsampled and the information in the top and bottom planes is removed in order to obtain clouds with less points and more robust results. Finally, the transformation matrix is calculated. This algorithm works well in environments where the characterization using regular algorithms is difficult. Most of the current registration algorithms do not work well in this kind of environments due to the fact that they are not able to extract reliable features. Additionally, although the registration algorithms are commonly based on a coarse alignment (using visual information), a fine alignment (using depth information) and an optional optimization, the method proposed here is able

to find accurate enough results just using the depth information. The results show that this algorithm works successfully for the majority of the cases and also lasts few seconds to reach the solution.

Furthermore, as the second contribution, a novel algorithm is proposed to solve the cases in which the alignment between consecutive poses was not possible. This algorithm tries to align the poses which were not well aligned with other poses within the environment. The results show that this method successfully aligns the majority of cases. Hence, this can be useful for online localization processes where it is desirable to ensure that no information is lost in order to determine the exact path followed by the robot within the environment. Although extra computing time is required, almost no pose information is lost and an accurate global map can be created.

The percentage of correspondences along with the EFS value permit calculating a validation value. The third contribution consists in tuning this validation threshold and testing its influence on the accuracy of the validation process. Good results were obtained when $\gamma = 0.7$ and $\gamma = 0.6$ and we conclude that the users should choose a value within this range according to their needs. When this coefficient is closer to 0.6, the validation step tends to be right in the majority of cases. If it is closer to 0.7, the validation step reduces the chance of detecting unsuccessful alignments as correct.

The results presented in this paper show the robustness and effectiveness of the proposed methods and their ability to cope with such challenging underfloor environments. The team are now working on improving the robustness of these algorithms by enhancing some steps. For instance, to optimize the points selection step through a more restrictive selection which provides a more reliable characterization of the environment and reduces the computing time. Additionally, to reduce the number of unsuccessful cases through the improvement of the registration step and to cut down the time to find a good alignment between two poses. Finally, to establish a validation step which provides a global reliable value for a variety of environments.



## Acknowledgement

This work has been supported by the Spanish government through the project DPI 2016-78361-R: "Creación de mapas mediante métodos de

apariencia visual para la navegación de robots." Q-Bot and this project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 698607.

## References

[1] D. J. Harris, S. J.-M. Dudek, Heat losses from suspended timber floors, Building Research & Information 25 (4) (1997) 226–233. arXiv:http://dx.doi.org/10.1080/096132197370354, doi:10.1080/096132197370354.
URL http://dx.doi.org/10.1080/096132197370354

[2] Department of Energy and Climate Change. Estimates of Home Insulation Levels in Great Britain, https://www.gov.uk/government/statistical-data-sets/estimates-of-home-insulation-levels-in-great-britain, accessed: 2017-05-03 (2013).

[3] M. Holloway, M. Julia, P. Childs, A robot for spray applied insulation in underfloor voids, in: Proceedings of ISR 2016: 47st International Symposium on Robotics, 2016, pp. 1–7, accessed: 2017-11-09.
URL http://ieeexplore.ieee.org/document/7559133/

[4] M. Julia, M. Holloway, O. Reinoso, P. R. Childs, Autonomous surveying of underfloor voids, in: ISR 2016: 47st International Symposium on Robotics; Proceedings of, VDE VERLAG GmbH, 2016, pp. 1–7, accessed: 2017-11-09.
URL http://ieeexplore.ieee.org/abstract/document/7559123/

[5] D. M. Cole, P. M. Newman, Using laser range data for 3d slam in outdoor environments, in: Robotics and Automation, 2006. IEEE International Conference on Robotics and Automation (ICRA) 2006. Proceedings 2006 IEEE International Conference on, IEEE, 2006, pp. 1556–1563. doi:10.1109/ROBOT.2006.1641929.

[6] S. Fu, H.-y. Liu, L.-f. Gao, Y.-x. Gai, Slam for mobile robots using laser range finder and monocular vision, in: Mechatronics and Machine Vision in Practice, 2007. M2VIP 2007. 14th International Conference on, IEEE, 2007, pp. 91–96. doi:10.1109/MMVIP.2007.4430722.

[7] J.-P. Tardif, Y. Pavlidis, K. Daniilidis, Monocular visual odometry in urban environments using an omnidirectional camera, in: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, IEEE, 2008, pp. 2531–2538. doi:10.1109/IROS.2008.4651205.

[8] D. Valiente, A. Gil, L. Fernández, O. Reinoso, Visual slam based on single omnidirectional views, in: Informatics in Control, Automation and Robotics, Springer, 2014, pp. 131–146. doi:https://doi.org/10.1007/978-3-319-03500-0_9.

[9] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, Monoslam: Real-time single camera slam, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (6) (2007) 1052–1067. doi:10.1109/TPAMI.2007.1049.

[10] S.-N. Yu, J.-H. Jang, C.-S. Han, Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel, Automation in Construction 16 (3) (2007) 255 – 261. doi:http://dx.doi.org/10.1016/j.autcon.2006.05.003.
URL http://www.sciencedirect.com/science/article/pii/S0926580506000197

[11] Y.-H. Tseng, S.-C. Kang, J.-R. Chang, C.-H. Lee, Strategies for autonomous robots to inspect pavement distresses, Automation in Construction 20 (8) (2011) 1156 – 1172. doi:http://dx.doi.org/10.1016/j.autcon.2011.04.018.
URL http://www.sciencedirect.com/science/article/pii/S0926580511000720

[12] N. Tan, R. E. Mohan, A. Watanabe, Toward a framework for robot-inclusive environments, Automation in Construction 69 (2016) 68 – 78. doi:http://dx.doi.org/10.1016/j.autcon.2016.06.001.
URL http://www.sciencedirect.com/science/article/pii/S0926580516301194

[13] H. Hamledari, B. McCabe, S. Davari, Automated computer vision-based detection of components of under-construction indoor partitions, Automation in Construction 74 (2017) 78 – 94. doi:http://dx.doi.org/10.1016/j.autcon.2016.11.009.

URL http://www.sciencedirect.com/science/article/pii/S0926580516304046

[14] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110. doi:10.1023/B:VISI.0000029664.99615.94.
URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[15] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, Speeded-up robust features (surf), Computer Vision and Image Understanding 110 (3) (2008) 346 – 359, similarity Matching in Computer Vision and Multimedia. doi:http://dx.doi.org/10.1016/j.cviu.2007.09.014.
URL http://www.sciencedirect.com/science/article/pii/S1077314207001555

[16] S. Leutenegger, M. Chli, R. Y. Siegwart, Brisk: Binary robust invariant scalable keypoints, in: 2011 International Conference on Computer Vision, 2011, pp. 2548–2555. doi:10.1109/ICCV.2011.6126542.

[17] P. Besl, N. D. McKay, A method for registration of 3-d shapes, Pattern Analysis and Machine Intelligence, IEEE Transactions on 14 (2) (1992) 239–256. doi:10.1109/34.121791.

[18] A. Segal, D. Haehnel, S. Thrun, Generalized-icp., Robotics: Science and Systems 2 (4), accessed: 2017-11-09.
URL https://docs.google.com/viewer?url=http%3A%2F%2Fwww.robots.ox.ac.uk%2F~avsegal%2Fresources%2Fpapers%2FGeneralized_ICP.pdf

[19] K. S. Arun, T. S. Huang, S. D. Blostein, Least-squares fitting of two 3-d point sets, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9 (5) (1987) 698–700. doi:10.1109/TPAMI.1987.4767965.

[20] T. Jost, H. Hügli, Pattern Recognition: 24th DAGM Symposium Zurich, Switzerland, September 16–18, 2002 Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, Ch. Fast ICP Algorithms for Shape Registration, pp. 91–99. doi:10.1007/3-540-45783-6_12.
URL http://dx.doi.org/10.1007/3-540-45783-6_12

[21] C.-S. Chen, Y.-P. Hung, J.-B. Cheng, A fast automatic method for registration of partially-overlapping range images, in: Computer Vision, 1998. Sixth International Conference on, 1998, pp. 242–248. doi:10.1109/ICCV.1998.710725.

[22] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard, An evaluation of the rgb-d slam system, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, 2012, pp. 1691–1696. doi:10.1109/ICRA.2012.6225199.

[23] P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox, Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments, International Journal of Robotics Research 31 (5) (2012) 647–663. doi:10.1177/0278364911434148.
URL http://dx.doi.org/10.1177/0278364911434148

[24] D. R. dos Santos, M. A. Basso, K. Khoshelham, E. de Oliveira, N. L. Pavan, G. Vosselman, Mapping indoor spaces by adaptive coarse-to-fine registration of rgb-d data, IEEE Geoscience and Remote Sensing Letters 13 (2) (2016) 262–266. doi:10.1109/LGRS.2015.2508880.

[25] K.-L. Low, Linear least-squares optimization for point-to-plane icp surface registrationAccessed: 2017-11-09.
URL https://docs.google.com/viewer?url=https%3A%2F%2Fwww.iscs.nus.edu.sg%2F~lowkl%2Fpublications%2Flowk_point-to-plane_icp_techrep.pdf

[26] E. Cappelletto, P. Zanuttigh, G. M. Cortelazzo, 3d scanning of cultural heritage with consumer depth cameras, Multimedia Tools and Applications 75 (7) (2016) 3631–3654. doi:10.1007/s11042-014-2065-4.
URL http://dx.doi.org/10.1007/s11042-014-2065-4

[27] J. Xie, Y.-F. Hsu, R. S. Feris, M.-T. Sun, Fine registration of 3d point clouds fusing structural and photometric information using an rgb-d camera, Journal of Visual Communication and Image Representation 32 (2015) 194 – 204. doi:http://dx.doi.org/10.1016/j.jvcir.2015.08.007.
URL http://www.sciencedirect.com/science/article/pii/S1047320315001509

[28] A. Wilkowski, T. Kornuta, W. Kasprzak, Point-Based Object Recognition in RGB-D Images, Springer International Publishing, Cham, 2015, pp. 593–604. doi:10.1007/978-3-319-11310-4_51.
URL http://dx.doi.org/10.1007/978-3-319-11310-4_51

[29] R. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, 2011, pp. 1–4. doi:10.1109/ICRA.2011.5980567.

applied sciences

MDPI

Article

# Evaluation of Clustering Methods in Compression of Topological Models and Visual Place Recognition Using Global Appearance Descriptors

**Sergio Cebollada [1,*,†], Luis Payá [1,†], Walterio Mayol [2,†] and Oscar Reinoso [1,†]**

[1] Department of Systems Engineering and Automation, Miguel Hernández University, 03202 Elche, Spain; lpaya@umh.es (L.P.); o.reinoso@umh.es (O.R.)
[2] Department of Computer Science, University of Bristol, Bristol BS81TH, UK; wmayol@cs.bris.ac.uk
* Correspondence: sergio.cebollada@umh.es
† These authors contributed equally to this work.

Received: 11 December 2018; Accepted: 17 January 2019; Published: 22 January 2019


**Abstract:** This paper presents an extended study about the compression of topological models of indoor environments. The performance of two clustering methods is tested in order to know their utility both to build a model of the environment and to solve the localization task. Omnidirectional images are used to create the compact model, as well as to estimate the robot position within the environment. These images are characterized through global appearance descriptors, since they constitute a straightforward mechanism to build a compact model and estimate the robot position. To evaluate the goodness of the proposed clustering algorithms, several datasets are considered. They are composed of either panoramic or omnidirectional images captured in several environments, under real operating conditions. The results confirm that compression of visual information contributes to a more efficient localization process through saving computation time and keeping a relatively good accuracy.

**Keywords:** mapping; localization; clustering; omnidirectional images; global appearance descriptors

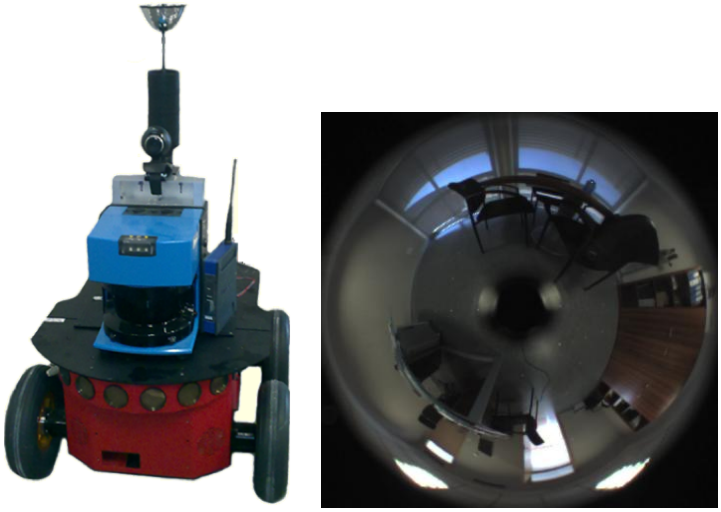
## 1. Introduction

The presence of mobile robots in many kinds of environments has increased substantially during the past few years. Robots need a high degree of autonomy to develop their tasks. In the case of autonomous mobile robots, this means that they must be able to localize themselves and to navigate through environments that are a priori unknown. Hence, the robot will have to carry out the mapping task, which consists of obtaining information from the environment and creating a model. Once this task is done, the robot will be able to address the localization task, i.e., estimating its position within the environment with respect to a specific reference system.

Vision sensors have been widely used for mapping, navigation, and localization purposes. According to the number of cameras and the field of view, different configurations have been proposed. Some authors (such as Okuyama et al. [1]) have used monocular configurations. Others proposed stereo cameras by using binocular (such as Yong-Guo et al. [2] or Gwinner et al. [3]) or even trinocular systems (such as Jia et al. [4]).

Despite stereo cameras permitting measuring depth from the images, these systems present a limitation related to their field of view. In order to obtain complete information from the environment, several images must be captured. In this respect, omnidirectional cameras constitute a good alternative. They can provide a big amount of information with a field of view of 360 deg. around them, and their cost is relatively low in comparison with other kinds of sensors. Furthermore, omnidirectional vision

systems present further advantages. For instance, the features in the images are more stable (because they stay longer as the robot moves), and they permit estimating both the position and the orientation of the robot. Omnidirectional cameras have been successfully used by different authors for mapping and localization [5–9]. A wide study was carried out by Payá et al. [10], who introduced a state-of-the-art of the most relevant mapping and localization algorithms developed with omnidirectional visual information. An example of a mobile robot that has an omnidirectional camera mounted on it is shown in Figure 1a, and an example of an omnidirectional image is shown in Figure 1b.



(a) (b)

**Figure 1.** (**a**) Example of a robot Pioneer P3-AT® equipped with an omnidirectional vision system and a laser range finder. In this work, only the omnidirectional camera is used. (**b**) Example of an omnidirectional image captured from one office.

In the related literature, two main frameworks have been proposed in order to carry out the mapping task: the metric maps, which represent the environment with geometric accuracy; and the topological maps, which describe the environment as a graph containing a set of locations with the related links among them. Regarding the second option, some authors have proposed to arrange the information in the map hierarchically, into a set of layers. The way a robot solves the localization task efficiently in hierarchical maps is as follows: first, a rough, but fast localization is carried out using the high-level layers; second, a fine localization is tackled in a local area using the low-level layers. Therefore, in order to address the mapping and localization issue, hierarchical maps constitute an efficient alternative (like the works [11–13] show).

Visual mapping and localization have been solved mainly by using two main approaches to extract the most relevant information from scenes; either by detection, description, and tracking of some relevant landmarks or working with global appearance algorithms, i.e., building a unique descriptor per image. On the one hand, the methods based on local features consist of extracting some outstanding points from each scene and creating a descriptor for each point, using the information around it (Figure 2a). The most popular description methods used for this purpose are SIFT (Scale-Invariant Feature Transform) [14] and SURF (Speeded-Up Robust Features) [15]. More recently, descriptors such as BRIEF (Binary Robust Independent Elementary Features) [16] or ORB (Oriented FAST and Rotated BRIEF) [17] have been proposed, trying to overcome some drawbacks such as the computational time and invariance against rotation. These descriptors have become very popular in visual mapping and localization, and many authors have proposed methods that use them, such as Angeli et al., who

employed SIFT [18], or Murillo et al., who used SURF [8]. Nonetheless, these methods present some disadvantages. For instance, to obtain reliable landmarks, the environments must be rich in details. Furthermore, keypoints' detection is not always robust against changes in the environments (e.g., changes of lighting conditions), and sometimes, the description is not totally invariant to changes in the robot position. Moreover, these approaches might be computationally complex; hence, in those cases, it would not be possible to build models in real time. On the other hand, the methods based on the global appearance of scenes consist of treating each image as a whole. Each image is represented by a unique descriptor, which contains information about its global appearance (Figure 2b). These methods lead to simpler mapping and localization algorithms, due to the fact that each scene is described by only one descriptor. Hence, mapping and localization can be carried out by just storing and comparing the descriptors pairwise. Besides, they could be more robust in dynamic and unstructured environments. However, as drawbacks, these methods present a lack of metric information (they are commonly employed to build topological maps). Visual aliasing also might have a negative impact on the mapping and localization tasks, due to the fact that indoor environments are prone to present repetitive visual structures. Additionally, modelling large environments would require a big amount of images, and this can introduce serious issues when these techniques have to be used in real-time applications. Therefore, global appearance is an intuitive alternative to solve the mapping and localization problem, but its robustness against these issues must be tested. Many authors have addressed mapping and localization using global appearance descriptors (Figure 2b). For instance, Menegatti et al. [19] used the Fourier signature in order to build a visual memory of a relatively small environment from a set of panoramic images. Liu et al. [20] proposed a descriptor based on colour features and geometric information. Through this descriptor, a topological map can be built. Payá et al. [21] proposed a mapping method from global appearance and solved the localization in a probabilistic fashion, using a Monte Carlo approach. Furthermore, they developed a comparative analysis of some description methods. Rituerto et al. [22] proposed the use of the descriptor *gist* [23,24] to create topological maps from omnidirectional images. More recently, Berenguer et al. [6] proposed the Radon transform [25] as the global appearance descriptor of omnidirectional images and a hierarchical localization method. Through this method, first, a rough localization is obtained; after that, a local topological map of a region is created and used to refine the localization of the robot.

In light of the previous information, in the present paper, the use of hierarchical models is proposed to solve the localization task efficiently. In this sense, compression methods are used as a solution to generate the high-level layers of the hierarchical model. Some authors have used clustering algorithms to carry out the compression task. For instance, Zivkovic et al. [26] used spectral clustering to obtain higher level models, which improved the efficiency of the path-planning. Grudic and Mulligan [27] built topological maps through the use of an unsupervised learning algorithm, which worked with spectral clustering. Valgren et al. [28] tackled an on-line topological mapping through the use of incremental spectral clustering. Štimec et al. [29] used an unsupervised clustering based on the multiple eigenspaces algorithm to carry out topological mapping hierarchically using omnidirectional images. More recently, Shi et al. [30] proposed the use of a differential clustering method to improve the compression of telemetry data.

We propose a method to build hierarchical maps through a combination of clustering methods and global appearance descriptors. We compare the performance of spectral and self-organizing maps' clustering. In addition, an exhaustive experimental evaluation is carried out to assess the performance of the method in mapping and localization tasks, and we evaluate the influence of the most relevant parameters in the results. This is an interesting problem in the field of mobile robotics because, as pointed out before, global appearance descriptors are a straightforward way of describing visual information, but they contain no metric information, comparing to local-features' descriptors. Additionally, no deep study to assess the performance of global-appearance descriptors in hierarchical mapping can be found in the literature. The experiments show that the proposal that we present is a

feasible alternative to build robust compact maps, despite the phenomenon of visual aliasing, which is present in the sets of images that we have used in the experiments.

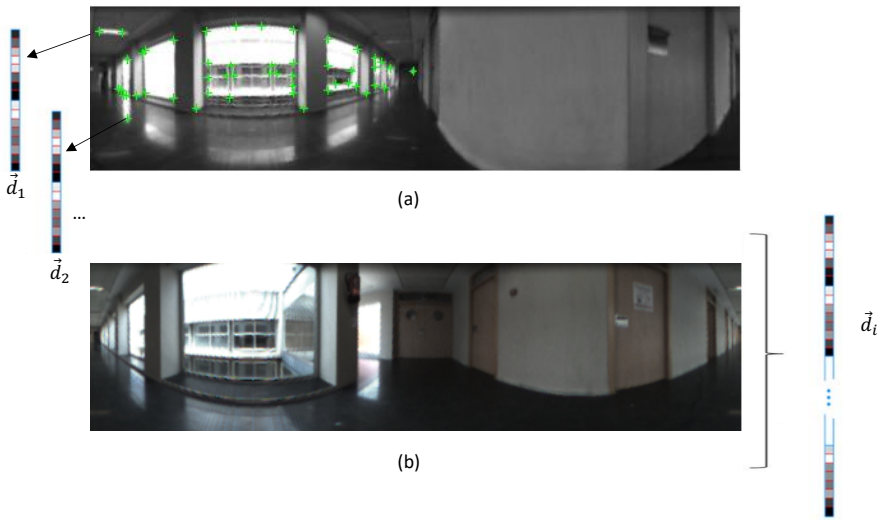



**Figure 2.** Two main methods to extract the most relevant information from the images for mapping and localization purposes. (**a**) Detection, description, and tracking of some relevant landmarks along a set of scenes. (**b**) Building a unique descriptor per image that contains information on its global appearance.

The present paper continues and extends the study presented in [31], which is a comparative evaluation in which the performance of some descriptors was assessed to create compact models and estimate the position of the robot. The contributions of the present paper are the following: (a) a new method to compact the visual model is proposed; (b) the trade-off compactness-accuracy-computational cost is addressed, and the performance of the compact models is compared to raw models (with no compaction); (c) a comparison between compression through direct methods and compression through clustering methods to solve the localization task is evaluated; and (d) new indoor environments with different topologies are included in the experimental section.

The remainder of the paper is structured as follows: Section 2 outlines the global appearance descriptors that will be tested throughout the paper. After that, Section 3 shows the clustering approaches used to compress the models. Next, Section 4 presents the method to obtain the localization within the compact models. Section 5 presents the experimental results of clustering and localization and also the discussions about the results. Finally, Section 6 outlines the conclusions and future research lines.

## 2. Global Appearance Descriptors

As mentioned in the previous section, global appearance descriptors constitute an interesting alternative for mapping and localization. In this work, the robot moves along the floor plane, and it captures images using a hyperbolic mirror, which is mounted over a camera along the vertical axis. This section details three methods to describe the global appearance of a set of panoramic scenes $IM = \{im_1, im_2, ..., im_N\}$ where $im_j \in \mathbb{R}^{M_x \times M_y}$. After using each description method, a descriptor for each image is calculated; thus, the database is composed of a set of descriptors, $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ where each descriptor is $\vec{d_j} \in \mathbb{C}^{l \times 1}$ and corresponds to the image $im_j$.

The remainder of the section presents the global appearance techniques used throughout the paper and the homomorphic filter, which is used as a pre-treatment for the images.

### *2.1. Fourier Signature Descriptor*

The Fourier signature descriptor was firstly used by Menegatti et al. [19] to create an image-based memory for robot navigation. Payá et al. [21] studied the computational cost and the error in localization by using Fourier Signature (FS) and proposed a Monte Carlo approach to solve the localization problem in indoor environments.

This description method is based on the use of the Discrete Fourier Transform (DFT). After calculating the FS of a panoramic image, a new complex matrix is obtained $IM(u,v)$. It can be decomposed into two real matrices, one containing the magnitudes and the other the arguments. The steps to obtain a global appearance descriptor from a panoramic image through the Fourier Signature (FS) are: First, departing from the intensity matrix of the original image, the DFT of each row is calculated. The result is a complex matrix with the same size as the original image ($IM(u,v) \in \mathbb{C}^{N_x \times N_y}$). Second, only the $k_1$ first columns of this matrix are retained since the main information is in the low frequency components. Third, the resultant matrix ($IM(u,v) \in \mathbb{C}^{N_x \times k_1}$) is decomposed into the magnitudes and arguments matrices. The matrix of magnitudes ($A(u,y) \in \mathbb{R}^{N_x \times k_1}$) is invariant against changes of the robot orientation in the movement plane if the image is panoramic. In the last step, the global appearance descriptor is obtained by arranging the $k_1$ columns of the magnitudes matrix in one single column ($\vec{d} \in \mathbb{R}^{N_x \cdot k_1 \times 1}$).

### *2.2. Histogram of Oriented Gradients Descriptor*

The Histogram of Oriented Gradients (HOG) is a description method used in computer vision to detect objects. This descriptor is remarkable due to the fact that it is easy to build, leads to successful results in detection tasks, and also requires a low computational cost. It is built from the orientation of the gradient in localized parts of the panoramic image. The development consists of dividing the image into small regions ($k_2$ horizontal cells in this work) and compiling a histogram with $b$ bins for the pixels, which are included inside each cell using their gradient orientation. The combination of this information provides the desired descriptor ($\vec{d} \in \mathbb{R}^{b \cdot k_2 \times 1}$). This method has been used by some authors such as Mekonnen et al. [32] to develop a person detection tool, or Dong et al. [33], who proposed an HOG-based multi-stage approach for object detection and pose recognition in the field of service robots. This method was firstly used in mobile robotics by Dalal and Triggs [34] to solve people detection task. Zhu et al. [35] presented an improved version with respect to computational time and efficiency to detect people.

The HOG version proposed in this work is described in detail in [36].

### *2.3. Gist Descriptor*

The *gist*description was introduced by Oliva et al. [37], and it has been commonly used to recognize scenes. Since then, several versions can be found, which work with different features from the images, such as colour, texture, orientation, etc. [38]. Some researchers have used *gist* in mobile robotics. For instance, Chang et al. [39] used this global appearance descriptor for localization and navigation. Murillo et al. [40] also used the *gist* descriptor to solve the localization problem, but in this case, the *gist* descriptor was a reduced version obtained with Principal Components Analysis (PCA).

The version we use throughout this paper is described in [36] and works with the orientation information obtained through a set of Gabor filters. From the panoramic image, $m$ different resolution levels are obtained. Then, $n_{masks}$ orientation filters are applied over each level. Finally, the pixels of every image are grouped into $k_3$ horizontal blocks, and the information is arranged in a vector ($\vec{d} \in \mathbb{R}^{n_{masks} \cdot m \cdot k_3 \times 1}$).

### 2.4. Homomorphic Filter

In order to solve the localization task, typical situations may happen such as lighting variations and changes in the position of some objects (chairs, tables, open doors, etc.). Hence, descriptors must be robust against these circumstances.

Fernandez et al. [41] showed that some pre-treatments could improve the localization accuracy in indoor environments with different lighting levels. Among the studied techniques, the use of the homomorphic filter [42] can be highlighted. The homomorphic filter permits filtering the luminance and reflectance components from an image separately.

The use of this filter has proven to provide especially good results when it is used in combination with the HOG descriptor [31], whereas in the FS and *gist* cases, the results were similar to or worse than without this pre-treatment filter. Hence, in the present paper, the following configurations will be used throughout the experiments: FS without filter, HOG with filter, and *gist* without filter.

## 3. Clustering Methods to Compact the Visual Information

In this section, the creation of topological models and how to compact them will be addressed. Subsequently, these models will be utilized to solve the localization problem. Only visual information and global appearance descriptors will be used in both tasks. This way, the problem will be addressed through the next two steps.

1. Learning: creating a map of the environment and compacting it. A set of omnidirectional images is captured from different positions, and a global appearance descriptor for each image is calculated. After that, a clustering method is used to determine the structure and compact the model.
2. Validation: Once the map is built, the robot obtains a new image from an unknown position, calculates the descriptor, and compares it with the set of descriptors obtained in the learning step. Through this comparison, the robot must be able to estimate its position.

Focusing on the learning step, the robot moves around the environment and captures some images from different positions to cover the whole environment. This way, a set of omnidirectional images is collected $I = \{im_1, im_2, ..., im_N\}$ where $im_j \in \mathbb{R}^{N_x \times N_y}$. After that, a global appearance descriptor is calculated for each image; hence, a set of descriptors is obtained $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ where $\vec{d_j} \in \mathbb{C}^{l \times 1}$.

This set of descriptors can be considered as a straightforward model of the environments [43,44], as some previous works do. However, in this mapping strategy, important problems appear when the environment has considerable dimensions. The larger the environment is, the more images have to be captured to model it completely. This leads to the requirement of more computational time and also more memory space in order to process and collect the information related to each captured image and to solve the subsequent localization problem. This way, the model should be compacted in such a way that it retains most of the visual information and permits solving the localization problem efficiently.

In this work, we propose a clustering approach to compact the model, with the objective of creating a two-layer hierarchical structure. The low-level layer is composed of a set of descriptors and, to obtain the high-level layer, this set will be compacted via clustering. Each cluster is characterized by the common attributes of the instances that form that group. This way, the dataset $D = \{\vec{d_1}, \vec{d_2}, ..., \vec{d_N}\}$ is divided into $n_c$ clusters $C = \{C_1, C_2, ..., C_{n_c}\}$ under the conditions:

$$\begin{aligned} C_i \neq \emptyset, i = 1, ..., n_c \\ \bigcup_{i=0}^{m} C_i = D \\ C_i \bigcap C_j = \emptyset, i \neq j, i, j = 1, ..., n_c. \end{aligned} \tag{1}$$

After this, each cluster is reduced to a unique representative descriptor, which is obtained in this work as the average of all the descriptors that compose that cluster. A set of representatives is obtained $R = \{\vec{r_1}, \vec{r_2}, ..., \vec{r_{n_c}}\}$, and therefore, the model is compacted. This set of representatives composes the high-level layer.

Figure 3 shows how a sample map is compacted. Figure 3a shows the positions where panoramic images were captured to cover the whole environment. The result of the clustering process is presented in Figure 3b, and then, one representative per cluster is obtained (Figure 3c). The representative descriptor is obtained as the average descriptor among those grouped in the same cluster. Additionally, the position of this representative descriptor is calculated as the average position of the capture points of the images included in the same cluster. These positions are calculated just as a ground truth to test the performance of the compact map in a localization process, but they are not used either to build the map, nor to localize the robot. Only visual information is used with these aims. Different clustering methods will be analysed. These methods will only use visual information, and ideally, the objective is to group images captured from near positions despite visual aliasing. To evaluate the correctness of the approach, the geometrical compactness of the clusters and their utility to solve the localization task will be tested in Section 5.

Regarding the clustering process to compact the visual models, two methods are studied: spectral clustering and self-organizing maps.

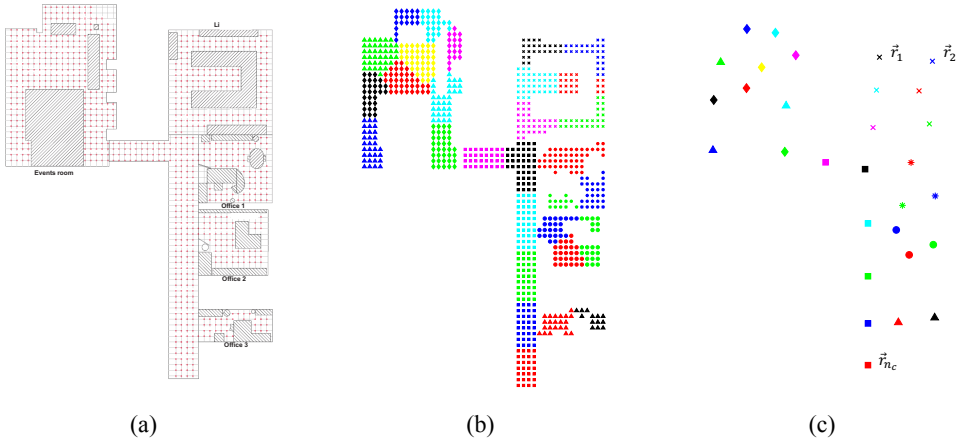



**Figure 3.** Example of an indoor map and a compression of the information. (**a**) Positions where the images were captured. (**b**) Result of the clustering process. (**c**) Each cluster is reduced to one representative.

### *3.1. Spectral Clustering Algorithm*

Spectral clustering algorithms [45] have proven to be suitable to process highly-dimensional data. In this work, a spectral normalized clustering algorithm is used as was introduced by Ng et al. [46]. This algorithm has been already used for mapping along with local features extracted from the scenes [29,47].

In our work, the algorithm departs from the set of global appearance descriptors $D = \{\vec{d_1}, \vec{d_2}, ...\vec{d_N}\}$ obtained from the images collected in the environment, and the parameter $n_c$ is the desired number of clusters. Initially, the similitude between descriptors is calculated. This parameter is calculated for each pair of descriptors; hence, a matrix of similitudes $S$ is obtained as $S_{ij} = e^{-\frac{|\vec{d_i} - \vec{d_j}|^2}{2\sigma^2}}$ where $\sigma$ is a parameter that controls the rapidity of the reduction of the similitude when the distance between $\vec{d_i}$ and $\vec{d_j}$ increases. The steps to carry out the clustering are the following:

1. Calculation of the normalized Laplacian matrix:

$$L = I - D^{-1/2} S D^{1/2} \quad (2)$$

where Dis a diagonal matrix $D_i = \sum_{j=1}^{N} S_{ij}$.

2. Calculation of the $n_c$ main eigenvectors of $L$, $\{\vec{u_1}, \vec{u_2}, ..., \vec{u_{n_c}}\}$. Arranging these vectors by columns, the matrix $U \in \mathbb{R}^{N \times n_c}$ is obtained.
3. Normalization of the matrix $U$ to obtain the matrix $T \in \mathbb{R}^{N \times n_c}$.
4. Extraction of vector $\vec{y_i} \in \mathbb{R}^{n_c}$ from the i$^{\text{th}}$ row of the matrix $T$. $i = 1, ..., N$.
5. Clustering of the $\vec{y_i}$ vectors by using a simple clustering algorithm (such as k-means or hierarchical clustering). Through this, the clusters $A_1, A_2, ..., A_{n_c}$ are obtained.
6. Obtaining the clusters with the original data as $C_1, C_2, ..., C_{n_c}$ where $C_i = \vec{d_j} \mid \vec{y_j} \in A_i$.

If the number of instances $N$ or the dimension $l$ is high, the computation of the $n_c$ eigenvectors (third step) will be computationally expensive. To solve this issue, Luxburg [45] proposed cancelling some components of the similitude matrix. This way, in the matrix $S$, only the components $S_{ij}$ so that $j$ is among the $t$ nearest neighbours of $i$ are retained. After this, the $n_c$ first eigenvectors of the Laplacian matrix $L$ are calculated by using the Lanczos/Arnoldi factorization [48].

Finally, for each cluster, a representative is obtained as the average visual descriptor of the set of descriptors that compose that cluster.

Spectral clustering may result in being more efficient than traditional methods such as k-means or hierarchical clustering in large environments due to the fact that spectral clustering considers the mutual similitude between the instances.

### *3.2. Cluster with a Self-Organizing Map Neural Network*

As a second alternative, Self-Organizing Maps (SOM) have been chosen to carry out the clustering evaluation in this work. This algorithm was introduced by Kohonen [49], and it is an effective option to carry out a mapping distribution when the data present a high dimensionality [50]. This algorithm has been commonly used for clustering or reducing the dimensionality of data. Therefore, in this work, the input data are the set of global appearance descriptors calculated with one of the methods described in Section 2. The size of the neural network map ($W_{SOM} \times H_{SOM} = n_c$) is chosen. After the training step, the data will be grouped into $n_c$ different clusters.

Self-organizing maps automatically learn to classify input vectors according to their similarity and topology in the input space. They differ from competitive layers in that neighbouring neurons in the SOM learn to recognize neighbouring sections of the input space. Thus, self-organizing maps learn both the distribution (as the competitive layers do) and topology of the input vectors with which they are trained. The neurons can be arranged in a grid, hexagonal, or random topology. The self-organizing map network identifies a wining neuron $i^*$ using the same procedure as employed by the competitive layer, but instead of updating only the winning neuron, all neurons within a certain neighbourhood $N_{i^*}(d)$ of the winning neuron are updated.

## 4. Using the Compact Topological Maps to Localize the Robot

At this point, the robot is provided with a model of the environment, which, in this case, is a hierarchical map. From it, the robot firstly uses the high-level layer to carry out a rough localization, and secondly, a fine localization is tackled through the use of the low-level layer. The visual localization problem has been solved by many authors through local features by using probabilistic approaches such as particle filters or Monte Carlo localization [51,52]. Nevertheless, the works developed with global appearance descriptors are scarce. Hence, this paper presents a comparison of this kind of descriptor to estimate hierarchically the position of the robot within a hierarchical map in a specific time instant. In order to test the accuracy of the localization method proposed in this work, the coordinates

where the images were captured within the environment are known (ground truth). Nevertheless, they are not used to estimate the position of the robot since, as mentioned before, the presented method only considers visual information. This decision permits studying the feasibility of visual sensors as the only source of information to create a compact topological map and, more concisely, of global appearance descriptors. Therefore, not using the position information in the mapping and localization algorithms permits isolating the effect of the main parameters of these descriptors and knowing the performance of this kind of information. The remainder of this section is structured as follows: Section 4.1 outlines the types of distances that have been used to calculate how different the global appearance descriptors are. Section 4.2 explains the localization step within maps that have not been compacted previously, i.e., no clustering has been carried out (the full information about the environment is provided). Finally, Section 4.3 explains the localization task within hierarchical topological maps.

### 4.1. Distance Measures between Descriptors

In order to know how similar two panoramic images are through their global appearance descriptors, some distance measurements have been used. This way, a comparison can be carried out by calculating the distance between the descriptors of two images captured from different positions of the environment. The lower the distance between those images is, the more similar they are. This kind of distance is used in the localization step. We consider two descriptors $\vec{a} \in \mathbb{R}^{l\times 1}$ and $\vec{b} \in \mathbb{R}^{l\times 1}$, where $a_i$ and $b_i$ are the $i^{\text{th}}$ components of $\vec{a}$ and $\vec{b}$ with $i = 1, ..., l$. The distances used in this work are:

- Euclidean distance: This a particular case of the the weighted metric distance and is defined as:

$$dist_{euclidean}(\vec{a}, \vec{b}) = \sqrt{\sum_{i=1}^{l}(a_i - b_i)^2} \tag{3}$$

- Cosine distance: Departing from a similitude metric, which is defined as the scalar product between two vectors, the distance is defined as:

$$\begin{aligned} dist_{cosine}(\vec{a}, \vec{b}) &= 1 - sim_{cosine}(\vec{a}, \vec{b}) \\ sim_{cosine}(\vec{a}, \vec{b}) &= \frac{\vec{a}^T \cdot \vec{b}}{|\vec{a}||\vec{b}|} \end{aligned} \tag{4}$$

- Correlation distance: Again, departing from a similitude metric, which is defined as a normalized version of the scalar product between two vectors, the distance is defined as:

$$\begin{aligned} dist_{correlation}(\vec{a}, \vec{b}) &= 1 - sim_{correlation}(\vec{a}, \vec{b}) \\ sim_{correlation}(\vec{a}, \vec{b}) &= \frac{(\vec{a} - \bar{a})^T(\vec{b} - \bar{b})}{\sqrt{(\vec{a} - \bar{a})^T(\vec{a} - \bar{a})}\sqrt{(\vec{b} - \bar{b})^T(\vec{b} - \bar{b})}} \end{aligned} \tag{5}$$

  where:

$$\bar{a} = \frac{1}{l}\sum_{i=1}^{l} a_i; \qquad \bar{b} = \frac{1}{l}\sum_{i=1}^{l} b_i \tag{6}$$

Previous research works [21,36] have evaluated the relation between the distance between the global appearance descriptors and the geometrical distance between capture points. These works show that even if the robot moves a short distance, the descriptor changes. Therefore, global appearance descriptors can be used to detect even small movements.

### *4.2. Resolution of the Localization Problem in a Model That Has Not Been Compacted*

In this case, the map is composed of a straightforward set of descriptors (i.e., this set has not been treated to create a hierarchical map through any clustering process). Once this straightforward map is available, the localization process starts:

1. The robot captures a new image at time instant $t$ from an unknown position ($im_t$).
2. It calculates the global appearance descriptor of the captured image $\vec{d_t}$.
3. The distances between this new descriptor and the set of descriptors in the map are obtained. The comparison between descriptors is carried out through one of the distance metrics presented in Section 4.1.
4. A distance vector $l_t = \{l_{t1}, ..., l_{tN}\}$ is obtained where $l_{tj} = dist\{\vec{d_t}, \vec{d_j}\}$ according to any distance measure.
5. Considering the position of the robot as the position of the closest neighbour within the map (the problem known as image retrieval [53]), the corresponding position of the robot is the position in the map that minimizes the distance $arg\min_j l_{tj}$. This way, the position $(x, y)$ of the robot in the instant $t$ is estimated.

### *4.3. Resolution of the Localization Problem in a Compact Model*

Image retrieval is an inefficient process due to the fact that the maps are usually composed by a huge number of images and the descriptors have a high dimensionality. Therefore, the computational cost could be a problem. In this case, clustering is used to compact the map. Additionally, indoor environments may present visual aliasing. As explained in Section 3, after clustering, the map $\mathfrak{M}$ will be formed only by a set of clusters $C = \{C_1, ..., C_{n_c}\}$, where $n_c$ is the number of clusters. For each cluster, a representative descriptor was calculated as the average of the descriptors in it and the coordinates of those representatives as the average coordinates of the descriptors that compose that cluster. Thus, a set of cluster representatives $\{\vec{r_1}, ..., \vec{r_{n_c}}\}$ and the coordinates of each representative $\{(x, y)_{r_1}, ..., (x, y)_{r_{n_c}}\}$ are known (ground truth).

The localization in this hierarchical map is carried out as follows. (1) The robot captures a new image $im_t$ from an unknown position $(x_t, y_t)$, which must be estimated, and (2) the descriptor corresponding to the new captured image is obtained ($\vec{d_t}$) by using any of the description algorithms explained in Section 2 (FS, HOG, or *gist*). (3) The distance vector is obtained $\vec{l_t} = \{l_{t1}, ..., l_{tn_c}\}$ where $l_{tj} = dist\{\vec{d_t}, \vec{r_j}\}$ is the distance (one of the three types explained in Section 4.1) between the descriptor $\vec{d_t}$ and each representative $\vec{r_j}$. Finally, (4) the estimated position of the robot $(x_e, y_e)$ is the position associated with the nearest neighbour $d_t^{nn}|t = arg\min_j l_{tj}$.

The coordinates of the representatives are not used in the localization step. However, to measure the goodness of the estimation, the geometric distance between $(x_t, y_t)$ and the centre of the corresponding cluster (obtained as the average position among the positions of the images that belong to that cluster) is calculated: $error = \sqrt{(x_e - x_t)^2 + (y_e - y_t)^2}$. Furthermore, the required computational cost to estimate the localization is calculated.

## 5. Experiments

### *5.1. Datasets*

Two different types of datasets were used to develop the experiments; QuorumV, which contains grid-distributed visual data, and the COsy Localization Database (COLD), which contains visual information along a trajectory. On the one hand, Quorum V is a publicly-available dataset [54], which consists of a set of omnidirectional images that have been captured in an indoor environment at Miguel Hernandez University (Spain). The database includes 3 offices, a library, a meeting room, and a corridor. It is composed by two datasets; the first one is a training dataset, and it is composed of 872 images,

which were captured on a dense 40 × 40 cm grid of points. As for the second dataset, the test dataset, it is composed of 77 images, which were captured in different parts of the environment, in half-way positions among the points of the training dataset, and including changes in the environment (e.g., people walking, position of furniture, etc.). Figure 4 shows the bird's eye view of theQuorum V database and the grid points captured by the robot for the training dataset.

On the other hand, COLD (COsy Localization Database) [55] (also publicly available) contains several sets of images captured in three different indoor environments, which are located in three different cities: Ljubliana (Slovenia), Saarbrücken, and Freiburg (Germany). This database contains omnidirectional images captured while the robot traversed several paths within the environments under real operating conditions (with people that appear and disappear from scenes, changes in the furniture, etc.). In the present work, we use the two longest paths: Saarbrücken and Freiburg. Both datasets include several rooms such as corridors, personal offices, printer areas, kitchens, bathrooms, etc. In order to represent the same distance between images as the distance presented in the Quorum V database, a downsampling is carried out to obtain an acquisition distance between images of 40 cm approximately. Therefore, two training datasets are generated: $Freiburg_{training}$ and $Saarbrucken_{training}$, with 519 and 566 images, respectively. Moreover, from the remaining images, test datasets were created. Figure 5 shows the bird's eye view of the environments and the path that the robot traversed to obtain the images. To summarize, Table 1 shows the datasets used for this work and the number of images that each of them contains.

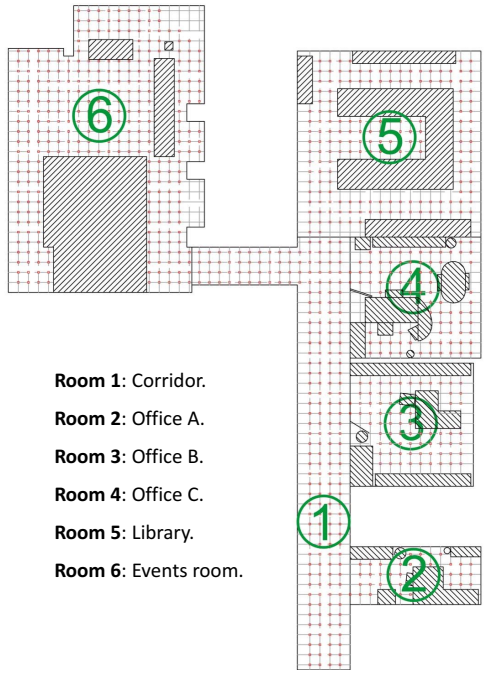
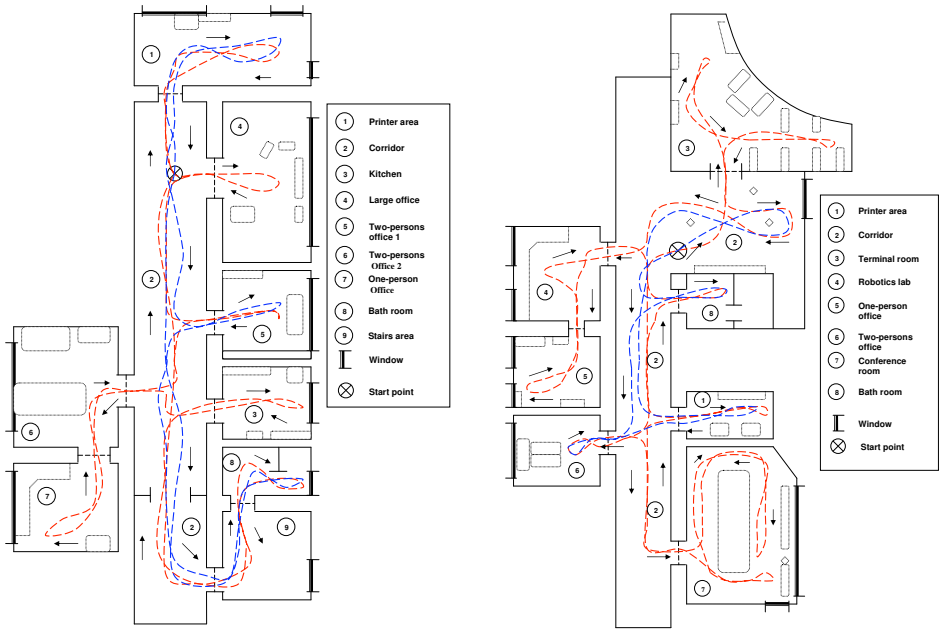



**Figure 4.** Bird's eye view of the Quorum V database.

Through evaluating these two types of datasets, an analysis of the localization in maps which are completely different is tackled: the first kind of map (Quorum V) is a grid-based map, and the second dataset (COLD) is a trajectory-based map. The Quorum V database presents a distance between images of 40 cm approximately. This distance is considered reasonable for indoor applications. In this case, the expected maximum error (when all the images are used for mapping) is around 28 cm (a case in which the test image is in the middle of four images of the map, which compose a square of a side of

40 cm). This is a reasonable accuracy to solve localization tasks, and additionally, the requirements of memory to store the images of the map are not excessively high in large environments. Regarding the downsampling that is carried out in COLD, this was done with the purpose of obtaining results that can be directly compared with the ones obtained through the Quorum V database (whose minimum available distance is 40 cm). Previous works [6] have shown that the distance between images has a direct relation with the accuracy of localization when global appearance descriptors are used. Lower distances tend to provide more accurate results. Therefore, if a specific application requires a lower error, a more dense initial dataset of images should be used to obtain the map.




