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

USO DE DESCRIPTORES HOLÍSTICOS PARA LA LOCALIZACIÓN Y CREACIÓN DE MAPAS: UNA APROXIMACIÓN AL GRAPH-SLAM MEDIANTE APARIENCIA VISUAL

Autor: Yerai Berenguer Fernández
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

2018

UNIVERSITAS
Miguel
Hernández

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:

- *Position Estimation and Local Mapping Using Omnidirectional Images and Global Appearance Descriptors*. [7]
  Y. Berenguer, L. Payá, M. Ballesta, O. Reinoso
  Sensors. Vol 15(10) (2015)
  ISSN: 1424-8220. Ed. MDPI.
  JCR-SCI Factor de impacto: 2.033, Cuartil **Q1**.
  Web: http://doi.org/10.3390/s151026368
  DOI: 10.3390/s151026368

- *Using Omnidirectional Vision to Create a Model of the Environment: A Comparative Evaluation of Global-Appearance Descriptors*. [55]
  L. Payá, O. Reinoso, Y. Berenguer, D. Úbeda
  Journal of Sensors. Vol 2016
  ISSN: 1687-725X. Ed. Hindawi.
  JCR-SCI Factor de impacto: 1.704, Cuartil **Q2**.
  Web: http://doi.org/10.1155/2016/1209507
  DOI: 10.1155/2016/1209507

UNIVERSITAS
Miguel
Hernández

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: ***Uso de descriptores holísticos para la localización y creación de mapas: una aproximación al graph-SLAM mediante apariencia visual.***

Autor: Yerai Berenguer Fernández

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 de de 2018.

Fdo: Dr. D. Óscar Reinoso García Fdo: Dr. D. Luis Payá Castelló

UNIVERSITAS
Miguel
Hernández

UNIVERSITAS
Miguel Hernández

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. Yerai Berenguer Fernández titulado ***Uso de descriptores holísticos para la localización y creación de mapas: una aproximación al graph-SLAM mediante apariencia visual*** 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 de de 2018.

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

UNIVERSITAS
Miguel
Hernández

Abstract

Throughout the last few years, the quantity of applications that use mobile robots have increased and they are present in very diverse fields. A robot must have an internal representation of the environment through which it is going to move. This representation will allow it to estimate its position and orientation, as well as the trajectory to follow during the operation. In this way it can carry out tasks, autonomously, within that environment. This information is essential to generate the necessary signals to feed the equipped actuators. These actuators generate the desired movement in the robot.

This environment representation is done by collecting information from the surroundings through different sensors. There are many types of sensors such as lasers, encoders, GPS, sonars, visual sensors, etc... All of them provide information to the robot. The robot uses these information to generate a representation of the environment. Among all of them, it is possible to highlight the visual sensors, due to different advantages such as its reduced weight, its limited energy consumption and, above all, its multiple possibilities of configuration, which makes it a suitable sensor for an infinity types of applications. These sensors provide images that have rich information that can be exploited in different ways.

Among all the possible configurations of visual sensors there is one that provides information in all directions around the robot. This configuration consists of a catadioptric system formed by a conventional camera pointing to the base of a convex mirror, which can be spherical, conical, elliptical, hyperbolic or parabolic. The entire environment is reflected in the mirror, and the camera captures this reflection, generating an omnidirectional image, which contains information about the environment with a 360° visual field around the axis of the catadioptric system. For the development of this Doctoral Thesis, the hyperbolic mirror has been chosen as a mirror of the system. This kind of mirror has different advantages that will be detailed throughout the document.

The visual information obtained is very rich and this can be a problem when working with it. For this reason, the use of thechniques to extract the most relevant information is essential to do possible its management. There are different methods to perform this task in order to generate maps of the environment based on descriptors that store the relevant information. The first one is based on the extraction and description of characteristic points of the scenes, which has reached a certain maturity because it has been used in many of the known mapping and localization algorithms. However, these systems have several disadvantages due to the high computational cost to manage it and their low robustness against changes in the environment. The second method consists of working with the general information of the scenes, generating a single descriptor that collects the information of each image as a whole, without the extraction of local characteristics. It permits to extract a single holistic descriptor per image that collects its global information. This approach is more recent than the previous one and, normally, leads to simpler localization algorithms, conceptually speaking. Due to the immaturity of these methods, it is necessary to carry out exhaustive studies to prove their validity in tasks of mapping and location. In this Doctoral Thesis, localization and mapping techniques are developed using this image description approach.