(a) (b)

**Figure 5.** Bird's eye view of the COsy Localization Database (COLD). (**a**) Freiburg and (**b**) Saarbrücken environment. Extracted from https://www.nada.kth.se/cas/COLD/.

**Table 1.** Datasets used to carry out the experiments.

| Dataset Name | Number of Images | Number of Rooms |
|---|---|---|
| *QuorumV_training* | 872 | 6 |
| *QuorumV_test* | 77 | |
| *Freiburg_training* | 519 | 9 |
| *Freiburg_test* | 52 | |
| *Saarbrucken_training* | 566 | 8 |
| *Saarbrucken_test* | 57 | |

### *5.2. Creating Compact Maps through Clustering*

This section focuses on the evaluation of clustering methods to compact the information contained in a set of global appearance descriptors. To carry out the experiments, two clustering methods were studied for each environment, and three global appearance descriptors were considered. The first

method (Method 1) consists of spectral clustering along with k-means as was explained in Section 3.1. Other configurations were tested, such as to use of SOM instead of k-means to solve Step 5 of the spectral clustering, but the results were quite similar; thus, only the spectral clustering along with k-means to cluster the normalized matrix of the $n_c$ eigenvectors is shown. The second method (Method 2) consists of the use of SOM, which was explained in Section 3.2. Therefore, for the two proposed methods, several experiments were carried out to study the influence of the parameters of the three global appearance descriptors. Table 2 summarizes the experiments developed.

**Table 2.** Summary of the parameters that have been varied to carry out the clustering experiments. FS, Fourier Signature.

| Parameter | Values |
|---|---|
| Environment | Quorum V<br>Freiburg (COLD)<br>Saarbrücken (COLD) |
| Descriptor | FS<br>HOG<br>*gist* |
| Descriptor parameters | FS: $k_1$ = 4, 8, 16, 32, 64, 128, 256<br>HOG: $k_2$ = 2, 4, 16, 32, 64, 128<br>*gist*: $k_3$ = 2, 4, 8, 16, 32, 64<br>*gist*: $n_{masks}$ = 2, 4, 8, 16, 32, 64 |
| Number of clusters | Quorum V: $n_c$ = 15, 25, 40, 60, 80, 100<br>Freiburg: $n_c$ = 10, 20, 30, 40, 50, 60, 70<br>Saarbrücken: $n_c$ = 10, 20, 30, 40, 50, 60, 70 |

The values $k_1$, $k_2$, and $k_3$ define the length of each descriptor, but their meaning is not the same (equal values of $k_1$, $k_2$, and $k_3$ would not lead to the same descriptor size). Therefore, as our aim is to study the correct tuning of these values to use each descriptor as efficiently as possible, we do not apply the same values for all the descriptors in the experiments.

Once the compact map has been produced, it may be interesting to provide some measures that permit quantifying the compactness of the map. In this context, the concept of the silhouette is commonly used. Silhouette values point out the degree of similarity between the instances within the same cluster and at the same time the dissimilarity with the instances that belong to other clusters. The silhouette takes values in the range $[-1, 1]$, and it provides information about how compact the clusters are. Therefore, in order to quantify the goodness of each method, three parameters are considered:

a The average moment of inertia of the cluster.
b The average silhouette of the points.
c The average silhouette of the descriptors.

These values are collected after the clustering process. As for the moment of inertia, it measures the compactness of the clusters (if the clusters group images captured from geometrically-close points) and is calculated as:

$$M = \sum_{i=1}^{n_c} \frac{\sum_{j=1}^{n_i} dist((x,y)_{r_i}, (x_j, y_j))^2}{n_i} \quad (7)$$

where $dist((x,y)_{r_i}, (x_j, y_j))$ is the Euclidean distance between the coordinates of the representative $\vec{r_i}$ and the position of the j$^{th}$ image that belongs to the cluster $C_i$, and $n_i$ is the number of images within this cluster.

As for the silhouettes values, two types of silhouette are used: the average silhouette of points is defined as:

$$S_{points} = \frac{\sum_{w=1}^{N} s_w}{N} \quad (8)$$

$N$ is the number of instances (images), and $s_w$ is the silhouette of each instance; it is calculated as:

$$s_w = \frac{b_w - a_w}{max(a_w, b_w)} \tag{9}$$

where $a_w$ is the average distance between the capture point of the instance $\vec{d}_w$ and the capture points of the other instances in the same cluster, and $b_w$ is the minimum average distance between the capture point of the instance $\vec{d}_w$ and the capture point of the instances in the other clusters.

Differently, the average silhouette of descriptors is traditionally obtained through:

$$S_{descr} = \frac{\sum_{k=1}^{N} s_k}{N} \tag{10}$$

where $N$ is the total number of instances and $s_k$ is the silhouette of each instance. This value is calculated as:

$$s_k = \frac{b_k - a_k}{max(a_k, b_k)} \tag{11}$$

where $a_k$ is the average distance between the descriptor $\vec{d}_k$ and the descriptor of the rest of the entities contained in the same cluster, and $b_k$ is the minimum average distance between $\vec{d}_k$ and the instances contained in the other clusters.

The silhouette of descriptors has been traditionally used to measure the compactness of the clusters. However, it does not measure the geometrical compactness. This is why we introduce the silhouette of points, which can provide more proper information since we are interested in knowing whether the clusters have grouped images captured nearby.

#### 5.2.1. Clustering in the Quorum V Environment

Figure 6 shows the results of the two clustering methods using FS as the descriptor depending on the parameter $k_1$. Figure 7 shows the results using HOG depending on the parameter $k_2$. Figure 8 shows the results using *gist* depending on the parameter $k_3$ and with $n_{masks}$ = 16. These figures present the graphs that determine the goodness of each configuration to carry out the mapping task through clustering. The three figures show the moment of inertia and average silhouettes vs. the number of clusters. In all cases, the range of the vertical axis is the same, for comparison purposes. Furthermore, Figure 9 shows the computing time necessary to cluster the environment through the two clustering methods.

Regarding the parameters used to measure the compactness of the maps, the lower the moment of inertia and the higher the silhouettes are, the more compact the map is. Generally, Method 1 (spectral clustering) produces the best results. Method 2 (SOM) does not improve these results. As for the use of the global appearance descriptor with the spectral clustering method, FS is not capable of creating reliable clusters. As for HOG, the moment of inertia and silhouettes depend considerably on the value of $k_2$. When $k_2$ is low, the results are poor, but when $k_2 > 8$, the moment of inertia, as well as the silhouettes improve significantly. At last, regarding the *gist* descriptor, low values of $k_3$ produce low silhouettes and high moments of inertia, and high values of this parameter imply better results.

As for the computation time required to carry out the clustering through the two methods, the SOM method presents the highest values. The computing time required for the clustering process through the FS descriptor is the highest, whereas the time through HOG or *gist* is lower, and the fastest one would be determined by the value of either $k_2$ or $k_3$.

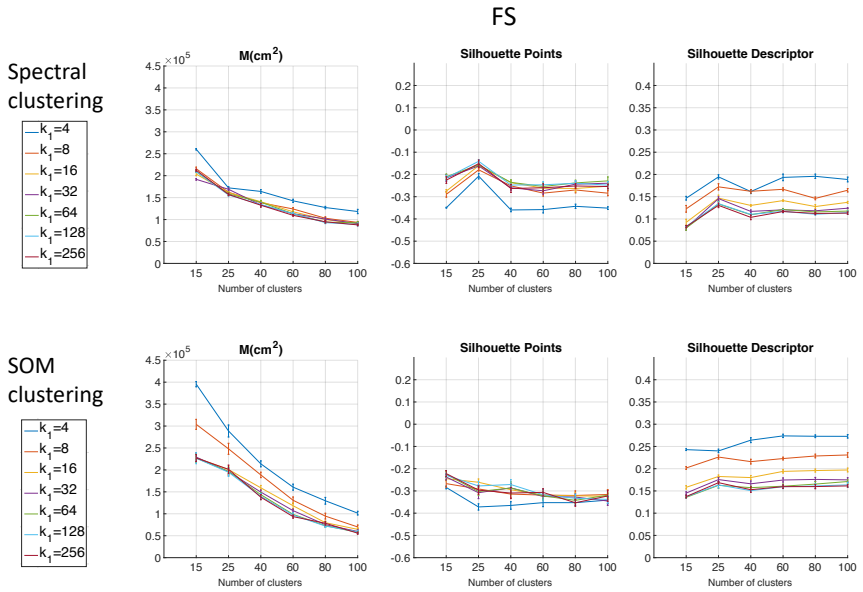



**Figure 6.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using FS in the Quorum V environment. SOM, Self-Organizing Maps.

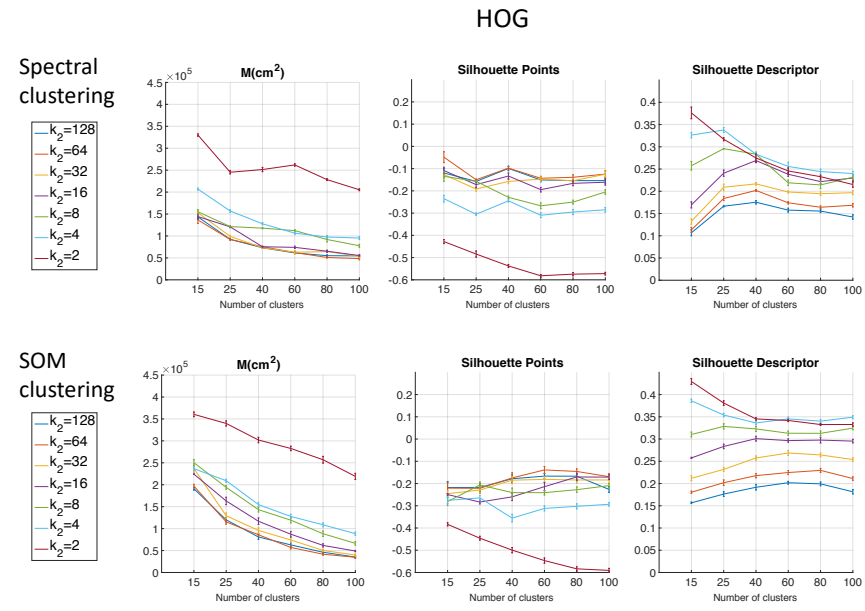



**Figure 7.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using HOG in the Quorum V environment.

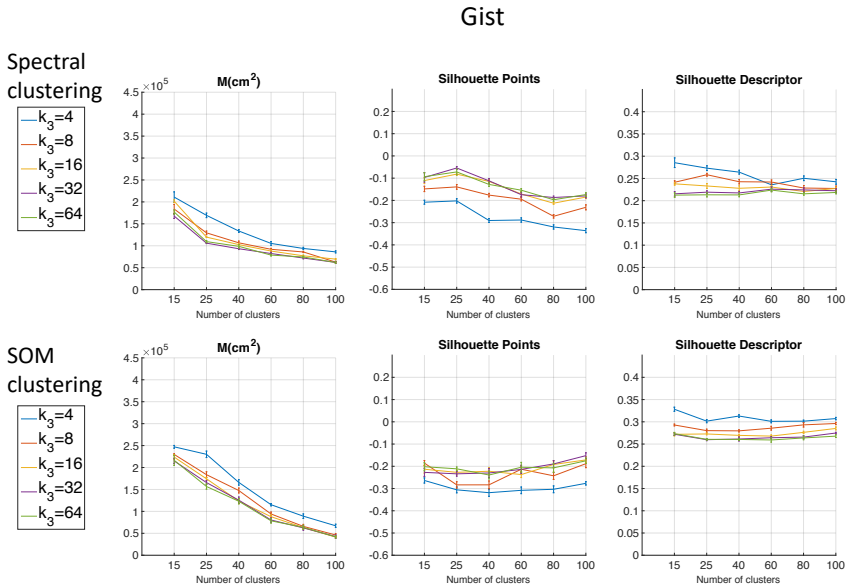



**Figure 8.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using *gist* in the Quorum V environment.

As expected, the more components the descriptor has, the more time is required. In Section 5.3, the trade-off descriptor size-localization accuracy will be studied.

Therefore, in the case of HOG, a value of $k_2 = 32$ or $k_2 = 64$ could be a good choice to achieve a compromise between compactness and computing time, and in the case of *gist*, an intermediate value of $k_3$ could be also a good choice for the same purpose. The FS descriptor presents, in general, the worst results: the moment of inertia is higher, and the silhouettes are lower, in general. Hence, the best clustering results are obtained through the use of the spectral clustering method and the use of HOG (for a configuration of $k_2 = [32, 64]$) or *gist* (for a configuration of $k_3 = [16, 32]$ and $n_{masks} = 16$) as the global appearance descriptor. Figure 10 shows a bird's eye view of the clusters obtained with spectral clustering and gistwith $k_3 = 32$ and $n_{masks} = 16$.

### 5.2.2. Clustering in COLD Environments

The previous results have shown that the use of FS for clustering is less suitable. Considering this, only HOG and *gist* descriptors are analysed in the experiments with the COLD environment. Figure 11 shows the results using HOG depending on the parameter $k_2$ in the Freiburg environment. Figure 12 shows the results of the clustering methods using *gist* depending on the parameter $k_3$ and with $n_{masks}$=16 in the Freiburg environment. In the same way, for the Saarbrücken environment, Figure 13 shows the results using HOG, and Figure 14 shows the results with *gist*. Regarding the use of HOG with the second method (using SOM), it was not able to solve the clustering task for $k_2 = [4, 16]$ when $n_c > 60$.

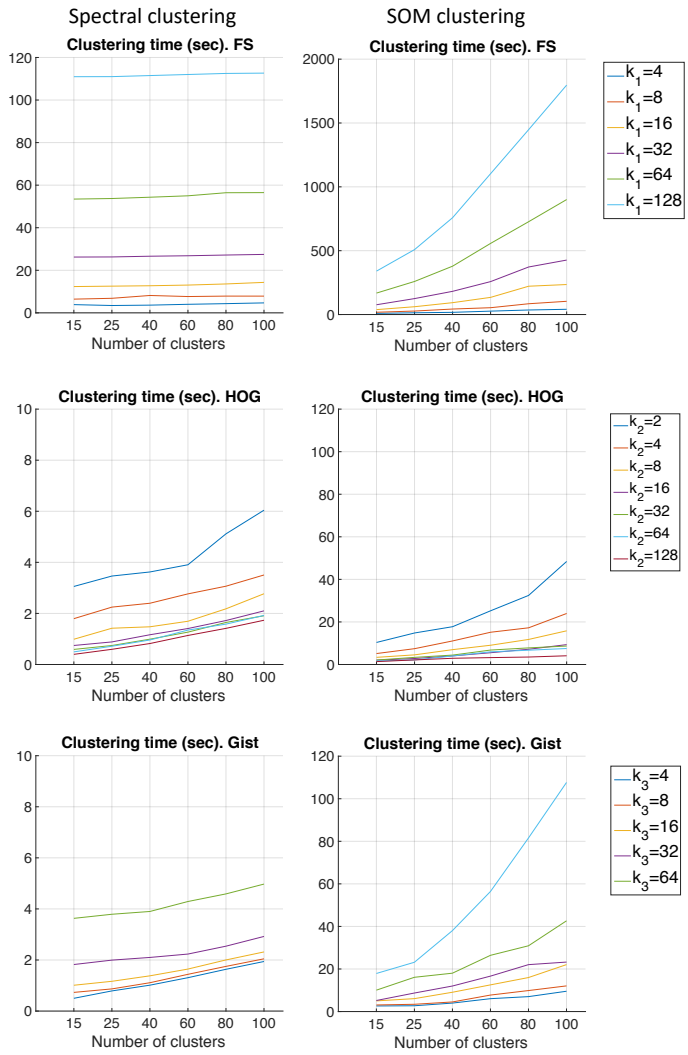



**Figure 9.** Results of the two clustering methods: computing time vs. number of clusters, when using FS, HOG, and *gist* descriptors in the Quorum V environment.

Again, spectral clustering is the best method, and in this case, *gist* presents better clustering outcomes. Hence, through the experiments carried out in the environments of the COLD database, a confirmation of the results obtained in Quorum V is reached (see Figure 15). Therefore, the proposed method is generalizable despite the use of different types of models (linear or grid). As a conclusion, the best option to carry out the compression of visual maps is reached when spectral clustering with *gist* is applied.

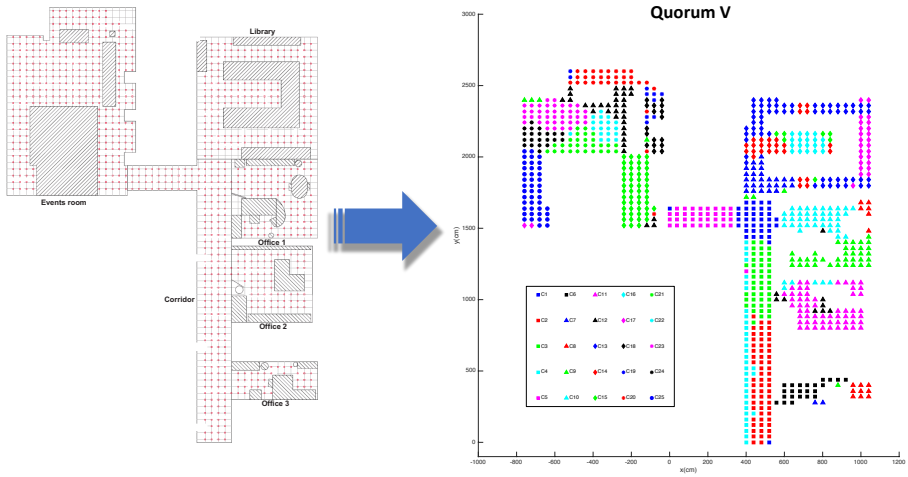



**Figure 10.** Quorum V environment. Cluster obtained with spectral clustering and *gist* description ($k_3 = 32, n_{masks} = 16$).

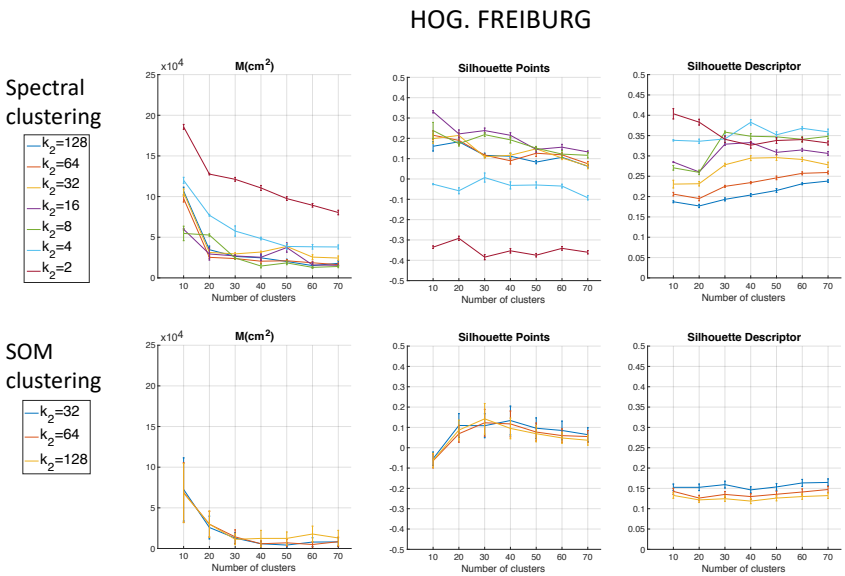



**Figure 11.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using HOG in the Freiburg environment.

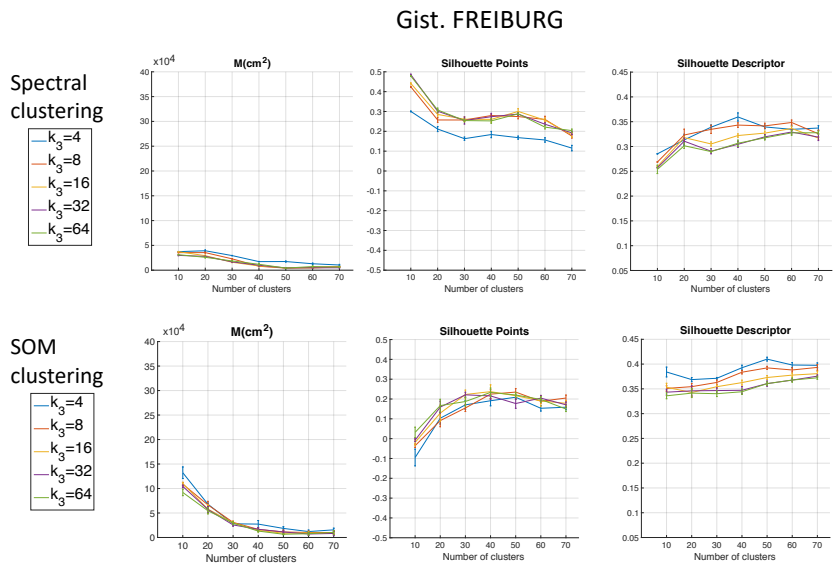



**Figure 12.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using *gist* in the Freiburg environment.

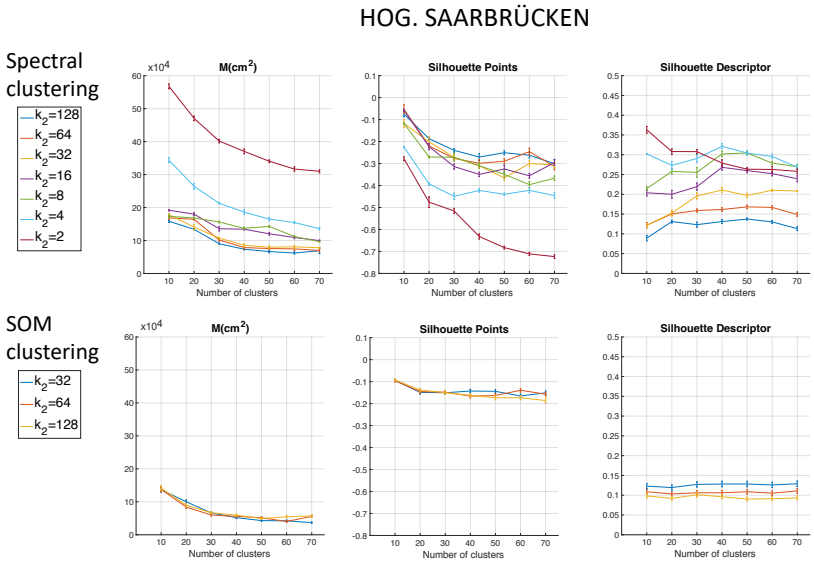



**Figure 13.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using HOG in the Saarbrücken environment.

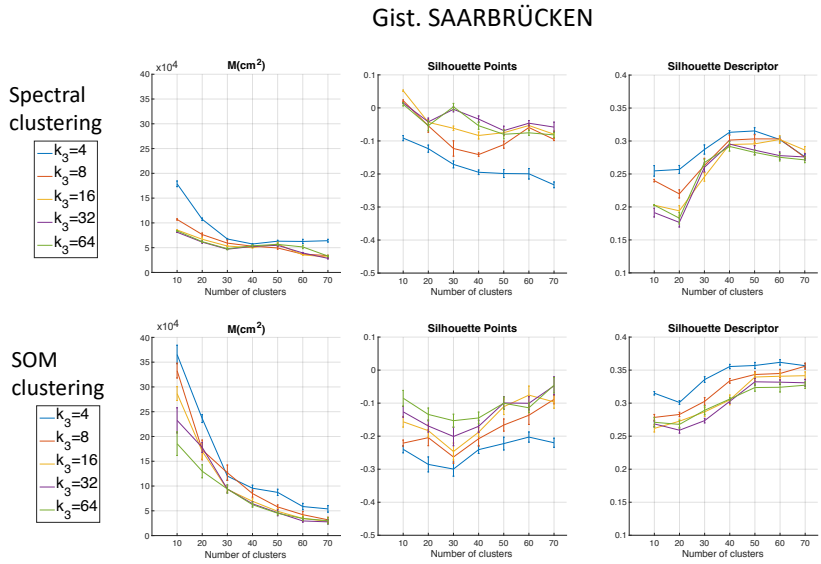
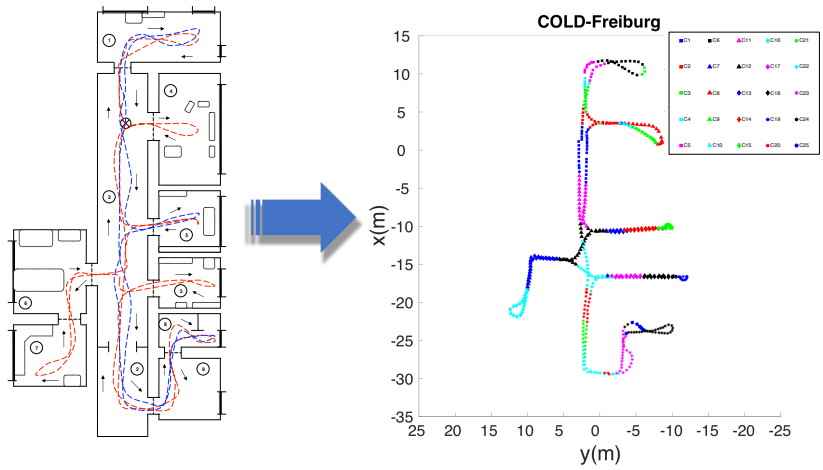


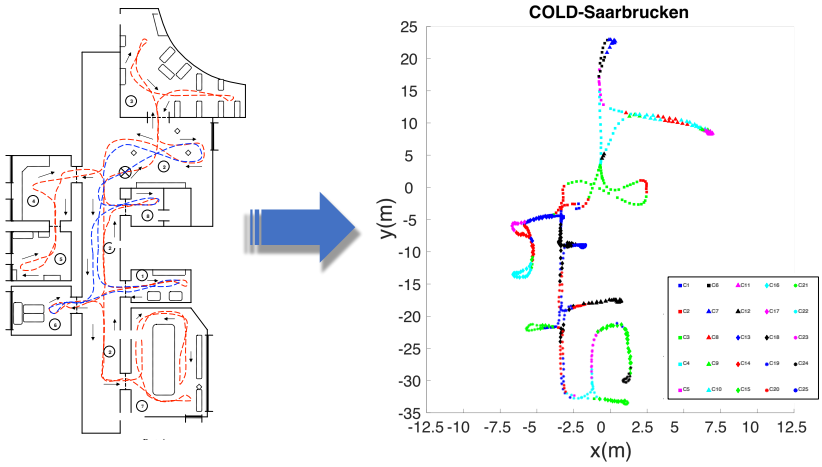

**Figure 14.** Results of the two clustering methods: average moment of inertia, average silhouette of points, and average silhouette of descriptors vs. number of clusters, when using *gist* in the Saarbrücken environment.

### *5.3. Localization Using the Compact Maps*

This section evaluates the performance of the compact maps to solve the localization problem. The objective is to achieve a compactness that presents a balance between computing time and accuracy of localization. To carry out the evaluation, among the mapping results, the spectral clustering algorithm is selected with the *gist* descriptor ($k_3 = 32$ and $n_{masks} = 16$). With this configuration, a map per environment is built, using the training images. After that, the test images are used to solve the localization problem. The previous subsection proved that the best option to build the compressed map was through the use of the *gist* descriptor. Nevertheless, the three proposed global appearance descriptors are proposed again to solve the localization task (because mapping and localization are two independent processes, and the performance of the descriptors could be different in a localization framework). For each test image, its descriptor is calculated (either by FS, HOG, or *gist*), and then, it is compared with the cluster representatives of the compact map. Afterwards, the most similar cluster is retained. Three distance measures are considered for this comparison: (1) the correlation distance, (2) the cosine distance, and (3) the Euclidean distance. In order to carry out a realistic comparison, despite the real position of the robot being provided by the database, only visual information will be used to estimate the position of the robot. The metric information will be used only as ground truth, for comparison purposes.




(a)




(b)

**Figure 15.** Clusters obtained in the COLD environments through the use of Spectral clustering and *gist* description. (**a**) Freiburg and (**b**) Saarbrücken environment.

### 5.3.1. Localization in the Quorum V Environment

Figure 16 shows the average localization error (cm) obtained when FS (first row), HOG (second row), and *gist* (third row) are used, respectively, as the descriptor. Figure 17 presents the computational time (s). In the case of HOG, the effect of homomorphic filtering adds a constant time of 0.02 s per test image. Regarding the number of clusters, $n_c = 872$ is considered since this value provides the case in which the localization is solved without compacting the map. This value is used as a reference to know the relative utility of the compacted map.

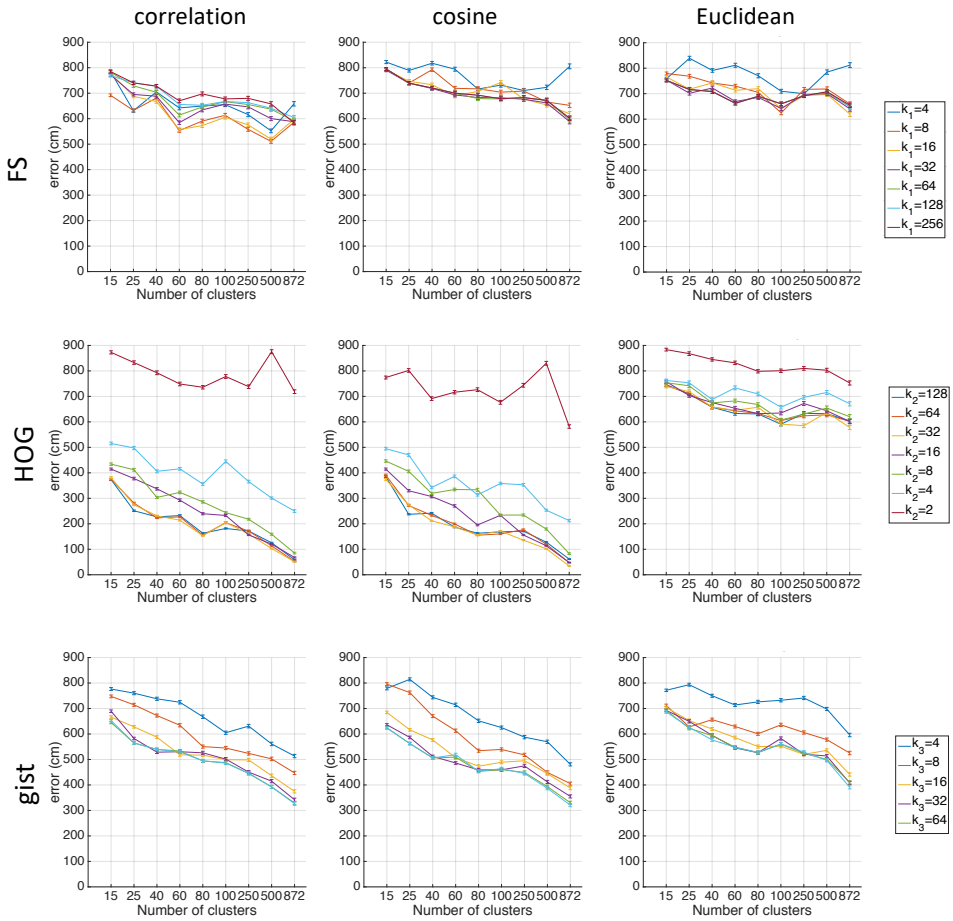



**Figure 16.** Results of the localization process with FS, HOG, and *gist* used to describe the representatives of the clusters and the test images: average localization error (cm) vs. number of clusters. Quorum V environment.

The FS descriptor is not good for localization since the best choice (correlation distance) presents errors between 650 cm and 800 cm depending on the number of clusters and the size of the descriptor. HOG clearly improves the localization task. Except for the case $k_2 = 2$, the average localization error decreases as the number of clusters increases, and these values go from 500 cm when $n_c$ is low and achieve values under 100 cm (when $n_c$ is high). As for the *gist* descriptor, it also produces relatively good results, but they are not as good as those obtained through the use of HOG. The localization task achieves the best results when the correlation distance is used.

Regarding the computation time, with the FS descriptor, as the number of clusters increases, the computational time required for the localization task increases substantially. With HOG, the time is much lower than FS, and it keeps constant independently of the number of clusters. This means that the time to calculate the descriptor is higher than the time to compare it with the map. The computation time required for *gist* is also worse than HOG. The time required by *gist* is around twice the time with HOG.

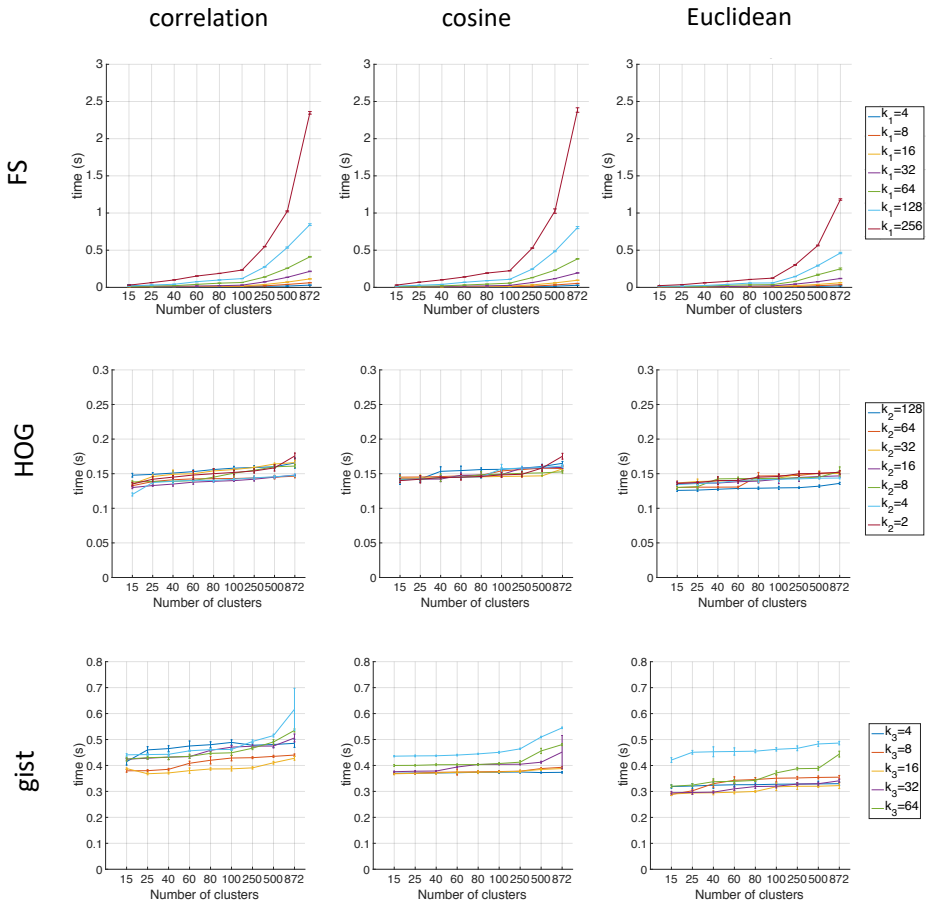



**Figure 17.** Results of the localization process with FS, HOG, and *gist* used to describe the representatives of the clusters and the test images: average computing time vs. number of clusters. Quorum V environment.

In general, as the number of clusters increases, the computation time required for the localization task also increases, and the average localization error decreases. This is an expected behaviour due to the fact that a high number of clusters means that the map is less compact and the information stays in representatives of the clusters whose distance to the test image is lower. Hence, the more clusters, the more comparisons with representatives must be carried out. This leads to a higher computation time and lower average localization error distance. Thus, a balance between these behaviours must be achieved. Therefore, in order to solve the localization in an environment whose properties are similar to the Quorum V environment (grid-distributed data), the optimal values are reached through the use of a HOG descriptor with $k_2 = [32, 64]$ and correlation distance.

5.3.2. Localization in the Freiburg Environment

As in the previous case (clustering task), with the aim of corroborating the results obtained in Quorum V, an evaluation of the localization task is carried out in the COLD environments. These environments present trajectory maps instead of grid maps. The two COLD environments present

a similar configuration and also similar results. This way, only the results obtained in one of them are shown. Freiburg is chosen because this environment presents more rooms and also is more challenging due to the fact that the building presents many glass walls. Moreover, as Figure 16 shows, since the FS descriptor has presented the worst results, this descriptor is discarded in subsequent localization experiments. Furthermore, the Euclidean distance results are omitted in this section because it presented the worst outcomes. Figure 18 shows the average localization error (cm) obtained when HOG (first row) and *gist* (second row) are used respectively as the descriptor. The case of no compaction is also considered ($n_c = 519$).

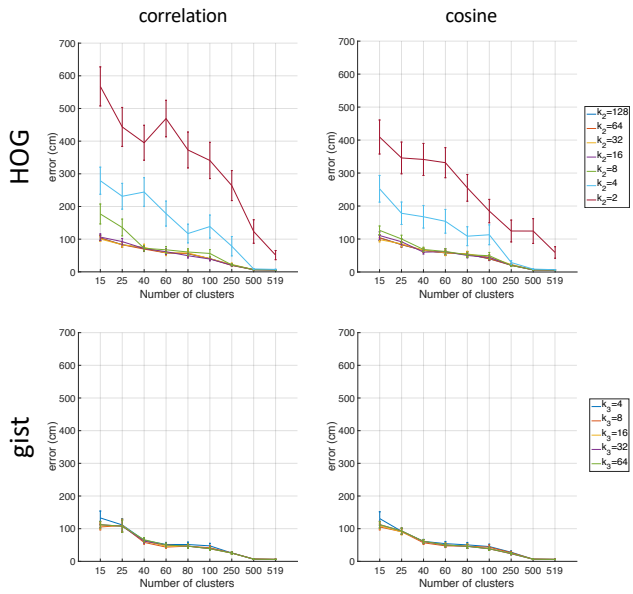



**Figure 18.** Results of the localization process with HOG and *gist* used to describe the representatives of the clusters and the test images: average localization error (cm) vs. number of clusters. Freiburg environment.

In this case, some differences are noticed between the results collected in the Quorum V environment and the results in the Freiburg environment. When the number of clusters is low ($n_c = [15, 25, 40]$), the localization task presents a lower average localization error with *gist*. If this number is higher than 40, the localization error is very similar for HOG and *gist*. Comparing the results obtained with the two evaluated types of distances, no remarkable differences are found. Nevertheless, a slight improvement can be noticed when the cosine distance is used. For instance, the average error value when $n_c = 40$ in HOG is lower with the cosine than with correlation.

Additionally, the value of $k_2$ in HOG is very important. The average error varies significantly according to it. Therefore, in order to solve the localization in an environment whose properties are similar to Freiburg or Saarbrücken (information along a trajectory), the optimal values are reached through the use of HOG descriptor with $k_2 = [16, 32]$ and cosine distance.

### 5.3.3. Localization When Several Maps Are Available

In some applications, several maps of some different environments are initially available. If the robot has no information about the environment it is located in, first, it has to use the visual information to select the correct environment. After that, the localization can be solved in the selected environment, as presented in Section 4. Considering this, in this section, the ability to select the right environment

is studied. In order to check the goodness of the descriptors for this purpose, the two COLD maps built in Section 5.2.2 are considered. Additionally, a test dataset is created as a combination of images from the Freiburg and Saarbrüken environments. A total of 60 test images compose the test dataset (34 from Freiburg and 26 from Saarbrücken). In this experiment, only HOG and *gist* are tested again. Furthermore, since the cosine distance presented the best solutions for COLD, only this kind of distance is applied. Figure 19 shows the percentage of success in selecting the right environment for the two descriptors.

By and large, the correct environment selection is almost always done. Many cases are given in which 100% success is reached, whereas the worst cases do not present a success rate under 75%. If the environment selection is carried out with HOG, results depend substantially on the chosen $k_2$ value. For instance, the worst cases are presented for $k_2 = 2, 4$. However, for $k_2 = 32 - 128$, 100% success is reached. Through the use of *gist* descriptor, 100% success is given independently of the number of clusters or the $k_3$ value.

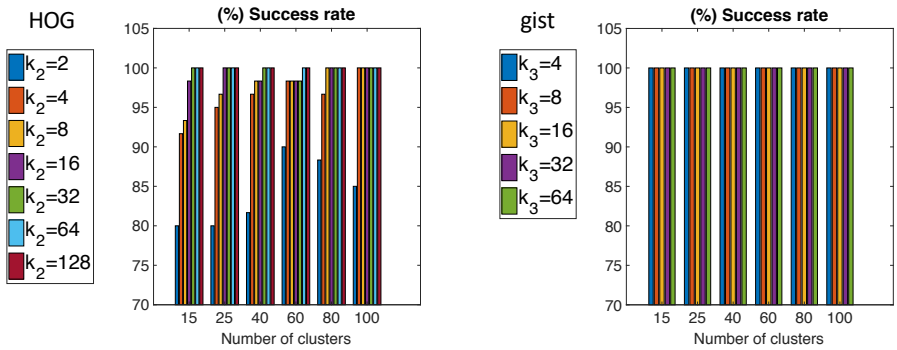



**Figure 19.** Percentage of success to detect the correct environment between Freiburg and Saarbrücken with FS, HOG, and *gist* used to describe the representatives of the clusters and the test images: percentage of success vs. number of clusters.

### *5.4. A Comparative Study of Localization with Straightforward and with Compact Maps*

Compact maps obtained after clustering present an effective solution to carry out the localization task in a high-level map, as shown in the previous experiments. This process requires capturing a high number of images from the environment to map, prior to the clustering process. At this point, we could ask the following question: is it necessary to capture this high number of images, or could we create a compact model directly, capturing only a limited number of images from the environment? In this section, this issue is studied. Two kinds of models are considered: (a) a compact model obtained after clustering a high number of images and (b) a straightforward model obtained by just capturing a limited number of views from the environment. Both kinds of models will be used to solve the high-level localization task. The straightforward method we propose to retain representatives is downsampling the databases: the COLD databases are downsampled, and only a certain number of images are retained (one of every $x$ images is retained).

The utility of this straightforward model will be compared with the utility of the optimal compact model obtained in Section 5.2.2 with spectral clustering.

Therefore, two models are used as departing points to carry out the localization task: (Model 1) departing from the representative instances obtained through the spectral clustering algorithm and (Model 2) departing from the instances obtained through sampling the databases. Afterwards, the localization task is studied in the Freiburg environment in the same way as was done in Section 5.3.2.

Figure 20 compares the utility of the two models in localization tasks. The cosine distance is selected to show these results, due to the fact that this distance presented good results in previous localization experiments. The two best global appearance descriptors for localization (HOG and *gist*) are shown.

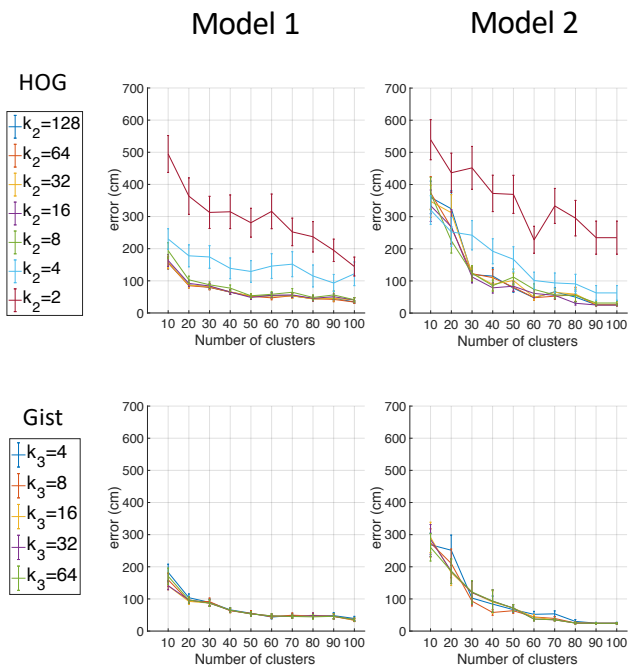


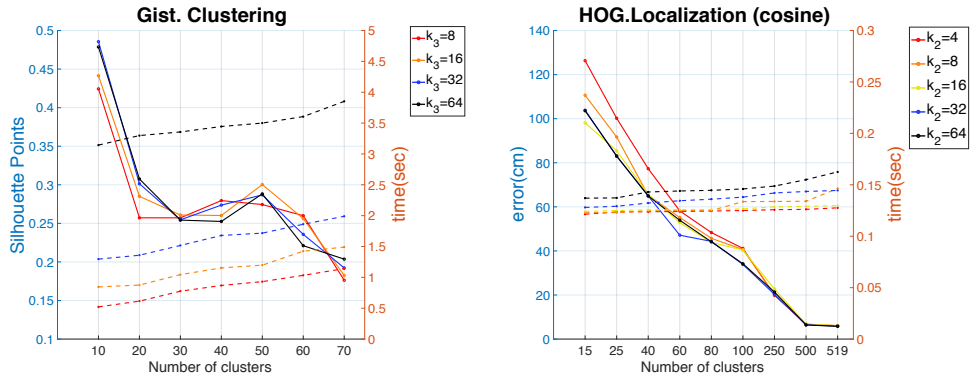

**Figure 20.** Results of the localization process in the Freiburg environment by using two types of models to retain visual representatives. Average localization error (cm) vs. number of clusters. Model 1 uses representatives obtained through spectral clustering, and Model 2 obtains the representatives through sampling the dataset. The localization task has been carried out with HOG and *gist*, and the distances are calculated through the cosine distance.

As can be seen, the localization error worsens when the straightforward map is used. When the number of clusters is low, the model that has been obtained through spectral clustering presents the best localization results. For example, independent of the descriptor, the average localization error is less than 100 cm when $n_c > 20$ for Model 1 and $n_c > 40$ for Model 2. The average localization error is lower for Model 2 only when the number of clusters is substantially high, $n_c > 80$ (HOG case) and $n_c > 70$ (*gist* case). This outcome means that the proposed alternative to spectral clustering may only be interesting when a low compactness is required. However, if the number of clusters is low (high compactness), spectral clustering provides better results. Therefore, as a conclusion, this experiment has proven that the use of straightforward methods to retain visual representatives is less efficient than using spectral clustering methods. Spectral clustering is able to create compact models that provide accurate localization results.

### *5.5. Discussion of the Results*

This subsection includes a brief discussion related to the results obtained throughout the present work. Regarding the use of methods to compress visual models, spectral clustering has proven to be, in general, more efficient than the SOM clustering. Furthermore, the global appearance descriptor, which presented better behaviour to carry out the clustering task, is *gist*. About the localization task,

HOG presented generally the best outcomes independently of the type of map. The best results are summarized in Figure 21. The best clustering results in Freiburg were obtained with *gist* ($k_3 = [32, 64]$ and $n_{masks} = 16$) and using spectral clustering. Moreover, the best localization outcome in this environment was obtained through the use of HOG ($k_2 = [16, 32]$) with the cosine distance.




(a) (b)

**Figure 21.** Best results of the clustering and localization processes. (**a**) Clustering with *gist* and spectral clustering: silhouette of points (left axis, solid lines) and computing time (right axis, dashed lines) vs. number of clusters. (**b**) Localization with HOG and cosine distance: average localization error (cm) (left axis, solid lines) and computing time (right axis, dashed lines) vs. the number of clusters. Freiburg environment.

Furthermore, comparing the localization results obtained after compaction and through using raw models, with no compaction ($n_c = 872$, $n_c = 519$, and $n_c = 566$ respectively for Quorum, Freiburg, and Saarbrücken), compact models have proven to be a successful tool to reduce computing time and keep the localization accuracy (see Section 4).

Regarding the use of the global appearance descriptor to select the right map among several options (Section 5.3.3), *gist* has proven to be the most efficient choice. Using this descriptor, 100% of success was reached independently of the number of clusters and the value $k_3$.

Finally, straightforward methods to compress the information can be discarded since they are not capable of keeping more information about the environment than the proposed spectral clustering method (Section 5.4). Despite that straightforward methods might be faster and easier, the localization outcomes obtained departing from spectral clustering proved to be, in general terms, more accurate.

## 6. Conclusions and Future Works

This paper proposes two different methods to compact topological maps. With this aim, three datasets from indoor environments were used. These datasets were composed by either panoramic images or omnidirectional images that were transformed to panoramic. During the experiments, with the objective of compacting the information, the number of instances was reduced to a value in the interval from 10–100. That means a reduction of instances up to between 1.1% and 11.5% of the original number. The proposed methods were (1) spectral clustering and (2) self-organizing maps. Moreover, three global appearance descriptors were used since they presented a good solution for environments whose data dimensionality was high. The work shows that it is possible to reduce the visual information drastically from the original model. Among these combinations of method-descriptor, spectral clustering along with the *gist* descriptor was proven to be the best choice to compact the model.

Once the original model is compacted, the resultant map can be used to solve the localization task. Hence, an evaluation is carried out with the aim of measuring the goodness of the localization task through the use of compact maps and global appearance descriptors. In this case, three descriptors and two indoor environments are evaluated. Furthermore, a mixture between indoor environments is created with the aim of evaluating whether it is possible, first, to detect the right environment and, second, estimate the position of the instance. From this study, HOG is the description method whose localization results were the best. Additionally, *gist* presented the most successful results in order to select the correct environment of a test instance from a combined dataset. Finally, the use of clustering methods to tackle the compression step has proven to be more efficient than carrying out a downsampling of the images directly from the database.

The team is now working on how the localization task through compact maps is affected by illumination changes. Additionally, other compacting methods will be studied in order to achieve the Simultaneous Localization And Mapping task (SLAM).

**Author Contributions:** Conceptualization, L.P. and O.R.; Methodology, L.P. and W.M.; Software, S.C.; Validation, L.P., S.C. and W.M.; Formal Analysis, O.R. and L.P.; Investigation, S.C. and O.R.; Resources, L.P. and W.M.; Data Curation, S.C. and W.M.; Writing—Original Draft Preparation, S.C.; Writing—Review & Editing, L.P. and O.R.; Visualization, S.C. and O.R.; Supervision, L.P.; Project Administration, O.R.; Funding Acquisition, L.P., O.R. and S.C.

**Funding:** This research was funded by the Generalitat Valenciana through Grant ACIF/2017/146 and by the Spanish government through the project DPI2016-78361-R (AEI/FEDER, UE): "Creación de mapas mediante métodos de apariencia visual para la navegación de robots".

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Okuyama, K.; Kawasaki, T.; Kroumov, V. Localization and position correction for mobile robot using artificial visual landmarks. In Proceedings of the 2011 International Conference on Advanced Mechatronic Systems, Zhengzhou, China, 11–13 August 2011; pp. 414–418.
2. Zhao, Y.; Cheng, W.; Liu, G. The navigation of mobile robot based on stereo vision. In Proceedings of the 2012 Fifth International Conference on Intelligent Computation Technology and Automation, Zhangjiajie, China, 12–14 January 2012; pp. 670–673.
3. Gwinner, K.; Jaumann, R.; Hauber, E.; Hoffmann, H.; Heipke, C.; Oberst, J.; Neukum, G.; Ansan, V.; Bostelmann, J.; Dumke, A.; et al. The High Resolution Stereo Camera (HRSC) of Mars Express and its approach to science analysis and mapping for Mars and its satellites. *Planet. Space Sci.* **2016**, *126*, 93–138. [CrossRef]
4. Jia, Y.; Li, M.; An, L.; Zhang, X. Autonomous navigation of a miniature mobile robot using real-time trinocular stereo machine. In Proceedings of the IEEE International Conference on Robotics, Intelligent Systems and Signal, Changsha, China, 8–13 Octorber 2003.
5. Valiente, D.; Gil, A.; Reinoso, Ó.; Juliá, M.; Holloway, M. Improved Omnidirectional Odometry for a View-Based Mapping Approach. *Sensors* **2017**, *17*, 325. [CrossRef] [PubMed]
6. Berenguer, Y.; Payá, L.; Ballesta, M.; Reinoso, O. Position Estimation and Local Mapping Using Omnidirectional Images and Global Appearance Descriptors. *Sensors* **2015**, *15*, 26368–26395. [CrossRef] [PubMed]
7. Tardif, J.P.; Pavlidis, Y.; Daniilidis, K. Monocular visual odometry in urban environments using an omnidirectional camera. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 Septtmber 2008; pp. 2531–2538.
8. Murillo, A.; Guerrero, J.; Sagues, C. SURF features for efficient robot localization with omnidirectional images. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 3901–3907.
9. Menegatti, E.; Pretto, A.; Scarpa, A.; Pagello, E. Omnidirectional vision scan matching for robot localization in dynamic environments. *IEEE Trans. Robot.* **2006**, *22*, 523–535. [CrossRef]
10. Payá, L.; Gil, A.; Reinoso, O. A State-of-the-Art Review on Mapping and Localization of Mobile Robots Using Omnidirectional Vision Sensors. *J. Sens.* **2017**, *2017*, 3497650. [CrossRef]

11. Pantazi, X.E.; Tamouridou, A.A.; Alexandridis, T.; Lagopodi, A.L.; Kashefi, J.; Moshou, D. Evaluation of hierarchical self-organising maps for weed mapping using uas multispectral imagery. *Comput. Electron. Agric.* **2017**, *139*, 224–230. [CrossRef]
12. Hagiwara, Y.; Inoue, M.; Kobayashi, H.; Taniguchi, T. Hierarchical Spatial Concept Formation Based on Multimodal Information for Human Support Robots. *Front. Neurorobot.* **2018**, *12*, 11. [CrossRef] [PubMed]
13. Hwang, Y.; Choi, B. Hierarchical System Mapping for Large-Scale Fault-Tolerant Quantum Computing. *arXiv* **2018**, arXiv:1809.07998.
14. Lowe, D.G. Object recognition from local scale-invariant features. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; pp. 1150–1157.
15. Bay, H.; Tuytelaars, T.; Gool, L. SURF: Speeded Up Robust Features. In *Computer Vision at ECCV 2006*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
16. Calonder, M.; Lepetit, V.; Strecha, C.; Fua, P. Brief: Binary robust independent elementary features. In Proceedings of the 11th European Conference on Computer Vision, Crete, Greece, 5–11 September 2010.
17. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
18. Angeli, A.; Doncieux, S.; Meyer, J.; Filliat, D. Visual topological SLAM and global localization. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 2029–2034.
19. Menegatti, E.; Maeda, T.; Ishiguro, H. Image-based memory for robot navigation using properties of omnidirectional images. *Robot. Autom. Syst.* **2004**, *47*, 251–267. [CrossRef]
20. Liu, M.; Scaramuzza, D.; Pradalier, C.; Siegwart, R.; Chen, Q. Scene recognition with omnidirectional vision for topological map using lightweight adaptive descriptors. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 Octorber 2009; pp. 116–121.
21. Payá, L.; Fernández, L.; Gil, A.; Reinoso, O. Map Building and Monte Carlo Localization Using Global Appearance of Omnidirectional Images. *Sensors* **2010**, *10*, 11468–11497. [CrossRef]
22. Rituerto, A.; Murillo, A.C.; Guerrero, J. Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robot. Autom. Syst.* **2014**, *62*, 685–695. [CrossRef]
23. Leonardis, A.; Bischof, H. Robust recognition using eigenimages. *Comput. Vis. Image Understand.* **2000**, *78*, 99–118. [CrossRef]
24. Oliva, A.; Torralba, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int. J. Comput. Vis.* **2001**, *42*, 145–175. [CrossRef]
25. Radon, J. Uber die bestimmug von funktionen durch ihre integralwerte laengs geweisser mannigfaltigkeiten. *Ber. Saechsishe Acad. Wiss. Math. Phys.* **1917**, *69*, 262.
26. Zivkovic, Z.; Bakker, B.; Krose, B. Hierarchical map building and planning based on graph partitioning. In Proceedings of the 2006 IEEE International Conference on Robotics and Automation, Orlando, FL, USA, 15–19 May 2006; pp. 803–809.
27. Grudic, G.Z.; Mulligan, J. Topological Mapping with Multiple Visual Manifolds. In Proceedings of the Robotics Science and Systems 2005 Workshop, Cambridge, MA, USA, 8–11 June 2005.
28. Valgren, C.; Lilienthal, A. SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments. *Robot. Autom. Syst.* **2010**, *58*, 149–156.
29. Stimec, A.; Jogan, M.; Leonardis, A. Unsupervised learning of a hierarchy of topological maps using omnidirectional images. *Int. J. Pattern Recognit. Artif. Intell.* **2007**, *22*, 639–665. [CrossRef]
30. Shi, X.; Shen, Y.; Wang, Y.; Bai, L. Differential-Clustering Compression Algorithm for Real-Time Aerospace Telemetry Data. *IEEE Access* **2018**, *6*, 57425–57433. [CrossRef]
31. Payá, L.; Mayol, W.; Cebollada, S.; Reinoso, O. Compression of topological models and localization using the global appearance of visual information. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017.
32. Mekonnen, A.A.; Briand, C.; Lerasle, F.; Herbulot, A. Fast HOG based person detection devoted to a mobile robot with a spherical camera. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 631–637.

33. Dong, L.; Yu, X.; Li, L.; Hoe, J.K.E. HOG based multi-stage object detection and pose recognition for service robot. In Proceedings of the 2010 11th International Conference on Control Automation Robotics & Vision, Singapore, 7–10 December 2010; pp. 2495–2500.
34. Dalal, N.; Triggs, B. Histograms of Oriented Gradients fot Human Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
35. Zhu, Q.; Avidan, S.; Yeh, M.; Cheng, K. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; pp. 1491–1498.
36. Payá, L.; Amorós, F.; Fernández, L.; Reinoso, O. Performance of global-appearance descriptors in map building and localization using omnidirectional vision. *Sensors* **2014**, *14*, 3033–3064. [CrossRef] [PubMed]
37. Oliva, A.; Torralba, A. Building the gist of ascene: The role of global image features in recognition. *Prog. Brain Res.* **2006**, *155*, 23–36.
38. Siagian, C.; Itti, L. Biologically Inspired Mobile Robot Vision Localization. *IEEE Trans. Robot.* **2009**, *25*, 861–873. [CrossRef]
39. Chang, C.; Siagian, C.; Itti, L. Mobile robot vision navigation and localization using Gist and Saliency. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 Octorber 2010; pp. 4147–4154.
40. Murillo, A.C.; Singh, G.; Kosecka, J.; Guerrero, J.J. Localization in Urban Environments Using a Panoramic Gist Descriptor. *IEEE Trans. Robot.* **2013**, *29*, 146–160. [CrossRef]
41. Fernández, L.; Payá, L.; Reinoso, Ó.; Gil, A.; Juliá, M. Robust Methods for Robot Localization under Changing Illumination Conditions-Comparison of Different Filtering Techniques. *ICAART* **2010**, *1*, 223–228.
42. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*; Prentice Hall: Upper Saddle River, NJ, USA, 2002.
43. Payá, L.; Reinoso, O.; Berenguer, Y.; Úbeda, D. Using Omnidirectional Vision to Create a Model of the Environment: A Comparative Evaluation of Global-Appearance Descriptors. *J. Sens.* **2016**, *2016*, 1–21. [CrossRef] [PubMed]
44. Fernández, L.; Paya, L.; Amoros, F.; Reinoso, O. Using Global Appearance Descriptors to Solve Topological Visual SLAM. In *Encyclopedia of Information Science and Technology*, 4th ed.; IGI Global: Philadelphia, PA, USA, 2018; pp. 6894–6905.
45. Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416. [CrossRef]
46. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On Spectral Clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2001; pp. 849–856.
47. Valgren, C.; Duckett, T.; Lilienthal, A. Incremental spectral clustering and its application to topological mapping. In Proceedings of the IEEE International conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 4283–4288.
48. Sorensen, D.C. Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations. In *Parallel Numerical Algorithms*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 119–165.
49. Kohonen, T. The self-organizing map. *Neurocomputing* **1998**, *21*, 1–6. [CrossRef]
50. Van Gassen, S.; Callebaut, B.; Van Helden, M.J.; Lambrecht, B.N.; Demeester, P.; Dhaene, T.; Saeys, Y. FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data. *Cytometry Part A* **2015**, *87*, 636–645. [CrossRef] [PubMed]
51. Thrun, S.; Fox, D.; Burgard, W.; Dellaert, F. Robust Monte Carlo localization for mobile robots. *Artif. Intell.* **2001**, *128*, 99–141. [CrossRef]
52. Pérez, J.; Caballero, F.; Merino, L. Enhanced Monte Carlo localization with visual place recognition for robust robot localization. *J. Intell. Robot. Syst.* **2015**, *80*, 641–656. [CrossRef]
53. Rui, Y.; Huang, T.S.; Chang, S.F. Image retrieval: Current techniques, promising directions, and open issues. *J. Vis. Commun. Image Represent.* **1999**, *10*, 39–62. [CrossRef]
54. Automation, Robotics and Computer Vision Research Group. Quorum 5 Set of Images. Available online: http://arvc.umh.es/db/images/quorumv/ (accessed on 1 June 2018).
55. Pronobis, A.; Caputo, B. COLD: COsy Localization Database. *IJRR* **2009**, *28*, 588–594. [CrossRef]


IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 22, 2019, accepted April 4, 2019, date of publication April 11, 2019, date of current version April 24, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2910581*

# Hierarchical Localization in Topological Models Under Varying Illumination Using Holistic Visual Descriptors

**SERGIO CEBOLLADA, LUIS PAYÁ, VICENTE ROMÁN, AND OSCAR REINOSO, (Senior Member, IEEE)**
Department of Systems Engineering and Automation, Miguel Hernández University, 03202 Elche, Spain

Corresponding author: Sergio Cebollada (sergio.cebollada@umh.es)

This work was supported in part by the Generalitat Valenciana under Grant ACIF/2017/146 and Grant ACIF/2018/224, and in part by the Spanish government through the project (AEI/FEDER, UE): "Creación de mapas mediante métodos de apariencia visual para la navegación de robots" under Grant DPI 2016-78361-R.

**ABSTRACT** In this paper, a hierarchical localization framework within indoor environments is proposed and evaluated, considering severe variations of the illumination conditions. The only source of information both to build a model of the environment and to solve the localization problem is a catadioptric vision system, which is mounted on the mobile robot. The images captured by this system are processed globally to obtain holistic descriptors. The position of the robot is estimated by comparing these descriptors with the information contained in a topological visual model, which is previously created using a clustering approach and is composed of a hierarchy of layers. Compacting the information via clustering proves to be an efficient alternative to estimate the position of the robot hierarchically and with robustness. The proposed localization strategy is tested with some sets of panoramic images, captured in large indoor environments under real operating conditions, including illumination changes that change substantially the appearance of the scenes. The results show a reasonable tradeoff computation time-accuracy when the localization is addressed in a hierarchical way.

**INDEX TERMS** Localization, omnidirectional visual information, global appearance descriptors, clustering, illumination changes.

## I. INTRODUCTION

Nowadays, the use of omnidirectional vision sensors in mobile robotics for solving mapping and localization has considerably increased. They have been successfully used by different authors for these purposes. For instance, Valiente *et al.* [1] used the local features extracted from omnidirectional images to generate a reliable visual odometry to improve the Simultaneous Localization And Mapping (SLAM) task. Marinho *et al.* [2] used feature extractions and machine learning techniques to solve localization using omnidirectional images. Faessler *et al.* [3] present a vision-based quadrotor system to map a dense three-dimensional area online with the purpose of removing delay between the quadrotor and external systems. Berenguer *et al.* [4] considered the global appearance of omnidirectional images to create local maps and to estimate the position of a robot within these maps. This kind of images covers a field of view of 360 *deg* around the robot. Hence, they offer a huge amount of information from the surroundings of the robot which permits both building rich maps and estimating the robot position. Working with images requires a step to obtain functional, robust and relevant information from them. Commonly, two methods to extract relevant information have been considered in the related literature: either detecting, describing and tracking some relevant landmarks over the image (such as [5]–[7]) or creating a unique descriptor per image which contains global information about it (for instance, [8]–[10]). As for the second proposed method, on the one hand, it usually leads to more direct localization algorithms. Basically, they consist in a pairwise comparison between descriptors. On the other hand, it presents a lack of metric information. Therefore, this kind of descriptors are usually used to build topological maps (such as [11]–[13]).

The associate editor coordinating the review of this manuscript and approving it for publication was Yongqiang Zhao.

In order to address the mapping and localization issue, arranging the topological information hierarchically constitutes an efficient alternative. This framework consists in creating a map which is composed of several layers with a hierarchical structure. The high-level ones present a relatively compact amount of information, which permits a rough but quick localization. The low-level layers have usually more information and are used to refine the position. A good example of this issue was developed by Stimec *et al.* [14], who proposed an unsupervised hierarchical mapping method. Garcia-Fidalgo and Ortiz [15] presented a review about the main approaches considered to carry out topological mapping and localization through visual information in the last years. recently, da Silva *et al.* [16] propose a localization and navigation approach for mobile robots using topological maps and using CNN to obtain descriptors from omnidirectional images.

Considering this information, the main objective of this work consists in proposing an approach to solve the localization problem using hierarchical models. Moreover, a comparative evaluation of some global descriptors is carried out to know which one behaves more robustly against illumination changes. The results obtained throughout this work permit selecting the best global descriptor method and also tuning correctly its parameters in order to obtain optimal results (the maximum accuracy and the lowest computational time). Additionally, the use of approaches based on deep learning are also considered to describe the scenes globally. The aim consists in evaluating which method solves more efficiently the localization task under the conditions previously exposed.

An omnidirectional vision sensor [17] is the unique source of information used to carry out mapping and localization in this work. The images used in the experiments are obtained from an indoor dataset (explained in IV-A.1) and they are described through global appearance descriptors. The present work continues and expands the research framework presented in [18], where an approach is proposed to build compact topological models of the environment. The approach consists in the use of clustering algorithms, which are non-supervised techniques, along with holistic visual descriptors, and both the correctness of the model and its utility to solve the localization problem is assessed. An exhaustive evaluation of different clustering methods was carried out in [18]. In that work, Spectral Clustering along with the holistic descriptor *gist* was chosen as the configuration which best tackled the mapping task. Hence, in this work, the localization algorithms are tested with the compact maps obtained with this combination of methods (*gist* + spectral clustering). These compact models are the basis of the present work, whose main differences and contributions are: (a) solving the localization problem hierarchically, with different degrees of granularity, (b) making an exhaustive comparative evaluation of the method and testing its robustness under severe illumination variations and (c) including in the evaluation a new holistic description method, based on deep learning (obtained through convolutional neural networks).

The remainder of the paper is structured as follows. Section II outlines the global appearance descriptors used along this work. After that, section III explains briefly the clustering approach used to compress the information and section IV presents the experiments carried out to test the validity of the proposed methods to solve the localization under changing lighting conditions. At last, the conclusions are presented in section V.

## II. THE GLOBAL APPEARANCE DESCRIPTOR

This section focuses on the methods used to describe the global appearance of the set of images. Four methods are evaluated in this paper: the Fourier Signature (FS), the Histogram of Oriented Gradients (HOG), the *gist* of the scenes and a global descriptor based on a Convolutional Neural Network (CNN). In order to reduce the effect of changing lighting conditions, the homomorphic filter [19] is applied over the images before describing them with HOG, since previous works [20] concluded that this pre-filtering improves the localization results when HOG is used.

The panoramic image $im(x, y) \in R^{N_x \times N_y}$ is the starting point, hence, a conversion from omnidirectional to panoramic must be carried out. After that, one of the four proposed description methods is used to calculate the global appearance descriptor vector $d \in R^{l \times 1}$. A deep description of FS, HOG and *gist* methods can be found in [21]. As for the use of CNN as global feature extractor, a wide explanation is presented in [22].

Regarding the FS descriptor, it was firstly used by Menegatti *et al.* [8]. This method calculates the discrete Fourier Transform of each row of the panoramic image and a complex matrix is obtained $IM(u, v)$. The $k_1$ first columns are retained (compression effect) $IM(u, v) \in C^{N_x \times k_1}$. Finally, a decomposition is tackled to obtain just the magnitudes information (the resulting matrix is invariant to robot orientation changes) and the rows of the resultant matrix are arranged to create a vector, obtaining the global appearance descriptor $d \in R^{N_x \cdot k_1 \times 1}$.

As for the HOG descriptor, it was firstly used by Dalal and Triggs [23] for a pedestrians detection task. The version used in this work consists in splitting the panoramic image into $k_2$ horizontal cells and compiling a histogram of gradients orientation per each cell with $b$ bins per histogram [24]. The set of histograms compose the final descriptor $d \in R^{b \cdot k_2 \times 1}$.

With regard to the *gist* descriptor, Oliva and Torralba [25] introduced this method, which has been widely used for scenes recognition. Several versions can be found depending on the features of the image used. In this case, firstly, $m_2$ different resolution images are created from the original panoramic one. Secondly, Gabor filters are applied over the $m_2$ images with $m_1$ different orientations each. Thirdly, the pixels of each image are grouped into $k_3$ horizontal blocks and finally, the obtained orientation information is grouped to create a vector, which is the resultant descriptor $d \in R^{m_1 \cdot m_2 \cdot k_3 \times 1}$. This descriptor has already been used in mobile robot localization. For instance, Murillo *et al.* [26] used

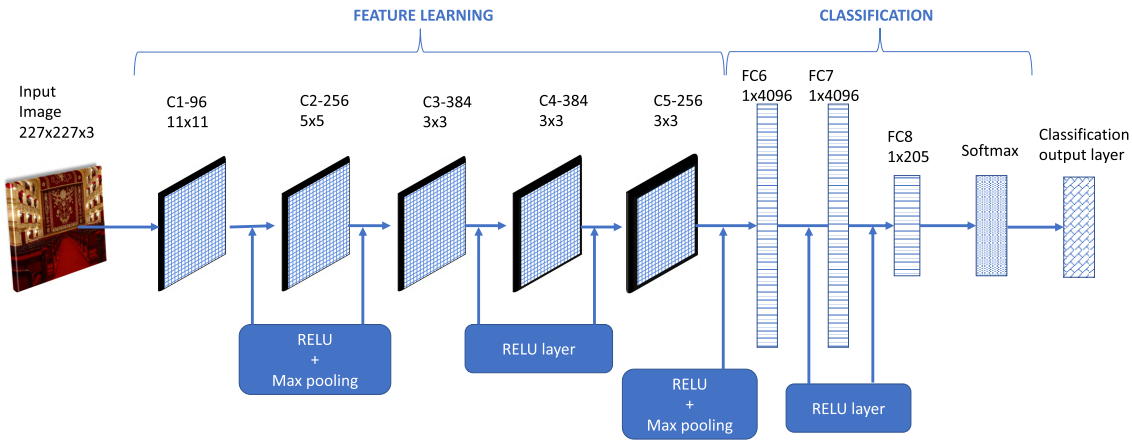



**FIGURE 1.** CNN *places* architecture design based on the pre-trained *'Caffe'* model. Layers *'fc7'* and *'fc8'* are used in this work as a method to obtain holistic descriptors from the original input image.

it with panoramic images for localization in urban regions including loop closure detection.

Last, concerning the use of the CNN-based descriptor, this method comes from the use of deep learning for classification, as Krizhevsky *et al.* [27] do. The neural network tackles two steps. First, it carries out a learning process, i.e., a set of images (which are already labeled) are collected and introduced to the network. Second, once trained, the network receives new images (also labeled) and tunes its internal parameters to optimize the results. After that, the network is available to face the classification task: a new image is introduced and the CNN returns the most likely label option. During the process of classification, descriptors are obtained by the fully connected layers which are within the neural network. These descriptors can be seen as global appearance descriptors of the input image. Therefore, they may be also used to carry out the localization task in the same way as the previously proposed global appearance descriptors. The neural network architecture that we use in this work is *places* [28], which was trained with around 2.5 million images to categorize 205 possible kinds of scenes. The fig. 1 shows the architecture of this CNN. To obtain holistic descriptors from these layers, the networks is directly used with the pre-training done by the creators, hence, a re-training is not necessary. The CNN is used directly as it appears in [29]. The descriptors extracted from this network correspond to the ones calculated in the layers *'fc7'* and *'fc8'*. These descriptors contain respectively 4096 ($d \in R^{4096\times 1}$) and 205 ($d \in R^{205\times 1}$) components. This kind of descriptor has been used by other authors such as Mancini *et al.* [30], who use them to carry out place categorization with the Naïve Bayes classifier. As for mobile robot localization, Payá *et al.* [22] proposed CNN-based descriptors to create hierarchical visual models. In a different way, Xu *et al.* [31] propose the use of a CNN which detects objects from the images and establishes relationships between the detected objects. Afterwards, the relationships established are used to calculate similitude between images. Nevertheless, in our work, CNNs are used just with the purpose of obtaining a holistic descriptor per scene.

## III. CLUSTERING THE VISUAL INFORMATION

This section outlines the clustering method used to compact the model. The clustering process departs from a set of images $I = \{im_1, ..., im_n\}$. These images were captured from different positions within the environment to map. The image capturing positions are known, but they are only used as *ground truth* $P = \{(x_1, y_1), ..., (x_n, y_n)\}$. Then, a set of global appearance descriptors $D = \{d_1, ..., d_n\}$ is calculated, one per image (through one of the description methods explained in section II). To create a compact model, a clustering process will be carried out with the components of $D$.

Several studies about clustering have been carried out. For instance, Theodoridis and Koutroumbas [32] developed a wide study about clustering and von Luxburg [33] provided a complete tutorial about the most common spectral clustering methods. This kind of algorithms have proved to be more effective than the traditional ones when the data size is high. Furthermore, the Spectral Clustering developed by Ng *et al.* [34] confirmed to be a good solution in these situations. This algorithm only considers the similitudes between instances $d_i$ and $d_j$: $S_{i,j} = e^{-\frac{|d_i - d_j|^2}{2\sigma^2}}$, where $\sigma$ is a parameter which controls the rapidity of reduction of the similitude when the distance between $d_i$ and $d_j$ increases. The clustering process is as follows:

1) Calculation of the normalized Laplacian matrix:

$$L = I - D^{-1/2} S D^{1/2} \tag{1}$$

where D is a diagonal matrix $D_i = \sum_{j=1}^{N} S_{ij}$.

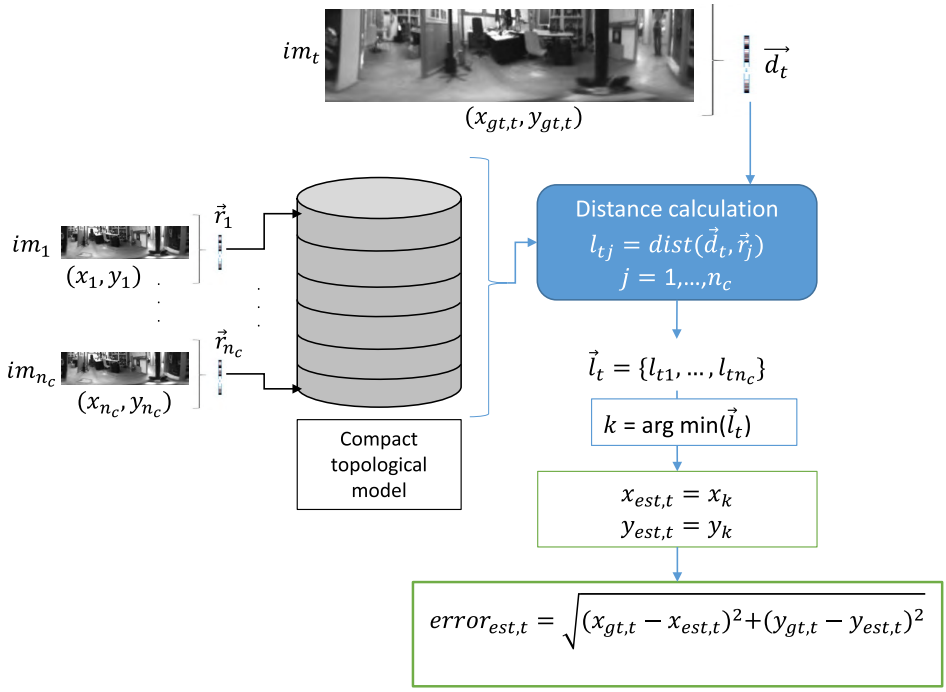



**FIGURE 2.** Block diagram regarding the steps to carry out the localization task through compact models.

2) Calculation of the $n_c$ main eigenvectors of $L$, $\{u_1, u_2, ..., u_{n_c}\}$. Arranging these vectors by columns, the matrix $U \in R^{n \times n_c}$ is obtained.
3) The matrix $U$ is normalized to obtain the matrix $T \in \mathbb{R}^{n \times n_c}$.
4) Extraction of vector $y_i \in R^{n_c}$ from the i-th row of the matrix $T$. $i = 1, ..., n$.
5) The $y_i$ vectors are clustered by using a simple clustering algorithm (k-means in this work). The clusters $A_1, A_2, ..., A_{n_c}$ are obtained.
6) Last, the clusters with the original data are obtained as $C_1, C_2, ..., C_{n_c}$ where $C_i = d_j$ such that $y_j \in A_i$.

After the clustering process, the representative of each cluster is calculated as the average of the descriptors which compose a specific cluster. The final result is a set of representatives $R = \{r_1, ..., r_{n_c}\}$, which constitutes the compressed map (i.e. the high-level layer of the hierarchical map). It can be used to carry out the localization task in a more efficient way.

## IV. LOCALIZATION UNDER CHANGING LIGHTING CONDITIONS

In a previous work [20], among the traditional global appearance descriptors, *gist* proved to be the most efficient to compact the model in indoor environments. Now, a comparative study of the proposed descriptors for localization under illumination changes is tackled including also the description method based on CNN.

### A. LOCALIZATION THROUGH COMPACT MODELS

The localization step is carried out once the compressed map is built. Hence, the starting point is a compact topological model, which consists of a set of $n_c$ representatives $\{r_1, ..., r_{n_c}\}$ and the coordinates of each cluster $\{(x, y)_1, ..., (x, y)_{n_c}\}$, where $n_c$ is the number of clusters. Nevertheless, the coordinates are only used as ground truth to test the accuracy. Only visual information is used during the localization. It allows us to carry out a pure evaluation of the visual description methods through avoiding the influence of other type of information. The accuracy is evaluated through the following error equation $error_{est,t} = \sqrt{(x_{gt,t} - x_{est,t})^2 + (y_{gt,t} - y_{est,t})^2}$, where $(x_{gt,t}, y_{gt,t})$ is the pose provided by the ground truth and $(x_{est,t}, y_{est,t})$ is the pose estimated by the algorithm for the test image $t$.

The localization task is performed through the following steps: first, it is assumed that the previous position of the robot is unknown; second, the robot captures a new image $im_t$ (an image from the test dataset which is different from the images used to create the map) and describes that image to obtain the descriptor $d_t$; third, the distance between $d_t$ and each representative descriptor is calculated, obtaining a distances vector $l_t = \{l_{t1}, ..., l_{tn_c}\}$ where $l_{tj} = dist(d_t, r_j)$; fourth, the minimum value of $l_t$ indicates the cluster which corresponds to the current position of the robot. A block diagram about these steps is shown in fig. 2.

### 1) EXPERIMENTS

To carry out the experiments, the COLD (COsy Localization Database) database is used [35]. This database is composed of several sets of omnidirectional images captured with a catadioptric vision system composed of a *Videre Design MDCS2* camera and a hyperbolic mirror. The images were collected under three different illumination conditions (cloudy days, sunny days and at nights). The Freiburg and Saarbücken datasets (images acquired at indoor laboratory environments located in those cities) were used to develop the experiments in this work. The images captured during cloudy weather are used to build a compact model through spectral clustering since they are the ones which are less affected by brightness, reflections, dark areas and thus, they provide more information. The sunny weather images and also the images captured at night are used as test images to evaluate the localization task under lighting changes.

Both datasets are composed of several rooms, such as corridors, personal offices, printer areas, kitchens, bathrooms, etc. The selected Freiburg dataset covers 9 different rooms and the Saarbrücken dataset covers 8. This dataset includes different changes in the environment such as people walking or position of furniture and objects. The datasets contain also images which do not provide much information due to the acquisition position and blurry images. All these handicaps make these datasets suitable to carry out experiments under real operating conditions. From the original cloudy dataset, a downsampling is carried out in order to obtain an acquisition distance between images of 40 cm approximately. This downsamplig is carried out because it is desirable to keep the model configuration which was used in previous works ( [20] and [21]). Hence, after downsampling, a training dataset composed of 519 (in Freiburg) and 566 (in Saarbrücken) images are considered. Furthermore, departing from the sunny and night datasets, three test datasets are created. Those images were selected randomly across the whole map. The table 1 summarizes the sets created for the experiments. The fig. 3 shows some examples of omnidirectional images in both environments under the proposed illumination conditions.

**TABLE 1. Datasets created from the COLD database to carry out the experiments.**

| Dataset name | Illumination condition | Number of images | Path length (m) |
|---|---|---|---|
| Freiburg_training | Cloudy | 519 | 104.2 |
| Freiburg_test_night | Night | 58 | |
| Freiburg_test_sunny | Sunny | 45 | |
| Saarbrücken_training | Cloudy | 566 | 156.6 |
| Saarbrücken_test | Night | 57 | |

As mentioned before, the localization experiment departs from the compact model. Several compact maps have been built, considering different numbers of clusters $n_c = [10, 20, ..., 100, N_{env}]$ where $N_{env}$ is the total number of images which compose the model (519 in the Freiburg environment and 566 in Saarbücken, table 1). It will enable us to analyze the localization process considering different granularities in the high-level layer of the map. The case $n_c = N_{env}$ provides information about the localization process when all the images of the original model are considered (i.e., no compression is performed and the localization is addressed as an image retrieval problem). This way, it can be seen as a reference to test the utility of the compact maps. To create the clusters, *gist* was chosen since it has provided the best results in previous works [20] and its parameters are tuned to $k_3 = 32$ and $n_{masks} = 16$. The fig. 4 shows examples of a sample clustering experiment applied to the datasets, according to the spectral clustering method. The images of these datasets are under cloudy illumination conditions.

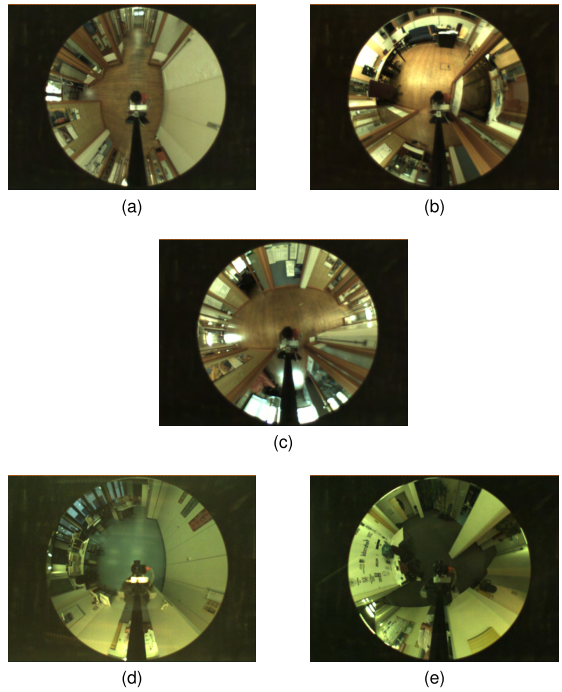



**FIGURE 3. Some sample omnidirectional images belonging to the Freiburg environment under (a) cloudy, (b) night and (c) sunny illumination conditions and also images which belong to the Saarbrücken environment under (d) cloudy and (e) night illumination conditions.**

Once the compact map is available (i.e. the clusters' representatives have been calculated), the localization is estimated as follows. Among the $n_c$ clusters, the node whose distance presents the minimum value of $l_t$ is chosen as the one which the captured image belongs to. Therefore, to estimate the goodness of the localization task, the Euclidean distance between the position where the test image was captured and the position of the nearest neighbour is calculated. Additionally, the computation time is measured since the scope is to reach a balance between accuracy and computational time. The experiments have been carried out in a PC with two CPU Quad-Core Intel Xeon® at 2,8 GHz and through Matlab® programming.

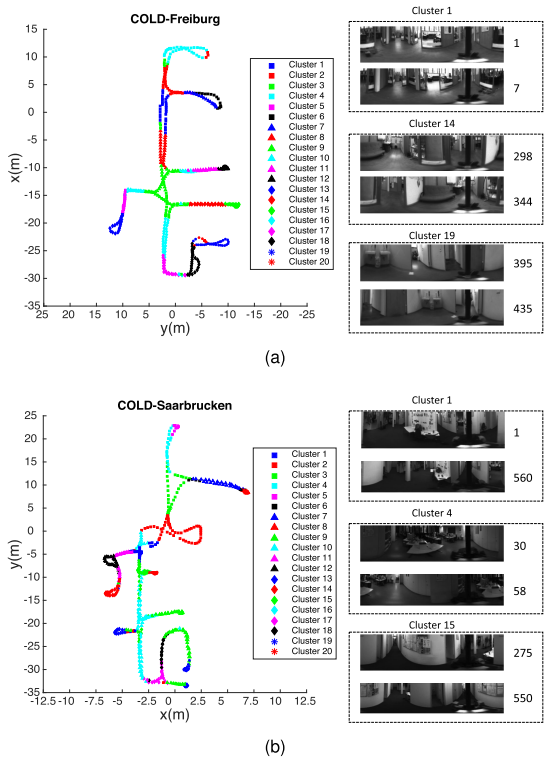



**FIGURE 4.** Results of a sample clustering considering $n_c = 20$ clusters and *gist* descriptor with $k_3 = 32$ and $n_{masks} = 16$. Some sample panoramic images belonging to the (a) Freibug and (b) Saarbrücken dataset under cloudy illumination conditions are shown.

To calculate the distance between descriptors, three kinds of distances are considered: the correlation distance, the cosine distance and the Euclidean distance. Also, three illumination conditions are considered: the cloudy condition, the night condition and the sunny condition. The dataset under cloudy condition is the one used to create the compact map (clustering and obtaining the representatives). Night and sunny conditions are used to evaluate the localization task under illumination changes. Fig. 5 shows the average localization error (cm) vs. the number of clusters $n_c$ obtained in the Freiburg environment when the test dataset was night and fig. 6 when the test dataset was sunny; fig. 7 shows the average localization error (cm) obtained in the Saarbrücken environment when the test dataset was night. In all cases, the localization error is expressed in cm, and the colorbar that expresses this error has the same range, to facilitate a comparative evaluation between figures.

In general, as the number of clusters increases, the average localization error tends to decrease. This behavior was expected and was also remarked in previous works [20]. When there is a low number of clusters, the plots present high error values, as expected. This is due to the fact that

**TABLE 2.** Computation time (sec) required to obtain the global appearance descriptor (HOG and CNN) per each test image. Freiburg test dataset under night conditions.

| descriptor | | time (ms) |
|---|---|---|
| HOG | k2=2 | $131.0 \pm 0.58$ |
| | k2=4 | $145.8 \pm 0.34$ |
| | k2=8 | $148.3 \pm 0.12$ |
| | k2=16 | $158.1 \pm 0.19$ |
| | k2=32 | $177.8 \pm 0.93$ |
| | k2=64 | $192.3 \pm 0.41$ |
| CNN | 'fc7' | $444.7 \pm 5.62$ |
| | 'fc8' | $453.3 \pm 4.51$ |

despite the matching between test images and representatives has been successful, the representatives are too sparse among them. Moreover, as for the illumination conditions, if we compare the outputs obtained under night conditions and the ones obtained under sunny conditions (see fig. 5 and fig. 6), generally, sunny conditions have a more negative impact upon the localization. For example, when using the CNN descriptor layer *'fc7'*, if $n_c = 10$, the error under night conditions is over 200 cm whereas under sunny conditions, it is over 300 cm. If $n_c = 60$, the error under night conditions is under 100 cm and under sunny conditions, it is over 200 cm.

Among the four studied global appearance descriptors, FS is the one which presents worst localization results in general. As for HOG, this descriptor presents relatively good localization error results. For example, for night conditions in the Freiburg environment (see fig. 5), when a correct tuning of the $k_2$ parameter is carried out and for more than 50 clusters, the localization error values are under 100 cm. It can be considered a successful result considering the size of the environment (table 1) and the granularity of the compact map. Moreover, the best results are obtained when the cosine distance is applied. In the case of the Saarbrücken environment, HOG presents slightly worse results than in Freiburg (fig. 7). *Gist* presents also relatively good localization results and they are not as influenced by the $k_3$ parameter (number of horizontal blocks) as the HOG results with $k_2$. For instance, in the *gist* results presented in the fig. 5, the error decreases until the number of clusters is 40 and after that value, the average localization error keeps almost constant. For the *gist* descriptor, the Euclidean distance presents the worst results whereas the cosine and correlation distances are quite similar. The CNN descriptor presents as good localization results as using HOG in Freiburg at night. Through the use of CNN with the layer *'fc7'*, the localization error is lower than 100 cm when $n_c > 30$ (using either correlation or cosine distance). Moreover, the results in the Saarbrücken environment are the best. Nevertheless, under sunny conditions (see fig. 6), CNN is more affected than HOG.

Among the two best descriptors which present best localization outputs (HOG and CNN), a computation time evaluation is obtained. With this aim, the time required to calculate the global appearance descriptors is performed. Table 2 shows

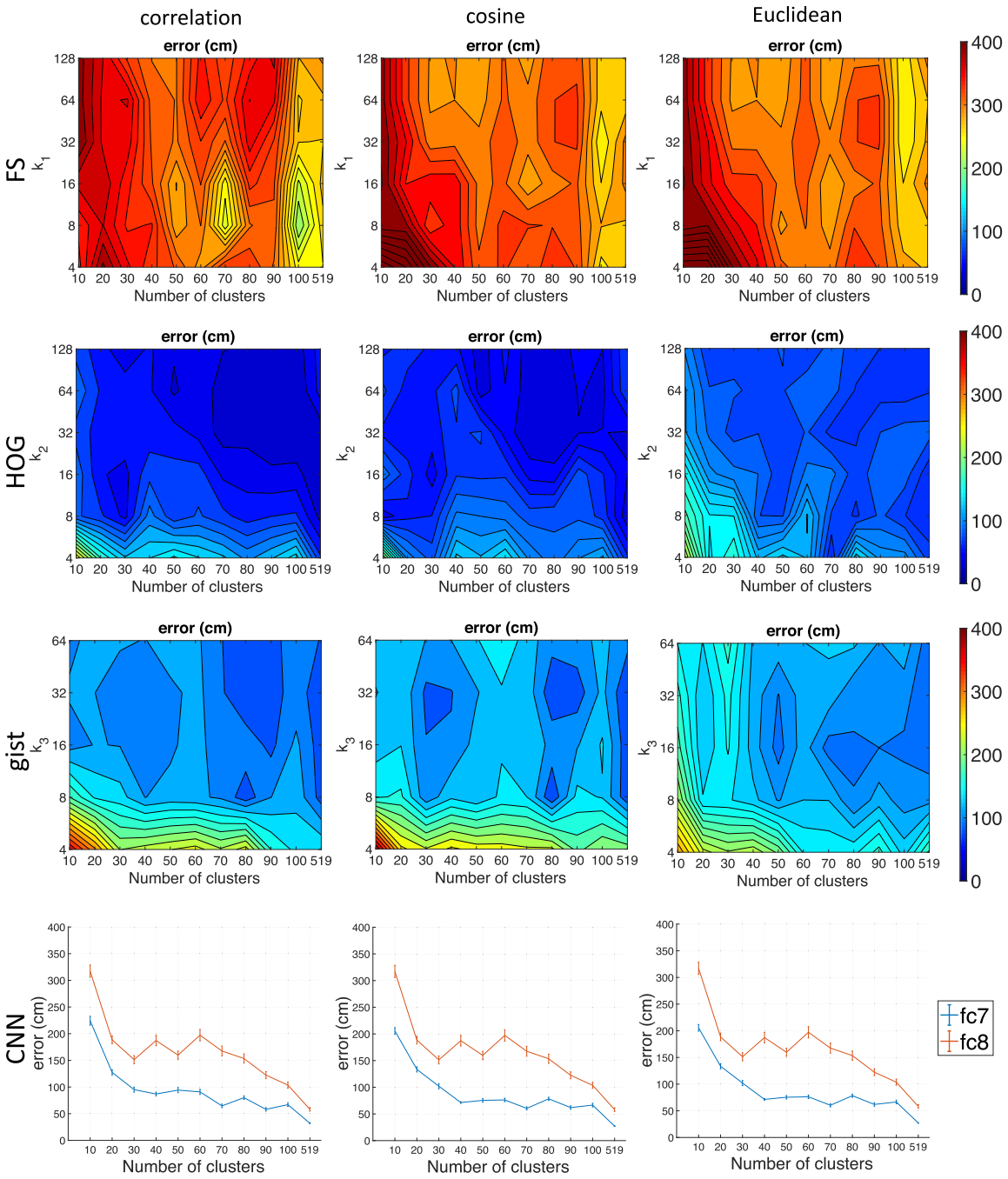



**FIGURE 5.** Results of the localization task when the night illumination conditions affect the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

the average computational time (sec) to compute the global appearance descriptor for the *Freiburg_test_night* dataset. As for HOG, the obtained values keep almost constant independently on the value of $k_2$. Regarding the use of the CNN descriptor, the related time values are higher (around 0.4 sec).

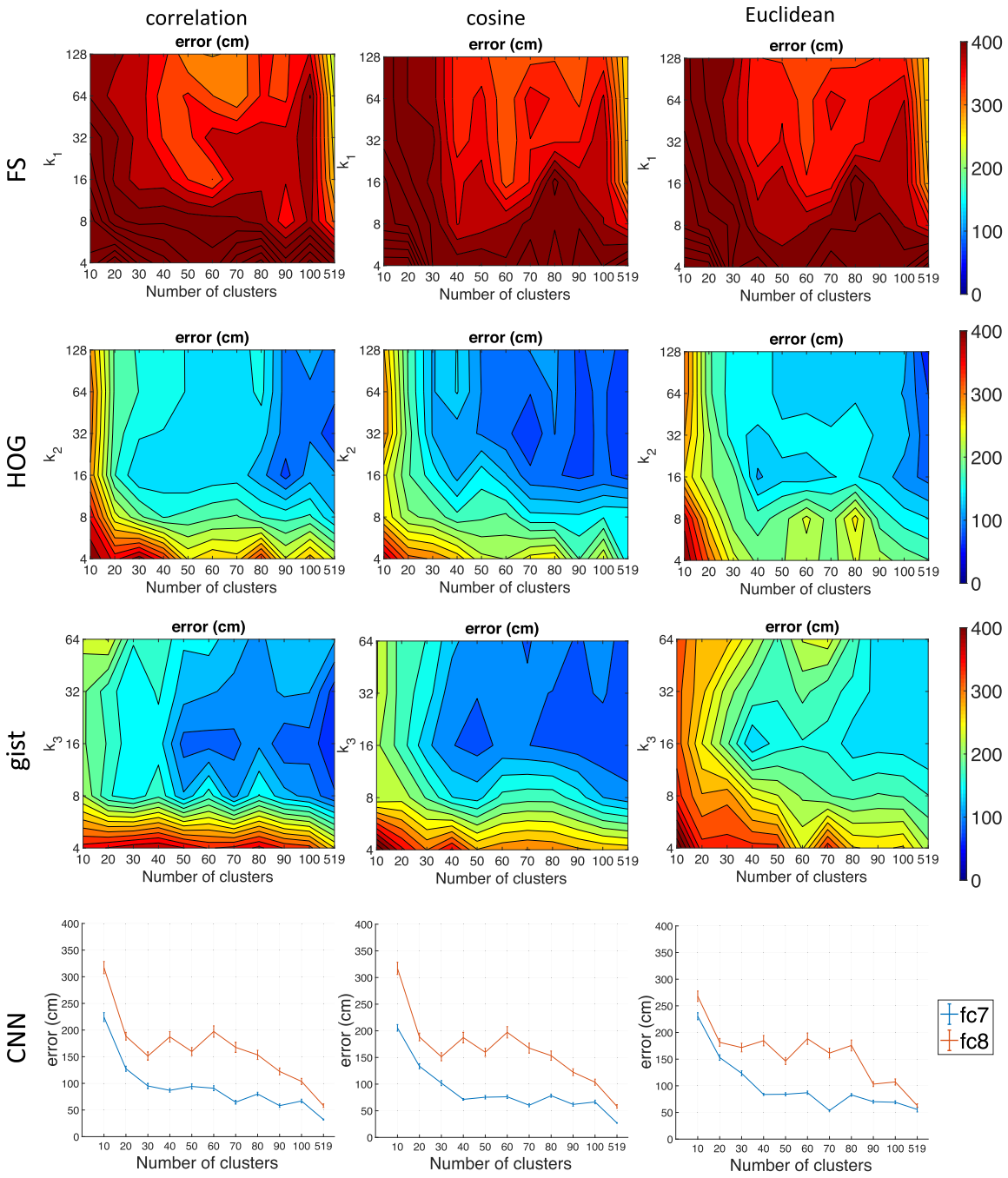



**FIGURE 6.** Results of the localization task when the sunny illumination conditions affect the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

In conclusion, among the different global appearance descriptors studied to solve the localization task in environments which present changes of illumination, CNN will be the optimal option. FS and *gist* localization values are relatively worse. HOG presents better results in the Freiburg environment, but in the Saarbrücken environment, results for

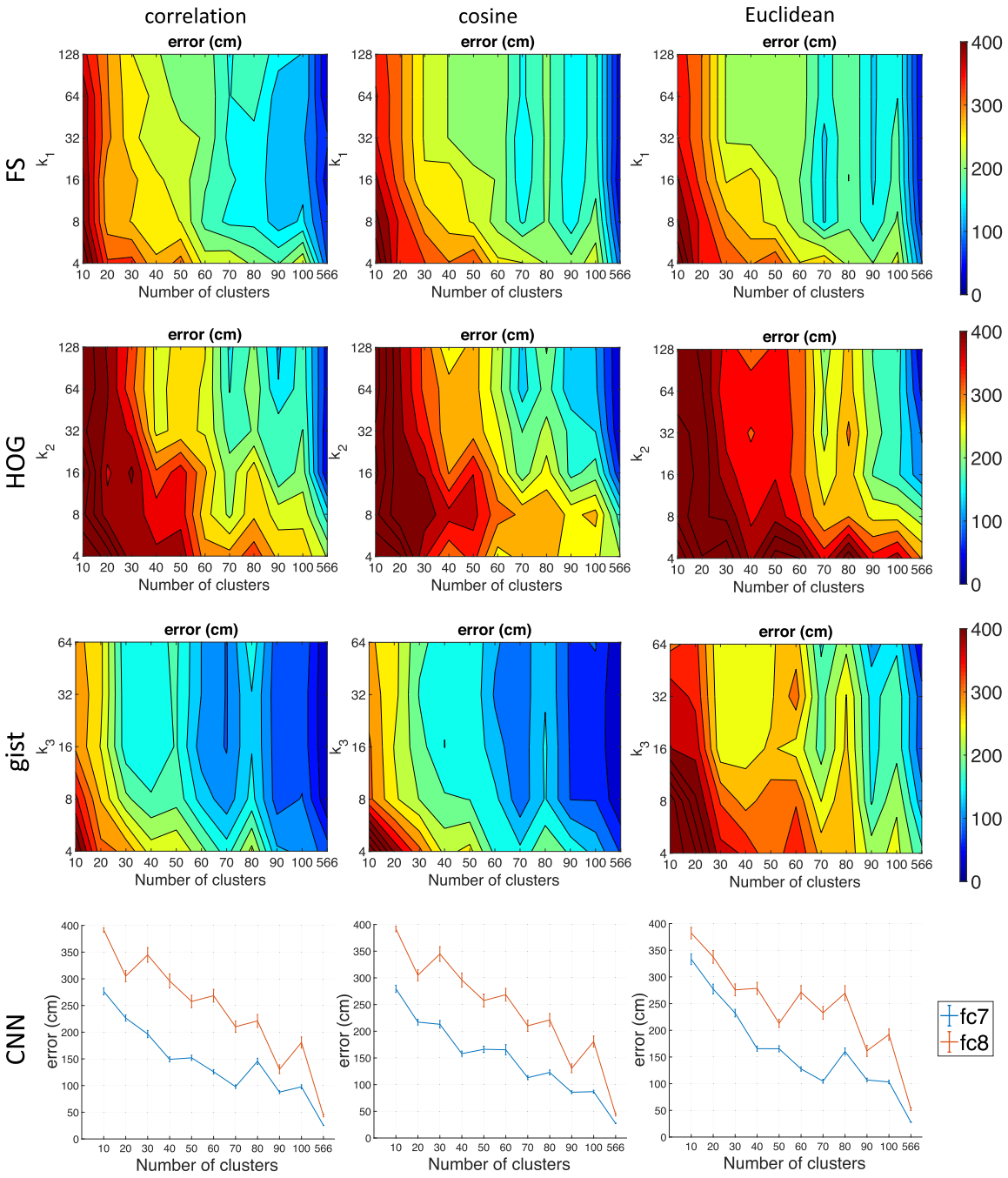



**FIGURE 7.** Results of the localization task when the night illumination conditions affect the Saarbrücken environment. Average localization error (cm) vs. number of clusters and descriptor size. Different description methods (FS, HOG, *gist* and CNN) and distances (correlation, cosine and Euclidean) are considered.

HOG are poor, whereas CNN keeps being also good. Despite the computing time is not as low as the HOG one, it is not substantially higher than the HOG results. As for the illumination changes, HOG is less affected by the sunny conditions than the rest of descriptors. Regarding which type of distance measure is better to calculate the distance

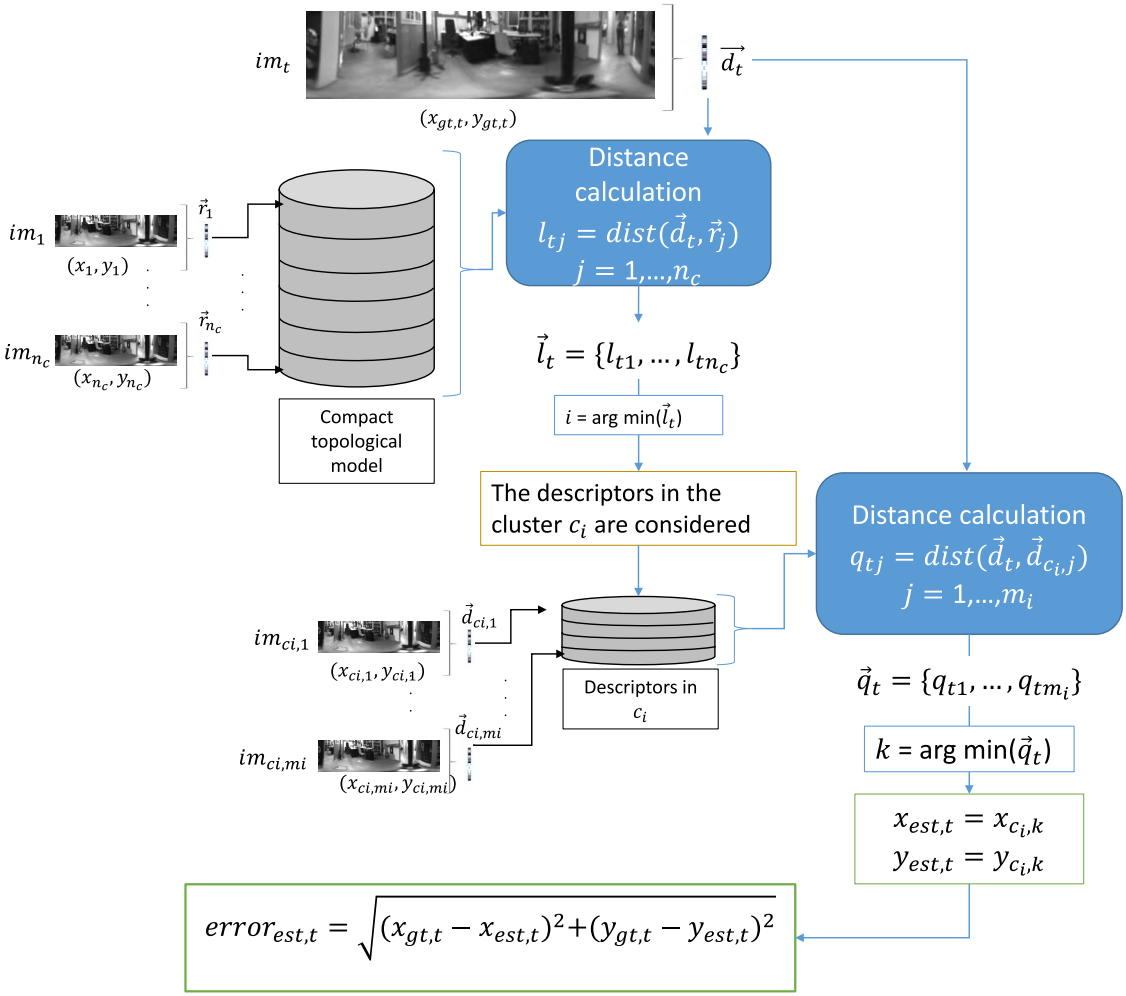



**FIGURE 8.** Block diagram regarding the steps to carry out the hierarchical localization task through compact models.

between descriptors, both corrrelation and cosine present similar outputs.

### B. HIERARCHICAL LOCALIZATION

In subsection IV-A, the localization has been solved using only the compact map (i.e., only the high level layer is used, and the result is a coarse localization). It has allowed us to analyze how different compression levels have an influence on the localization error (i.e. the tradeoff map granularity - localization accuracy).

In this subsection, we go one step beyond, and the localization is addressed hierarchically. First, a coarse localization is performed, as in subsection IV-A. Once the nearest cluster has been retrieved, a second step is carried out to refine the estimation.

Therefore, the hierarchical localization task consists of the following processes: first, the robot describes the image captured at time instant $t$ (test image) $im_t \rightarrow d_t$. After that, the distances vector is again obtained $l_t = \{l_{t1}, ..., l_{tn_c}\}$. Next, the most likely cluster is selected as the one which presents the minimum value of $l_t$. At this step, a new comparison is carried out between the descriptor of the test image $d_t$ and the descriptors of the images which belong to the chosen cluster. From this step, a new distances vector is obtained $q_t = \{q_{t1}, ..., q_{tm_i}\}$ where $m_i$ is the number of images within the selected cluster $i$. Finally, the minimum value of $q_t$ indicates the most similar image and hence, it corresponds to the current position of the robot with a higher accuracy. Fig. 8 shows the block diagram about these steps. It should be mentioned that more than one cluster may be selected. The higher

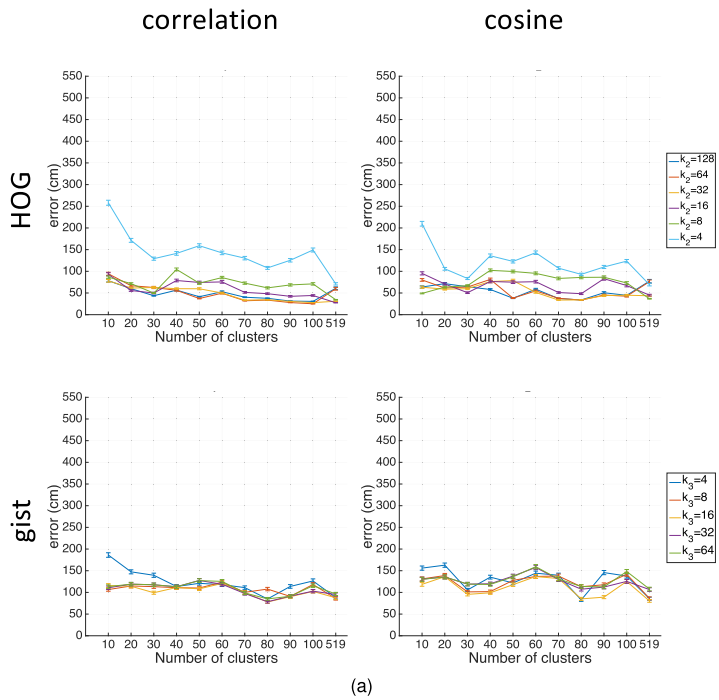



**FIGURE 9.** Results of the complete hierarchical localization task when the night condition of illumination is affecting the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Pre-selection of either (a) one ($c = 1$) or (b) two ($c = 2$) clusters as the most likely options.

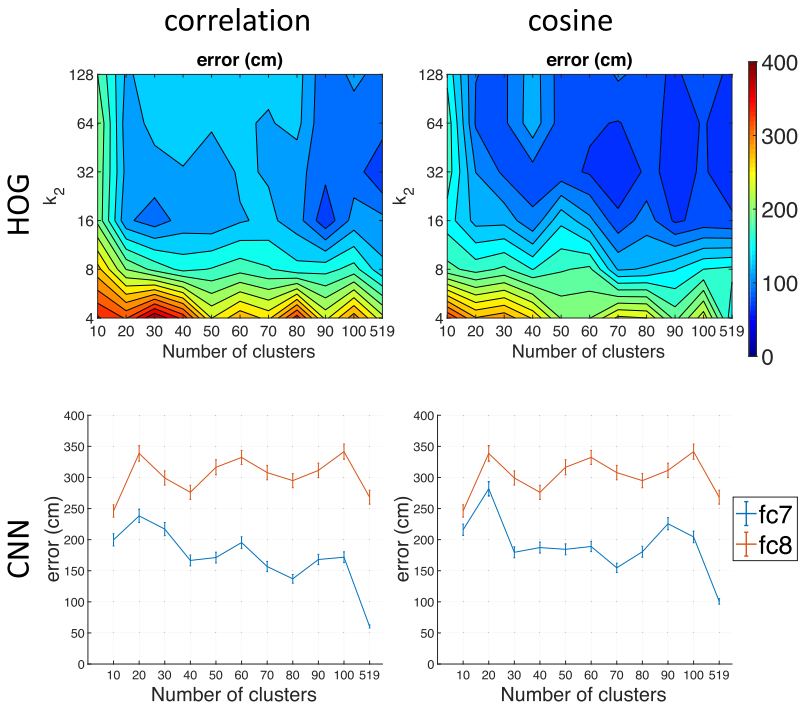



**FIGURE 10.** Results of the complete hierarchical localization task when the sunny condition of illumination is affecting the Freiburg environment. Average localization error (cm) vs. number of clusters and descriptor size. Pre-selection of one cluster as the most likely option.

the number of selected clusters, the more comparisons with images will be tackled.

### 1) EXPERIMENTS

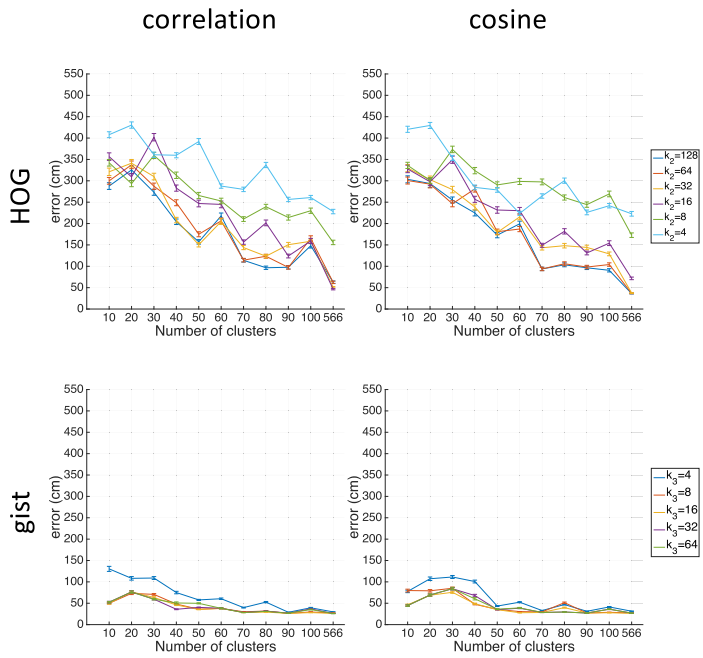
As in the sub-subsection IV-A.1, the experiments were carried out through the use of the COLD database with the same characteristics previously commented. Again, the starting point of the localization experiment is the compact model through *gist* ($k_3 = 32$ and $n_{masks} = 16$).

Since the Euclidean distance presented the worst localization error results in the experiment 1, this distance is discarded. Furthermore, neither FS nor *gist* descriptor related results are shown in this experiment because, as was shown in the previous subsection, those results are worse for localization purposes. Therefore, to sum up, the hierarchical localization is evaluated in the Freiburg and Saarbrücken environments under two illumination conditions (night and sunny) calculating two types of distances (correlation and cosine) and using two kind of descriptors (HOG and CNN).

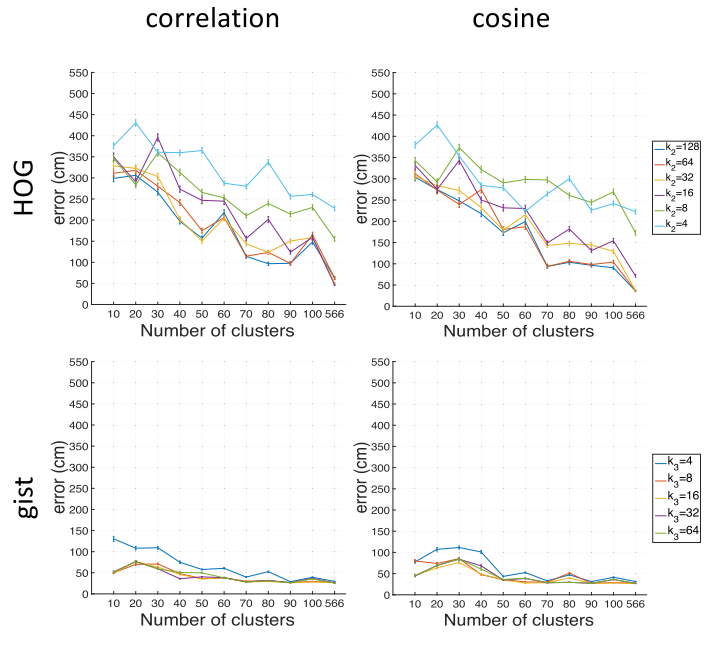
Fig. 9 shows the average localization error (cm) vs. the number of clusters $n_c$ obtained in the Freiburg environment when the test dataset was night and either one or two clusters are selected to carry out the fine localization. Fig. 10 shows the average localization error (cm) obtained in the Freiburg environment results when the test dataset was sunny and one cluster is selected for fine localization; fig. 11 shows the average localization error (cm) vs. the number of clusters $n_c$ obtained in the Saarbrücken environment when the test dataset was night and one and two clusters are selected to carry out the fine localization.

The evaluation of these results is carried out from three points of view. Firstly, a comparison of the results obtained through hierarchical localization against the localization tackled in the subsection IV-A.1 is performed. In general, the localization error obtained through hierarchical localization clearly improves when the number of clusters is low (see fig. 9, 10 and 11). However, when the number of clusters increases, no improvements are noticed. This behaviour is due to the fact that a low number of clusters implies a very rough initial localization. This way, the second step really permits refining this estimation. However if the number of clusters is relatively high, the initial estimation is quite fine and the improvement achieved in the second step is not substantial. This allows us to conclude that the high-level layer can be built with a high degree of compression and the localization can be refined with the low-level layer, through an efficient process.

Secondly, after proving that hierarchical localization presents improvements when a high compression is carried out, an analysis about how illumination conditions affect to




(a)




(b)

**FIGURE 11.** Results of the complete hierarchical localization task when the night condition of illumination is affecting the Saarbrücken environment. Average localization error(cm) vs. number of clusters and descriptor size. Pre-selection of either (a) one ($c = 1$) or (b) two ($c = 2$) clusters as the most likely options.

**TABLE 3. Summary of the minimum localization error values obtained through the two localization methods and the four descriptors evaluated throughout this work.**

| Localiz. Metthod | Descriptor | Minimum localization error (cm) | | | | |
|---|---|---|---|---|---|---|
| | | $n_c = 20$ | $n_c = 40$ | $n_c = 60$ | $n_c = 80$ | $n_c = 100$ |
| Localization through compact models | Fs | 403.25 | 331.88 | 350.88 | 351.81 | 267.60 |
| | HOG | 149.75 | 128.19 | 118.53 | 112.20 | 90.85 |
| | gist | 201.33 | 125.77 | 149.45 | 125.92 | 153.38 |
| | CNN | 133.54 | 71.39 | 76.23 | 78.27 | 66.51 |
| Hierarchical Localization | Fs | 360.73 | 308.40 | 327.13 | 328.32 | 252.17 |
| | HOG | 61.41 | 81.27 | 54.89 | 33.59 | 42.52 |
| | gist | 136.27 | 99.10 | 136.58 | 85.02 | 125.31 |
| | CNN | 69.52 | 38.68 | 50.57 | 49.01 | 50.73 |

hierarchical localization is tackled. Comparing the results obtained when a hierarchical localization process is developed under night conditions (see fig. 9, $c = 1$) and sunny conditions (see fig. 10); several conclusions can be extracted. For a localization task carried out through the use of HOG descriptor and correlation distance in Freiburg under night conditions, the average localization error is between less than 50 and 250 cm, whereas, under sunny conditions, this value is between 70 and 400 cm. This analysis can be extended to the CNN descriptor, which is again highly affected by the sunny conditions. Therefore, collecting results from both experiments, the conclusion is that the sunny condition affects to a greater extent the localization task.

Thirdly, an evaluation about varying the number of clusters selected to carry out the fine localization is done. If we compare the hierarchical localization results in Freiburg for one selected cluster and the results for two selected clusters (see fig. 9), a slight improvement is appreciated in the case $c = 2$ when the number of clusters is low. This behavior means that for few clusters, selecting the right one can be more challenging. Hence, selecting more than one cluster for fine localization may result beneficial when a huge compression was carried out. For the Saarbrücken environment, no improvements have been noticed between selecting one and selecting two clusters (see fig.11). This lack of improvement means that the instances are very well represented even when there is a high level of compression and thus, selecting more than one cluster does not provide a higher probability to find the more accurate position of the test image.

### C. DISCUSSION OF RESULTS

The scope of these experiments is to evaluate the robustness of global appearance descriptors to solve the localization problem using hierarchical maps either (1) by using a localization method which estimates the position through compact models, or (2) by solving also a fine localization step (hierarchical localization method). For the sake of completeness, the experimental section considers several methods to obtain the global appearance descriptors (FS, HOG, *gist* and CNN), different configuration parameters of these descriptors and also a variety of illumination conditions. Regarding the localization method through compact models, the average accuracy improves as less compression is tackled. As for the hierarchical localization, this method produces an efficient process to refine the localization in the low-level layer. Nevertheless, this method only improves when the number of clusters is low but no substantial differences exist when the number of clusters is relatively high. Selecting more than one cluster to carry out the fine localization is only interesting when a huge compression is carried out, otherwise, selecting only one cluster produces more efficient results because its computing time is relatively low.

Concerning the global appearance descriptors, FS always outputs the worst results and HOG usually leads to the best solutions. We have found out that the CNN-based descriptor also presents good results. Nevertheless, CNN is more affected by the sunny illumination conditions than HOG is and CNN also needs more computing time to calculate the descriptor than HOG. In general, the sunny illumination conditions affect more negatively the performance of the methods than the night conditions.

## V. CONCLUSION

In this work, a study is carried out about the utility to solve the localization task hierarchically in mobile robotics when substantial illumination changes are present. This task is tackled once a compact model of the environment is created. Two indoor sets of 519 and 566 panoramic images have been respectively used. A clustering approach through Spectral Clustering with a number of clusters between 10 and the total number of instances was considered. Therefore, a reduction between 1.77% and 17.67% of information contained in the initial set of images is considered. Additionally, we also analyze the localization when no compression is done, as a reference.

The work has shown that it is possible to keep a good localization error departing from a compact model. The issue is solved through the use of global appearance of panoramic scenes. A comparative evaluation between four methods to globally describe images has been carried out: FS, HOG, *gist* and a CNN-based descriptor. The CNN-based descriptor and cosine distance has been proved to be the best choice. The table 3 summarizes the localization error obtained along this work. Through this table, it is easy to conclude that the CNN-based descriptor provides the best results to carry out

the localization task for both localization methods although HOG also presents good results when the localization is addressed hierarchically.

This work has also shown the efficiency of this localization framework under severe changes of illumination. Moreover, it has proved that the test images under sunny conditions affect more negatively the results than the night conditions.

As for the use of hierarchical localization, it may result interesting for high levels of compression and just selecting one cluster as candidate may be enough for most cases.

Future works will include the study of other methods to compress the models and the study of other disadvantageous issues which may be presented in real operating conditions, such as occlusions, changes of furniture, etc.

## ACKNOWLEDGMENTS

The authors declare that there are no competing interests regarding the publication of this paper.

## REFERENCES

[1] D. Valiente, A. Gil, Ó. Reinoso, M. Juliá, and M. Holloway, "Improved omnidirectional odometry for a view-based mapping approach," *Sensors*, vol. 17, no. 2, p. 325, 2017.

[2] L. B. Marinho, J. S. Almeida, J. W. M. Souza, V. H. C. Albuquerque, and P. P. R. Filho, "A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images," *Expert Syst. Appl.*, vol. 72, pp. 1–17, Apr. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417416306790

[3] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, "Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle," *J. Field Robot.*, vol. 33, no. 4, pp. 431–450, 2016.

[4] Y. Berenguer, L. Payá, M. Ballesta, and O. Reinoso, "Position estimation and local mapping using omnidirectional images and global appearance descriptors," *Sensors*, vol. 15, no. 10, pp. 26368–26395, 2015. [Online]. Available: http://www.mdpi.com/1424-8220/15/10/26368

[5] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Visual topological SLAM and global localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2009, pp. 2029–2034.

[6] A. C. Murillo, J. J. Guerrero, and C. Sagues, "Surf features for efficient robot localization with omnidirectional images," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3901–3907.

[7] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization by means of local invariant features and map refinement," *Robotica*, vol. 33, no. 7, pp. 1446–1470, 2015.

[8] E. Menegatti, T. Maeda, and H. Ishiguro, "Image-based memory for robot navigation using properties of omnidirectional images," *Robot. Auton. Syst.*, vol. 47, no. 4, pp. 251–267, 2004.

[9] M. Liu, D. Scaramuzza, C. Pradalier, R. Siegwart, and Q. Chen, "Scene recognition with omnidirectional vision for topological map using lightweight adaptive descriptors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2009, pp. 116–121.

[10] H. Korrapati and Y. Mezouar, "Multi-resolution map building and loop closure with omnidirectional images," *Auto. Robots*, vol. 41, no. 4, pp. 967–987, Mar. 2016.

[11] L. Gerstmayr-Hillen, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller, "Dense topological maps and partial pose estimation for visual control of an autonomous cleaning robot," *Robot. Auton. Syst.*, vol. 61, no. 5, pp. 497–516, 2013.

[12] H. Korrapati and Y. Mezouar, "Vision-based sparse topological mapping," *Robot. Auton. Syst.*, vol. 62, no. 9, pp. 1259–1270, 2014.

[13] F. Amigoni *et al.*, "A standard for map data representation: IEEE 1873–2015 facilitates interoperability between robots," *IEEE Robot. Autom. Mag.*, vol. 25, no. 1, pp. 65–76, Mar. 2018.

[14] A. Štimec, M. Jogan, and A. Leonardis, "Unsupervised learning of a hierarchy of topological maps using omnidirectional images," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 22, no. 4, pp. 639–665, 2007. doi: 10.1142/S0218001408006430.

[15] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robot. Auton. Syst.*, vol. 64, pp. 1–20, Feb. 2015. doi: 10.1016/j.robot.2014.11.009.

[16] S. P. P. da Silva, R. V. M. da Nóbrega, A. G. Medeiros, L. B. Marinho, J. S. Almeida, and P. P. R. Filho, "Localization of mobile robots with topological maps and classification with reject option using convolutional neural networks in omnidirectional images," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[17] A. Rituerto, L. Puig, and J. J. Guerrero, "Visual SLAM with an omnidirectional camera," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 348–351.

[18] S. Cebollada, L. Payá, W. Mayol, and O. Reinoso, "Evaluation of clustering methods in compression of topological models and visual place recognition using global appearance descriptors," *Appl. Sci.*, vol. 9, no. 3, p. 377, 2019.

[19] R. González and R. Woods, *Digital Image Processing*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2008.

[20] L. Payá, W. Mayol, S. Cebollada, and O. Reinoso, "Compression of topological models and localization using the global appearance of visual information," in *Proc. ICRA*, 2017, pp. 5630–5637.

[21] L. Payá, O. Reinoso, Y. Berenguer, and D. Úbeda, "Using omnidirectional vision to create a model of the environment: A comparative evaluation of global-appearance descriptors," *J. Sensors*, vol. 2016, Feb. 2016, Art. no. 1209507. doi: 10.1155/2016/1209507.

[22] L. Payá, A. Peidró, F. Amorós, D. Valiente, and O. Reinoso, "Modeling environments hierarchically with omnidirectional imaging and global-appearance descriptors," *Remote Sens.*, vol. 10, no. 4, p. 522, 2018.

[23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, San Diego, CA, USA, vol. 2, Jun. 2005, pp. 886–893.

[24] A. Leonardis and H. Bischof, "Robust recognition using eigenimages," *Comput. Vis. Image Understand.*, vol. 78, no. 1, pp. 99–118, 2000.

[25] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Prog. Brain Res.*, vol. 155, pp. 23–36, 2006.

[26] A. Murillo, G. Singh, J. Košecká, and J. J. Guerrero, "Localization in urban environments using a panoramic gist descriptor," *IEEE Trans. Robot.*, vol. 29, no. 1, pp. 146–160, Feb. 2013.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[28] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 487–495.

[29] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. *Places-CNN Model From MIT*. Accessed: Feb. 28, 2019 [Online]. Available: https://github.com/BVLC/caffe/wiki/Model-Zoo#places-cnn-model-from-mit

[30] M. Mancini, S. R. Bulò, E. Ricci, and B. Caputo, "Learning deep NBNN representations for robust place categorization," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1794–1801, Jul. 2017.

[31] Y. Xu, M. Sun, Z. Cao, J. Liang, and T. Li, "Multi-object tracking for mobile navigation in outdoor with embedded tracker," in *Proc. 7th Int. Conf. Natural Comput. (ICNC)*, vol. 3, 2011, pp. 1739–1743.

[32] S. Theodoridis and K. Koutroumbas, "Pattern recognition and neural networks," in *Proc. Mach. Learn. Appl.*, 2001, pp. 169–195.

[33] U. von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

[34] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001, pp. 849–856.

[35] A. Pronobis and B. Caputo, "COLD: COSY localization database," *Int. J. Robot. Res.*, vol. 28, no. 5, pp. 588–594, May 2009. [Online]. Available: http://www.pronobis.pro/publications/pronobis2009ijrr

**SERGIO CEBOLLADA** received the M.Eng. degree in telecommunication engineering from Miguel Hernández University, in 2014, where he is currently pursuing Ph.D. degree. His research interests include omnidirectional vision and global appearance algorithms, map building and localization of mobile robots, and deep learning. He received the Ph.D. Candidate Scholarship Supported by Valencian Government (ACIF/2017/146), in 2017.

**LUIS PAYÁ** received the M.Eng. degree in industrial engineering in Spain, in 2002, and the Ph.D. degree in industrial technologies in Spain, in 2014. He is currently an Associate Professor with the Department of Systems Engineering and Automation, Miguel Hernández University, Spain. He teaches some subjects related to the fields of automatic control, electronics, and robotics. He has authored several books, papers, and communications in the cited topics. His current research interests include omnidirectional vision and global appearance algorithms, topological map building and localization of mobile robots, and also implementation and testing of remote laboratories.

**VICENTE ROMÁN** received the Degree in electronics and automation engineering, in 2016 and the M.Sc. degree in robotics from Miguel Hernandez University, Elche, Spain, in 2017, where he is currently pursuing the Ph.D. degree. He teaches some subjects related to control, robotics, and computer vision with the Department of Systems Engineering and Automation, UMH. His current research interests include mobile robotics, omnidirectional vision, and global appearance algorithms.

**OSCAR REINOSO** received the Degree in industrial engineering and the Ph.D. degree from the Polytechnic University of Madrid, in 1991 and 1996, respectively. From 1994 to 1997, he was with the Research and Development Department, Protos Desarrollo in a visual inspection system. Since 1997, he has been with Miguel Hernández University, as a Professor in control, robotics, and computer vision. He has authored several books, papers, and communications in the cited topics. His research interests include robotics, teleoperated robots, climbing robots, visual serving, and visual inspection systems. He is a member of the CEA-IFAC.

ARTICLE IN PRESS

Expert Systems With Applications xxx (xxxx) xxx

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Review

# A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data☆

Sergio Cebollada *, Luis Payá, María Flores, Adrián Peidró, Oscar Reinoso

*Department of Systems Engineering and Automation, Miguel Hernández University, Elche, 03202, Spain*

ARTICLE INFO

*Keywords:*
Mobile robotics
Visual information
Artificial intelligence
Mapping
Localization
Navigation
SLAM
Exploration

ABSTRACT

Nowadays, the field of mobile robotics has experienced an important evolution and these robots are more commonly proposed to solve different tasks autonomously. The use of visual sensors has played an important role in mobile robotics tasks during the past few years due to the advances in computer vision hardware and algorithms. It is worth remarking the use of AI tools to solve a variety of problems in mobile robotics based on the use of images either as the only source of information or combining them with other sensors such as laser or GPS. The improvement of the autonomy of mobile robots has attracted the attention of the scientific community. A considerable amount of works have been proposed over the past few years, leading to an extensive variety of approaches. Building a robust model of the environment (mapping), estimating the position within the model (localization) and controlling the movement of the robot from one place to another (navigation) are important abilities that any mobile robot must have. Considering this, this review focuses on analyzing these problems; how researchers have addressed them by means of AI tools and visual information; and how these approaches have evolved in recent years. This topic is currently open and a large number of works can be found in the related literature. Therefore, it can be of interest making an analysis of the current state of the topic. From this review, we can conclude that AI has provided robust solutions to some specific tasks in mobile robotics, such as information retrieval from scenes, mapping, localization and exploration. However, it is worth continuing to develop this line of research to find more integral solutions to the navigation problem so that mobile robots can increase their autonomy in large, complex and heterogeneous environments.

## 1. Introduction

Over the past few years, the use of mobile robots has significantly increased. Nowadays, they can be used for a wide range of applications and they can be found in diverse kinds of environments, such as industrial, household, educational and healthcare. Regarding mobile autonomous robots, they must be able to navigate through an environment which is usually *a priori* unknown, while simultaneously tackle the task they have been designed for. Hence, the robot must be capable of building a model of the environment, estimating its current position and orientation within the environment by using this model and also navigating throughout the environment to arrive to the target points.

Mapping, localization and navigation are the classical problems in mobile robotics. They have attracted a great attention and nowadays continue being a prominent research area, since a robust solution to these problems is fundamental to increase the autonomy of mobile robots and subsequently expand their use for other applications.

To conduct the mobile robotics tasks, it is necessary to provide the robot with relevant information about the environment. For this purpose, robots are equipped with sensors that allow them to obtain such information. Subsequently, the robots need to process the data captured from the environment and transform them in useful information for their tasks.

Concerning the mapping task, two main frameworks can be highlighted from the related literature: metric and topological maps. On the one hand, the metric maps represent the environment with geometric accuracy. On the other hand, the topological maps lead typically to a graph representation, that is, the environment is described as a graph that contain representative locations and links that connect them. As

☆ This work has been supported by the Generalitat Valenciana, Spain and the FSE, European Union, through the grant ACIF/2017/146 and the project AICO/2019/031: "Creación de modelos jerárquicos y localización robusta de robots móviles en entornos sociales"; and by the Spanish government through the project DPI 2016-78361-R (AEI/FEDER, UE): "Creación de mapas mediante métodos de apariencia visual para la navegación de robots.".

* Corresponding author.
*E-mail addresses:* sergio.cebollada@umh.es (S. Cebollada), lpaya@umh.es (L. Payá), maria.flores03@umh.es (M. Flores), apeidro@umh.es (A. Peidró), o.reinoso@umh.es (O. Reinoso).

https://doi.org/10.1016/j.eswa.2020.114195
Received 5 June 2020; Received in revised form 10 August 2020; Accepted 29 October 2020
Available online 2 November 2020

for the localization task, it tries to estimate the current position and orientation of the robot using a model of the environment. In order to carry out the localization task, the environment must be previously modeled. Hence, in this way, firstly the robot carries out the mapping task and after that, once the map is available, the localization can be done. Nonetheless, the related literature has also studied a blend of both tasks that can be developed at the same time. This concept is known as Simultaneous Localization And Mapping (SLAM) and consists in modeling the environment as the robot moves through it and, at the same time, estimating its position and orientation.

Additionally to the mapping and localization tasks, there are other tasks that are included within the mobile robotics challenges. The navigation task includes the ability of the robot to determine its position within the map, to plan a path to reach a target position and to send the necessary commands to the actuators to move the robot while avoiding dynamic obstacles. Hence, the robot must be capable of mapping the environment and interpreting the agents included in it. The main objectives of the robot navigation consist in avoiding collisions such as objects, walls, human beings, etc; and avoiding unsafe places or conditions such as radioactive places, exposing to hazardous places due to high temperatures or other environmentally dangerous conditions. Reinoso and Payá (2020a) present a special issue about the current frameworks in the mobile robots navigation field and a variety of approaches related to this task.

Similar to the concept of SLAM, the mobile robotics field also considers the combination of mapping and navigation. This combination is known as exploration. The exploration task consists basically in guiding a robot in such a way that it covers the environment with its sensors (Stachniss & Burgard, 2003). Exploration approaches are relevant to address surveillance or surface inspection mine sweeping, among others.

Concerning the sensors provided to address the mobile robotics tasks, this work focuses on the use of cameras. This type of sensors have been widely used for these purposes. In this way, Reinoso and Payá (2020b) present a special issue about some of the possibilities that vision systems offer, focusing on the different configurations that can be used and novel applications in fields of application, from mapping for navigation of mobile robots to object recognition or scene reconstruction. Through this kind of sensors, the amount of information collected may be enough to carry out most of the problems related to mobile robotics. Moreover, these sensors present a relatively good relation "quantity of information - cost". However, these approaches present downsides, such as their sensitivity to changes of lighting conditions. For example, underfloor environments may be completely dark and the illumination is only provided by light sources installed either on the robot and/or in a specific position of the environment. Hence, the shadows may generate inaccuracies (Cebollada, Payá, Juliá, Holloway, & Reinoso, 2018; Parra, Cebollada, Payá, Holloway, & Reinoso, 2020).

According to the number of cameras and the field of view, different configurations have been proposed. Some authors such as Okuyama, Kawasaki, and Kroumov (2011) have used monocular configurations. Others proposed stereo cameras by using binocular (such as Yongguo, Wei, and Guang-liang (2012) or Gwinner et al. (2016)) or even trinocular systems such as Jia, Li, An, and Zhang (2003). In order to obtain complete information from the environment, several images must be captured. In this respect, omnidirectional cameras constitute a good alternative. They can provide a big amount of information with a field of view of 360 deg. around them and their cost is relatively low in comparison with other kinds of sensors. Furthermore, omnidirectional vision systems present further advantages. For instance, the features in the images are more stable (because they stay longer as the robot moves) and they permit estimating both the position and the orientation of the robot. Omnidirectional cameras have been successfully used by different authors for mapping and localization (Menegatti, Pretto, Scarpa, & Pagello, 2006; Murillo, Guerrero, & Sagues, 2007; Payá, Peidró, Amorós, Valiente, & Reinoso, 2018; Tardif, Pavlidis, & Daniilidis, 2008; Valiente, Payá, Jiménez, Sebastián, & Reinoso, 2018). A wide study was carried out by Payá, Gil, and Reinoso (2017), who introduce a state of the art of the most relevant mapping and localization algorithms developed with omnidirectional visual information.

With this aim, a wide range of approaches have emerged to process the information obtained by the sensory systems. As a result, a variety of methods have been proposed regarding sensory information and processing techniques. Additionally, in recent years, several tools and techniques based on artificial intelligence (AI) have proved their ability to solve a variety of problems with a profound data treatment. The use of these techniques has become very popular among the mobile robotics works and it is worth studying their main applications to solve mapping and localization. Within the AI field, Machine Learning (ML) is a subfield whose algorithms attempt to improve automatically through experience (Mitchell, 1997). More precisely, these algorithms build a mathematical model based on sample data with the aim of carrying out predictions or decisions without being explicitly programmed for a specific purpose (Bishop, 2006). During the past few years, ML has received considerable attention, since this technique is capable of addressing a wide variety of complex problems accurately. There are many machine learning algorithms and they depend on the approach they use, the data type of the input, the data type of the output, and the type of problem that they are designed to solve. On the other hand, supervised learning algorithms consist on building a mathematical model to represent a set of data which contains the inputs and also the desired outputs. By addressing iterative optimization of an objective function, these algorithms learn a function that can be used to predict outputs associated with new inputs (Mehryar, Rostamizadeh, & Talwalkar, 2012). On the other hand, unsupervised learning algorithms take a set of data that contains only inputs. These algorithms try to find a structure in the training data such as grouping or clustering of data points (Hinton, Sejnowski, & Poggio, 1999).

Currently, deep learning has become the dominant approach for many works in the field of machine learning. Like ML, deep learning algorithms build mathematical models based on sample data to carry out predictions/decisions without being explicitly programmed for a certain purpose using typically an architecture of multiple hidden layers in an artificial neural network. These algorithms try to construct automatically high level data models by using a matrix of initial data and architectures that allow linear, non-linear, multiple and iterative transformations (Bengio, Courville, & Vincent, 2013). These architectures have been commonly applied during the past few years to fields like computer vision, speech recognition, natural language processing, social network filtering, machine translation, and bioinformatics among others. Its use has proved to provide results comparable to and in some cases surpassing human expert performance (Goodfellow, Bengio, & Courville, 2016).

The fields of mobile robotics and computer vision have progressed considerably during the past few years. Notwithstanding that, there are still some issues that need to be addressed more robustly to enable mobile robots to move and perform their tasks more autonomously in complex environments, under real operation conditions (Payá et al., 2017). Additionally, the continuous improvement in the capacity and performance of computing devices has extended the use of different AI methods. Therefore, it is worth knowing the contribution of AI to mobile robotics and computer vision, and the research gaps, opportunities and solutions which are contributing to the development of these fields.

Some researchers have presented reviews in the field of mobile robotics. Parker (2000) develops a survey about the methods used to solve tasks in distributed mobile robotics. Muhammad, Fofi, and Ainouz (2009) introduce a review about SLAM methods that make use of visual information. Garcia-Fidalgo and Ortiz (2015) focus on approaches to solve the topological mapping problem. Also, Fuentes-Pacheco, Ruiz-Ascencio, and Rendón-Mancha (2015) and Kuutti et al. (2018) present a state of the art of the localization techniques for autonomous vehicle applications. Finally, Payá et al. (2017) develop a review of mapping

and localization techniques using omnidirectional vision. With respect to these previous reviews, and considering the great development of AI techniques and their use in mobile robotics, in the present work we focus on the use of such techniques. We present the most relevant AI tools in mobile robotics, how they can be used to extract relevant information from the scenes and their application to solve specific problems in mobile robotics.

To summarize, the aim of this review is to present the most relevant works conducted in the field of mapping, localization, navigation, SLAM and exploration by using AI, paying more attention to the developments that are based on visual information. The remainder of the paper is structured as follows. First, Section 2 presents the main AI tools used in the robotics field. Then, Section 3 presents the main methods to describe the information. After that, Section 4 shows the AI techniques that have been proposed to solve the mobile robotics tasks with visual data. Last, Section 5 presents a final discussion about the existing approaches.

## 2. Artificial intelligence tools

This section depicts the concept of AI and the areas where this science has been commonly applied in the field of robotics during the last few years. Also, the most relevant techniques are outlined. Section 2.1 defines some areas of use in the field of robotics; Section 2.2 presents a variety of applications that contribute to the autonomy of mobile robots and Section 2.3 focuses on the problems of mapping and localization and the AI tools used to address these problems.

### 2.1. Definition and areas of use

In the related literature, some definitions of AI can be found. For instance, Schleichert (1970) defined AI as "the science of making machines do things that would require intelligence if done by men". Charniak, McDermott, and McDermott (1985) define AI as "the study of mental faculties through the use of computational models". More recently, according to Schalkoff (1990), AI is "a field of study that seeks to explain and emulate intelligent behavior in terms of computational process". If we focus on the use of AI to solve robotics tasks, we can describe this science as a set of techniques that are applied in computer programming to solve problems whose difficulty requires a certain degree of intelligence. As for the birth of the AI, many researchers consider that this happened during the Second World War, when the scientist Alan Turing worked to crack the 'Enigma' code that was used by German forces to send messages securely. Alan Turing and his team created the Bombe machine, which was used to decipher Enigma's messages. Both Enigma and Bombe Machines are considered the foundations for Artificial Intelligence (Ray, 2018).

Artificial Intelligence has been widely used in different areas. According to the type of manipulation, we can establish two categories. First, the **physical manipulation**, which covers the fields of computer vision, robotics and control systems. For example, Vyborny and Giger (1994) have used successfully computer vision together with AI in mammography to detect or to characterize abnormalities on digital images. Thanks to it, Radiologists are able to detect anomalies better and then, the errors in mammography interpretation are considerably reduced. Another example of computer vision and AI is proposed by Wachs, Kölsch, Stern, and Edan (2011), who propose a vision based hand gesture recognition for human computer interaction based on an artificial neural network, fuzzy logic, and genetic algorithms. Regarding the use of Artificial Intelligence for Robotics, Singh and Parhi (2011) propose a neural network to solve the path and time optimization problem of mobile robots. The inputs to the proposed neural controller consist of distances to the obstacles with respect to the position of the robot and target angle. The output of the neural network is the steering angle. De Momi and Ferrigno (2010) propose a backpropagation algorithm in the healthcare field that is used to train the network. The goal is to assist surgeons with a robotic system controlled by an intelligent high-level controller (HLC) able to gather and integrate information from the surgeon, from diagnostic images, and from an array of on-field sensors. Last, regarding control systems, Wong, Tam, Li, and Vong (2010) propose a novel modeling and optimization approach for steady state and transient performance tune-up of an engine at idle speed. In terms of electric control, a genetic algorithm and particle swarm optimization are applied to obtain an optimal control unit setting automatically. Gadoue, Giaouris, and Finch (2009) present a comparison between four different speed controller design strategies based on AI techniques; two are based on tuning of conventional PI (Proportional–Integral) controllers, the third makes use of a fuzzy logic controller and the last is based on hybrid fuzzy sliding mode control theory.

Regarding the **thinking manipulation**, this branch covers fields such as Natural Language Processing, Data Mining, Neural Networks, Automatic Learning, and Pattern Recognition. There are also a substantial amount of works related to them. To cite some examples, Kohavi and Quinlan (2002) present data mining tasks by using decision-tree discovery for classification. Xing, Xie, and Yang (2015) propose a learning-based framework for robust and automatic nucleus segmentation with shape preservation. Krittanawong, Zhang, Wang, Aydar, and Kitai (2017) review the recent AI applications in cardiovascular clinical care and discuss its potential role in facilitating precision cardiovascular medicine. Calderon-Cordova, Ramírez, Barros, Quezada-Sarmiento, and Barba-Guamán (2016) conduct the design and development of the system architecture to recognition of Electromyography signal patterns by using Feedforward backpropagation Artificial Neural Network. Minaei-Bidgoli, Kashy, Kortemeyer, and Punch (2003) introduce an approach to classify students in order to predict their final grade based on features extracted from logged big data in an education Web-based system and they also propose the use of a genetic algorithm to learn an appropriate weighting of the features.

### 2.2. Applications of AI

In Section 2.1, some definitions were introduced as well as some examples of use in a variety of fields. This subsection focuses on the main applications of AI which are closely related to mobile robotics and that have been developed during the past few years.

#### 2.2.1. Self-driving navigation

Self-driving navigation means that a vehicle is able to plan its path and execute its plan without human intervention. This task is carried out through the use of data captured from sensors aboard the vehicle and sometimes through the use of remote navigation aids (Michelson, 2000). This application is quite common in vehicles such as cars and lorries but also for other kinds of ground, underwater or aerial robots. The interest in such applications has increased in recent years due to the desire to develop a full self-driving vehicle with the main aim of reducing the traffic accidents provoked by human beings. These systems basically consist of a mobile platform that integrates a set of sensors. The data collected by the sensors provide the perception of the environment. This information can be processed through AI algorithms with the aim of tackling the path-planning task to move through the environment with minimal human intervention. The autonomous navigation also takes into consideration tasks such as move-on-route, obstacle detection and avoidance and leader/follower capabilities.

Regarding the use of AI for this application, a considerable number of works have been developed in recent years. Li et al. (2017) introduce a fully autonomous navigation system for a smart microvehicle with a microscope-coupled CCD (Charge-Coupled Device) camera, an AI planner, and a magnetic field generator. The AI planner is split into three functional modules: a computer vision module for tracking the microvehicle and detecting obstacles in its environment; a motion planner to generate an optimal obstacle-free path between starting point and destination; and a magnetic motion controller to manipulate

the microvehicle movement along a predesigned path. Polvara, Sharma, Wan, Manning, and Sutton (2018) propose collision detection and path planning methods for autonomous Unmanned Surface Vehicles by using artificial neural networks and evolutionary algorithms. Sharma, Liu, Wang, and Zhang (2017) apply AI to secure wireless communications of Connected Vehicles, which facilitates exchange of safety messages for collision avoidance in self-driving cars. The AI system learns to augment its ability to discern and recognize its surroundings. Badue et al. (2019) carry out a survey about the state-of-the-art on self-driving cars focusing on works published since the birth of the DARPA (Defense Advanced Research Projects Agency) challenges. This survey focuses on the perception systems and the decision-making systems based on methods that make use of AI.

#### 2.2.2. Face detection and recognition

**Face Detection** is the preliminary step for face recognition, and it consists basically in detecting faces in the images. These tasks have played an important role in robotics concerning problems such as surveillance (Ahuja, Krishnan, Kiran, Dalin, & Sagar, 2018) and home service robots (Jiang & Wang, 2017). According to Yang, Kriegman, and Ahuja (2002), the solution to the face detection problem can be divided in two steps:

1. Finding out whether there is any face in a given image or not.
2. If there is any face within the image, then, calculate where it is located.

In the related literature, many works can be found in this field. Romdhani, Torr, Scholkopf, and Blake (2001) propose a face detector that is based on running an observation window at all possible positions and using a Support Vector Machine (SVM) to determine whether a face is contained within the window. Ahuja et al. (2018) propose Local Binary Patterns (LBP) to detect the ROI (Region Of Interest) of the face inside the image and Haar feature-based cascade classifiers for developing the face recognition. Nevertheless, the revolution in this task arrives when Viola and Jones (2004) introduced a real-time face detector, which is able to detect faces in real-time with high accuracy. This work is based basically on three contributions. The first is the introduction of a new image representation that allows quick computations. The second contribution is a simple and efficient classifier built through the AdaBoost learning algorithm to select a small number of critical visual features from a very large set of potential features (Freund & Schapire, 1995). The third contribution is a method that combines classifiers in cascade, allowing quickly background regions rejection and spending more computation on promising face-like regions.

Face detection frameworks are commonly proposed to identify multiple appearances in smartphone cameras like Hadid, Heikkila, Silvén, and Pietikainen (2007) explain. Nowadays, face detection is commonly used in any kind of storing system or social networks. For instance, Facebook, Google, etc. are using face detection in the images uploaded in social networks (Rabbath, Sandhaus, & Boll, 2012).

Second, face recognition is defined by Jafri and Arabnia (2009) as a system to verify the identity of a person among a set of identities by using as input a face image and a database of face images of known individuals . This task has attracted an enormous interest in automatic processing of digital images in order to solve a variety of applications such as biometric authentication or surveillance (Jain & Li, 2011). Face recognition has been proposed during the past few years as an identification system in the same way that fingerprint and iris were proposed before. According to Abate, Nappi, Riccio, and Sabatino (2007), face recognition systems fall into two categories: verification and identification. As for face verification, it is a one-to-one match that compares an image of a face, whose identity has to be recovered, against a template face. On the other hand, concerning face identification, it is a one-to-many problem that compares a candidate face image against all the image templates that are contained in a face database with the objective of determining the identity of the candidate face.

The face recognition task has been used in a high number of applications. For example, Kim (2005) proposes a security system that carries out automatic recognition for verification between the picture of the passport and the face of the individual; this work proposes a clustering algorithm that creates adaptive clusters to the variations of input patterns and it is applied to the extracted areas for the recognition. Regarding surveillance, CCTVs (Closed-circuit television) can be used to look for someone. Wang, Bao, Ding, and Zhu (2017) use face recognition in real-world surveillance. They propose a convolutional neural network which is trained with a labeled dataset and subsequently proposed to recognize individuals from the campus surveillance system.

#### 2.2.3. Objects recognition and categorization

These tasks have played an important role in robotics concerning building object-based representations of the environment and manipulation of objects. Object recognition basically consists in detecting an object instance and object categorization consists in classifying a specific object (such as a cup of tea) (Loncomilla, Ruiz-del Solar, & Martínez, 2016). For instance, Gao et al. (2018) propose an object classification method using RGB-D data to train a Convolutional Neural Network (CNN) with the objective of detecting and categorizing usual objects in an autonomous vehicle environment such as other cars, cyclists, pedestrians and trucks. Zhu et al. (2016) introduce a CNN to detect and classify traffic signs. Furthermore, there are several works that use this application to additionally carry out a pose estimation of the objects detected. Kanezaki, Matsushita, and Nishida (2018) use a CNN to categorize objects from multi-view images and estimating their position. Wei, Xie, Wu, and Shen (2018) developed an end-to-end Mask-CNN model that selects deep convolutional descriptors for fine-grained object recognition. Zaki, Shafait, and Mian (2019) propose a multi-scale feature representation based on a convolutional hypercube pyramid (HP-CNN) that is able to carry out viewpoint invariant semantic object and scene categorization.

#### 2.2.4. Objects manipulation

A manipulation planning or object manipulation is a task related to the motion planning, but the focus is not on the movement of the robot, but on the objects to be manipulated. This task consists basically in changing the position and/or the orientation of a specific object (or set of objects), while avoiding collisions or breaking the object/s (Jiménez, 2012). The interest for this application has increased substantially over the past few decades with the aim of replacing human workers in challenging (due to the required accuracy) or hazardous tasks, specially in industrial, health care and domestic environments (Smith et al., 2012).

In the bibliography, many works about manipulations and planning can be found which are sustained by AI tools. For instance, Boularias, Bagnell, and Stentz (2015) introduce a robot system for grasping objects in dense clutter by using depth images. For this purpose, the robot learns to manipulate the objects by trial and error through a decision-making problem based on a reinforcement learning framework. Yang, Li, Fermuller, and Aloimonos (2015) propose a system that learns manipulation action by processing videos from the internet by using two CNNs, one for classifying the hand grasp type and the other for object recognition. Matas, James, and Davison (2018) present a combination of state-of-the-art deep reinforcement learning algorithms to solve the problem of manipulating deformable objects.

### 2.3. Frameworks commonly proposed for mapping and localization

After presenting some definitions and the main applications of AI in the robotics field, the present subsection introduces some of the most popular AI tools used to address mapping and localization in mobile robotics.

#### 2.3.1. Machine learning classifiers

Classification is a task that predicts the class or category which an 'object' belongs to. The object is also known as pattern and it is assumed to pertain to a unique class among a set of categories. Each pattern is represented by a set of measurements known as features, that must provide enough class-discriminatory information to predict the category of the pattern with high probability (Theodoridis, 2015). Usually, $n$ feature variables, $x_1, \ldots, x_n$, are selected and arranged in a feature vector, $x \in \mathbb{R}^n$. The objective is to train a classifier whose function (or set of functions) $f(x)$ in $\mathbb{R}^n$ is able to predict the class which the pattern belongs to.

This technique has been widely used to solve a range of problems. For example, Atkinson and Campos (2016) propose a feature-based emotion recognition model by using a multi-class SVM with EEG-based Brain–Computer interfaces. Narudin, Feizollah, Anuar, and Gani (2016) use different machine learning classifiers to detect malware in mobile phones using the anomaly-based approach. Concerning the computer vision field, there are many works that use classifiers. For instance, Korytkowski, Rutkowski, and Scherer (2016) introduce a fuzzy classifier with local image features to carry out objects classification. Zhang, Huang, Gong, Li, Zhao, Liu, et al. (2015) propose an automatic defective apple detection method by using a weighted relevance vector machine (RVM) classifier. Aguilar et al. (2017) propose a pedestrian detector for UAVs (Unmanned Aerial Vehicles) based on a combination of Haar-LBP features with Adaboost and using cascade classifiers with Meanshift.

#### 2.3.2. Clustering

Classifiers, as described in the previous subsection, is a supervised technique, that is, it needs correctly labeled data to carry out the training process. In this case, clustering is an unsupervised technique, where class labeling of the training patterns is not available. Hence, the main objective consists in finding out the organization of patterns into clusters (groups). To organize specific data into clusters, a clustering criterion or several clustering criteria must be established. Then, each pattern is categorized in a group and each cluster is characterized by the common attributes of the data that belong to it (Theodoridis & Koutroumbas, 1999).

This AI tool has been commonly proposed in a wide range of problems related to robotics and computer vision. For example, Dhanachandra, Manglem, and Chanu (2015) propose an image segmentation using k-means clustering. Schroff, Kalenichenko, and Philbin (2015) propose a clustering and face recognition approach based on a system that directly learns a measure of face similarity. Fan, Zheng, Yan, and Yang (2018) propose a progressive unsupervised learning method based on pedestrian clustering and fine-tuning of a CNN to transfer pre-trained deep representations to unseen domains. Wang, Pelillo, and Siddiqi (2019) propose an improvement of 3D object recognition by introducing a view clustering and pooling layer based on dominant sets.

#### 2.3.3. Deep feedforward networks

Deep feedforward networks, also known as feedforward neural networks or multilayer perceptrons (MLPs), are deep learning models whose objective is to approximate some function $f^*$. This network defines a mapping $y = f(x;\phi)$ where $x$ and $y$ are the input and output (or target) data respectively. This network learns the value of the parameters $\phi$ that best approximate the function $f$ (Goodfellow et al., 2016). These models carry out a flow through the function $f$ evaluating from $x$ to the output $y$. Nevertheless, these models do not provide feedback connections, that is, outputs of the model are not fed back into the model itself. During training, the aim is to drive $f(x)$ to match $f^*(x)$: the training data are provided and $f^*(x)$ is evaluated with those data. Moreover, a label $y$ is included with each example $x$ to achieve $y \approx f^*(x)$.

These networks present an extreme importance in machine learning, since they are the basis of many applications. For example, many object recognition approaches are based on this kind of models, such as the work by Mostajabi, Yadollahpour, and Shakhnarovich (2015), who propose a feed-forward architecture for semantic segmentation to tackle a rich feature representation that is used for object recognition.

#### 2.3.4. Autoencoders

An autoencoder is a neural network architecture composed basically of an encoder and a decoder system whose aim is to find a compressed representation of the given input data. The process consists in finding a representation or code to carry out useful transformations on the input data. Traditionally, autoencoders were proposed for dimensionality reduction or even for feature learning (Goodfellow et al., 2016). Denoising autoencoders try to find a code that can convert noisy data into clean ones. Moreover, autoencoders are also used to perform colorization, feature-level arithmetic, detection, tracking, and segmentation among others. As shown in Fig. 1, regarding the encoder, it transforms the input data $x$ into a low-dimensional latent representation $h = f(x)$. This latent representation is a vector of lower dimension. The encoder learns to extract the most important features of the input data. As for the decoder, it recovers the input data from the latent representation, $r = g(h)$ with the objective that $g(f(x)) = r$, being $r$ as close as possible to $x$. In general the encoder and decoder are non-linear functions and the dimension of the latent representation $h$ is considerably smaller than the input dimensions. Similar to other neural networks, the autoencoder tries to minimize a loss function during the training process. The loss function is established to measure how dissimilar the input $x$ and the reconstructed input $r$ are. For example, the Mean Squared Error (MSE) can be used for this purpose.

$$L(x, r) = MSE = \frac{1}{m}\sum_{i=1}^{i=m}(x_i - r_i) \tag{1}$$

where $m$ the number of components of the input and the output ($m = width \times height \times channels$).

Autoencoders have been a successful tool for dimensionality reduction and also for information retrieval. Regarding dimensionality reduction, this tool has provided reconstructions with an error rate lower than using other techniques such as PCA (Principal Components Analysis) (Hinton and Salakhutdinov, 2006). Therefore, through the improvement of lower-dimensional representations, other related tasks have also been improved. First, in classification tasks, autoencoders provide a model with less memory requirements and computing time consumption (Ma, Wang, & Geng, 2016). Second, by means of dimensionality reduction, information retrieval can be carried out more efficiently. With the use of autoencoders and its related dimensionality reduction, exhaustive searching becomes more efficient. For example, Pfeiffer, Broscheit, Gemulla, and Göschl (2018) present a study about learning-to-rank and query refinement approaches for information retrieval in the pharmacogenomic domain. Zhu et al. (2016) propose using an autoencoder for feature learning from 2D images with the aim of carrying out 3D shape retrieval. Moreover, autoencoders have been widely proposed to produce codifications that are low-dimensional and binary. In this way, entries of a database can be stored in a hash table and information retrieval can be carried out by returning all the entries that have the same binary code as the query. To cite one example of this approach, Carreira, Calado, Cardeira, and Oliveira (2015) introduce a fast search in image databases with binary hashing, where each high-dimensional, real-valued image is mapped with an autoencoder onto a low-dimensional, binary vector and the search is done in this binary space. Apart from these examples, many others can be found in the related literature concerning the use of autoencoders for mobile robotics. Sergeant, Sünderhauf, Milford, and Upcroft (2015) address a navigation task by using a deep autoencoder which learns how to navigate from the sensory data stored in a dataset. Wang, Yang, Huang, Lin, and Tang (2018) propose an autoencoder for fusion and extraction of multiple visual features from different sensors with the aim of carrying out motion planning based on deep reinforcement learning.

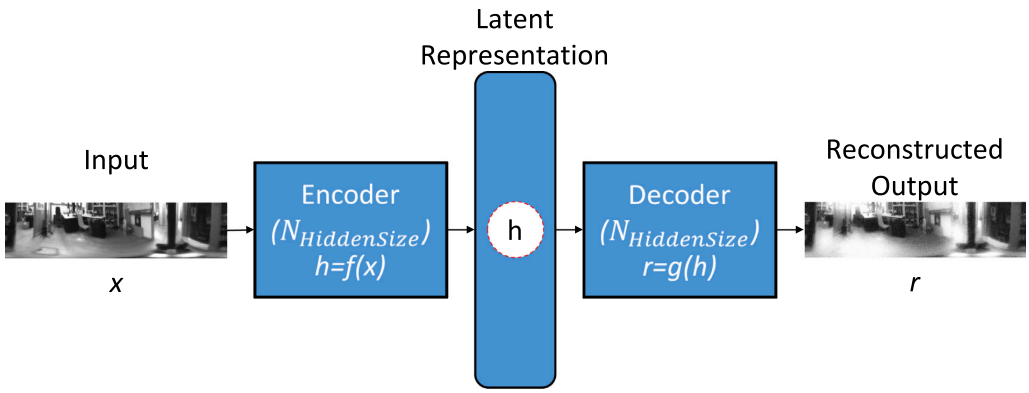



**Fig. 1.** Autoencoder structure. An input $x$ is mapped to obtain a reconstruction $r$ by means of using a latent representation $h$. $f$ is a function that encodes or maps $x$ to $h$ and $g$ is a function that decodes, that is, maps $h$ to $r$.

*2.3.5. Convolutional neural networks*

Convolutional Neural Networks, commonly known as CNNs, are currently the most popular tool among the deep learning techniques, since they have led to successful results in many practical applications. They are a specialized kind of neural network for processing data that present an already known topology. These networks are commonly designed to receive images as input and they have different applications such as classification or objects detection. This kind of networks are based on the use of convolutions, which is a specialized kind of linear mathematical operation (Goodfellow et al., 2016). That means, whereas traditional neural networks use matrix multiplication with a separate parameter that describes the interaction between inputs and outputs, CNNs present sparse interactions, i.e., using specific and meaningful features obtained from the input data. CNNs consist of local connections between neurons and hierarchically organized transformations of the data. Basically, CNNs are composed by three types of neural layers: convolutional layers, pooling layers and fully connected layers. Every layer transforms the input and generates an output according to the parameters established. This process is tackled throughout several layers until reaching the last layer, which is a fully connected layer that outputs a 1D feature vector, which provides the most likely prediction.

There are very well known CNNs whose architectures have been used as starting point to develop new computer vision tasks. For instance, AlexNet was introduced by Krizhevsky, Sutskever, and Hinton (2012). This network consists of eight layers (five convolutional layers and three fully connected layers) with a final 1000-way softmax and three pooling layers. The input image has a size of 227 × 227 × 3 and the network was trained to identify objects in the input images. It is able to identify 1000 object categories, such as keyboard, pencil, and a variety of animals. Fig. 2 shows the architecture of this network. GoogLeNet was proposed by Szegedy et al. (2015). This network has 22 layers, it is also trained for object classification but it uses 12 times fewer parameters than AlexNet. A wide review of the most outstanding CNNs can be found in Pak and Kim (2017). Moreover, Table 1 shows a summary table of the most popular CNNs until the present date.

Additionally, there are other options that permit reusing robust CNNs which have provided successful results, to solve different problems from the input images. On the one hand, the **transfer learning** technique consists in the process of retraining a pre-trained network to classify a new set of images, that is, reusing the architecture, weights and parameters of a CNN which already works properly as starting point to build a new CNN with a different purpose. The main idea is to get profit of most of the intermediate layers, because their parameters have been tuned with a large number of images. The problem, then, is reduced to changing the final layers (in order to re-adapt them to the new task proposed) and, perhaps, the initial layers (if the size of the images does not match the size used previously). Once the "new" network architecture is established, the training process starts through using the new labeled training data. Hence, this technique can save a considerable amount of time for training and even output better results than creating a new network from scratch. This idea has been used by many authors. For example, Han, Liu, and Fan (2018) use CNN transfer learning together with data augmentation in order to achieve good solutions despite the small size of the datasets used. Also, as mentioned previously, Wozniak, Afrisal, Esparza, and Kwolek (2018) use the transfer learning technique to retrain the VGG-F network to classify places among 16 rooms acquired by a humanoid robot. On the other hand, many authors have also proposed the use of intermediate layers to generate global-appearance descriptors of the input image. In this sense, once the network is properly available to face the desired task, the hidden layers perform vector description which can be used to characterize the input data. This idea has been exploited by some authors such as Arroyo, Alcantarilla, Bergasa, and Romera (2016), who use a CNN that automatically learns to generate visual descriptors which are robust against changes of seasons, in order to carry out a robust topological localization. Wozniak et al. (2018) also use the features extracted from the FC-6 layer to train a linear SVM (Singular Vector Machine) classifier. Mancini, Bulò, Ricci, and Caputo (2017) use this visual information to carry out place categorization with a Naïve Bayes classifier.

Regarding the use of CNNs to solve robotics tasks through visual information, there are many works that have provided successful results by using this technique. For instance, Sinha, Patrikar, Dhekane, Pandey, and Kothari (2018) propose a CNN to process data from a monocular camera and tackle an accurate robot relocalization in GPS-denied indoor and outdoor environments. Payá et al. (2018) propose using CNN-based descriptors to create hierarchical visual models for mobile robot localization. More recently, Chaves, Ruiz-Sarmiento, Petkov, and Gonzalez-Jimenez (2019) propose a CNN to build a semantic map. Concretely, they use the network to detect objects in images and, after that, the results are placed within a geometric map of the environment. Xu, Chou, and Dong (2019) propose a multi-sensor-based indoor global localization system integrating visual localization aided by CNN-based image retrieval with a Monte Carlo probabilistic localization approach.

*2.3.6. Regression neural networks*

Apart from the main techniques based on deep learning showed previously, there are other options that have been used by researchers. Despite its use is less common to solve mobile robotics tasks, they have also provided successful results.

Regression Neural Networks (R-CNN) are among these deep learning techniques. Deep neural networks are well known for classification

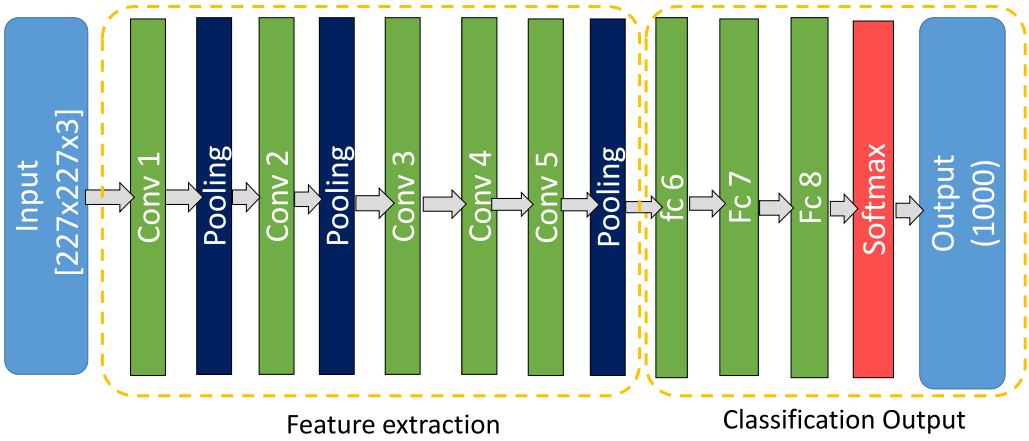



**Fig. 2.** Architecture of the CNN AlexNet. Input images have a size of 227 × 227 × 3 and the output is able to classify objects into 1000 categories.

**Table 1**
Summary of the most popular CNNs developed during the past few years.

| CNN | Year | Developed by | No. of convolutional layers | No. of parameters |
|---|---|---|---|---|
| LeNet | 1998 | LeCun, Bottou, Bengio, and Haffner (1998) | 5 | 60,000 |
| AlexNet | 2012 | Krizhevsky et al. (2012) | 8 | 60 million |
| GoogLeNet | 2014 | Szegedy et al. (2015) (Google company) | 22 | 4 million |
| VGG Net | 2014 | Simonyan and Zisserman (2014) | 19 | 138 million |
| Inception | 2015 | Szegedy et al. (2015) | 65 | 5 million |
| ResNet | 2016 | He, Zhang, Ren, and Sun (2016) | 152 | 25.6 million |
| Xception | 2017 | Chollet (2017) | 42 | 23 million |

problems, where the goal is to predict a single discrete label of an input vector. Nevertheless, the regression problem consists in obtaining a continuous value instead. Therefore, this type of network has commonly been proposed for continuous predictions such as forecasting. Bilgili and Sahin (2010) propose an analysis of regression neural network models to predict wind speed; and Kumar, Aggarwal, and Sharma (2015) introduce a study about regression neural networks to estimate the monthly average global solar radiation. These models have also been proposed for other types of predictions such as medical diagnoses. For instance, Kayaer and Yıldırım (2003) propose using a general regression neural network to diagnose diabetes. Ferreira, Amaral, Pires, Crisostomo, and Coimbra (2004) use a general regression neural network to construct the base of an adaptive neuro-fuzzy system and carry out a walking control of an autonomous biped robot. As for mobile robotics, the related literature also presents different examples of application. For example, Wang, Wang, and Zhuang (2007) use a general regression neural network for approximating the functional relationship between high-dimensional map features and states of the robot. Rahman, Park, and Kim (2012) propose a location estimation algorithm using generalized regression neural network and Wireless Sensor Network (WSN). Dezfoulian, Wu, and Ahmad (2013) propose a method to interpret the data from various types of 2-dimensional range sensors and a regression neural network to perform the navigation task.

### 2.4. Recurrent neural networks

Recurrent Neural Networks (RNNs) are a kind of neural network specialized in processing a sequence of values. This type of networks share parameters in a different way. Each member of the output is a function of the previous outputs and the connections between nodes form a directed graph along a temporal sequence. RNNs can use their internal state to process sequences of inputs and therefore they exhibit temporal dynamic behavior. RNNs are flexible in their use of context information, because they can learn what to store and what to ignore and they can recognize sequential patterns in the presence of distortions (Sak, Senior, & Beaufays, 2014). More detailed information about this type of networks can be found in the work developed by Graves (2012). An example of RNN is the Long short-term memory (LSTM), which has feedback connections and can process entire sequences of data. For instance, LSTM is commonly applicable to tasks such as unsegmented, connected handwriting recognition, as Messina and Louradour (2015) do to recognize lines of handwritten Chinese text, or speech recognition such as Graves, Mohamed, and Hinton (2013) do. Additionally, this tool has also been proposed to solve mobile robotics tasks. For example, Otte, Weiss, Scherer, and Zell (2016) introduce an extension of Long Short Term Memories (LSTMs) for ground robots based on vibration data classification with the aim of carrying out recognition of the ground reliably condition for mobile robot navigation. Rahmatizadeh, Abolghasemi, Behal, and Bölöni (2016) carry out a deep learning controller based on LSTM with the aim of learning manipulation tasks for assistance robotics and wheelchair mobile robots. Otte et al. (2016) propose an extension of LSTMs for classification of 14 different ground types based on vibration data, since recognizing the condition of the ground may be key in mobile robot navigation systems. Sun, Yan, Mellado, Hanheide, and Duckett (2018) present a 3-DOF pedestrian trajectory prediction approach for autonomous mobile robots by means of range-finder sensors with an LSTM network.

### 2.5. Deep reinforcement learning

Reinforcement learning is a branch of machine learning that has gained a lot of attention since it was proposed to play Atari games (Mnih et al., 2013). In reinforcement learning, an autonomous *agent* receives information from the environment and takes actions to maximize a notion of cumulative reward (Chollet, 2017). Deep reinforcement learning consists in the use of deep learning and reinforcement learning principles with the aim of creating efficient algorithms. This field of research has been able to solve complex decision making tasks that

were hard to solve by means of conventional methods. François-Lavet, Henderson, Islam, Bellemare, and Pineau (2018) introduce this deep learning model and focus on the aspects related to generalization and how deep reinforcement learning can be used for practical applications. Despite the related algorithms have been scarcely applied to solve real situation tasks, the state of the art already presents some examples, such as Lillicrap et al. (2015), who introduce a deep learning reinforcement algorithm that solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. As for the mobile robotics tasks, Zhang, Springenberg, Boedecker and Burgard (2017) propose a successor-feature-based deep reinforcement learning algorithm that can learn to transfer knowledge from previously mastered navigation tasks to new problem instances. Tai, Paolo and Liu (2017) propose a learning-based mapless motion planner based on an asynchronous deep reinforcement learning method to estimate the target steering commands with respect to the mobile robot coordinate frame. Zhu et al. (2017) introduce a target-driven visual navigation system in indoor scenes by using deep reinforcement learning with the aim of improving the lack of capability of generalizing new goals and the data inefficiency. Kahn, Villaflor, Ding, Abbeel, and Levine (2018) carry out a self-supervised deep reinforcement learning for robot navigation to improve the need to learn complex policies from the environment with few samples. Their robotic system captures raw monocular images and it is able to tackle the navigation task by means of learning by a fully autonomous reinforcement learning.

## 3. Description of the visual information by using AI tools

Vision sensors have been widely used for mobile robotics purposes. However, images are highly dimensional data and they also change for several reasons apart from the movement of the robot, such as change of illumination or position of some objects that constitute the environment. Hence, the approaches that work with these data consist commonly in extracting the most relevant and invariant information from scenes. In this sense, two main approaches have been commonly proposed; either by detection and description of local features, or working with global-appearance extraction methods. On the one hand, the methods based on local features consist in extracting some outstanding points from each scene and creating a descriptor for each point, using the information around it (Fig. 3(a)). On the other hand, global-appearance description methods consist in building a unique descriptor per image. Fig. 3 illustrates (a) local features extraction and description and (b) global-appearance description.

In the literature, many examples can be found using local features as well as global-appearance descriptors to solve mobile robotics tasks. To cite some examples, concerning local features, Kunii, Kovacs, and Hoshi (2017) propose a robust landmark tracking method for mobile robot operation in natural environments, where ORB (Oriented FAST and rotated BRIEF), CenSurE (Center Surround Extremas) are used for feature extraction and SURF (Speeded-Up Robust Features), ORB, FREAK (Fast Retina Keypoint) for feature description. Su et al. (2017) propose a global localization approach with the capability of addressing the kidnapped robot problem, where the ORB local descriptor is used to further improve localization accuracy. Regarding global-appearance descriptors, Payá, Reinoso, Berenguer, and Úbeda (2016) present a comparative analysis of some global-appearance descriptors for mapping. Rituerto, Murillo, and Guerrero (2014) propose the use of the *gist* (Oliva & Torralba, 2001) descriptor to build topological maps departing from omnidirectional images. Murillo, Singh, Kosecká, and Guerrero (2013) use a panoramic gist descriptor to address the localization task in urban environments. More recently, Faessler et al. (2016) present a vision-based quadrotor system to map a dense three-dimensional area. Korrapati and Mezouar (2017) propose the use of omnidirectional images through global appearance descriptors to build topological maps and also a loop closure detection method. Both local features and global-appearance descriptors have been commonly calculated by means of analytical methods. Traditionally, initial works in mobile robotics tried to extract and describe local features from the scenes. Later, a number of works proposed using the information as a whole, creating a holistic descriptor per scene. More recently, the development of new AI techniques and the evolution of the computing devices has made it possible to extract relevant information by means of such AI techniques. Fig. 4 shows this evolution of methods to extract relevant information from the scenes in mobile robotics, and includes a number of relevant works that make use of each approach. In the next subsections, some of the most popular methods are detailed.

### 3.1. Local features

Since the emergence of SIFT (Scale-Invariant Feature Transform) (Lowe, 2004), local features have played an important role in image matching, for example, to solve the image retrieval problem (Se, Lowe, & Little, 2005; Zheng, Yang, & Tian, 2017). Nevertheless, local features have not been used only for image retrieval, but they have also been proposed as a powerful tool for other computer vision problems such as wide baseline stereo matching or object detection. Typically, methods based on local features comprise two main stages: extracting a set of outstanding points, objects or regions from each scene and creating a descriptor for each. That means that every feature is described by means of a data vector, which is typically invariant against changes in the position and orientation of the camera. Once the extraction and description of the features has been addressed, they are usually tracked and matched along a set of scenes.

A considerable amount of local features extraction and description methods have been developed since the appearance of SIFT. Many subsequent developments focused on reducing its computational requirements or improving the invariability to other effects. For example, SURF (Bay, Ess, Tuytelaars, & Van Gool, 2008) presents lower computational cost and higher robustness against image transformation and BRIEF (Binary Robust Independent Elementary Features) is designed to be used in real time at expense of a lower tolerance to image distortion and transformations (Calonder, Lepetit, Strecha, & Fua, 2010). All these examples are known as traditional or hand-crafted features, since they are based on the detection of visual structures such as corners. A deep survey of these tools can be found in Payá et al. (2017) and Mukherjee, Wu, and Wang (2015) carried out an exhaustive comparative experimental study.

Concerning the development of local features based on AI techniques, they are widely known as learned features and like hand-crafted ones, the AI methods typically consist in either detecting or describing local features, or even both (detecting and describing). Within the learned local features, two main blocks can be established: features based on machine learning and based on deep learning techniques. As for the first block, FAST (Features From Accelerated Segment Test) was one of the first successful methods and it is designed for high-speed corners detection (Rosten & Drummond, 2006). Despite being principally constructed for speed purposes, this method also proved to outperform existing corner detectors. Later, simulated annealing was proposed to optimize the parameters of the FAST detector to achieve higher repeatability (Rosten, Porter, & Drummond, 2008). This work shows that using machine learning produces significant improvements in repeatability, speed and quality. Early attempts were based on genetic algorithms. In this sense, Trujillo and Olague (2006) present an approach for extracting automatically low-level features by applying genetic programming. These authors introduce a Genetic Programming implementation that is capable of discovering a modified version of a feature operator which presents an improved performance. This work also highlights the balance between genetic programming and domain knowledge expertise to obtain results that improve hand-crafted solutions. More recently, machine learning tools have been typically used in feature detection to imitate and/or accelerate previously defined

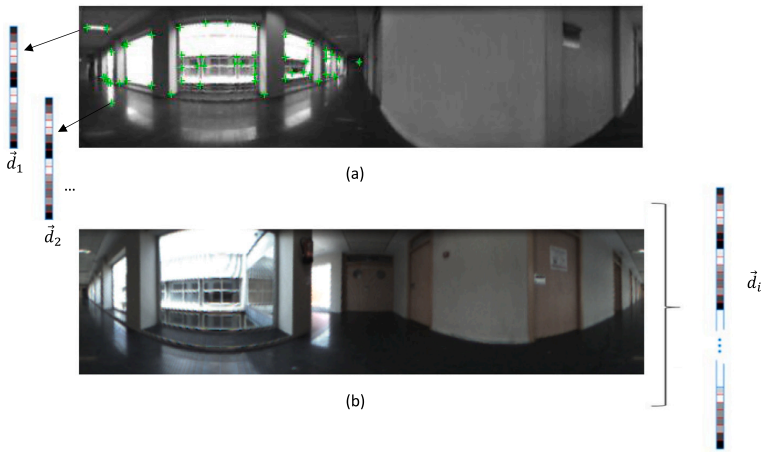



**Fig. 3.** Two main methods to extract the most relevant information from the images for mapping and localization purposes. (a) Detection, description and tracking of some relevant landmarks along a set of scenes. (b) Building a unique descriptor per image that contains information on its global-appearance.

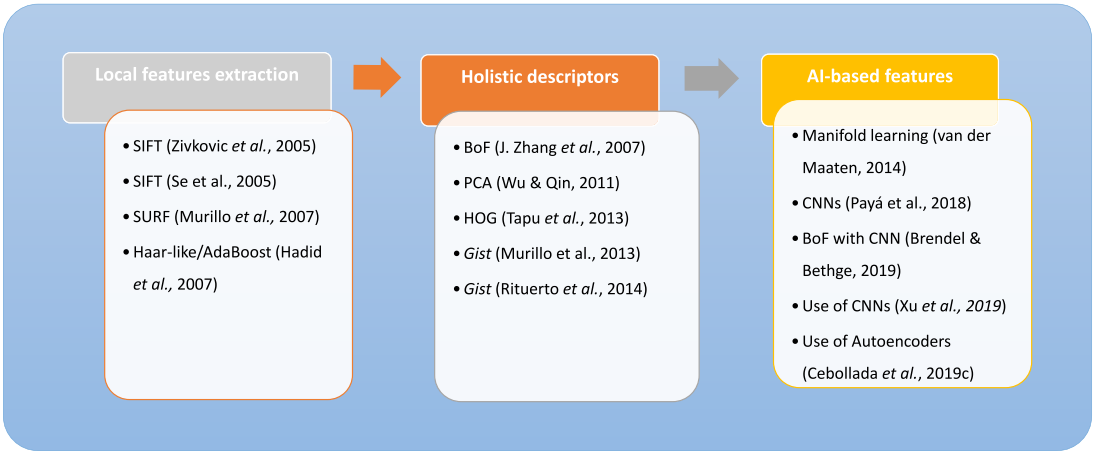



**Fig. 4.** Evolution of methods to extract relevant information from the scenes, with the purpose of solving problems in mobile robotics.

methods. Šochman and Matas (2009) propose a faster version of binary decision algorithms by using a WaldBoost classifier . This classifier learns to minimize the decision time of the classifier while guaranteeing predefined precision. Holzer, Shotton, and Kohli (2012) address the Interest Point (IP) detection as a regression problem by using machine learning. A regression forest (RF) model learns to detect if there is an IP in the center of a given image patch. Other researchers use machine learning to reduce the size of the descriptor, such as Strecha, Bronstein, Bronstein, and Fua (2011), who propose metric learning to reduce the size of the descriptors by representing them as short binary strings. In short, they map the descriptor vectors into the Hamming space, which is used to compare the resulting representations. This way, the size of the descriptors is reduced by representing them as short binary strings. Simonyan and Zisserman (2014) develop a learned local feature descriptor by using convex optimization. This work shows that learning the pooling regions for the descriptor can be formulated as a convex optimization problem. It also shows a descriptor dimensionality reduction by using Mahalanobis matrix nuclear norm regularization. Both formulations are based on discriminative large margin learning constraints.

Regarding the learned features based on deep learning techniques, they have been often used to improve rather than to replace hand-crafted local features. For instance, they have been used to learn covariant feature detectors invariant against viewpoint changes without supervision. For example, Lenc and Vedaldi (2016) propose a general machine learning formulation for covariant feature detectors. Moreover, many other improvements can be done, such as including explicitly modeling detection confidence, predicting multiple features in a patch, or jointly training detectors and descriptors. Mishkin, Radenovic, and Matas (2018) introduce a method for learning local affine-covariant regions. The proposed affine shape estimator is trained considering the loss function, descriptor type, geometric parametrization, etc. Furthermore, the training process does not require aligned patches geometrically accurate.

Most of the related works are focused on feature descriptors, nevertheless, more recently, there has been also an important progress in the development of detectors. For example, Verdie, Yi, Fua, and Lepetit (2015) use deep neural networks to learn a feature detector robust against illumination changes. The process consists in firstly identifying good keypoint candidates in multiple training images and

secondly training a regressor to predict a score map whose maxima are those points. Yi, Trulls, Lepetit and Fua (2016) propose an end-to-end framework based on the use of a deep network architecture to detect keypoints, estimate orientation and compute descriptors. These authors also developed a work to train a CNN to estimate the canonical orientation of a local feature given an image patch centered on the feature point and extended it to several different description methods (Yi, Verdie, Fua and Lepetit, 2016). They propose siamese networks to avoid the task of finding a target orientation to learn. Furthermore, they also propose a new activation function. Concerning CNNs to obtain local features, its use has been widely proposed by several authors. This technique is specially successful for large-scale image retrieval applications. For instance, Noh, Araujo, Sim, and Han (2016) propose a CNN-based local feature that is trained for instance-level recognition tasks without the need of object and patch-level annotations and it is suitable to replace hand-crafted descriptors. This framework can be used in image retrieval problems, enabling more accurate feature matching and geometric verification.

Nonetheless, despite the wide use of deep convolutional networks, local viewpoint invariant features based on hand-crafted techniques still play an important role in applications such as motion and image retrieval. Recently, Lenc and Vedaldi (2018) have carried out a deep evaluation of local feature detectors by evaluating a range of state-of-the-art local feature detectors. Through this study, they concluded that machine-learning-based detectors help to improve illumination invariance, that traditional methods are still competitive and they suggest also that a significant progress regarding deep-learning-based detectors can be done.

### 3.2. Global-appearance description

The approach based on global-appearance or holistic descriptors consists on working with the image as a whole, i.e., without extracting any local information. For this type of approaches, each image is represented by a unique descriptor that contains information on its global appearance (Payá et al., 2017). Concerning mobile robotics, this description method presents advantages in dynamic and poorly structured environments, where extracting stable local features may result difficult. Additionally, due to the fact that each image is represented by a unique descriptor, global-appearance descriptors lead to simpler mapping and localization algorithms (Amorós, Payá, Marín, & Reinoso, 2018; Berenguer, Payá, Valiente, Peidró, & Reinoso, 2019; Cebollada, Payá, Mayol and Reinoso, 2019; Cebollada, Payá, Román and Reinoso, 2019).

A wide range of works have been proposed during the past few years to develop holistic descriptors by using AI techniques. This method is known by some authors as feature engineering and it tries to take advantage of human prior knowledge to compensate the weaknesses that may present the algorithms (Bengio et al., 2013). One of the main objectives of developing methods to learn descriptors is to achieve faster solutions to proposed AI problems. Furthermore, AI applications have proved to be able to understand the environment that surrounds the camera, thank to their capability of identifying interesting and rejecting unprofitable information from the sensory data. Remarking the global-appearance descriptors based on deep architectures, they are usually effective to train robust models and introduce two advantages in this topic: first, deep architectures promote the reuse of features and second, they lead to more abstract features in higher layers that are typically invariant to local changes. A profound review about a wide range of unsupervised feature learning techniques can be found in the work presented by Bengio et al. (2013).

Among the early techniques proposed, **PCA** was one of the first alternatives that presented robustness. PCA basically performs a linear transformation $h = f(x) = W^T x + b$ of the input $x \in \mathbb{R}^n$ and the results are $d_h$ features that are the first components of the representation $h$ (Kirby, 2000). Similar to PCA, Independent Component Analysis (**ICA**) performs a linear analysis to obtain distinctive features based on linear generative models with non-Gaussian independent variables. Like sparse coding, ICA and its variants have also been used to obtain nonlinear features such as in the works developed by Bell and Sejnowski (1997), Jutten and Herault (1991) and Le, Zou, Yeung, and Ng (2011).

Successful feature learning algorithms and related applications are used in many works using a variety of approaches such as **RBMs** (Restricted Boltzmann Machines). For example, Hinton, Osindero and Teh (2006) propose a technique that consists in stacking pre-trained RBMs into deep belief networks (DBN), where the top layer is interpreted as an RBM and the lower layers as a directed sigmoid belief network. This work has proved to give better digit classification than discriminative learning algorithms. Salakhutdinov and Hinton (2009) propose to combine RBM parameters into DBM (Deep Boltzmann Machines) by halving the RBM weights to obtain the DBM weights and train it by approximate maximum likelihood. This way, this work shows that DBM learn good generative models and perform well on handwritten digit and visual object recognition tasks. Larochelle, Bengio, Louradour, and Lamblin (2009) carry out an empirical study about the use of different RBM input unit distributions. This study confirms the hypothesis that the greedy layer-wise unsupervised training strategy improves the optimization by initializing weights in a region near a good local minimum and it also brings better generalization of the input. Another important perspective on global-appearance descriptors is based on the **manifold learning**, a geometric notion whose premise is based on the concentration of high-dimensional input space in the vicinity of a manifold $M$ of lower dimensionality. The majority of the methods based on this technique lead to a non-parametric approach based on neighbor graphs. Belkin and Niyogi (2003) propose a geometrical algorithm for representing the high-dimensional data that provides a computationally efficient reduction of dimensionality. This reduction has locality-preserving properties and a natural connection to clustering. Donoho and Grimes (2003) propose a Hessian-based locally linear embedding method for recovering the underlying parametrization of scattered data. Weinberger and Saul (2006) introduce an algorithm for unsupervised learning of image manifolds by semidefinite programming. The algorithm computes a low dimensional representation of each image so that distances between nearby images are preserved. Accelerating t-SNE using tree-based algorithms, author=van der Maaten, L. (2014) proposes variants of the Barnes–Hut algorithm with the t-SNE (t-distributed Stochastic Neighbor Embedding) algorithm to learn embeddings of data sets with millions of objects. More recently, some authors have proposed to use free energy functions, that is, without explicit latent variables. For instance, Ngiam, Chen, Koh, and Ng (2011) use a hybrid of Monte Carlo to train the free energy function. In brief, they propose using deep feedforward neural networks to model the energy landscapes that define probabilistic models. The lower layers of the model adapt the training of the higher layers, and thereby this produces better generative models. By means of this method, all the layers of the model are simultaneously and efficiently trained. Kingma and Cun (2010) propose denoising score matching. Differentiating the loss with respect to the model parameters is automated with an extended version of a double-backpropagation algorithm.

Apart from the previously mentioned techniques, it is worth remarking the use of deep neural networks, specially **CNNs**, to obtain holistic descriptors, since a number studies have proved that these networks can learn more transferable features for domain adaptation and produce successful results in a wide range of scenarios and applications (Glorot, Bordes, & Bengio, 2011; Yosinski, Clune, Bengio, & Lipson, 2014). For instance, Donahue et al. (2014) propose the use of features extracted from the activation of a deep convolutional network trained in a fully supervised fashion on a large, fixed set of object recognition tasks and use it for a completely different task. The works focuses on investigating the semantic clustering of deep convolutional features with respect to a variety of tasks such as scene recognition, domain adaptation, and

fine-grained recognition. The study addresses an efficacy comparison relying on various network levels to define a fixed feature. Yosinski et al. (2014) carry out a deep study about how transferable features are in deep neural networks. They conclude that features obtained from the initial layers appear not to be specific to a particular dataset or task and the features become more specific as the selected layer approaches to the last one. Additionally, they conclude that initializing a network with transferred features from almost any number of layers can produce a boost to generalization. Long, Cao, Wang, and Jordan (2015) propose a Deep Adaption Network (DAN) architecture that generalizes deep CNNs to the domain adaptation scenario. The DAN architecture learns transferable features and can scale linearly by unbiased estimate of kernel embedding. Arandjelovic, Gronat, Torii, Pajdla, and Sivic (2016) solve the image retrieval problem by developing a CNN and using it to obtain global-appearance descriptors. This network incorporates a new layer which is inspired by the "Vector of Locally Aggregated Descriptors" (VLAD) image representation. VLAD is commonly used in image retrieval tasks. Gordo, Almazán, Revaud, and Larlus (2016) introduce a method that employs a region proposal network to learn which regions should be pooled to form the final global descriptor. This approach produces a global image representation in a single forward pass. Most recently, Xu et al. (2019) propose a transfer learning method based on a pre-trained model to transform general features into special features, which are adapted to the desired task. The pre-trained model of Faster R-CNN is used to extract the high dimensional convolution features of images.

Another technique widely used in recent years within the deep learning has been the use of **autoencoders** and variations of this tool. As outlined in Section 2.3.4, in the autoencoder frameworks, the starting point is a feature-extracting function in a specific parameterized closed form. This function, $f(x)$, is the encoder and tackles the straightforward computation of a feature vector from an input $x$ through $h = f(x)$. So, for each sample of the dataset $X = \{x_1, \dots, x_N\}$, we define $h(i) = f(x_i)$, where $h(i)$ is the feature vector computed from $x_i$. For example, Vincent, Larochelle, Lajoie, Bengio, and Manzagol (2010) propose a denoising autoencoder (trained to denoise corrupted versions of the inputs) to obtain a robust global-appearance descriptor, which is used successfully to solve classification problems. This work shows that the denoising autoencoders are able to learn Gabor-like edge detectors from natural image patches. The descriptor generated permits performing classification task similar than using deep belief networks. Coates, Ng, and Lee (2011) carried out a detailed analysis of the effect of changes in the model setup (receptive field size, number of hidden nodes, the step-size) and compare the extracted features (sparse autoencoders, sparse RBMs, K-means clustering, and Gaussian mixtures) with a whitening process. They conclude that complex algorithms can have greater representational power and simple but fast algorithms can be highly competitive. Le (2013) trains a deep sparse autoencoder on a large dataset of images to build high-level features. The experiments show that this feature detector is robust against translation, scaling and out-of-plane rotation. Gao and Zhang (2017) propose an approach based on the Stacked Denoising Autoencoder (SDA) to detect loops for a visual SLAM system. The descriptors are calculated by using SDA over patches of the original images. This autoencoder can lead to very complicated structures, since the learned features reflect the inner patterns of the data, whereas traditional hand-crafted feature descriptors are usually not able to show that.

Within this subsection, it is worth to mention the **Bag of local Features (BoF)** method, which can be considered as a blended method between local features and global-appearance descriptors. This method comes from Bag of Words (BoW) representation, which basically consists of a model for representing text data with machine learning algorithms. The bag-of-words model is simple to understand and implement and has seen great success in problems such as language modeling and document classification. Concerning BoF, many approaches have been developed in computer vision during the last few decades (Csurka, Dance, Fan, Willamowski, & Bray, 2004; Jurie & Triggs, 2005; Lazebnik, Schmid, & Ponce, 2006; Zhang, Marszałek, Lazebnik, & Schmid, 2007). Bag of Features methods have been applied to image classification, object detection, image retrieval, and even visual localization for robots. BoF approaches consists basically in a characterization based on the use of a collection of image local features to form a vector that characterizes the input image. Despite the lack of structure or spatial information, this image representation can be good enough for many state-of-the-art applications. A detailed explanation of this technique can be found in O'Hara and Draper (2011).

Like local features and global-appearance descriptors, many works have proposed the use of AI techniques in bag-of-features frameworks. In the majority of these works, AI (specially deep learning) is used to replace previously hand-crafted features extraction methods in BoF. For example, Gong, Wang, Guo, and Lazebnik (2014) introduce a multi-scale orderless pooling (MOP-CNN), which extracts CNN activations for local patches at multiple scale levels, performs orderless VLAD pooling of these activations at each level separately, and concatenates the result. The proposed method can be used as a generic feature for either supervised or unsupervised recognition tasks. Ng, Yang, and Davis (2015) present an approach for extracting convolutional features from different layers of the networks OxfordNet and GoogLeNet, and adopt VLAD encoding to encode features into a single vector for each image. Mohedano et al. (2016) propose an image retrieval pipeline based on encoding the convolutional features to obtain BoF by assigning each local array of activations in a convolutional layer to a visual word. Feng, Liu, and Wu (2017) propose CNNs for optimizing the feature extraction to perform BoF for geographical scene classification. Cao, Huang, and Shen (2017) build an effective BoF model using deep local features. They show how to use the CNN as a combination of local feature detector and extractor without the need of feeding multiple image patches to the network. Khan et al. (2018) propose two strategies to encode multi-scale information explicitly during the image encoding stage. The aim of this approach is to recognize human actions. The first approach is based on a multi-scale image representation with scale encoded with respect to the image size. The second approach, instead, encodes feature scale relative to the size of the bounding box corresponding to the person instance. Scale coding of bag of deep features is performed by applying the coding strategies to the convolutional features from the pre-trained VGG-19 network. Brendel and Bethge (2019) implement a variant of the ResNet-50 CNN that classifies images by using BoF as input and compare their method with other high-performance deep neural networks (VGG-16, ResNet-50 and DenseNet-169) to carry out the classification task with the ImageNet dataset.

From this section, the conclusions achieved are that the use of techniques based on AI present some advantages: compared to other description methods, they lead to semantic, objects and geometric forms interpretation, and once trained, the model is easy and quick to use and obtain the necessary descriptors. Nonetheless, these descriptors are not based on a closed mathematical process and the method to obtain such description is not known *a priori*; it depends on the parameters configuration during the training process. Therefore, the modeling will be sensitive to the specific training dataset and it is expected to work well under similar environments, but it may lead to less robust descriptors under different circumstances, what may limit further results.

About the performance of AI and deep learning tools in the process of extracting information from the scenes, as detailed in the previous paragraph, it typically depends on a training process which requires a high number of data vectors. Therefore, this process can be computationally expensive and require large computational resources and a long period of time to learn how to extract significant data from the set of training images. However, once the model has been trained, using this model to extract information from new images (as the robot performs a task) is a relatively fast process. Quantitative data about these processes can be found in the works by Cebollada, Payá, Valiente, Jiang and Reinoso (2019) and Cebollada et al. (2020).

## 4. Mobile robotics tasks using vision and AI

This section presents a review of recent works related to the mapping, localization, navigation, SLAM and exploration tasks in robotics using visual information and AI tools.

### 4.1. Map building

Mapping consists basically in creating a map, model or representation of the environment using the data provided by the sensors mounted on the robot. Such models are useful to solve, subsequently, other tasks, such as localization, path planning or navigation. These problems can be solved by comparing the information provided by the sensors of the robot with the model. Thrun (2002) presented an exhaustive explanation to the robotic mapping concept. In the related literature, two main frameworks have been proposed in order to carry out this task: the metric maps, which represent the environment with geometric accuracy; and the topological maps which describe the environment as a graph containing a set of locations with the related links among them. For example, Tanzmeister, Thomas, Wollherr, and Buss (2014) propose an approach that estimates a uniform, low-level, grid-based world model including dynamic and static objects. da Silva et al. (2018) propose a localization and navigation approach for mobile robots using topological maps and using a CNN to obtain descriptors from omnidirectional images. Apart from these options, arranging the information hierarchically constitutes an efficient alternative. This framework consists in creating a map which is composed of several layers with a hierarchical structure. The high-level layers contain a relatively compact amount of information, which permits a rough but quick localization. The low-level layers have usually more information and are used to refine the position. Kuipers, Modayil, Beeson, MacMahon, and Savelli (2004) propose a hierarchical hybrid map, which consists in using a metrical approach to build local maps of small-scale space and topological maps to represent the structure of large-scale space. This approach is proposed to solve the SLAM task in an environment with multiple nested large-scale loops. Cebollada, Payá, Mayol et al. (2019) propose a study about clustering methods to carry out efficiently the data compaction of metric and topological maps based on omnidirectional images with the aim of building hierarchical maps. They have also tested the robustness of the hierarchical maps proposed under different illumination conditions to tackle the localization task (Cebollada, Payá, Román et al., 2019).

Concerning mapping by using visual data, the models are commonly created using either local or global features. Beyond these frameworks, the use of AI with vision systems has contributed to the emergence of new paradigms to create visual maps. Zivkovic, Bakker, and Krose (2005) build a hierarchical model based on omnidirectional images. The characterization of the data is done through local features (SIFT) and to carry out the graph partitioning, and define the map hierarchy, a cluster algorithm is proposed. Peretroukhin, Clement, and Kelly (2017) propose the use of Bayesian Convolutional Neural Networks (BCNN) to train and implement a sun detection model from a single RGB image to incorporate global orientation information from the sun into a visual odometry pipeline. They also propose an uncertainty associated with each prediction by using a Monte Carlo dropout scheme. Clark, Wang, Markham, Trigoni, and Wen (2017) carry out a mapping and a posterior relocalization task by means of feeding an LSTM network with holistic descriptors obtained from a CNN. The proposed model estimates the current pose within an environment departing from short sequences of monocular frames. Similarly to this work, concerning the use of the CNNs to obtain global-appearance descriptors, many authors have proposed this strategy. For instance, Iyer, Murthy, Gupta, Krishna, and Paull (2018) propose a self-supervised visual odometry estimation. The approach first obtains global-appearance descriptors from the fully connected layer of the VGG-11 CNN. Second, an LSTM network is used to regress pose transformations between monocular frame-pair sequences. Kopitkov and Indelman (2018) propose an approach to estimate the robot position via CNN holistic descriptors and using neural networks to learn a generative viewpoint-dependent model of CNN features given the robot pose and approximate this model by a spatially-varying Gaussian distribution. Furthermore, once developed the proposed model, it is utilized within a Bayesian framework for probabilistic inference to solve the localization problem. Sarlin, Cadena, Siegwart, and Dymczyk (2019) propose a hierarchical model using a CNN. This network simultaneously extracts local features and global descriptors that are used for accurate 6-DOF localization. Once the model is built, the coarse localization is solved by using global retrieval through a k-nearest neighbor algorithm and the holistic descriptors. The fine localization is solved through evaluating matching points from the local features.

Another widely developed strategy is the use of neural networks to model a system that is capable of estimating the position directly from the raw data. For example, Kuse, Jaiswal, and Shen (2017) propose a deep residual network to model the environment representation. Naseer and Burgard (2017) develop a model that allows a 6-DOF localization using a regression neural network and a single monocular RGB image. The resulting map size is constant with respect to the size of the dataset and during the localization task, the time complexity is also constant and independent of the dataset size. Walch et al. (2017) introduce a CNN+LSTM model to estimate the pose in both indoor and outdoor environments. Raw data are introduced to the network and it is trained in such a way that the CNN layers learn suitable local features and they are then used by the LSTM layers to improve the pose estimation. In this way, the whole network learns to optimize the localization task. Brahmbhatt, Gu, Kim, Hays, and Kautz (2018) propose a mapping model based on a regression neural network, which enables learning a data-driven map representation. Furthermore, the proposed network can be updated with unlabeled data. Payá et al. (2018) use a CNN to obtain holistic descriptors and create a hierarchical visual model with that information. Sinha et al. (2018) also propose a mapping and a subsequent localization task based on regression neural networks. The proposed method first trains a CNN that takes RGB images from a monocular camera as input and performs regression for robot pose estimation. It then incorporates the relocalization output of the CNN in an Extended Kalman Filter to tackle the localization task. Moolan-Feroze, Karachalios, Nikolaidis, and Calway (2019) propose the deployment of a model to map the environment that surrounds wind turbines. For this purpose, a CNN is trained to extract an estimate of the projection of the 3D skeleton representation departing from monocular images. After that, the localization task is solved by means of a pose graph optimization that uses the 3D representation outputs from the CNN.

### 4.2. Localization

As denoted in 4.1, localization is the task that tries to estimate the current position and orientation of the robot in the environment and to carry out this, a model of the environment must be available prior to start the localization. Filliat and Meyer (2003) presented an exhaustive review about the state-of-the-art strategies to carry out the localization in mobile robots. Regarding the use of vision systems together with AI, a wide range of works have been proposed in recent years. For example, Kendall, Grimes, and Cipolla (2015) present a robust and real-time monocular 6-DOF relocalization system. The proposed system trains a CNN to regress the 6-DOF camera pose from a single RGB image in an end-to-end manner without additional graph optimization. Neto (2015) proposes a topological localization system based on monocular images, learning classifier systems and self-organizing maps (SOM). The whole system carries out a localization task through detecting and avoiding obstacles by means of both local and holistic features. Meng et al. (2017) address the localization issue by using methods based on Random Forests that directly estimate 3D positions with

SIFT features as input. Li, Liu, Gui, Gu, and Hu (2018) introduce an indoor localization approach using a dual-stream regression CNN by introducing color data as well as depth data from monocular images. This system is tested under night illumination conditions and also under blur effects. As in mapping, there are also a wide range of works proposed during the last few years that propose and evaluate the use of intermediate layers from several CNNs to obtain local features or holistic descriptors. For instance, Sünderhauf, Shirazi, Dayoub, Upcroft, and Milford (2015) introduce a real-time place recognition algorithm by using different layers from CNNs to carry out the localization in large maps by integrating a variety of existing optimization techniques such as semantic search space partitioning.

Cascianelli et al. (2017) propose a strategy for mapping and posterior localization that relies on the use of a CNN to obtain local features that are robust to appearance changes. Similar to this work, Unicomb, Ranasinghe, Dantanarayana, and Dissanayake (2018) also use a CNN to extract local features; in this case, they extract ground plane edges and then estimate a 6-DOF position through an EKF (Extended Kalman Filter) algorithm. Moolan-Feroze and Calway (2018) present a framework that uses CNNs to predict object feature points that are out-of-view in the input image. These feature points are then fed to estimate more robustly the pose of the robot in the environment. Holliday and Dudek (2018) propose a combination of deep-learning-based hierarchical object features and SIFT features. These points are used to perform more robust localization tasks. Regression networks have been extensively proposed to directly estimate the position within the map. For example, Sommer, Kim, Kim, and Jo (2017) carry out a 6-DOF localization with CNN by applying transfer learning over pre-trained CNN Google's Inception-V4. Xu et al. (2019) introduce a multi-sensor-based indoor global localization system using visual localization aided by CNN-based image retrieval with a Monte Carlo probabilistic approach. Cebollada, Payá and Valiente et al. (2019) propose the use of autoencoders and also a CNN to obtain holistic descriptors from omnidirectional images and then use them to solve the localization task in an indoor environment. Cattaneo et al. (2019) develop a regression network, which learns to localize an RGB-D image of a scene in a map built from LIDAR (Laser Imaging Detection and Ranging) data. In a similar way, Weinzaepfel, Csurka, Cabon, and Humenberger (2019) introduce a regression strategy based on CNN for visual localization from a single RGB image that relies on densely matching a set of objects of interest. Given a query image, the network model detects the objects, segments them and finds a dense set of 2D–2D matches between each detected object and its corresponding one in the reference image. Given these 2D–2D matches, a Perspective-n-Point problem is used to estimate the pose.

### 4.3. Navigation

The navigation task basically consists in solving the problem of how the robot can get to other places from its current position (Levitt & Lawton, 1990). That is, to perform a trajectory to reach a certain place. This trajectory is calculated by a path-planning system, and the robot is subsequently commanded by the control system (Barber, Crespo, Gómez, Hernámdez, & Galli, 2018). During the past few years, a wide number of works about navigation using visual sensors and AI can be found in the related literature. Maier, Bennewitz, and Stachniss (2011) propose a machine learning classifier to address the autonomous navigation for a humanoid equipped with a monocular vision and a sparse laser data system. The classifier proposed for visual information is based on the hue and saturation values from the HSV color space with the aim of being less sensitive to illumination changes. Additionally, a classifier is also trained with texture-based information. The classifiers learn to estimate from the images which parts of the surroundings of the robot are traversable. The goal is to make the system as independent as possible of the 3D scan data. Tapu, Mocanu, and Zaharia (2013) introduce a visual navigation system based on HOG (Histogram of Oriented Gradients) (Dalal & Triggs, 2005) and BoW for obstacle classification. This approach basically consists in detecting objects from an image. For each object, its related HOG descriptor is introduced into the BoW retrieval framework. After that, an SVM classifier is applied to retrieve which object is. Object detection plays a vital role to carry out the navigation approach successfully. Giusti et al. (2015) presents a method to autonomously navigate through a man-made trail in the mountain for UAVs. This approach is based on a CNN classifier with the aim of operating the main direction of the trail.

Yang et al. (2017) propose a two-stage CNN with intermediate perception. The first stage CNN predicts the depth and surface normal from images. The second CNN predicts a path from the depth and normal maps using another CNN model. Smolyanskiy, Kamenev, Smith, and Birchfield (2017) propose several deep learning tools to address the autonomous navigation of Micro Aerial Vehicles (MAVs). A Recurrent Neural Network is used to estimate the view orientation and lateral offset of the MAV with respect to the trail center. In addition, another network is used to estimate depth with the aim of carrying out low-level obstacle detection. Puthussery et al. (2017) propose an autonomous navigation approach that uses the Inception v3 CNN to classify the objects detected by the camera. This classification is included within the Marker Detection Phase, which is done prior to the Robot Navigation Phase. Richter and Roy (2017) introduce a navigation system based on deep learning. On the one hand, they propose a fully connected feedforward network to model collision probability. On the other hand, they propose the use of an autoencoder to recognize when a query image is novel and requires a priority treatment. This is due to the fact that neural networks may not be efficient to provide accurate estimations when queried input data are very different from training data. Concerning the use of autoencoders for navigation, Mancini, Costante, Valigi, and Ciarfuglia (2016) introduce an object detection method that is able to detect obstacles at very long range and at a very high speed without making motion assumptions. This method is based on the use of an autoencoder which is trained with real and synthetic images and performs depth predictions. Deepika and Variyar (2017) propose a method which uses an autoencoder architecture for pixel-wise semantic segmentation of the image followed by an obstacle detection algorithm. The aim of this approach is to develop a robust vision based autonomous navigation system for self-driving cars. Walker, Graham, and Philippides (2017) propose this tool to build a compressed representation of visual data. Images reconstructed from the compressed representation retain enough information to be used as a visual compass (an image is matched with another to recall a movement direction).

Zhao et al. (2018) present a hybrid structure with a CNN and local image features to achieve first-person vision pedestrian navigation. They also developed a novel global pooling operator which improves the results obtained by the CNN for real-time scene recognition. A SIFT-based tracking algorithm is designed for movement calculation, then the mixture of both threads perform a robust trajectory tracking. Anderson, Wu, Teney, Bruce, Sünderhauf, Reid, et al. (2018) present a reinforcement learning approach based on vision and natural language in a large-scale environment. The aim of this work is to provide a visually-grounded natural language navigation which is able to work properly in real buildings. Hui, Bian, Zhao, and Tan (2018) propose an autonomous navigation approach for UAVs in outdoor environments surrounded by transmission towers and power lines. This method introduces the use of a CNN trained from end to end to address semantic segmentation and detected the power lines. Mansouri, Karvelis, Kanellakis, Kominiak, and Nikolakopoulos (2019) present a CNN to address autonomous navigation of low-cost Micro Aerial Vehicle platforms along dark underground mine environments. The proposed CNN provides online heading rate commands for the MAV by utilizing the image stream from the on-board camera, thus allowing the platform to follow a collision-free path along the tunnel axis. Ma, Chen, and Liu (2019) introduce a navigation system approach based on reinforcement

learning. Moreover, they use a variational autoencoder to obtain visual features from the input images, these features are put into the network together with the target and motion information. Ruan, Ren, Zhu, and Huang (2019) propose an approach for the navigation of mobile robots in an unknown environment using deep reinforcement learning. Through a dueling network architectures based on the double deep q network (D3QN) algorithm, the robot learns the environment and also models to navigate autonomously to the target destination with an RGB-D camera only.

### 4.4. Simultaneous localization and mapping

Additionally to the mapping and the subsequent localization task, the SLAM presents a blended alternative. This process consists in building continuously a map and updating it as the robot simultaneously estimates its position within the model. Fuentes-Pacheco et al. (2015) presented an exhaustive review about the state-of-the-art strategies to carry out SLAM. The related literature shows that not many approaches have been proposed to solve this task by using visual information and AI tools. Apart from some of the examples commented in Sections 4.1 and 4.2 , which propose mapping or localization tasks with the aim of developing subsequent SLAM, there are other examples that propose complete SLAM systems. For instance, Wu and Qin (2011) propose a SLAM algorithm based on omnidirectional images. This algorithm uses incremental landmark appearance learning to provide posterior probability distribution for estimating the robot pose under a particle filtering framework. The major contribution of the work is to represent the posterior estimation of the robot pose by incremental probabilistic PCA, which can be incorporated into the particle filtering algorithm for SLAM. Lu et al. (2015) propose a machine learning tool known as multi-task point retrieval to develop a regression model based on 3D points local features extracted from monocular images. Garg, BG, Carneiro, and Reid (2016) carry out an unsupervised deep convolutional network which behaves like an autoencoder, since it does not require annotated ground-truth data. This network is trained with the aim of predicting the depth map for the source image. During the training step, a pair of images (source and target) are fed into the network. Schmidt, Newcombe, and Fox (2017) introduce a CNN to produce robust local features and then use them for dense correspondence estimation and solve the SLAM task. An interesting work was developed by Gao and Zhang (2017) in which they carry out a loop detection by training an autoencoder that calculates local features from monocular images. Tateno, Tombari, Laina, and Navab (2017) propose an approach which consists in two CNNs based on monocular RGB images. The first network is trained with the aim of predicting depth. CNN-predicted dense depth maps are naturally fused together with depth measurements obtained from direct monocular SLAM, based on a scheme that privileges depth prediction in image locations. The second network is trained to address semantic segmentation. Once the information is obtained from the CNNs, this is fused with more data to address the SLAM task in a highly accurate way. Mukasa, Xu, and Stenger (2017) introduce a SLAM framework that integrates the geometrical measurements obtained from a monocular vision system with depth information predicted by means of a CNN. Tang, Ren, and Liu (2017) introduce the use of CNNs fed with visual information and human voice commands with the aim of solving the SLAM task with a mobile robot. Focusing on the visual information, the raw data is fed into two CNNs, the first network is used to produce accurate localization updates and the second is used to perform an object recognition task. The recognized objects are used to reinforce the mapping task. Zhang, Su and Zhu (2017) propose a loop closure detection framework based on CNNs. In this way, the images are fed into a pre-trained CNN model to extract holistic descriptors and, after that, these descriptors are pre-processed with PCA.

More recently, Milz, Arbeiter, Witt, Abdallah, and Yogamani (2018) explore the use of deep learning tools to enhance the visual SLAM. On the one hand, they propose the use of CNN to carry out the depth estimation. On the other hand, on the other hand, they propose the use of CNNs to address an end-to-end approach for learning of feature matching. This technique can learn diversity and distribution instead of just picking the top high textured features. Zhong, Wang, Zhang, and Wang (2018) address the SLAM and object detection by using a Single Shot multi-box object Detector (SSD). The RGB-D information is introduced to the SSD and it detects moving and static objects inside the image. After this, the detected moving objects are eliminated and the rest of the data are used to build a semantic map composed of all the detected static objects in the mapping thread. Simultaneously, the dynamic objects are used to update the tracking and local mapping thread. Liang, Tie, Qi, and Bi (2018) propose a CNN based on 360 degrees panoramic images to carry out the visual navigation and SLAM tasks in outdoor environments. Bloesch, Czarnowski, Clark, Leutenegger, and Davison (2018) present a compact but dense representation of the scene geometry based on a deep autoencoder. This method is suitable to solve a keyframe-based monocular dense SLAM task. Liu, Mo, and Jiao (2019) propose a feature-based visual SLAM. They use a CNN to obtain a more robust object location information. Lu and Lu (2019) propose a SLAM approach that uses a regression CNN for pose estimation without ground truth data.

### 4.5. Exploration

As stated by Burgard, Moors, Stachniss, and Schneider (2005) the problem of exploring an environment belongs to the fundamental problems in mobile robotics and it consists basically in covering the whole environment in a minimum amount of time. Hence, the robot must keep track of the already visited areas. This task is a blend of the navigation and mapping tasks, that is, the robot has to construct a global map in order to plan their paths and to coordinate its actions. This problem has been commonly solved by using a team of robots (Ferri, Munafò, Tesei, Braca, Meyer, Pelekanakis, et al., 2017; Michel & McIsaac, 2012; Pawgasame, 2016), since the use of multiple robots is often suggested to have several advantages over single robot systems (Cao, Fukunaga, Kahng, & Meng, 1995).

During the past few years, a wide number of works about exploration have been proposed by using visual sensors and AI techniques. For example, Krishnan and Krishna (2010) present a vision based exploration algorithm that invokes semantic cues for constructing a hybrid map. The approach proposes semantic labeling of the input images through a probabilistic SVM classifier that runs over a BoWs. The objective is to provide the robot with hybrid understanding of its surroundings from the lower metric characterizations to higher semantic recognition. Mukhija, Tourani, and Krishna (2012) proposes a segmentation and classification method based on visual information to support the laser to solve the obstacle avoidance task. The classification is addressed with a Gaussian Mixture Models algorithm. Craye, Filliat, and Goudou (2015) propose a method for incrementally learning a mechanism of visual saliency. The proposed system is trained and learns the visual aspect of salient elements within their context. The RGB-D data are used to train a random forest classifier, which learns to determine whether the area is salient or not. Tai and Liu (2016) introduce a reinforcement learning method to address the exploration task in a corridor environment. The learning model receives information from a CNN fed with RGB-D data. The Q-network is used in the robot controller.

More recently, Choudhury et al. (2017) propose an algorithm which trains a policy to gather information to carry out a exploration task with UAVs. When the distribution corresponds to a scene containing ladders, the learned policy executes a helical motion around parts of the observed. On the contrary, when the distribution corresponds to a scene from a construction site, the learned policy executes a large sweeping motion. The proposed algorithm is based on a classifier among other tools. Liu et al. (2017) introduce an end-to-end learning model based CNN that converts directly the raw visual data to steering commands.

**Table 2**
Summary of works that use AI together with vision systems to solve mapping, localization, navigation,SLAM or exploration tasks.

| Reference | Task | Type of image | AI tool | Input data |
|---|---|---|---|---|
| Zivkovic et al. (2005) | 4.1 | Omnidirectional | Clustering | Local features |
| Peretroukhin et al. (2017) | 4.1 | Stereo | BCNN, Monte Carlo | Local features |
| Clark et al. (2017) | 4.1, 4.2 | Monocular | CNN, LSTM | Holistic descriptors |
| Kuse et al. (2017) | 4.1 | Monocular | Regression CNN | Raw images |
| Naseer and Burgard (2017) | 4.1, 4.2 | Monocular | Regression CNN | Raw images |
| Walch et al. (2017) | 4.1, 4.2 | Monocular | CNN, LSTM | Raw images |
| Iyer et al. (2018) | 4.1 | Monocular | CNN, LSTM | Holistic descriptors |
| Brahmbhatt et al. (2018) | 4.1 | Monocular | Regression CNN | Raw images |
| Sinha et al. (2018) | 4.1, 4.2 | Monocular | Regression CNN | Raw images |
| Kopitkov and Indelman (2018) | 4.1, 4.2 | Monocular | CNN, Gaussian prametrization | Holistic descriptors |
| Moolan-Feroze et al. (2019) | 4.1, 4.2 | Monocular | CNN | Raw images |
| Sarlin et al. (2019) | 4.1, 4.2 | Monocular | k-nearest neighbors, CNN | Local features, holistic descriptors |
| Kendall et al. (2015) | 4.2 | Monocular | CNN | Raw images |
| Neto (2015) | 4.2 | Monocular | Classifier, Kohonen SOM | Local features, holistic descriptors |
| Sünderhauf et al. (2015) | 4.2 | Monocular | CNN | Holistic descriptors |
| Li et al. (2018) | 4.2 | Monocular | Regression CNN | Raw images |
| Cascianelli et al. (2017) | 4.1, 4.2 | Monocular | CNN | Local features |
| Sommer et al. (2017) | 4.2 | Monocular | Regression CNN | Raw images |
| Meng et al. (2017) | 4.2 | Monocular | Random forests | Local features |
| Unicomb et al. (2018) | 4.2 | Monocular | CNN | Local features |
| Moolan-Feroze and Calway (2018) | 4.2 | Monocular | Recurrent neural network | Local features |
| Holliday and Dudek (2018) | 4.2 | Monocular | CNN | Local features |
| Cattaneo et al. (2019) | 4.2 | Stereo | Regression CNN | Raw RGB-D data |
| Cebollada, Payá and Valiente et al. (2019) | 4.2 | Omnidirecional | Autoencoder, CNN | Holistic descriptors |
| Weinzaepfel et al. (2019) | 4.2 | | Regression CNN | Local features |
| Wu and Qin (2011) | 4.4 | Omnidirectional | Incremental landmark appearance learning | Raw images |
| Lu et al. (2015) | 4.4 | Monocular | Multi-task learning | Local features |
| Garg et al. (2016) | 4.4 | Panoramic | Autoencoder | Raw images |
| Schmidt et al. (2017) | 4.4 | Monocular | CNN | Local features |
| Gao and Zhang (2017) | 4.4 | Monocular | Autoencoder | Local features |
| Tateno et al. (2017) | 4.4 | Monocular | CNN | Raw images |
| Mukasa et al. (2017) | 4.4 | Monocular | CNN | Raw images |
| Tang et al. (2017) | 4.4 | stereo | CNN | Raw images |
| Zhang, Su et al. (2017) | 4.4 | Monocular | CNN | Raw images |
| Milz et al. (2018) | 4.4 | Monocular | CNN | Raw images |
| Zhong et al. (2018) | 4.4 | Monocular | SSD | Raw RGB-D data |
| Liang et al. (2018) | 4.3, 4.4 | Panoramic | CNN | Raw images |
| Bloesch et al. (2018) | 4.4 | Monocular | Autoencoder | Raw images |
| Liu et al. (2019) | 4.4 | Monocular | CNN | Local features |
| Lu and Lu (2019) | 4.4 | Monocular | Recurrent CNN | Raw images |
| Maier et al. (2011) | 4.3 | Monocular | Classifier | HSV and texture data |
| Tapu et al. (2013) | 4.3 | Monocular | SVM classifier | HOG and BoW |
| Giusti et al. (2015) | 4.3 | Monocular | CNN | Raw images |
| Mancini et al. (2016) | 4.3 | Monocular | Autoencoder | Raw images and Optical Flow |
| Deepika and Variyar (2017) | 4.3 | Monocular | Autoencoder | Raw images |
| Yang et al. (2017) | 4.1, 4.3 | Monocular | CNN | Raw images |
| Smolyanskiy et al. (2017) | 4.3 | Monocular | Recurrent neural network | Raw images |
| Puthussery et al. (2017) | 4.3 | Monocular | CNN | Raw RGB-D data |
| Richter and Roy (2017) | 4.3 | Monocular | CNN and autoencoder | Raw images |
| Walker et al. (2017) | 4.3 | Panoramic | Autoencoder | Raw images |
| Anderson et al. (2018) | 4.3 | Monocular | Reinforcement learning | Raw images |
| Zhao et al. (2018) | 4.3 | Monocular | CNN | Raw images, local features |
| Hui et al. (2018) | 4.3 | Monocular | CNN | Raw images |
| Mansouri et al. (2019) | 4.3 | Monocular | CNN | Raw images |
| Ma et al. (2019) | 4.3 | Monocular | Reinforcement learning, autoencoder | Raw images |
| Ruan et al. (2019) | 4.3 | Monocular | Deep reinforcement learning | Raw RGB-D data |
| Krishnan and Krishna (2010) | 4.5 | Monocular | SVM classifier | BoW |
| Mukhija et al. (2012) | 4.5 | Monocular | Gaussian mixture models classifier | Raw images |
| Craye et al. (2015) | 4.5 | Monocular | Random forest classifier | Local features |
| Tai and Liu (2016) | 4.5 | Monocular | Deep reinforcement learning, CNN | Raw RGB-D data |
| Choudhury, Kapoor, Ranade, and Dey (2017) | 4.5 | Monocular | Classifier | Local features |
| Liu et al. (2017) | 4.5 | Monocular | CNN | Raw images |
| Tai, Li and Liu (2017) | 4.5 | Monocular | CNN | Raw RGB-D data |
| Flaspohler, Roy, and Girdhar (2017) | 4.5 | Monocular | Autoencoder | Holistic descriptors |
| Wang et al. (2018) | 4.5 | Monocular | Autoencoder and Classifier | Holistic descriptors |
| Ly and Tsai (2019) | 4.5 | Monocular | CNN | Raw RGB-D data |

Tai, Li et al. (2017) propose an exploration algorithm which uses a hierarchical structure that fuses several CNN layers with decision-making process. The system is trained by taking RGB-D information as input and generates a sequence of main moving direction as output. Flaspohler et al. (2017) present an autoencoder which encodes information from visual data to carry out exploration with marine robots. Wang et al. (2018) propose an optimal light intensity optimization method to address efficiently the visual navigation. The proposed method is mainly based on a regression model to automatically predict optimal light intensity values for desired image quality when camera observation distances fluctuate. Nevertheless, a classification task is addressed rather than a regression. The query image is classified according to five levels of brightness. In this framework, simple features are extracted using intensity histogram and utilized as primary features to describe

the distribution of image intensity. After that, the primary features are made more discriminative by an evolution process with stacked autoencoders. The evaluation of brightness is solved by introducing the holistic descriptor into a softmax classifier. Ly and Tsai (2019) propose an autonomous exploration, reconstruction, and surveillance of 3D Environments by using deep learning. They establish a gain function for each issue. After that, they use CNNs to approximate the corresponding gain function.

To conclude, Table 2 presents an outline of the approaches presented in the present section and their main characteristics: the type of task (Sections Section 4.1, 4.2, 4.3, 4.4, or 4.5), the type of image, the AI tool used and the kind of visual data employed (local features, holistic descriptors or raw data).

## 5. Conclusions

Vision sensors constitute a robust alternative to capture the necessary information to solve a variety of tasks in the field of mobile robotics. Additionally, during the past few years, systems based on AI have been used extensively to carry out the tasks more efficiently. Consequently, the amount of works that use visual sensors and AI has increased substantially and many approaches can be found to solve the mapping, localization, navigation, SLAM and exploration tasks.

The present review presents a collection of the main proposals to solve the mobile robotics tasks by means of visual information and AI tools. To this end, this work started focusing on the AI tools which are more commonly used together with vision systems. After that, the review has focused on how visual information can be described and handled by means of AI techniques. Two main options are available: local features and methods based on holistic description. Finally, the present work has focused on the study of the mapping, localization, navigation, SLAM and exploration tasks in mobile robotics.

The huge amount of works regarding these topics show how vision systems, AI techniques and mobile robotics are three very active research areas and hence, the research on them is expected to continue increasing during the following years. This work has shown that a great variety of AI and deep learning tools can be used in mobile robotics and computer vision. Such tools have provided good solutions to some specific problems in these fields, such as the extraction and labeling of relevant information from the scenes; the creation of models of the environment and the estimation of the position and orientation of the robot from raw data; and the exploration of initially unknown environments. Good solutions to these problems have been proposed in specific scenarios, depending on the motion abilities of the robot and the characteristics of the surroundings (indoors–outdoors, aerial–terrestrial–underwater, etc.). Notwithstanding that, there are still some issues that need to be addressed more robustly to enable mobile robots to move and perform their tasks more autonomously in complex, heterogeneous and changing environments and circumstances, trying to provide a complete and more integral solution to the SLAM and navigation problems, and AI shows potential to address these challenges. In this sense, finding robust and fast solutions to overcome current problems will help to improve the autonomy of mobile robots. Therefore, their range of use will also increase. Nowadays, there are some technologies which are closely related to AI and can be of interest to researchers in the fields of mobile robotics and computer vision. As detailed in the survey, one of the major issues of AI and deep learning is the computationally expensive process to train the models with a large number of samples. In this sense, some current technologies, such as cloud computing, data science and big data provide robust tools and approaches to address this issue, and therefore they may contribute to a quicker development of AI techniques in mobile robotics (Allam & Dhunny, 2019; Gill et al., 2019; Zhu & Zheng, 2018). Second, as shown throughout the review, AI techniques are contributing to a major autonomy of the mobile robots in a wider variety of environments and circumstances. This increase of autonomy plays a crucial role in the development of other relevant technologies to current society, such as IoT (Internet of Things), smart cities and industry 5.0 (Chui, Lytras, & Visvizi, 2018; Özdemir & Hekim, 2018; Singh, Rathore, & Park, 2020).

The following abbreviations are used in this manuscript:

| | |
|---|---|
| AI | Artificial intelligence |
| BCNN | Bayesian Convolutional Neural Network |
| BoF | Bag of Features |
| BoW | Bag of Words |
| BRIEF | Binary Robust Independent Elementary Features |
| CCD | Charge-Coupled Device |
| CCTV | Closed-Circuit Television |
| CenSurE | Center Surround Extremas |
| CNN | Convolutional Neural Network |
| D3QN | Dueling Architecture based double deep Q Network |
| DAN | Deep Adaption Network |
| DARPA | Defense Advanced Research Projects Agency |
| DBM | Deep Boltzmann Machines |
| DBN | Deep Belief Networks |
| EEG | Electroencephalography |
| EKF | Extended Kalman Filter |
| FAST | Features From Accelerated Segment Test |
| FREAK | Fast Retina Keypoint |
| HLC | High-level controller |
| HOG | Histogram of Oriented Gradients |
| HP-CNN | Hypercube Pyramid Convolutional Neural Network |
| PCA | Principal Components Analysis |
| ICA | Independent Component Analysis |
| IoT | Internet of Things |
| IP | Interest point |
| LBP | Local Binary Pattern |
| LIDAR | Laser Imaging Detection and Ranging |
| LSTM | Long short-term memory |
| MAV | Micro Aerial Vehicles |
| ML | Machine Learning |
| MLP | Multilayer Perceptrons |
| MOP-CNN | Multi-Scale Orderless Pooling Convolutional Neural Network |
| MSE | Mean Squared Error |
| ORB | Oriented FAST and rotated BRIEF |
| PI | Proportional–integral |
| RBM | Restricted Boltzmann Machines |
| RF | Regression Forest |
| R-CNN | Regression Convolutional Neural Network |
| RNN | Recurrent Neural Network |
| ROI | Region Of Interest |
| RVM | Relevance Vector Machine |
| SDA | Stacked Denoising Autoencoder |
| SIFT | Scale-Invariant Feature Transform |
| SLAM | Simultaneous Localization And Mapping |
| SOM | Self-Organizing Maps |
| SSD | Single Shot multi-box object Detector |
| SURF | Speeded-Up Robust Features |
| SVM | Support Vector Machine |
| t-SNE | t-distributed Stochastic Neighbor Embedding |
| UAV | Unmanned Aerial Vehicle |
| VLAD | Vector of Locally Aggregated Descriptors |
| WSN | Wireless Sensor Network |

## References

Abate, A. F., Nappi, M., Riccio, D., & Sabatino, G. (2007). 2d and 3d face recognition: A survey. *Pattern Recognition Letters*, *28*(14), 1885–1906.

(2014). Accelerating t-SNE using tree-based algorithms, author=van der Maaten, L.. *Journal of Machine Learning Research*, *15*(1), 3221–3245.

Aguilar, W. G., Luna, M. A., Moya, J. F., Abad, V., Parra, H., & Ruiz, H. (2017). Pedestrian detection for UAVs using cascade classifiers with meanshift. In *2017 IEEE 11th international conference on semantic computing (ICSC)* (pp. 509–514). IEEE.

Ahuja, K., Krishnan, K. V. A., Kiran, A., Dalin, E., & Sagar, S. (2018). Smart office surveillance robot using face recognition. *International Journal of Mechanical and Production Engineering Research and Development*, *8*, 725–734. http://dx.doi.org/10.24247/ijmperdjun201877.

Allam, Z., & Dhunny, Z. A. (2019). On big data, artificial intelligence and smart cities. *Cities*, *89*, 80–91.

Amorós, F., Payá, L., Marín, J. M., & Reinoso, O. (2018). Trajectory estimation and optimization through loop closure detection, using omnidirectional imaging and global-appearance descriptors. *Expert Systems with Applications*, *102*, 273–290.

Anderson, P., Wu, Q., Teney, D., Bruce, J. J. M., Sünderhauf, N., Reid, I., et al. (2018). Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3674–3683).

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5297–5307).

Arroyo, R., Alcantarilla, P. F., Bergasa, L. M., & Romera, E. (2016). Fusion and binarization of CNN features for robust topological localization across seasons. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4656–4663). http://dx.doi.org/10.1109/IROS.2016.7759685.

Atkinson, J., & Campos, D. (2016). Improving BCI-based emotion recognition by combining EEG feature selection and kernel classifiers. *Expert Systems with Applications*, *47*, 35–41.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., et al. (2019). Self-driving cars: A survey. arXiv preprint arXiv:1901.04407.

Barber, R., Crespo, J., Gómez, C., Hernámdez, A. C., & Galli, M. (2018). Mobile robot navigation in indoor environments: Geometric, topological, and semantic navigation. In *Applications of mobile robots*. IntechOpen.

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, *110*(3), 346–359.

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*(6), 1373–1396.

Bell, A. J., & Sejnowski, T. J. (1997). The "independent components" of natural scenes are edge filters. *Vision Research*, *37*(23), 3327–3338.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Berenguer, Y., Payá, L., Valiente, D., Peidró, A., & Reinoso, O. (2019). Relative altitude estimation using omnidirectional imaging and holistic descriptors. *Remote Sensing*, *11*(3), 323.

Bilgili, M., & Sahin, B. (2010). Comparative analysis of regression and artificial neural network models for wind speed prediction. *Meteorology and Atmospheric Physics*, *109*(1–2), 61–72.

Bishop, C. M. (2006). *Pattern recognition and machine learning springer-verlag New York, Vol. 2006*.

Bloesch, M., Czarnowski, J., Clark, R., Leutenegger, S., & Davison, A. J. (2018). CodeSLAM—learning a compact, optimisable representation for dense visual SLAM. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2560–2568).

Boularias, A., Bagnell, J. A., & Stentz, A. (2015). Learning to manipulate unknown objects in clutter by reinforcement. In *Twenty-Ninth AAAI conference on artificial intelligence*.

Brahmbhatt, S., Gu, J., Kim, K., Hays, J., & Kautz, J. (2018). Geometry-aware learning of maps for camera localization. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 2616–2625). http://dx.doi.org/10.1109/CVPR.2018.00277.

Brendel, W., & Bethge, M. (2019). Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. arXiv preprint arXiv:1904.00760.

Burgard, W., Moors, M., Stachniss, C., & Schneider, F. E. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, *21*(3), 376–386.

Calderon-Cordova, C., Ramírez, C., Barros, V., Quezada-Sarmiento, P. A., & Barba-Guamán, L. (2016). EMG signal patterns recognition based on feedforward artificial neural network applied to robotic prosthesis myoelectric control. In *2016 future technologies conference (FTC)* (pp. 868–875). IEEE.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010* (pp. 778–792). Springer.

Cao, Y. U., Fukunaga, A. S., Kahng, A. B., & Meng, F. (1995). Cooperative mobile robotics: Antecedents and directions. In *Proceedings 1995 IEEE/RSJ international conference on intelligent robots and systems. human robot interaction and cooperative robots, Vol. 1* (pp. 226–234). IEEE.

Cao, J., Huang, Z., & Shen, H. T. (2017). Local deep descriptors in bag-of-words for image retrieval. In *Proceedings of the on thematic workshops of ACM multimedia 2017* (pp. 52–58). ACM.

Carreira, F., Calado, J. M., Cardeira, C., & Oliveira, P. (2015). Enhanced PCA-based localization using depth maps with missing data. *Journal of Intelligent and Robotic Systems*, *77*(2), 341–360.

Cascianelli, S., Costante, G., Bellocchio, E., Valigi, P., Fravolini, M. L., & Ciarfuglia, T. A. (2017). Robust visual semi-semantic loop closure detection by a covisibility graph and CNN features. *Robotics and Autonomous systems*, *92*, 53–65. http://dx.doi.org/10.1016/j.robot.2017.03.004, Retrieved from http://www.sciencedirect.com/science/article/pii/S0921889016304900.

Cattaneo, D., Vaghi, M., Ballardini, A. L., Fontana, S., Sorrenti, D. G., & Burgard, W. (2019). CMRNet: Camera to LiDAR-Map registration. In *2019 IEEE intelligent transportation systems conference (ITSC)* (pp. 1283–1289). http://dx.doi.org/10.1109/ITSC.2019.8917470.

Cebollada, S., Payá, L., Flores, M., Román, V., Peidró, A., & Reinoso, O. (2020). A deep learning tool to solve localization in mobile autonomous robotics. In *ICINCO 2020, 17th Intl. Conf. on informatics in control, automation and robotics (Online Streaming, 7-9 July 2020)* (pp. 232–241). Ed. INSTICC.

Cebollada, S., Payá, L., Juliá, M., Holloway, M., & Reinoso, O. (2018). Mapping and localization module in a mobile robot for insulating building crawl spaces. *Automation in Construction*, *87*, 248–262. http://dx.doi.org/10.1016/j.autcon.2017.11.007, Retrieved from http://www.sciencedirect.com/science/article/pii/S0926580517306726.

Cebollada, S., Payá, L., Mayol, W., & Reinoso, O. (2019). Evaluation of clustering methods in compression of topological models and visual place recognition using global appearance descriptors. *Applied Sciences*, *9*(3), 377.

Cebollada, S., Payá, L., Román, V., & Reinoso, O. (2019). Hierarchical localization in topological models under varying illumination using holistic visual descriptors. *IEEE Access*, *7*, 49580–49595. http://dx.doi.org/10.1109/ACCESS.2019.2910581.

Cebollada, S., Payá, L., Valiente, D., Jiang, X., & Reinoso, O. (2019). An evaluation between global appearance descriptors based on analytic methods and deep learning techniques for localization in autonomous mobile robots. In *ICINCO 2019, 16th international conference on informatics in control, automation and robotics (Prague, Czech Republic, 29-31 July, 2019)* (pp. 284–291). Ed. INSTICC.

Charniak, E., McDermott, D., & McDermott, D. (1985). Introduction to artificial intelligence. In *Addison-Wesley series in computer science and information processing*, Addison-Wesley, Retrieved from https://books.google.es/books?id=kPCAR2VEd1YC.

Chaves, D., Ruiz-Sarmiento, J., Petkov, N., & Gonzalez-Jimenez, J. (2019). Integration of CNN into a robotic architecture to build semantic maps of indoor environments. In *International work-conference on artificial neural networks* (pp. 313–324). Springer.

Chollet, F. (2017). *Deep learning with Python*. Manning Publications Company.

Choudhury, S., Kapoor, A., Ranade, G., & Dey, D. (2017). Learning to gather information via imitation. In *2017 IEEE international conference on robotics and automation (ICRA)* (pp. 908–915). IEEE.

Chui, K. T., Lytras, M. D., & Visvizi, A. (2018). Energy sustainability in smart cities: Artificial intelligence, smart monitoring, and optimization of energy consumption. *Energies*, *11*(11), 2869.

Clark, R., Wang, S., Markham, A., Trigoni, N., & Wen, H. (2017). VidLoc: A deep spatio-temporal model for 6-DoF video-clip relocalization. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2652–2660). http://dx.doi.org/10.1109/CVPR.2017.284.

Coates, A., Ng, A., & Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 215–223).

Craye, C., Filliat, D., & Goudou, J. F. (2015). Exploration strategies for incremental learning of object-based visual saliency. In *2015 joint IEEE international conference on development and learning and epigenetic robotics (ICDL-EpiRob)* (pp. 13–18). IEEE.

Csurka, G., Dance, C., Fan, L., Willamowski, J., & Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV, Vol. 1* (pp. 1–2). Prague.

da Silva, S. P., da Nóbrega, R. V. M., Medeiros, A. G., Marinho, L. B., Almeida, J. S., & Filho, P. P. R. (2018). Localization of mobile robots with topological maps and classification with reject option using convolutional neural networks in omnidirectional images. In *2018 international joint conference on neural networks (IJCNN)* (pp. 1–8). http://dx.doi.org/10.1109/IJCNN.2018.8489328.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition, San Diego, USA. Vol. II* (pp. 886-893).

De Momi, E., & Ferrigno, G. (2010). Robotic and artificial intelligence for keyhole neurosurgery: the robocast project, a multi-modal autonomous path planner. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, *224*(5), 715–727.

Deepika, N., & Variyar, V. (2017). Obstacle classification and detection for vision based navigation for autonomous driving. In *2017 international conference on advances in computing, communications and informatics (ICACCI)* (pp. 2092–2097). IEEE.

Dezfoulian, S. H., Wu, D., & Ahmad, I. S. (2013). A generalized neural network approach to mobile robot navigation and obstacle avoidance. In *Intelligent autonomous systems 12* (pp. 25–42). Springer.

Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, *54*, 764–771.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., et al. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning* (pp. 647–655).

Donoho, D., & Grimes, C. (2003). *Hessian eigenmaps: new locally linear embedding techniques for highdimensional data (Technical Report TR2003-08)*. Stanford.

Faessler, M., Fontana, F., Forster, C., Mueggler, E., Pizzoli, M., & Scaramuzza, D. (2016). Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, *33*(4), 431–450.

Fan, H., Zheng, L., Yan, C., & Yang, Y. (2018). Unsupervised person re-identification: Clustering and fine-tuning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, *14*(4), 83.

Feng, J., Liu, Y., & Wu, L. (2017). Bag of visual words model with deep spatial features for geographical scene classification. *Computational Intelligence and Neuroscience*, *2017*.

Ferreira, J. P., Amaral, T. G., Pires, V. F., Crisostomo, M. M., & Coimbra, P. (2004). A neural-fuzzy walking control of an autonomous biped robot. In *Proceedings world automation congress, 2004., Vol. 15* (pp. 253–258). IEEE.

Ferri, G., Munafò, A., Tesei, A., Braca, P., Meyer, F., Pelekanakis, K., et al. (2017). Cooperative robotic networks for underwater surveillance: an overview. *IET Radar, Sonar & Navigation*, *11*(12), 1740–1761.

Filliat, D., & Meyer, J. A. (2003). Map-based navigation in mobile robots:: I. a review of localization strategies. *Cognitive Systems Research*, *4*(4), 243–282. http://dx.doi.org/10.1016/S1389-0417(03)00008-1, Retrieved from http://www.sciencedirect.com/science/article/pii/S1389041703000081.

Flaspohler, G., Roy, N., & Girdhar, Y. (2017). Feature discovery and visualization of robot mission data using convolutional autoencoders and Bayesian nonparametric topic models. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1–8). IEEE.

François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, *11*(3–4), 219–354.

Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory* (pp. 23–37). Springer.

Fuentes-Pacheco, J., Ruiz-Ascencio, J., & Rendón-Mancha, J. M. (2015). Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, *43*(1), 55–81.

Gadoue, S. M., Giaouris, D., & Finch, J. (2009). Artificial intelligence-based speed control of DTC induction motor drives—A comparative study. *Electric Power Systems Research*, *79*(1), 210–219.

Gao, H., Cheng, B., Wang, J., Li, K., Zhao, J., & Li, D. (2018). Object classification using cnn-based fusion of vision and lidar in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics*, *14*(9), 4224–4231.

Gao, X., & Zhang, T. (2017). Unsupervised learning to detect loops using deep neural networks for visual SLAM system. *Autonomous Robots*, *41*(1), 1–18.

Garcia-Fidalgo, E., & Ortiz, A. (2015). Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, *64*, 1–20. http://dx.doi.org/10.1016/j.robot.2014.11.009, Retrieved from http://www.sciencedirect.com/science/article/pii/S0921889014002619.

Garg, R., BG, V. K., Carneiro, G., & Reid, I. (2016). Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European conference on computer vision* (pp. 740–756). Springer.

Gill, S. S., Tuli, S., Xu, M., Singh, I., Singh, K. V., Lindsay, D., et al. (2019). Transformative effects of IoT, Blockchain and artificial intelligence on cloud computing: Evolution, vision, trends and open challenges. *Internet of Things*, *8*, Article 100118.

Giusti, A., Guzzi, J., Cireşan, D. C., He, F. L., Rodríguez, J. P., Fontana, F., et al. (2015). A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, *1*(2), 661–667.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 513–520).

Gong, Y., Wang, L., Guo, R., & Lazebnik, S. (2014). Multi-scale orderless pooling of deep convolutional activation features. In *European conference on computer vision* (pp. 392–407). Springer.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2016). Deep image retrieval: Learning global representations for image search. In *European conference on computer vision* (pp. 241–257). Springer.

Graves, A. (2012). Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks* (pp. 5–13). Springer.

Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing* (pp. 6645–6649). IEEE.

Gwinner, K., Jaumann, R., Hauber, E., Hoffmann, H., Heipke, C., Oberst, J., et al. (2016). The high resolution stereo camera (HRSC) of mars express and its approach to science analysis and mapping for mars and its satellites. *Planetary and Space Science*, *126*, 93–138.

Hadid, A., Heikkila, J., Silvén, O., & Pietikainen, M. (2007). Face and eye detection for person authentication in mobile phones. In *2007 first ACM/IEEE international conference on distributed smart cameras* (pp. 101–108). IEEE.

Han, D., Liu, Q., & Fan, W. (2018). A new image classification method using CNN transfer learning and web data augmentation. *Expert Systems with Applications*, *95*, 43–56.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hinton, G. E., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Hinton, G. E., Sejnowski, T. J., & Poggio, T. A. (1999). *Unsupervised learning: foundations of neural computation*. MIT press.

Holliday, A., & Dudek, G. (2018). Scale-robust localization using general object landmarks. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1688–1694). http://dx.doi.org/10.1109/IROS.2018.8594011.

Holzer, S., Shotton, J., & Kohli, P. (2012). Learning to efficiently detect repeatable interest points in depth data. In *European conference on computer vision* (pp. 200–213). Springer.

Hui, X., Bian, J., Zhao, X., & Tan, M. (2018). Vision-based autonomous navigation approach for unmanned aerial vehicle transmission-line inspection. *International Journal of Advanced Robotic Systems*, *15*(1), Article 1729881417752821.

Iyer, G., Murthy, J. K., Gupta, G., Krishna, K. M., & Paull, L. (2018). Geometric consistency for self-supervised end-to-end visual odometry. In *2018 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW)* (pp. 380–3808). http://dx.doi.org/10.1109/CVPRW.2018.00064.

Jafri, R., & Arabnia, H. R. (2009). A survey of face recognition techniques.. *Jips*, *5*(2), 41–68.

Jain, A. K., & Li, S. Z. (2011). *Handbook of face recognition*. Springer.

Jia, Y., Li, M., An, L., & Zhang, X. (2003). Autonomous navigation of a miniature mobile robot using real-time trinocular stereo machine. In *Robotics, intelligent systems and signal processing, 2003. proceedings. 2003 IEEE international conference on, Vol. 1* (pp. 417–421). http://dx.doi.org/10.1109/RISSP.2003.1285610.

Jiang, W., & Wang, W. (2017). Face detection and recognition for home service robots with end-to-end deep neural networks. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 2232–2236). http://dx.doi.org/10.1109/ICASSP.2017.7952553.

Jiménez, P. (2012). Survey on model-based manipulation planning of deformable objects. *Robotics and Computer-Integrated Manufacturing*, *28*(2), 154–163.

Jurie, F., & Triggs, B. (2005). Creating efficient codebooks for visual recognition. In *Tenth IEEE international conference on computer vision (ICCV'05) Volume 1, Vol. 1* (pp. 604–610). IEEE.

Jutten, C., & Herault, J. (1991). Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, *24*(1), 1–10.

Kahn, G., Villaflor, A., Ding, B., Abbeel, P., & Levine, S. (2018). Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE international conference on robotics and automation (ICRA)* (pp. 1–8). IEEE.

Kanezaki, A., Matsushita, Y., & Nishida, Y. (2018). Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5010–5019).

Kayaer, K., & Yıldırım, T. (2003). Medical diagnosis on Pima Indian diabetes using general regression neural networks. In *Proceedings of the international conference on artificial neural networks and neural information processing (ICANN/ICONIP) Vol. 181* (p. 184).

Kendall, A., Grimes, M., & Cipolla, R. (2015). PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *2015 IEEE international conference on computer vision (ICCV)* (pp. 2938–2946). http://dx.doi.org/10.1109/ICCV.2015.336.

Khan, F. S., Van De Weijer, J., Anwer, R. M., Bagdanov, A. D., Felsberg, M., & Laaksonen, J. (2018). Scale coding bag of deep features for human attribute and action recognition. *Machine Vision and Applications*, *29*(1), 55–71.

Kim, K. (2005). Intelligent immigration control system by using passport recognition and face verification. In *International symposium on neural networks* (pp. 147–156). Springer.

Kingma, D. P., & Cun, Y. L. (2010). Regularized estimation of image statistics by score matching. In *Advances in neural information processing systems* (pp. 1126–1134).

Kirby, M. (2000). *Geometric data analysis: an empirical approach to dimensionality reduction and the study of patterns*. John Wiley & Sons, Inc..

Kohavi, R., & Quinlan, J. R. (2002). Data mining tasks and methods: Classification: decision-tree discovery. In *Handbook of data mining and knowledge discovery* (pp. 267–276). Oxford University Press, Inc..

Kopitkov, D., & Indelman, V. (2018). Bayesian information recovery from CNN for probabilistic inference. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 7795–7802). http://dx.doi.org/10.1109/IROS.2018.8594506.

Korrapati, H., & Mezouar, Y. (2017). Multi-resolution map building and loop closure with omnidirectional images. *Autonomous Robots*, *41*(4), 967–987.

Korytkowski, M., Rutkowski, L., & Scherer, R. (2016). Fast image classification by boosting fuzzy classifiers. *Information Sciences*, *327*, 175–182.

Krishnan, A. K., & Krishna, K. M. (2010). A visual exploration algorithm using semantic cues that constructs image based hybrid maps. In *2010 IEEE/RSJ international conference on intelligent robots and systems* (pp. 1316–1321). IEEE.

Krittanawong, C., Zhang, H., Wang, Z., Aydar, M., & Kitai, T. (2017). Artificial intelligence in precision cardiovascular medicine. *Journal of the American College of Cardiology*, *69*(21), 2657–2664.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Kuipers, B., Modayil, J., Beeson, P., MacMahon, M., & Savelli, F. (2004). Local metrical and global topological maps in the hybrid spatial semantic hierarchy. In *IEEE international conference on robotics and automation, 2004. proceedings. ICRA'04. 2004, Vol. 5* (pp. 4845–4851). IEEE.

Kumar, R., Aggarwal, R., & Sharma, J. (2015). Comparison of regression and artificial neural network models for estimation of global solar radiations. *Renewable & Sustainable Energy Reviews*, *52*, 1294–1299.

Kunii, Y., Kovacs, G., & Hoshi, N. (2017). Mobile robot navigation in natural environments using robust object tracking. In *2017 IEEE 26th international symposium on industrial electronics (ISIE)* (pp. 1747–1752). IEEE.

Kuse, M., Jaiswal, S. P., & Shen, S. (2017). Deep-mapnets : A residual network for 3D environment representation. In *2017 IEEE international conference on image processing (ICIP)* (pp. 2652–2656). http://dx.doi.org/10.1109/ICIP.2017.8296763.

Kuutti, S., Fallah, S., Katsaros, K., Dianati, M., Mccullough, F., & Mouzakitis, A. (2018). A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. *IEEE Internet of Things Journal*, *5*(2), 829–846.

Larochelle, H., Bengio, Y., Louradour, J., & Lamblin, P. (2009). Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, *10*(Jan), 1–40.

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06), Vol. 2* (pp. 2169–2178). IEEE.

Le, Q. V. (2013). Building high-level features using large scale unsupervised learning. In *2013 IEEE international conference on acoustics, speech and signal processing* (pp. 8595–8598). IEEE.

Le, Q. V., Zou, W. Y., Yeung, S. Y., & Ng, A. Y. (2011). Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR 2011* (pp. 3361–3368). IEEE.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lenc, K., & Vedaldi, A. (2016). Learning covariant feature detectors. In *European conference on computer vision* (pp. 100–117). Springer.

Lenc, K., & Vedaldi, A. (2018). Large scale evaluation of local image feature detectors on homography datasets. arXiv preprint arXiv:1807.07939.

Levitt, T. S., & Lawton, D. T. (1990). Qualitative navigation for mobile robots. *Artificial Intelligence*, *44*(3), 305–360.

Li, T., Chang, X., Wu, Z., Li, J., Shao, G., Deng, X., et al. (2017). Autonomous collision-free navigation of microvehicles in complex and dynamically changing environments. *ACS Nano*, *11*(9), 9268–9275.

Li, R., Liu, Q., Gui, J., Gu, D., & Hu, H. (2018). Indoor relocalization in challenging environments with dual-stream convolutional neural networks. *IEEE Transactions on Automation Science and Engineering*, *15*(2), 651–662. http://dx.doi.org/10.1109/TASE.2017.2664920.

Liang, C., Tie, Y., Qi, L., & Bi, C. (2018). Deep ViDAR: CNN based 360° panoramic video system for outdoor robot visual navigation and SLAM. In *Tenth international conference on digital image processing (ICDIP 2018), Vol. 10806*. International Society for Optics and Photonics, Article 1080663.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., et al. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Liu, W., Mo, Y., & Jiao, J. (2019). An efficient edge-feature constraint visual SLAM. In *Proceedings of the international conference on artificial intelligence, information processing and cloud computing* (pp. 1–7).

Liu, C., Zheng, B., Wang, C., Zhao, Y., Fu, S., & Li, H. (2017). CCNN-based vision model for obstacle avoidance of mobile robot. In *MATEC web of conferences, Vol. 139* (p. 00007). EDP Sciences.

Loncomilla, P., Ruiz-del Solar, J., & Martínez, L. (2016). Object recognition using local invariant features for robotic applications: A survey. *Pattern Recognition*, *60*, 499–514.

Long, M., Cao, Y., Wang, J., & Jordan, M. I. (2015). Learning transferable features with deep adaptation networks. arXiv preprint arXiv:1502.02791.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

Lu, Y., & Lu, G. (2019). Deep unsupervised learning for simultaneous visual odometry and depth estimation. In *2019 IEEE international conference on image processing (ICIP)* (pp. 2571–2575). http://dx.doi.org/10.1109/ICIP.2019.8803247.

Lu, G., Yan, Y., Ren, L., Song, J., Sebe, N., & Kambhamettu, C. (2015). Localize me anywhere, anytime: A multi-task point-retrieval approach. In *2015 IEEE international conference on computer vision (ICCV)* (pp. 2434–2442). http://dx.doi.org/10.1109/ICCV.2015.280.

Ly, L., & Tsai, Y. R. (2019). Autonomous exploration, reconstruction, and surveillance of 3d environments aided by deep learning. In *2019 international conference on robotics and automation (ICRA)* (pp. 5467–5473). IEEE.

Ma, L., Chen, J., & Liu, Y. (2019). Using RGB image as visual input for mapless robot navigation. arXiv preprint arXiv:1903.09927.

Ma, X., Wang, H., & Geng, J. (2016). Spectral–spatial classification of hyperspectral image based on deep auto-encoder. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *9*(9), 4073–4085.

Maier, D., Bennewitz, M., & Stachniss, C. (2011). Self-supervised obstacle detection for humanoid navigation using monocular vision and sparse laser data. In *2011 IEEE international conference on robotics and automation* (pp. 1263–1269). IEEE.

Mancini, M., Bulò, S. R., Ricci, E., & Caputo, B. (2017). Learning deep nbnn representations for robust place categorization. *IEEE Robotics and Automation Letters*, *2*(3), 1794–1801.

Mancini, M., Costante, G., Valigi, P., & Ciarfuglia, T. A. (2016). Fast robust monocular depth estimation for obstacle detection with fully convolutional networks. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4296–4303). IEEE.

Mansouri, S. S., Karvelis, P., Kanellakis, C., Kominiak, D., & Nikolakopoulos, G. (2019). Vision-based mav navigation in underground mine using convolutional neural network. In *IECON 2019-45th annual conference of the IEEE industrial electronics society, Vol. 1* (pp. 750–755). IEEE.

Matas, J., James, S., & Davison, A. J. (2018). Sim-to-real reinforcement learning for deformable object manipulation. arXiv preprint arXiv:1806.07851.

Mehryar, M., Rostamizadeh, A., & Talwalkar, A. (2012). Foundations of machine learning. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), Vol. 17*.

Menegatti, E., Pretto, A., Scarpa, A., & Pagello, E. (2006). Omnidirectional vision scan matching for robot localization in dynamic environments. *IEEE Transactions on Robotics*, *22*(3), 523–535. http://dx.doi.org/10.1109/TRO.2006.875495.

Meng, L., Chen, J., Tung, F., Little, J. J., Valentin, J., & de Silva, C. W. (2017). Backtracking regression forests for accurate camera relocalization. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 6886–6893). http://dx.doi.org/10.1109/IROS.2017.8206611.

Messina, R., & Louradour, J. (2015). Segmentation-free handwritten Chinese text recognition with LSTM-RNN. In *2015 13th international conference on document analysis and recognition (ICDAR)* (pp. 171–175). IEEE.

Michel, D., & McIsaac, K. (2012). New path planning scheme for complete coverage of mapped areas by single and multiple robots. In *2012 IEEE international conference on mechatronics and automation* (pp. 1233–1240). IEEE.

Michelson, R. C. (2000). Autonomous navigation. *Access Science*, http://dx.doi.org/10.1036/1097-8542.YB000130.

Milz, S., Arbeiter, G., Witt, C., Abdallah, B., & Yogamani, S. (2018). Visual slam for automated driving: Exploring the applications of deep learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 247–257).

Minaei-Bidgoli, B., Kashy, D. A., Kortemeyer, G., & Punch, W. F. (2003). Predicting student performance: an application of data mining methods with an educational web-based system. In *33rd annual frontiers in education, 2003. FIE 2003, Vol. 1* (pp. T2A–13). IEEE.

Mishkin, D., Radenovic, F., & Matas, J. (2018). Repeatability is not enough: Learning affine regions via discriminability. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 284–300).

Mitchell, T. M. (1997). *Machine learning*. McGraw-hill New York.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., et al. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Mohedano, E., McGuinness, K., O'Connor, N. E., Salvador, A., Marques, F., & Giro-i Nieto, X. (2016). Bags of local convolutional features for scalable instance search. In *Proceedings of the 2016 ACM on international conference on multimedia retrieval* (pp. 327–331). ACM.

Moolan-Feroze, O., & Calway, A. (2018). Predicting out-of-view feature points for model-based camera pose estimation. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 82–88). http://dx.doi.org/10.1109/IROS.2018.8594297.

Moolan-Feroze, O., Karachalios, K., Nikolaidis, D. N., & Calway, A. (2019). Improving drone localisation around wind turbines using monocular model-based tracking. In *2019 international conference on robotics and automation (ICRA)* (pp. 7713–7719). http://dx.doi.org/10.1109/ICRA.2019.8794156.

Mostajabi, M., Yadollahpour, P., & Shakhnarovich, G. (2015). Feedforward semantic segmentation with zoom-out features. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3376–3385).

Muhammad, N., Fofi, D., & Ainouz, S. (2009). Current state of the art of vision based SLAM. In *Image processing: Machine vision applications II, Vol. 7251* (p. 72510F). International Society for Optics and Photonics.

Mukasa, T., Xu, J., & Stenger, B. (2017). 3D scene mesh from CNN depth predictions and sparse monocular SLAM. In *Proceedings of the IEEE international conference on computer vision workshops* (pp. 921–928).

Mukherjee, D., Wu, Q. J., & Wang, G. (2015). A comparative experimental study of image feature detectors and descriptors. *Machine Vision and Applications*, *26*(4), 443–466.

Mukhija, P., Tourani, S., & Krishna, K. M. (2012). Outdoor intersection detection for autonomous exploration. In *2012 15th international IEEE conference on intelligent transportation systems* (pp. 218–223). IEEE.

Murillo, A., Guerrero, J., & Sagues, C. (2007). SURF features for efficient robot localization with omnidirectional images. In *Robotics and automation, 2007 IEEE international conference on* (pp. 3901–3907). http://dx.doi.org/10.1109/ROBOT.2007.364077.

Murillo, A. C., Singh, G., Kosecká, J., & Guerrero, J. J. (2013). Localization in urban environments using a panoramic gist descriptor. *IEEE Transactions on Robotics*, *29*(1), 146–160.

Narudin, F. A., Feizollah, A., Anuar, N. B., & Gani, A. (2016). Evaluation of machine learning classifiers for mobile malware detection. *Soft Computing*, *20*(1), 343–357.

Naseer, T., & Burgard, W. (2017). Deep regression for monocular camera-based 6-DoF global localization in outdoor environments. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1525–1530). http://dx.doi.org/10.1109/IROS.2017.8205957.

Neto, A. (2015). Short-term visual mapping and robot localization based on learning classifier systems and self-organizing maps. In *2015 IEEE intelligent vehicles symposium (IV)* (pp. 235–240). http://dx.doi.org/10.1109/IVS.2015.7225692.

Ng, J. Y. H., Yang, F., & Davis, L. S. (2015). Exploiting local features from deep networks for image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 53–61).

Ngiam, J., Chen, Z., Koh, P. W., & Ng, A. Y. (2011). Learning deep energy models.

Noh, H., Araujo, A., Sim, J., & Han, B. (2016). Image retrieval with deep local features and attention-based keypoints. CoRR.

O'Hara, S., & Draper, B. A. (2011). Introduction to the bag of features paradigm for image classification and retrieval. arXiv preprint arXiv:1101.3354.

Okuyama, K., Kawasaki, T., & Kroumov, V. (2011). Localization and position correction for mobile robot using artificial visual landmarks. In *Advanced mechatronic systems (ICAMechS), 2011 international conference on* (pp. 414–418).

Oliva, A., & Torralba, A. (2001). Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, *42*(3), 145–175.

Otte, S., Weiss, C., Scherer, T., & Zell, A. (2016). Recurrent neural networks for fast and robust vibration-based ground classification on mobile robots. In *2016 IEEE international conference on robotics and automation (ICRA)* (pp. 5603–5608). IEEE.

Özdemir, V., & Hekim, N. (2018). Birth of industry 5.0: Making sense of big data with artificial intelligence,"the internet of things" and next-generation technology policy. *Omics: a Journal of Integrative Biology*, *22*(1), 65–76.

Pak, M., & Kim, S. (2017). A review of deep learning in image recognition. In *2017 4th international conference on computer applications and information processing technology (CAIPT)* (pp. 1–3). IEEE.

Parker, L. E. (2000). Current state of the art in distributed autonomous mobile robotics. In *Distributed autonomous robotic systems 4* (pp. 3–12). Springer.

Parra, C., Cebollada, S., Payá, L., Holloway, M., & Reinoso, O. (2020). A novel method to estimate the position of a mobile robot in underfloor environments using RGB-D point clouds. *IEEE Access*, *8*, 9084–9101.

Pawgasame, W. (2016). A survey in adaptive hybrid wireless sensor network for military operations. In *2016 second Asian conference on defence technology (ACDT)* (pp. 78–83). IEEE.

Payá, L., Gil, A., & Reinoso, O. (2017). A state-of-the-art review on mapping and localization of mobile robots using omnidirectional vision sensors. *Journal of Sensors*, *2017*.

Payá, L., Peidró, A., Amorós, F., Valiente, D., & Reinoso, O. (2018). Modeling environments hierarchically with omnidirectional imaging and global-appearance descriptors. *Remote Sensing*, *10*(4), 522.

Payá, L., Reinoso, O., Berenguer, Y., & Úbeda, D. (2016). Using omnidirectional vision to create a model of the environment: A comparative evaluation of global-appearance descriptors. *Journal of Sensors*, *2016*.

Peretroukhin, V., Clement, L., & Kelly, J. (2017). Reducing drift in visual odometry by inferring sun direction using a Bayesian convolutional neural network. In *2017 IEEE international conference on robotics and automation (ICRA)* (pp. 2035–2042). http://dx.doi.org/10.1109/ICRA.2017.7989235.

Pfeiffer, J., Broscheit, S., Gemulla, R., & Göschl, M. (2018). *A neural autoencoder approach for document ranking and query refinement in pharmacogenomic information retrieval*. Association for Computational Linguistics.

Polvara, R., Sharma, S., Wan, J., Manning, A., & Sutton, R. (2018). Obstacle avoidance approaches for autonomous navigation of unmanned surface vehicles. *Journal of Navigation*, *71*(1), 241–256. http://dx.doi.org/10.1017/S0373463317000753.

Puthussery, A. R., Haradi, K. P., Erol, B. A., Benavidez, P., Rad, P., & Jamshidi, M. (2017). A deep vision landmark framework for robot navigation. In *2017 12th system of systems engineering conference (SoSE)* (pp. 1–6). IEEE.

Rabbath, M., Sandhaus, P., & Boll, S. (2012). Analysing facebook features to support event detection for photo-based facebook applications. In *Proceedings of the 2nd ACM international conference on multimedia retrieval* (p. 11). ACM.

Rahman, M. S., Park, Y., & Kim, K. D. (2012). RSS-based indoor localization algorithm for wireless sensor network using generalized regression neural network. *Arabian Journal for Science and engineering*, *37*(4), 1043–1053.

Rahmatizadeh, R., Abolghasemi, P., Behal, A., & Bölöni, L. (2016). Learning real manipulation tasks from virtual demonstrations using LSTM. arXiv preprint arXiv:1603.03833.

Ray, S. (2018). History of AI, in Towards Data Science. https://towardsdatascience.com/history-of-ai-484a86fc16ef.

Reinoso, O., & Payá, L. (2020a). Special issue on mobile robots navigation. *Applied Sciences*, *10*(4), http://dx.doi.org/10.3390/app10041317, Retrieved from https://www.mdpi.com/2076-3417/10/4/1317.

Reinoso, O., & Payá, L. (2020b). Special issue on visual sensors. *Sensors*, *20*(3), http://dx.doi.org/10.3390/s20030910, Retrieved from https://www.mdpi.com/1424-8220/20/3/910.

Richter, C., & Roy, N. (2017). *Safe visual navigation via deep learning and novelty detection*. Robotics: Science and Systems Foundation.

Rituerto, A., Murillo, A. C., & Guerrero, J. (2014). Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous systems*, *62*(5), 685–695.

Romdhani, S., Torr, P., Scholkopf, B., & Blake, A. (2001). Computationally efficient face detection. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001, Vol. 2* (pp. 695–700). IEEE.

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In *European conference on computer vision* (pp. 430–443). Springer.

Rosten, E., Porter, R., & Drummond, T. (2008). Faster and better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(1), 105–119.

Ruan, X., Ren, D., Zhu, X., & Huang, J. (2019). Mobile robot navigation based on deep reinforcement learning. In *2019 Chinese control and decision conference (CCDC)* (pp. 6174–6178). IEEE.

Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*.

Salakhutdinov, R., & Hinton, G. (2009). Deep boltzmann machines. In *Artificial intelligence and statistics* (pp. 448–455).

Sarlin, P., Cadena, C., Siegwart, R., & Dymczyk, M. (2019). From coarse to fine: Robust hierarchical localization at large scale. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 12708–12717). http://dx.doi.org/10.1109/CVPR.2019.01300.

Schalkoff, R. J. (1990). *Artificial intelligence: an engineering approach*. McGraw-Hill New York.

Schleichert, H. (1970). Marvin minsky (ed.)," semantic information processing"(book review). *Theory and Decision*, *1*(2), 222.

Schmidt, T., Newcombe, R., & Fox, D. (2017). Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, *2*(2), 420–427. http://dx.doi.org/10.1109/LRA.2016.2634089.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 815–823).

Se, S., Lowe, D. G., & Little, J. J. (2005). Vision-based global localization and mapping for mobile robots. *IEEE Transactions on Robotics*, *21*(3), 364–375.

Sergeant, J., Sünderhauf, N., Milford, M., & Upcroft, B. (2015). Multimodal deep autoencoders for control of a mobile robot. In *Proc. of Australasian Conf. for robotics and automation (ACRA)*.

Sharma, P., Liu, H., Wang, H., & Zhang, S. (2017). Securing wireless communications of connected vehicles with artificial intelligence. In *2017 IEEE international symposium on technologies for homeland security (HST)* (pp. 1–7). IEEE.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Singh, M. K., & Parhi, D. R. (2011). Path optimisation of a mobile robot using an artificial neural network controller. *International Journal of Systems Science*, *42*(1), 107–120.

Singh, S. K., Rathore, S., & Park, J. H. (2020). Blockiotintelligence: A blockchain-enabled intelligent IoT architecture with artificial intelligence. *Future Generation Computer Systems*, *110*, 721–743.

Sinha, H., Patrikar, J., Dhekane, E. G., Pandey, G., & Kothari, M. (2018). Convolutional neural network based sensors for mobile robot relocalization. In *2018 23rd international conference on methods models in automation robotics (MMAR)* (pp. 774–779). http://dx.doi.org/10.1109/MMAR.2018.8485921.

Smith, C., Karayiannidis, Y., Nalpantidis, L., Gratal, X., Qi, P., Dimarogonas, D. V., et al. (2012). Dual arm manipulation—A survey. *Robotics and Autonomous systems*, *60*(10), 1340–1353.

Smolyanskiy, N., Kamenev, A., Smith, J., & Birchfield, S. (2017). Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4241–4247). IEEE.

Šochman, J., & Matas, J. (2009). Learning fast emulators of binary decision processes. *International Journal of Computer Vision*, *83*(2), 149–163.

Sommer, K., Kim, K., Kim, Y., & Jo, S. (2017). Towards accurate kidnap resolution through deep learning. In *2017 14th international conference on ubiquitous robots and ambient intelligence (URAI)* (pp. 502–506). http://dx.doi.org/10.1109/URAI.2017.7992654.

Stachniss, C., & Burgard, W. (2003). Exploring unknown environments with mobile robots using coverage maps. In *IJCAI, Vol. 2003* (pp. 1127–1134).

Strecha, C., Bronstein, A., Bronstein, M., & Fua, P. (2011). Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(1), 66–78.

Su, Z., Zhou, X., Cheng, T., Zhang, H., Xu, B., & Chen, W. (2017). Global localization of a mobile robot using lidar and visual features. In *2017 IEEE international conference on robotics and biomimetics (ROBIO)* (pp. 2377–2383). IEEE.

Sun, L., Yan, Z., Mellado, S. M., Hanheide, M., & Duckett, T. (2018). 3DOF pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data. In *2018 IEEE international conference on robotics and automation (ICRA)* (pp. 1–7). IEEE.

Sünderhauf, N., Shirazi, S., Dayoub, F., Upcroft, B., & Milford, M. (2015). On the performance of ConvNet features for place recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4297–4304). http://dx.doi.org/10.1109/IROS.2015.7353986.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).

Tai, L., Li, S., & Liu, M. (2017). Autonomous exploration of mobile robots through deep neural networks. *International Journal of Advanced Robotic Systems*, *14*(4), Article 1729881417703571.

Tai, L., & Liu, M. (2016). Mobile robots exploration through cnn-based reinforcement learning. *Robotics and Biomimetics*, *3*(1), 24.

Tai, L., Paolo, G., & Liu, M. (2017). Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 31–36). IEEE.

Tang, J., Ren, Y., & Liu, S. (2017). Real-time robot localization, vision, and speech recognition on nvidia jetson TX1. arXiv preprint arXiv:1705.10945.

Tanzmeister, G., Thomas, J., Wollherr, D., & Buss, M. (2014). Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *2014 IEEE international conference on robotics and automation (ICRA)* (pp. 6090–6095). IEEE.

Tapu, R., Mocanu, B., & Zaharia, T. (2013). A computer vision system that ensure the autonomous navigation of blind people. In *2013 E-health and bioengineering conference (EHB)* (pp. 1–4). IEEE.

Tardif, J. P., Pavlidis, Y., & Daniilidis, K. (2008). Monocular visual odometry in urban environments using an omnidirectional camera. In *2008 IEEE/RSJ international conference on intelligent robots and systems* (pp. 2531–2538). http://dx.doi.org/10.1109/IROS.2008.4651205.

Tateno, K., Tombari, F., Laina, I., & Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6243–6252).

Theodoridis, S. (2015). *Machine learning: a Bayesian and optimization perspective*. Academic Press.

Theodoridis, S., & Koutroumbas, K. (1999). Pattern recognition and neural networks. In *Advanced course on artificial intelligence* (pp. 169–195). Springer.

Thrun, S. (2002). Robotic mapping: A survey. *Exploring Artificial Intelligence in the New Millennium*, *1*(1–35), 1.

Trujillo, L., & Olague, G. (2006). Synthesis of interest point detectors through genetic programming. In *Proceedings of the 8th annual conference on genetic and evolutionary computation* (pp. 887–894). ACM.

Unicomb, J., Ranasinghe, R., Dantanarayana, L., & Dissanayake, G. (2018). A monocular indoor localiser based on an extended Kalman filter and edge images from a convolutional neural network. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1–9). http://dx.doi.org/10.1109/IROS.2018.8594337.

Valiente, D., Payá, L., Jiménez, L. M., Sebastián, J. M., & Reinoso, O. (2018). Visual information fusion through bayesian inference for adaptive probability-oriented feature matching. *Sensors*, *18*(7), 2041.

Verdie, Y., Yi, K., Fua, P., & Lepetit, V. (2015). TILDE: a temporally invariant learned detector. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5279–5288).

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, *11*(Dec), 3371–3408.

Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, *57*(2), 137–154.

Vyborny, C. J., & Giger, M. L. (1994). Computer vision and artificial intelligence in mammography.. *AJR. American Journal of Roentgenology*, *162*(3), 699–708.

Wachs, J. P., Kölsch, M., Stern, H., & Edan, Y. (2011). Vision-based hand-gesture applications. *Communications of the ACM*, *54*(2), 60–71.

Walch, F., Hazirbas, C., Leal-Taixé, L., Sattler, T., Hilsenbeck, S., & Cremers, D. (2017). Image-based localization using LSTMs for structured feature correlation. In *2017 IEEE international conference on computer vision (ICCV)* (pp. 627–637). http://dx.doi.org/10.1109/ICCV.2017.75.

Walker, C., Graham, P., & Philippides, A. (2017). Using deep autoencoders to investigate image matching in visual navigation. In *Conference on biomimetic and biohybrid systems* (pp. 465–474). Springer.

Wang, Y., Bao, T., Ding, C., & Zhu, M. (2017). Face recognition in real-world surveillance videos with deep learning method. In *2017 2nd international conference on image, vision and computing (ICIVC)* (pp. 239–243). IEEE.

Wang, C., Pelillo, M., & Siddiqi, K. (2019). Dominant set clustering and pooling for multi-view 3d object recognition. arXiv preprint arXiv:1906.01592.

Wang, K., Wang, W., & Zhuang, Y. (2007). Appearance-based map learning for mobile robot by using generalized regression neural network. In *International symposium on neural networks* (pp. 834–842). Springer.

Wang, H., Yang, W., Huang, W., Lin, Z., & Tang, Y. (2018). Multi-feature Fusion for deep reinforcement learning: Sequential control of mobile robots. In *International conference on neural information processing* (pp. 303–315). Springer.

Wei, X., Xie, C. W., Wu, J., & Shen, C. (2018). Mask-CNN: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognition*, *76*, 704–714. http://dx.doi.org/10.1016/j.patcog.2017.10.002, Retrieved from http://www.sciencedirect.com/science/article/pii/S0031320317303990.

Weinberger, K. Q., & Saul, L. K. (2006). Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, *70*(1), 77–90.

Weinzaepfel, P., Csurka, G., Cabon, Y., & Humenberger, M. (2019). Visual localization by learning objects-of-interest dense match regression. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 5627–5636). http://dx.doi.org/10.1109/CVPR.2019.00578.

Wong, P., Tam, L., Li, K., & Vong, C. (2010). Engine idle-speed system modelling and control optimization using artificial intelligence. *Proceedings of the Institution of Mechanical Engineers, Part D (Journal of Automobile Engineering)*, *224*(1), 55–72.

Wozniak, P., Afrisal, H., Esparza, R. G., & Kwolek, B. (2018). Scene recognition for indoor localization of mobile robots using deep CNN. In *International conference on computer vision and graphics* (pp. 137–147). Springer.

Wu, H., & Qin, S. Y. (2011). An approach to robot SLAM based on incremental appearance learning with omnidirectional vision. *International Journal of Systems Science*, *42*(3), 407–427.

Xing, F., Xie, Y., & Yang, L. (2015). An automatic learning-based framework for robust nucleus segmentation. *IEEE Transactions on Medical Imaging*, *35*(2), 550–566.

Xu, S., Chou, W., & Dong, H. (2019). A robust indoor localization system integrating visual localization aided by CNN-based image retrieval with Monte Carlo localization. *Sensors*, *19*(2), 249.

Yang, S., Konam, S., Ma, C., Rosenthal, S., Veloso, M., & Scherer, S. (2017). Obstacle avoidance through deep networks based intermediate perception. arXiv preprint arXiv:1704.08759.

Yang, M., Kriegman, D. J., & Ahuja, N. (2002). Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, *24*(1), 34–58.

Yang, Y., Li, Y., Fermuller, C., & Aloimonos, Y. (2015). Robot learning manipulation action plans by "watching" unconstrained videos from the world wide web. In *Twenty-Ninth AAAI conference on artificial intelligence*.

Yi, K. M., Trulls, E., Lepetit, V., & Fua, P. (2016). Lift: Learned invariant feature transform. In *European conference on computer vision* (pp. 467–483). Springer.

Yi, K. M., Verdie, Y., Fua, P., & Lepetit, V. (2016). Learning to assign orientations to feature points. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 107–116).

Yong-guo, Z., Wei, C., & Guang-liang, L. (2012). The navigation of mobile robot based on stereo vision. In *Intelligent computation technology and automation (ICICTA), 2012 fifth international conference on* (pp. 670–673). IEEE.

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. In *Advances in neural information processing systems* (pp. 3320–3328).

Zaki, H. F., Shafait, F., & Mian, A. (2019). Viewpoint invariant semantic object and scene categorization with RGB-D sensors. *Autonomous Robots*, *43*(4), 1005–1022.

Zhang, B., Huang, W., Gong, L., Li, J., Zhao, C., Liu, C., et al. (2015). Computer vision detection of defective apples using automatic lightness correction and weighted RVM classifier. *Journal of Food Engineering*, *146*, 143–151.

Zhang, J., Marszałek, M., Lazebnik, S., & Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, *73*(2), 213–238.

Zhang, J., Springenberg, J. T., Boedecker, J., & Burgard, W. (2017). Deep reinforcement learning with successor features for navigation across similar environments. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 2371–2378). IEEE.

Zhang, X., Su, Y., & Zhu, X. (2017). Loop closure detection for visual SLAM systems using convolutional neural network. In *2017 23rd international conference on automation and computing (ICAC)* (pp. 1–6). IEEE.

Zhao, Q., Zhang, B., Lyu, S., Zhang, H., Sun, D., Li, G., & Feng, W. (2018). A CNN-SIFT hybrid pedestrian navigation method based on first-person vision. *Remote Sensing*, *10*(8), 1229.

Zheng, L., Yang, Y., & Tian, Q. (2017). SIFT meets CNN: A decade survey of instance retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(5), 1224–1244.

Zhong, F., Wang, S., Zhang, Z., & Wang, Y. (2018). Detect-slam: Making object detection and slam mutually beneficial. In *2018 IEEE winter conference on applications of computer vision (WACV)* (pp. 1001–1010). IEEE.

Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-sign detection and classification in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2110–2118).

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., et al. (2017). Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)* (pp. 3357–3364). http://dx.doi.org/10.1109/ICRA.2017.7989381.

Zhu, L., & Zheng, W. J. (2018). Informatics, data science, and artificial intelligence. *Jama*, *320*(11), 1103–1104.

Zivkovic, Z., Bakker, B., & Krose, B. (2005). Hierarchical map building using visual landmarks and geometric constraints. In *2005 IEEE/RSJ international conference on intelligent robots and systems* (pp. 2480–2485). http://dx.doi.org/10.1109/IROS.2005.1544951.

# Bibliography


[1] Department of Energy and Climate Change. Estimates of Home Insulation Levels in Great Britain. https://www.gov.uk/government/statistical-data-sets/estimates-of-home-insulation-levels-in-great-britain, 2013. Accessed: 2017-05-03. 43

[2] A. F. Abate, M. Nappi, D. Riccio, and G. Sabatino. 2d and 3d face recognition: A survey. *Pattern recognition letters*, 28(14):1885–1906, 2007. 26

[3] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Deep audio-visual speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018. 23, 176

[4] W. G. Aguilar, M. A. Luna, J. F. Moya, V. Abad, H. Parra, and H. Ruiz. Pedestrian detection for uavs using cascade classifiers with meanshift. In *2017 IEEE 11th international conference on semantic computing (ICSC)*, pages 509–514. IEEE, 2017. 27

[5] F. Amorós, L. Payá, J. M. Marín, and O. Reinoso. Trajectory estimation and optimization through loop closure detection, using omnidirectional imaging and global-appearance descriptors. *Expert Systems with Applications*, 102:273–290, 2018. 33

[6] A. Angeli, S. Doncieux, J.A. Meyer, and D. Filliat. Visual topological slam and global localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 2029-2034*. IEEE, 2009. 33, 86

[7] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 37

[8] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, and E. Romera. Fusion and binarization of cnn features for robust topological localization across seasons. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4656–4663, Oct 2016. 31, 177

[9] C. Arth, C. Pirchheim, J. Ventura, D. Schmalstieg, and V. Lepetit. Instant outdoor localization and slam initialization from 2.5 d maps. *IEEE transactions on visualization and computer graphics*, 21(11):1309–1318, 2015. 1

[10] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, Sept 1987. 21

[11] ARVC. Automation, Robotics and Computer Vision Research Group. Miguel Hernández University. Spain. Quorum 5 set of images. http://arvc.umh.es/db/images/quorumv/. 100

[12] J. Atkinson and D. Campos. Improving bci-based emotion recognition by combining eeg feature selection and kernel classifiers. *Expert Systems with Applications*, 47:35–41, 2016. 27

[13] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. F. R. Jesus, R. F. Berriel, T. M. Paixão, F. Mutz, et al. Self-driving cars: A survey. *arXiv preprint arXiv:1901.04407*, 2019. 25

[14] T. Baldawi. Fireman robot: a prototype design. *International Journal of Applied Engineering Research*, 12(23):13255–13264, 2017. 17

[15] B. Barshan, B. Ayrulu, and S. W. Utete. Neural network-based target differentiation using sonar for robotics applications. *IEEE transactions on Robotics and Automation*, 16(4):435–442, 2000. 137

[16] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008. 45, 62

[17] H. Bay, T. Tuytelaars, and L. Gool. Surf: Speeded up robust features. In *Computer Vision at ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin Heidelberg, 2006. 33

[18] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003. 36

[19] A. J. Bell and T. J. Sejnowski. The "independent components" of natural scenes are edge filters. *Vision research*, 37(23):3327–3338, 1997. 36

[20] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 23, 36, 176

[21] Y. Berenguer, L. Payá, M. Ballesta, and O. Reinoso. Position estimation and local mapping using omnidirectional images and global appearance descriptors. *Sensors*, 15(10):26368–26395, 2015. 2, 34, 87

[22] Y. Berenguer, L. Payá, D. Valiente, A. Peidró, and O. Reinoso. Relative altitude estimation using omnidirectional imaging and holistic descriptors. *Remote Sensing*, 11(3):323, 2019. 33

[23] P. J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):239–256, Feb 1992. 21

[24] P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, volume 3, pages 2743–2748. IEEE, 2003. 60

[25] M. Bilgili and B. Sahin. Comparative analysis of regression and artificial neural network models for wind speed prediction. *Meteorology and Atmospheric Physics*, 109(1-2):61–72, 2010. 32

[26] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart. Topomap: Topological mapping and navigation based on visual slam maps. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018. 19

[27] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018. 41

[28] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart. State estimation for legged robots-consistent fusion of leg kinematics and imu. *Robotics*, 17:17–24, 2013. 1

[29] J. Bodner, H. Wykypiel, G. Wetscher, and T. Schmid. First experiences with the da vinci$^{TM}$ operating robot in thoracic surgery. *European Journal of Cardio-thoracic surgery*, 25(5):844–851, 2004. 17

[30] A. Boularias, J. A. Bagnell, and A. Stentz. Learning to manipulate unknown objects in clutter by reinforcement. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 27

[31] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. Geometry-aware learning of maps for camera localization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, June 2018. 39

[32] A. Brunetti, D. Buongiorno, G. F. Trotta, and V. Bevilacqua. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing*, 300:17–33, 2018. 23, 176

[33] C. Calderon-Cordova, C. Ramírez, V. Barros, P. A. Quezada-Sarmiento, and L. Barba-Guamán. Emg signal patterns recognition based on feedforward artificial neural network applied to robotic prosthesis myoelectric control. In *2016 Future Technologies Conference (FTC)*, pages 868–875. IEEE, 2016. 24

[34] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010*, pages 778–792, 2010. 33

[35] F. M Campos, L. Correia, and J. MF Calado. Robot visual localization through local feature fusion: an evaluation of multiple classifiers combination approaches. *Journal of Intelligent & Robotic Systems*, 77(2):377–390, 2015. 33

[36] E. Cappelletto, P. Zanuttigh, and G. M. Cortelazzo. 3d scanning of cultural heritage with consumer depth cameras. *Multimedia Tools and Applications*, 75(7):3631–3654, 2016. 21

[37] F. Carreira, J. M.F. Calado, C. Cardeira, and P. Oliveira. Enhanced pca-based localization using depth maps with missing data. *Journal of Intelligent & Robotic Systems*, 77(2):341–360, 2015. 29

[38] S. Cascianelli, G. Costante, E. Bellocchio, P. Valigi, M. L. Fravolini, and T. A. Ciarfuglia. Robust visual semi-semantic loop closure detection by a covisibility graph and cnn features. *Robotics and Autonomous Systems*, 92:53 – 65, 2017. 40

[39] D. Cattaneo, M. Vaghi, A. L. Ballardini, S. Fontana, D. G. Sorrenti, and W. Burgard. Cmrnet: Camera to lidar-map registration. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1283–1289, Oct 2019. 40

[40] S. Cebollada, L. Payá, M. Flores, A. Peidró, and O. Reinoso. A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data. *Expert Systems with Applications*, page 114195, 2020. iii, 7, 42, 218

[41] S. Cebollada, L. Payá, M. Juliá, M. Holloway, and O. Reinoso. Mapping and localization module in a mobile robot for insulating building crawl spaces. *Automation in Construction*, 87:248 – 262, 2018. iii, 7, 22, 84, 217

[42] S. Cebollada, L. Payá, A. Peidró, L.M. Jiménez, and O. Reinoso. Estudio de descriptores holísticos basados en métodos analíticos y técnicas de deep learning para localización con robots móviles. 9, 167

[43] S. Cebollada, L. Payá, M. Flores, V. Román, A. Peidró, and O. Reinoso. A deep learning tool to solve localization in mobile autonomous robotics. In *ICINCO 2020, 17th International Conference on Informatics in Control, Automation and Robotics (Lieusaint-Paris, France, 7-9 July, 2020)*. Ed. INSTICC, 2020. 9, 200

[44] S. Cebollada, L. Payá, W. Mayol, and O. Reinoso. Evaluation of clustering methods in compression of topological models and visual place recognition using global appearance descriptors. *Applied Sciences*, 9(3):377, 2019. iii, 7, 126, 217

[45] S. Cebollada, L. Payá, V. Román, and O. Reinoso. Hierarchical localization in topological models under varying illumination using holistic visual descriptors. *IEEE Access*, 7:49580–49595, 2019. iii, 7, 127, 218

[46] S. Cebollada, V. Roman, L. Payá, M. Flores, L.M. Jiménez, and O. Reinoso. Uso de técnicas de machine learning para realizar mapping en robótica móvil. 9, 168

[47] C. K. Chang, C. Siagian, and L. Itti. Mobile robot vision navigation and localization using gist and saliency. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 4147–4154, 2010. 90

[48] E. Charniak, D. McDermott, and D.V. McDermott. *Introduction to Artificial Intelligence*. Addison-Wesley series in computer science and information processing. Addison-Wesley, 1985. 23, 133

[49] D. Chaves, J. R. Ruiz-Sarmiento, N. Petkov, and J. Gonzalez-Jimenez. Integration of cnn into a robotic architecture to build semantic maps of indoor environments. In *International Work-Conference on Artificial Neural Networks*, pages 313–324. Springer, 2019. 32, 177

[50] Y. Chen, W. Gan, L. Zhang, C. Liu, and X. Wang. A survey on visual place recognition for mobile robots localization. In *2017 14th Web Information Systems and Applications Conference (WISA)*, pages 187–192. IEEE, 2017. 19

[51] S. O. Chishti, S. Riaz, M. BilalZaib, and M. Nauman. Self-driving cars using cnn and q-learning. In *2018 IEEE 21st International Multi-Topic Conference (INMIC)*, pages 1–7. IEEE, 2018. 176

[52] F. Chollet. Deep learning with python, vol. 1. *Greenwich, CT: Manning Publications CO*, 2017. 31, 176

[53] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2652–2660, July 2017. 38, 135

[54] D. M. Cole and P. M. Newman. Using laser range data for 3d slam in outdoor environments. In *Robotics and Automation, 2006. IEEE International Conference on Robotics and Automation (ICRA) 2006. Proceedings 2006 IEEE International Conference on*, pages 1556–1563. IEEE, 2006. 45

[55] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995. 137

[56] F. A. X. Da Mota, M. X. Rocha, J. J. P. C. Rodrigues, V. H. C. De Albuquerque, and A. R. De Alexandria. Localization and navigation for autonomous mobile robots using petri nets in indoor environments. *IEEE Access*, 6:31665–31676, 2018. 20

[57] S. P. P. da Silva, R. V. M. da Nóbrega, A. G. Medeiros, L. B. Marinho, J. S. Almeida, and P. P. R. Filho. Localization of mobile robots with topological maps and classification with reject option using convolutional neural networks in omnidirectional images. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2018. 19, 38, 88

[58] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, USA. Vol. II, pp. 886-893*, 2005. 90

[59] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007. 45

[60] E. De Momi and G. Ferrigno. Robotic and artificial intelligence for keyhole neurosurgery: the robocast project, a multi-modal autonomous path planner. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 224(5):715–727, 2010. 24

[61] S. H. Dezfoulian, D. Wu, and I. S. Ahmad. A generalized neural network approach to mobile robot navigation and obstacle avoidance. In *Intelligent Autonomous Systems 12*, pages 25–42. Springer, 2013. 33

[62] N. Dhanachandra, K. Manglem, and Y. J. Chanu. Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54:764–771, 2015. 28

[63] J. Ding, B. Chen, H. Liu, and M. Huang. Convolutional neural network with data augmentation for sar target recognition. *IEEE Geoscience and remote sensing letters*, 13(3):364–368, 2016. 181

[64] Y. Doi, Y. Ji, Y. Tamura, Y. Ikeda, A. Umemura, Y. Kaneshima, H. Murakami, A. Yamashita, and H. Asama. Robust path planning against pose errors for mobile robots in rough terrain. In *International Conference on Intelligent Autonomous Systems*, pages 27–39. Springer, 2018. 20

[65] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 37

[66] L. Dong, X. Yu, L. Li, and J. K. E. Hoe. Hog based multi-stage object detection and pose recognition for service robot. In *2010 11th International Conference on Control Automation Robotics Vision*, pages 2495–2500, Dec 2010. 90

[67] D. Donoho and C. Grimes. Hessian eigenmaps: new locally linear embedding techniques for highdimensional data (technical report tr2003-08), 2003. 36

[68] D. R. dos Santos, M. A. Basso, K. Khoshelham, E. de Oliveira, N. L. Pavan, and G. Vosselman. Mapping indoor spaces by adaptive coarse-to-fine registration of rgb-d data. *IEEE Geoscience and Remote Sensing Letters*, 13(2):262–266, Feb 2016. 2

[69] M. Duguleana and G. Mogan. Neural networks based reinforcement learning for mobile robots obstacle avoidance. *Expert Systems with Applications*, 62:104 – 115, 2016. 23

[70] M. Dymczyk, I. Gilitschenski, J. Nieto, S. Lynen, B. Zeisl, and R. Siegwart. Landmarkboost: Efficient visualcontext classifiers for robust localization. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 677–684, Oct 2018. 4, 23, 133, 175

[71] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the rgb-d slam system. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1691–1696, May 2012. 2, 21

[72] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza. Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4):431–450, 2016. 85, 142

[73] F. H. Fan, Q. Ma, J. Ge, Q. Y. Peng, William W. Riley, and S. Z. Tang. Prediction of texture characteristics from extrusion food surface images using a computer vision system and artificial neural networks. *Journal of Food Engineering*, 118(4):426 – 433, 2013. 23

[74] H. Fan, L. Zheng, C. Yan, and Y. Yang. Unsupervised person re-identification: Clustering and fine-tuning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(4):83, 2018. 28

[75] L. Fernández, L. Payá, Ó. Reinoso, A. Gil, and M. Juliá. Robust methods for robot localization under changing illumination conditions-comparison of different filtering techniques. In *ICAART (1)*, pages 223–228, 2010. 91

[76] J. P. Ferreira, T. G Amaral, V. F. Pires, M. M. Crisostomo, and P. Coimbra. A neural-fuzzy walking control of an autonomous biped robot. In *Proceedings World Automation Congress, 2004.*, volume 15, pages 253–258. IEEE, 2004. 32

[77] D. Filliat and J. A. Meyer. Map-based navigation in mobile robots:: I. a review of localization strategies. *Cognitive Systems Research*, 4(4):243 – 282, 2003. 39

[78] F. Fraundorfer, C. Engels, and D. Nistér. Topological mapping, localization and navigation using image collections. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3872–3877. IEEE, 2007. 19

[79] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995. 25

[80] S. Fu, H. Liu, L. Gao, and Y. Gai. Slam for mobile robots using laser range finder and monocular vision. In *Mechatronics and Machine Vision in Practice, 2007. M2VIP 2007. 14th International Conference on*, pages 91–96. IEEE, 2007. 45

[81] J. Fuentes-Pacheco and J. Ruiz-Ascencio. Visual simultaneous localization and mapping: a survey. 1, 20, 40

[82] S. M. Gadoue, D. Giaouris, and J.W. Finch. Artificial intelligence-based speed control of dtc induction motor drives—a comparative study. *Electric Power Systems Research*, 79(1):210–219, 2009. 24

[83] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li. Object classification using cnn-based fusion of vision and lidar in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics*, 14(9):4224–4231, 2018. 26, 176

[84] X. Gao and T. Zhang. Unsupervised learning to detect loops using deep neural networks for visual slam system. *Autonomous robots*, 41(1):1–18, 2017. 41, 142

[85] E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, 64:1–20, 2015. 19, 87

[86] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016. 41

[87] A. Gil, O. Reinoso, M. Ballesta, M. Juliá, and L. Payá. Estimation of visual maps with a robot network equipped with vision sensors. *Sensors*, 10(5):5209–5232, 2010. 19

[88] R. Gonzalez, D. Apostolopoulos, and K. Iagnemma. Slippage and immobilization detection for planetary exploration rovers via machine learning and proprioceptive sensing. *Journal of Field Robotics*, 35(2):231–247, 2018. 4, 23, 133, 175

[89] R. C. Gonzalez and R. E. Woods. Digital image processing prentice hall. *Upper Saddle River, NJ*, 2002. 91

[90] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016. 28, 30, 141

[91] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European conference on computer vision*, pages 241–257. Springer, 2016. 37

[92] G. Z. Grudic and J. Mulligan. Topological mapping with multiple visual manifolds. In *Robotics: Science and Systems*, pages 185–192, 2005. 3

[93] J. Guo and S. Gould. Deep cnn ensemble with data augmentation for object detection. *arXiv preprint arXiv:1506.07224*, 2015. 181

[94] K. Gwinner, R. Jaumann, E. Hauber, H. Hoffmann, C. Heipke, J. Oberst, G. Neukum, V. Ansan, J. Bostelmann, A. Dumke, et al. The high resolution stereo camera (hrsc) of mars express and its approach to science analysis and mapping for mars and its satellites. *Planetary and Space Science*, 126:93–138, 2016. 2

[95] A. Hadid, J.Y. Heikkila, O. Silvén, and M. Pietikainen. Face and eye detection for person authentication in mobile phones. In *2007 First ACM/IEEE International Conference on Distributed Smart Cameras*, pages 101–108. IEEE, 2007. 26

[96] Y. Hagiwara, M. Inoue, H. Kobayashi, and T. Taniguchi. Hierarchical spatial concept formation based on multimodal information for human support robots. *Frontiers in neurorobotics*, 12:11, 2018. 3, 19

[97] D. Han, Q. Liu, and W. Fan. A new image classification method using cnn transfer learning and web data augmentation. *Expert Systems with Applications*, 95:43–56, 2018. 31, 177, 181, 188

[98] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys. 3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. *Image and Vision Computing*, 68:14–27, 2017. 1

[99] D. J. Harris and S. J.-M. Dudek. Heat losses from suspended timber floors. *Building Research & Information*, 25(4):226–233, 1997. 43

[100] J. Haugeland. *Artificial intelligence: The very idea*. MIT press, 1989. 133

[101] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 31

[102] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *International Journal of Robotics Research*, 31(5):647–663, April 2012. 2, 21

[103] G. E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 36

[104] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. 29, 142

[105] A. Holliday and G. Dudek. Scale-robust localization using general object landmarks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1688–1694, Oct 2018. 40

[106] M. Holloway, M. Julia, and P. Childs. A robot for spray applied insulation in underfloor voids. In *Proceedings of ISR 2016: 47st International Symposium on Robotics*, pages 1–7, June 2016. Accessed: 2017-11-09. 43, 44

[107] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke. Registration with the point cloud library: A modular framework for aligning in 3-d. *IEEE Robotics & Automation Magazine*, 22(4):110–124, 2015. 60

[108] S. Holzer, J. Shotton, and P. Kohli. Learning to efficiently detect repeatable interest points in depth data. In *European Conference on Computer Vision*, pages 200–213. Springer, 2012. 35

[109] N. Hubens. *Deep inside: Autoencoders*, 2018. `https://towardsdatascience.com/deep-inside-autoencoders-7e41f319999f`. Accessed February 11, 2019. 141

[110] Y. Hwang and B. Choi. Hierarchical system mapping for large-scale fault-tolerant quantum computing. *arXiv preprint arXiv:1809.07998*, 2018. 3, 19

[111] K. Iagnemma and C. C. Ward. Classification-based wheel slip detection and detector fusion for mobile robots on outdoor terrain. *Autonomous Robots*, 26(1):33–46, 2009. 137

[112] G. Iyer, J. K. Murthy, G. Gupta, K. M. Krishna, and L. Paull. Geometric consistency for self-supervised end-to-end visual odometry. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 380–3808, June 2018. 38

[113] R. Jafri and H. R. Arabnia. A survey of face recognition techniques. *Jips*, 5(2):41–68, 2009. 26

[114] A. K. Jain and S. Z. Li. *Handbook of face recognition*. Springer, 2011. 26

[115] Y. Jia, M. Li, L. An, and X. Zhang. Autonomous navigation of a miniature mobile robot using real-time trinocular stereo machine. In *Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003 IEEE International Conference on*, volume 1, pages 417–421 vol.1, 2003. 2

[116] W. Jiang and W. Wang. Face detection and recognition for home service robots with end-to-end deep neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2232–2236, March 2017. 25

[117] P. Jiménez. Survey on model-based manipulation planning of deformable objects. *Robotics and computer-integrated manufacturing*, 28(2):154–163, 2012. 27

[118] T. Jost and H. Hügli. *Pattern Recognition: 24th DAGM Symposium Zurich, Switzerland, September 16–18, 2002 Proceedings*, chapter Fast ICP Algorithms for Shape Registration, pages 91–99. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002. 21

[119] M. Julia, M. Holloway, O. Reinoso, and P. R. N. Childs. Autonomous surveying of underfloor voids. In *Proceedings of ISR 2016: 47st International Symposium on Robotics*, pages 1–7. VDE, 2016. 44, 46

[120] C. Jutten and J. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal processing*, 24(1):1–10, 1991. 36

[121] A. Kanezaki, Y. Matsushita, and Y. Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5010–5019, 2018. 26

[122] A. Karishma, K. V. Anand Krishnan, A. Kiran, E. M. Dalin, and S. Shivaji. Smart office surveillance robot using face recognition. *International Journal of Mechanical and Production Engineering Research and Development*, 8:725–734, 06 2018. 25

[123] K. Kayaer and T. Yıldırım. Medical diagnosis on pima indian diabetes using general regression neural networks. In *Proceedings of the international conference on artificial neural networks and neural information processing (ICANN/ICONIP)*, volume 181, page 184, 2003. 32

[124] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, Dec 2015. 39

[125] K. Kim. Intelligent immigration control system by using passport recognition and face verification. In *International Symposium on Neural Networks*, pages 147–156. Springer, 2005. 26

[126] S. Kim, C. Roh, S. Kang, and M. Park. Outdoor navigation of a mobile robot using differential gps and curb detection. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3414–3419. IEEE, 2007. 1

[127] D. P. Kingma and Y. L. Cun. Regularized estimation of image statistics by score matching. In *Advances in neural information processing systems*, pages 1126–1134, 2010. 37

[128] M. Kirby. *Geometric data analysis: an empirical approach to dimensionality reduction and the study of patterns*. John Wiley & Sons, Inc., 2000. 36

[129] R. Kohavi and J. R. Quinlan. Data mining tasks and methods: Classification: decision-tree discovery. In *Handbook of data mining and knowledge discovery*, pages 267–276. Oxford University Press, Inc., 2002. 24

[130] T. Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1 – 6, 1998. 95

[131] D. Kopitkov and V. Indelman. Bayesian information recovery from cnn for probabilistic inference. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7795–7802, Oct 2018. 38

[132] H. Korrapati and Y. Mezouar. Multi-resolution map building and loop closure with omnidirectional images. *Autonomous Robots*, 41(4):967–987, 2017. 142

[133] M. Korytkowski, L. Rutkowski, and R. Scherer. Fast image classification by boosting fuzzy classifiers. *Information Sciences*, 327:175–182, 2016. 27

[134] I. Kostavelis and A. Gasteratos. Semantic mapping for mobile robotics tasks: A survey. *Robotics and Autonomous Systems*, 66:86 – 103, 2015. 18

[135] C. Krittanawong, H. Zhang, Z. Wang, M. Aydar, and T. Kitai. Artificial intelligence in precision cardiovascular medicine. *Journal of the American College of Cardiology*, 69(21):2657–2664, 2017. 24

[136] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 30, 31, 90, 176

[137] B. Krose, R. Bunschoten, S. Hagen, B. Terwijn, and N. Vlassis. Visual homing in enviroments with anisotropic landmark distrubution. In *Autonomous Robots, 23(3), 2007, pp. 231-245*, 2007. 86

[138] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli. Local metrical and global topological maps in the hybrid spatial semantic hierarchy. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 5, pages 4845–4851. IEEE, 2004. 19, 38

[139] R. Kumar, R.K. Aggarwal, and J.D. Sharma. Comparison of regression and artificial neural network models for estimation of global solar radiations. *Renewable and Sustainable Energy Reviews*, 52:1294–1299, 2015. 32

[140] I. Kuric, V. Bulej, M. Saga, and P. Pokorny. Development of simulation software for mobile robot path planning within multilayer map system based on metric and topological maps. *International Journal of Advanced Robotic Systems*, 14(6):1729881417743029, 2017. 18

[141] M. Kuse, S. P. Jaiswal, and S. Shen. Deep-mapnets : A residual network for 3d environment representation. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2652–2656, Sep. 2017. 39

[142] K. Kyriakopoulos, K. M. Knill, and M. JF Gales. A deep learning approach to automatic characterisation of rhythm in non-native english speech. *Proc. Interspeech 2019*, pages 1836–1840, 2019. 23, 176

[143] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. Exploring strategies for training deep neural networks. *Journal of machine learning research*, 10(Jan):1–40, 2009. 36

[144] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR 2011*, pages 3361–3368. IEEE, 2011. 36

[145] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 31

[146] K. E. Lee, J. Rao, and Y. Youn. Endoscopic thyroidectomy with the da vinci robot system using the bilateral axillary breast approach (baba) technique: our initial experience. *Surgical Laparoscopy Endoscopy & Percutaneous Techniques*, 19(3):e71–e75, 2009. 17

[147] K. Lenc and A. Vedaldi. Learning covariant feature detectors. In *European Conference on Computer Vision*, pages 100–117. Springer, 2016. 35

[148] K. Lenc and A. Vedaldi. Large scale evaluation of local image feature detectors on homography datasets. *arXiv preprint arXiv:1807.07939*, 2018. 35

[149] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015. 176

[150] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78(1):99–118, 2000. 34, 86, 87

[151] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018. 176

[152] T. S. Levitt and D. T. Lawton. Qualitative navigation for mobile robots. *Artificial intelligence*, 44(3):305–360, 1990. 17

[153] L. Li, H. Kim, S. Jiang, Y. Kim, and T. Kuc. Feature saliency based slam of mobile robot. In *2018 International Conference on Electronics, Information, and Communication (ICEIC)*, pages 1–3. IEEE, 2018. 20

[154] R. Li, Q. Liu, J. Gui, D. Gu, and H. Hu. Indoor relocalization in challenging environments with dual-stream convolutional neural networks. *IEEE Transactions on Automation Science and Engineering*, 15(2):651–662, April 2018. 39

[155] T. Li, X. Chang, Z. Wu, J. Li, G. Shao, X. Deng, J. Qiu, B. Guo, G. Zhang, Q. He, et al. Autonomous collision-free navigation of microvehicles in complex and dynamically changing environments. *ACS nano*, 11(9):9268–9275, 2017. 25

[156] C. Liang, Y. Tie, L. Qi, and C. Bi. Deep vidar: Cnn based 360°panoramic video system for outdoor robot visual navigation and slam. In *Tenth International Conference on Digital Image Processing (ICDIP 2018)*, volume 10806. International Society for Optics and Photonics, 2018. 41

[157] K. Lingemann, A. Nüchter, J. Hertzberg, and H. Surmann. High-speed laser localization for mobile robots. *Robotics and autonomous systems*, 51(4):275–296, 2005. 1

[158] K. Lingemann, H. Surmann, A. Nuchter, and J. Hertzberg. Indoor and outdoor localization for fast mobile robots. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2185–2190. IEEE, 2004. 2

[159] M. Liu, D. Scaramuzza, C. Pradalier, R. Siegwart, and Q. Chen. Scene recognition with omnidirectional vision for topological map using lightweight adaptive descriptors. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 116–121. IEEE, 2009. 34, 87

[160] W. Liu, Y. Mo, and J. Jiao. An efficient edge-feature constraint visual slam. In *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*, pages 1–7, 2019. 42

[161] P. Loncomilla, J. Ruiz-del Solar, and L. Martínez. Object recognition using local invariant features for robotic applications: A survey. *Pattern Recognition*, 60:499–514, 2016. 26

[162] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015. 37

[163] D. Longo and G. Muscato. A modular approach for the design of the alicia3 climbing robot for industrial inspection. *Industrial Robot: An International Journal*, 31(2):148–158, 2004. 17

[164] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999. 33

[165] G. Lu, Y. Yan, L. Ren, J. Song, N. Sebe, and C. Kambhamettu. Localize me anywhere, anytime: A multi-task point-retrieval approach. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2434–2442, Dec 2015. 41

[166] Y. Lu and G. Lu. Deep unsupervised learning for simultaneous visual odometry and depth estimation. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2571–2575, Sep. 2019. 42

[167] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007. 93, 95, 135

[168] X. Ma, H. Wang, and J. Geng. Spectral–spatial classification of hyperspectral image based on deep auto-encoder. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(9):4073–4085, 2016. 29

[169] S. Mahendran, H. Ali, and R. Vidal. 3d pose regression using convolutional neural networks. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2174–2182, Oct 2017. 177

[170] M. Mancini, S. R. Bulò, E. Ricci, and B. Caputo. Learning deep nbnn representations for robust place categorization. *IEEE Robotics and Automation Letters*, 2(3):1794–1801, 2017. 31, 91, 140, 177, 184

[171] L. B. Marinho, J. S. Almeida, J. W. M. Souza, V. H. C. Albuquerque, and P. P. R. Filho. A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images. *Expert Systems with Applications*, 72:1 – 17, 2017. 85

[172] M. A. Markom, A. H. Adom, E. S. M. M. Tan, S. Aniza A. Shukor, N. A. Rahim, and A. Y. M. Shakaff. A mapping mobile robot using rp lidar scanner. In *2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, pages 87–92. IEEE, 2015. 18

[173] M. E. Maron. Automatic indexing: An experimental inquiry. *J. ACM*, 8(3):404–417, July 1961. 136

[174] O. Martinez Mozos, K. Nakashima, H. Jung, Y. Iwashita, and R. Kurazume. Fukuoka datasets for place categorization. *The International Journal of Robotics Research*, 38(5):507–517, 2019. g, 3

[175] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:1806.07851*, 2018. 27

[176] R. Meattini, S. Benatti, U. Scarcia, D. De Gregorio, L. Benini, and C. Melchiorri. An semg-based human–robot interface for robotic hands using machine learning and synergies. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 8(7):1149–1158, July 2018. 175

[177] S. F. Memon, I. H. Kalwar, I. Grout, E. Lewis, and Y. N. Panhwar. Prototype for localization of multiple fire detecting mobile robots in a dynamic environment. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 395–400. IEEE, 2016. 19

[178] E. Menegatti, T. Maeda, and H. Ishiguro. Image-based memory for robot navigation using properties of omnidirectional images. *Robotics and Autonomous Systems*, 47(4):251 – 267, 2004. 34, 61, 89

[179] E. Menegatti, A. Pretto, A. Scarpa, and E. Pagello. Omnidirectional vision scan matching for robot localization in dynamic environments. *IEEE Transactions on Robotics*, 22(3):523–535, June 2006. 2, 22

[180] L. Meng, J. Chen, F. Tung, J. J. Little, J. Valentin, and C. W. de Silva. Backtracking regression forests for accurate camera relocalization. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6886–6893, Sep. 2017. 39

[181] R. C. Michelson. Autonomous navigation. *Access Science*, 2000. 24

[182] S. Milz, G. Arbeiter, C. Witt, B. Abdallah, and S. Yogamani. Visual slam for automated driving: Exploring the applications of deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 247–257, 2018. 41

[183] B. Minaei-Bidgoli, D. A. Kashy, G. Kortemeyer, and W. F. Punch. Predicting student performance: an application of data mining methods with an educational web-based system. In *33rd Annual Frontiers in Education, 2003. FIE 2003.*, volume 1, pages T2A–13. IEEE, 2003. 24

[184] M. Minsky. Semantic information processing. 1982. 23, 133

[185] D. Mishkin, F. P. Radenovic, and J. Matas. Repeatability is not enough: Learning affine regions via discriminability. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 284–300, 2018. 35

[186] O. Moolan-Feroze and A. Calway. Predicting out-of-view feature points for model-based camera pose estimation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 82–88, Oct 2018. 40

[187] O. Moolan-Feroze, K. Karachalios, D. N. Nikolaidis, and A. Calway. Improving drone localisation around wind turbines using monocular model-based tracking. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7713–7719, May 2019. 39

[188] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3376–3385, 2015. 28

[189] T. Mukasa, J. Xu, and B. Stenger. 3d scene mesh from cnn depth predictions and sparse monocular slam. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 921–928, 2017. 41

[190] D. Mukherjee, Q.M. J. Wu, and G. Wang. A comparative experimental study of image feature detectors and descriptors. *Machine Vision and Applications*, 26(4):443–466, 2015. 34

[191] A. C. Murillo, J. J. Guerrero, and C. Sagues. Surf features for efficient robot localization with omnidirectional images. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3901 –3907, april 2007. 2, 22, 33, 86

[192] A. C. Murillo, G. Singh, J. Kosecka, and J. J. Guerrero. Localization in urban environments using a panoramic gist descriptor. *IEEE Transactions on Robotics*, 29(1):146–160, 2013. 90

[193] G. R. S. Murthy and R. S. Jadon. Hand gesture recognition using neural networks. In *2010 IEEE 2nd International Advance Computing Conference (IACC)*, pages 134–138, Feb 2010. 22

[194] A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE transactions on pattern analysis and machine intelligence*, 32(12):2262–2275, 2010. 60

[195] F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani. Evaluation of machine learning classifiers for mobile malware detection. *Soft Computing*, 20(1):343–357, 2016. 27

[196] T. Naseer and W. Burgard. Deep regression for monocular camera-based 6-dof global localization in outdoor environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1525–1530, Sep. 2017. 39

[197] T. Naseer, W. Burgard, and C. Stachniss. Robust visual localization across seasons. *IEEE Transactions on Robotics*, 34(2):289–302, April 2018. 140

[198] P. Nazemzadeh, D. Fontanelli, D. Macii, and L. Palopoli. Indoor localization of mobile robots through qr code detection and dead reckoning data fusion. *IEEE/ASME Transactions on Mechatronics*, 22(6):2588–2599, 2017. 19

[199] A. M. Neto. Short-term visual mapping and robot localization based on learning classifier systems and self-organizing maps. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 235–240, June 2015. 39, 135

[200] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001. 28, 93

[201] J. Ngiam, Z. Chen, P. W. Koh, and A. Y. Ng. Learning deep energy models. 2011. 37

[202] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart. A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1929–1934. IEEE, 2005. 2

[203] K. Okuyama, T. Kawasaki, and V. Kroumov. Localization and position correction for mobile robot using artificial visual landmarks. In *Advanced Mechatronic Systems (ICAMechS), 2011 International Conference on*, pages 414–418, 2011. 2

[204] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. In *International Journal of Computer Vision, Vol. 42(3): 145-175.*, 2001. 34, 87

[205] A. Oliva and A. Torralba. Building the gist of ascene: the role of global image features in recognition. In *Progress in Brain Reasearch: Special Issue on Visual Perception.Vol. 155.*, 2006. 90

[206] M. Pak and S. Kim. A review of deep learning in image recognition. In *2017 4th international conference on computer applications and information processing technology (CAIPT)*, pages 1–3. IEEE, 2017. 31, 177

[207] X. E. Pantazi, A. A. Tamouridou, TK Alexandridis, A. L. Lagopodi, J. Kashefi, and D. Moshou. Evaluation of hierarchical self-organising maps for weed mapping using uas multispectral imagery. *Computers and Electronics in Agriculture*, 139:224–230, 2017. 3, 19

[208] C. Parra, S. Cebollada, L. Payá, M. Holloway, and O. Reinoso. A novel method to estimate the position of a mobile robot in underfloor environments using rgb-d point clouds. *IEEE Access*, 8:9084–9101, 2020. 9, 22, 84

[209] L. Payá, F. Amorós, L. Fernández, and O. Reinoso. Performance of global-appearance descriptors in map building and localization using omnidirectional vision. *Sensors*, 14(2):3033–3064, 2014. 11, 90, 96

[210] L. Payá, L. Fernández, A. Gil, and O. Reinoso. Map building and monte carlo localization using global appearance of omnidirectional images. *Sensors*, 10(12):11468–11497, 2010. 34, 87, 89, 96

[211] L. Payá, A. Gil, and O. Reinoso. A state-of-the-art review on mapping and localization of mobile robots using omnidirectional vision sensors. *Journal of Sensors*, 2017, 2017. 2, 22, 33, 34, 85

[212] L. Payá, W. Mayol, S. Cebollada, and O. Reinoso. Compression of topological models and localization using the global appearance of visual information. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5630–5637. IEEE, 2017. 9, 126

[213] L. Payá, A. Peidró, F. Amorós, D. Valiente, and O. Reinoso. Modeling environments hierarchically with omnidirectional imaging and global-appearance descriptors. *Remote Sensing*, 10(4):522, 2018. 22, 32, 91, 135, 178, 179

[214] L. Payá, O. Reinoso, Y. Berenguer, and D. Úbeda. Using omnidirectional vision to create a model of the environment: A comparative evaluation of global-appearance descriptors. *Journal of Sensors*, 2016, 2016. 61

[215] V. Peretroukhin, L. Clement, and J. Kelly. Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2035–2042, May 2017. 38

[216] J. Pérez, F. Caballero, and L. Merino. Enhanced monte carlo localization with visual place recognition for robust robot localization. *Journal of Intelligent & Robotic Systems*, 80(3-4):641–656, 2015. 95

[217] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017. 181

[218] J. Pfeiffer, S. Broscheit, R. Gemulla, and M. Göschl. A neural autoencoder approach for document ranking and query refinement in pharmacogenomic information retrieval. Association for Computational Linguistics, 2018. 29, 142

[219] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berlles. Stereo parallel tracking and mapping for robot localization. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1373–1378. IEEE, 2015. 1

[220] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton. Obstacle avoidance approaches for autonomous navigation of unmanned surface vehicles. *Journal of Navigation*, 71(1):241–256, 2018. 25

[221] L. F. Posada, K. K. Narayanan, F. Hoffmann, and T. Bertram. Floor segmentation of omnidirectional images for mobile robot visual navigation. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 804–809. IEEE, 2010. 137

[222] A. Pronobis and B. Caputo. COLD: COsy Localization Database. *The International Journal of Robotics Research (IJRR)*, 28(5):588–594, May 2009. 11, 99, 146, 180

[223] M. Rabbath, P. Sandhaus, and S. Boll. Analysing facebook features to support event detection for photo-based facebook applications. In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, page 11. ACM, 2012. 26

[224] J. Radon. Uber die bestimmug von funktionen durch ihre integralwerte laengs geweisser mannigfaltigkeiten. *Berichte Saechsishe Acad. Wissenschaft. Math. Phys., Klass*, 69:262, 1917. 87

[225] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008. 143

[226] M. S. Rahman, Y. Park, and K. D. Kim. Rss-based indoor localization algorithm for wireless sensor network using generalized regression neural network. *Arabian journal for science and engineering*, 37(4):1043–1053, 2012. 33

[227] H. S. Rao, V. H. Desai, R. Bhat, S. Jayaprakash, and Y. Sampangi. A study and implementation of mapping and speech recognition techniques for an autonomous mobile robot based on ros. *International Journal of Advanced Mechatronic Systems*, 7(5):303–310, 2017. 18

[228] S. Ray. History of ai, in towards data science. https://towardsdatascience.com/history-of-ai-484a86fc16ef, Aug 2018. 23

[229] O. Reinoso and L. Payá. Special issue on visual sensors, 2020. 22

[230] A. Rituerto, A. C. Murillo, and J.J. Guerrero. Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous Systems*, 62(5):685–695, 2014. 34, 87

[231] S. Romdhani, P. Torr, B. Scholkopf, and A. Blake. Computationally efficient face detection. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 695–700. IEEE, 2001. 25

[232] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer, 2006. 34

[233] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2008. 34

[234] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *International Conference on Computer Vision*, pages 2564–2571. IEEE, 2011. 33, 86

[235] Y. Rui, T. S. Huang, and S. F. Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1):39–62, 1999. 97

[236] R.B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1–4, May 2011. 52, 64

[237] D. Valiente X. Jiang O. Reinoso S. Cebollada, L. Paya. An evaluation between global appearance descriptors based on analytic methods and deep learning techniques for localization in autonomous mobile robots. In *ICINCO 2019, 16th International Conference on Informatics in Control, Automation and Robotics (Prague, Czech Republic, 29-31 July, 2019)*, pages 284–291. Ed. INSTICC, 2019. 9, 168

[238] M. Juliá M. Holloway L.M. Jiménez O. Reinoso S. Cebollada, C. Parra. Alineamiento 3d desde posiciones no cercanas de un robot para trabajos en interiores a partir de imágenes rgb-d. 9, 84

[239] R. Sahdev and J. K. Tsotsos. Indoor place recognition system for localization of mobile robots. In *2016 13th Conference on Computer and Robot Vision (CRV)*, pages 53–60. IEEE, 2016. 19

[240] R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455, 2009. 36

[241] J. Salamon and J. P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, March 2017. 181

[242] P. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12708–12717, June 2019. 38

[243] R. J. Schalkoff. *Artificial intelligence: an engineering approach*. McGraw-Hill New York, 1990. 23, 133

[244] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, April 2017. 41

[245] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 28

[246] A. Segal, D. Haehnel, and S. Thrun. Generalized-icp. *Robotics: Science and Systems*, 2(4), 2009. Accessed: 2017-11-09. 21

[247] J. Sergeant, N. Sünderhauf, M. Milford, and B. Upcroft. Multimodal deep autoencoders for control of a mobile robot. In *Proc. of Australasian Conf. for Robotics and Automation (ACRA)*, 2015. 29, 142

[248] P. Sharma, H. Liu, H. Wang, and S. Zhang. Securing wireless communications of connected vehicles with artificial intelligence. In *2017 IEEE international symposium on technologies for homeland security (HST)*, pages 1–7. IEEE, 2017. 25

[249] D. Shen, Y. Huang, Y. Wang, and C. Zhao. Research and implementation of slam based on lidar for four-wheeled mobile robot. In *2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE)*, pages 19–23, Aug 2018. 20

[250] X. Shi, Y. Shen, Yo. Wang, and L. Bai. Differential-clustering compression algorithm for real-time aerospace telemetry data. *IEEE Access*, 6:57425–57433, 2018. 3, 88

[251] A. A. Shvets, A. Rakhlin, A. A. Kalinin, and V. I. Iglovikov. Automatic instrument segmentation in robot-assisted surgery using deep learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 624–628, Dec 2018. 176

[252] C. Siagian and L. Itti. Biologically inspired mobile robot vision localization. *Robotics, IEEE Transactions on*, 25(4):861–873, 2009. 90

[253] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. Autonomous mobile robots. *A Bradford Book*, 2011. 17

[254] J. Silva Almeida, L. Bezerra Marinho, J. W. Mendes Souza, E. A Assis, and P. P. Reboucas Filho. Localization system for autonomous mobile robots using machine learning methods and omnidirectional sonar. *IEEE Latin America Transactions*, 16(2):368–374, Feb 2018. 20

[255] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 31, 35

[256] M. K. Singh and D. R. Parhi. Path optimisation of a mobile robot using an artificial neural network controller. *International Journal of Systems Science*, 42(1):107–120, 2011. 24

[257] H. Sinha, J. Patrikar, E. G. Dhekane, G. Pandey, and M. Kothari. Convolutional neural network based sensors for mobile robot relocalization. In *2018 23rd International Conference on Methods Models in Automation Robotics (MMAR)*, pages 774–779, Aug 2018. 32, 39, 177

[258] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic. Dual arm manipulation—a survey. *Robotics and Autonomous systems*, 60(10):1340–1353, 2012. 27

[259] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4297–4304, Sep. 2015. 40

[260] J. Šochman and J. Matas. Learning fast emulators of binary decision processes. *International Journal of Computer Vision*, 83(2):149–163, 2009. 35

[261] K. Sommer, K. Kim, Y. Kim, and S. Jo. Towards accurate kidnap resolution through deep learning. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 502–506, June 2017. 40

[262] D. C. Sorensen. Implicitly restarted arnoldi/lanczos methods for large scale eigenvalue calculations. In *Parallel Numerical Algorithms*, pages 119–165. Springer, 1997. 95

[263] B. Stanczyk and M. Buss. Development of a telerobotic system for exploration of hazardous environments. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2532–2537. IEEE, 2004. 17

[264] A. Štimec, M. Jogan, and A. Leonardis. Unsupervised learning of a hierarchy of topological maps using omnidirectional images. *International Journal of Pattern Recognition and Artificial Intelligence*, 22(04):639–665, 2008. 3, 19, 28, 88, 93

[265] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. Ldahash: Improved matching with smaller descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 34(1):66–78, 2011. 35

[266] Z. Su, X. Zhou, T. Cheng, H. Zhang, B. Xu, and W. Chen. Global localization of a mobile robot using lidar and visual features. In *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2377–2383. IEEE, 2017. 140

[267] N. Sünderhauf, F. Dayoub, S. McMahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford. Place categorization and semantic mapping on a mobile robot. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 5729–5736. IEEE, 2016. 18

[268] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 30, 31, 177

[269] J. Tang, Y. Ren, and S. Liu. Real-time robot localization, vision, and speech recognition on nvidia jetson tx1. *arXiv preprint arXiv:1705.10945*, 2017. 41

[270] G. Tanzmeister, J. Thomas, D. Wollherr, and M. Buss. Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6090–6095. IEEE, 2014. 38

[271] J. P. Tardif, Y. Pavlidis, and K. Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2531–2538, Sept 2008. 2, 22

[272] K. Tateno, F. Tombari, I. Laina, and N. Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6243–6252, 2017. 41

[273] S. Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic Press, 2015. 27

[274] S. Theodoridis and K. Koutroumbas. Pattern recognition and neural networks. In *Advanced Course on Artificial Intelligence*, pages 169–195. Springer, 1999. 28

[275] S. Thrun. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, 1(1-35):1, 2002. 38

[276] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1-2):99–141, 2001. 95

[277] R. Triebel, H. Grimmett, R. Paul, and I. Posner. *Driven Learning for Driving: How Introspection Improves Semantic Mapping*, pages 449–465. Springer International Publishing, Cham, 2016. 23, 135

[278] L. Trujillo and G. Olague. Synthesis of interest point detectors through genetic programming. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 887–894. ACM, 2006. 35

[279] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1023–1029. IEEE, 2000. 86

[280] J. Unicomb, R. Ranasinghe, L. Dantanarayana, and G. Dissanayake. A monocular indoor localiser based on an extended kalman filter and edge images from a convolutional neural network. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, Oct 2018. 40

[281] C. Valgren, T. Duckett, and A. Lilienthal. Incremental spectral clustering and its application to topological mapping. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4283–4288. IEEE, 2007. 28, 93

[282] C. Valgren and A.J. Lilienthal. Sift, surf & seasons: Appearance-based long-term localization in outdoor environments. *Robotics and Autonomous Systems*, 58:149–156, 2010. 3, 86, 88

[283] D. Valiente, A. Gil, Ó. Reinoso, M. Juliá, and M. Holloway. Improved omnidirectional odometry for a view-based mapping approach. *Sensors*, 17(2), 2017. 2, 85

[284] D. Valiente, L. Payá, L. M. Jiménez, J. M. Sebastián, and O. Reinoso. Visual information fusion through bayesian inference for adaptive probability-oriented feature matching. *Sensors*, 18(7):2041, 2018. 22

[285] L. van der Maaten. Accelerating t-sne using tree-based algorithms. *The Journal of Machine Learning Research*, 15(1):3221–3245, 2014. 37

[286] S. Van Gassen, B. Callebaut, M. J. Van Helden, B. N. Lambrecht, P. Demeester, T. Dhaene, and Y. Saeys. Flowsom: Using self-organizing maps for visualization and interpretation of cytometry data. *Cytometry Part A*, 87(7):636–645, 2015. 95

[287] M. van Gerven and S. Bohte. *Artificial neural networks as models of neural information processing*. Frontiers Media SA, 2018. 137

[288] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004. 25

[289] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018. 23, 176

[290] C. J. Vyborny and M. L. Giger. Computer vision and artificial intelligence in mammography. *AJR. American journal of roentgenology*, 162(3):699–708, 1994. 23

[291] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan. Vision-based hand-gesture applications. *Communications of the ACM*, 54(2):60–71, 2011. 24

[292] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 627–637, Oct 2017. 39

[293] C. Wang, M. Pelillo, and K. Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. *arXiv preprint arXiv:1906.01592*, 2019. 28

[294] H. Wang, W. Yang, W. Huang, Z. Lin, and Y. Tang. Multi-feature fusion for deep reinforcement learning: Sequential control of mobile robots. In *International Conference on Neural Information Processing*, pages 303–315. Springer, 2018. 29, 142

[295] K. Wang, W. Wang, and Y. Zhuang. Appearance-based map learning for mobile robot by using generalized regression neural network. In *International Symposium on Neural Networks*, pages 834–842. Springer, 2007. 32

[296] Y. Wang, T. Bao, C. Ding, and M. Zhu. Face recognition in real-world surveillance videos with deep learning method. In *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pages 239–243. IEEE, 2017. 26, 176

[297] X. S. Wei, C. W. Xie, J. Wu, and C. Shen. Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognition*, 76:704 – 714, 2018. 26

[298] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International journal of computer vision*, 70(1):77–90, 2006. 37

[299] P. Weinzaepfel, G. Csurka, Y. Cabon, and M. Humenberger. Visual localization by learning objects-of-interest dense match regression. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5627–5636, June 2019. 40

[300] B. H. Wilcox. Robotic vehicles for planetary exploration. *Applied Intelligence*, 2(2):181–193, 1992. 17

[301] A. Wilkowski, T. Kornuta, and W. Kasprzak. *Point-Based Object Recognition in RGB-D Images*, pages 593–604. Springer International Publishing, Cham, 2015. 21

[302] P. K. Wong, L. M. Tam, K. Li, and C. M. Vong. Engine idle-speed system modelling and control optimization using artificial intelligence. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 224(1):55–72, 2010. 24

[303] P. Wozniak, H. Afrisal, R. G. Esparza, and B. Kwolek. Scene recognition for indoor localization of mobile robots using deep cnn. In *International Conference on Computer Vision and Graphics*, pages 137–147. Springer, 2018. 31, 32, 177

[304] H. Wu and S. Y. Qin. An approach to robot slam based on incremental appearance learning with omnidirectional vision. *International journal of systems science*, 42(3):407–427, 2011. 40

[305] F. Xing, Y. Xie, and L. Yang. An automatic learning-based framework for robust nucleus segmentation. *IEEE transactions on medical imaging*, 35(2):550–566, 2015. 24

[306] S. Xu, W. Chou, and H. Dong. A robust indoor localization system integrating visual localization aided by cnn-based image retrieval with monte carlo localization. *Sensors*, 19(2):249, 2019. 32, 37, 40, 134

[307] Y. Xu, M. Sun, Z. Cao, J. Liang, and T. Li. Multi-object tracking for mobile navigation in outdoor with embedded tracker. In *Natural Computation (ICNC), 2011 Seventh International Conference on*, volume 3, pages 1739–1743, 2011. 91

[308] M. H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 24(1):34–58, 2002. 25

[309] Q. Yang, D. Qu, F. Xu, F. Zou, G. He, and M. Sun. Mobile robot motion control and autonomous navigation in gps-denied outdoor environments using 3d laser scanning. *Assembly Automation*, 39, 10 2018. 21

[310] Y. Yang, Y. Li, C. Fermuller, and Y. Aloimonos. Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 27

[311] Z. Yong-guo, C. Wei, and L. Guang-liang. The navigation of mobile robot based on stereo vision. In *Intelligent Computation Technology and Automation (ICICTA), 2012 Fifth International Conference on*, pages 670–673. IEEE, 2012. 2

[312] J. Yosinski, Je. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014. 37

[313] H. F. Zaki, F. Shafait, and A. Mian. Viewpoint invariant semantic object and scene categorization with rgb-d sensors. *Autonomous Robots*, 43(4):1005–1022, 2019. 26

[314] B. Zhang, W. Huang, L. Gong, J. Li, C. Zhao, C. Liu, and D. Huang. Computer vision detection of defective apples using automatic lightness correction and weighted rvm classifier. *Journal of Food Engineering*, 146:143–151, 2015. 27

[315] X. Zhang, Y. Su, and X. Zhu. Loop closure detection for visual slam systems using convolutional neural network. In *2017 23rd International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, 2017. 41

[316] Y. Zhang and L. Wu. Classification of fruits using computer vision and a multiclass support vector machine. *Sensors*, 12(9):12489–12505, 2012. 22

[317] F. Zhong, S. Wang, Z. Zhang, and Y. Wang. Detect-slam: Making object detection and slam mutually beneficial. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1001–1010. IEEE, 2018. 41

[318] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014. 90, 91, 140

[319] Q. Zhu, S. Avidan, M.C. Yeh, and K.T. Cheng. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498, 2006. 90

[320] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3357–3364, May 2017. 176

[321] Z. Zhu, X. Wang, S. Bai, C. Yao, and X. Bai. Deep learning representation using autoencoder for 3d shape retrieval. *Neurocomputing*, 204:41–50, 2016. 26, 29, 142

[322] A. Ziegler, C. John Morse, Duane L. G. Jr., A. Jones, S. Pratt, P. E. Sandin, and N. Dussault-Smith. Autonomous surface cleaning robot for wet and dry cleaning, February 2019. US Patent 10,213,081. 17

[323] Z. Zivkovic, B. Bakker, and B. Krose. Hierarchical map building using visual landmarks and geometric constraints. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2480–2485, Aug 2005. 38

[324] Z. Zivkovic, B. Bakker, and B. Krose. Hierarchical map building and planning based on graph partitioning. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 803–809, May 2006. 3