With this approach in mind, various algorithms, collected in each of the chapters of the Doctoral Thesis, have been proposed. First, a program to generate omnidirectional images in virtual environments is proposed to solve the problem of acquiring omnidirectional im-

ages with a real catadioptric system. This program permits to confgurate the catadioptric system by testing it using virtual images. This saves on costs and time because the acquisition of different actual databased using different system configurations is not necesary. Secondly, two 2D localization algorithms are proposed. they use a descriptor based on the Radon transform, starting from different initial hypotheses. These algorithms are compared with methods based on the extraction of characteristics and with methods based on other known global appearance descriptors. Third, a relative height estimation algorithm has been developed using the global appearance of the scenes and compared with an alternative method based on the description of the local characteristics of the scenes. Finally, a method of graph-SLAM (Creation of topological maps and localization simultaneously) is proposed. All the developed algorithms have been validated through experiments that use diverse databases formed by omnidirectional images.-

Resumen

A lo largo de los últimos años, las aplicaciones de los robots móviles han aumentado de una forma considerable y cada vez están más presentes en muy diversos ámbitos. Para que estos robots sean capaces de realizar tareas de forma autónoma dentro de diferentes entornos, es necesario que posean una representación interna del mismo con el objetivo de poder estimar su posición y orientación dentro de él, así como la trayectoria seguida durante su funcionamiento. Esta información es imprescindible para poder generar las señales necesarias para alimentar a los actuadores equipados y que generen el movimiento deseado en el robot.

Esta representación de entorno se realiza recogiendo información de los alrededores a través de diferentes sensores. Existen multitud de tipos de sensores como pueden ser láseres, encoders, GPS, sonars, sensores visuales, etc... Todos ellos proporcionan información de diferente tipo al robot para que sea capaz de generar una representación del entorno. De entre todos ellos se puede destacar los sensores visuales, debido a diversas características como su reducido peso, su consumo energético limitado y sobre todo a su infinidad de posibilidades de configuración, lo cual los convierte en unos sensores aptos para infinidad de aplicaciones. Estos sensores proporcionan imágenes que poseen información de gran riqueza que puede ser aprovechada de diferentes formas.

De entre todas las posibles configuraciones de sensores visuales existe una que proporciona información en todas las direcciones alrededor del robot: los sistemas visuales catadióptricos. Estos sistemas están formados por una cámara convencional apuntando a la base de un espejo convexo, el cual puede ser esférico, cónico, elíptico, hiperbólico o parabólico. El entorno completo se refleja en el espejo, y la cámara captura este reflejo, dando lugar a una imagen omnidireccional, que contiene información del entorno con un campo visual de 360ž alrededor del eje del sistema catadióptrico. Para el desarrollo de esta Tesis Doctoral se ha elegido el espejo hiperbólico como espejo del sistema debido a diferentes ventajas que se detallarán a lo largo del documento.

La información visual obtenida es muy rica y esto puede suponer un problema a la hora de trabajar con ella. Por este motivo se hace indispensable el uso de técnicas de extracción de la información más relevante para que su manejo sea posible. Existen diferentes métodos que permiten realizar esta tarea con el fin de generar mapas del entorno basados en descriptores que almacenan la información relevante. El primero de ellos está basado en la extracción y descripción de puntos característicos de las escenas, el cual ha alcanzado cierta madurez debido a que ha sido utilizado en muchos de los algoritmos de creación de mapas y localización conocidos hoy en día. Sin embargo, estos sistemas presentan diversos inconvenientes debido al elevado coste computacional que requiere su manejo al trabajar con mapas extensos y su baja robustez frente a cambios en el entorno. El segundo método consiste en trabajar con la información general de las escenas, generando un único descriptor que recoja la información de cada imagen de manera conjunta, sin la extracción de características locales. Con esto lo que se consigue es extraer un único descriptor holístico por imagen que recoge su información global. Este enfoque es más reciente que el anterior y, normalmente, conduce a algoritmos de localización mas sencillos, conceptualmente hablando. El desarrollo de estos enfoques no ha sido tan amplio, por lo que se requiere realizar exhaustivos estudios para probar su validez en tareas

de creación de mapas y localización. En la presente Tesis Doctoral se desarrollan técnicas de localización y creación de mapas utilizando este enfoque de descripción de imágenes.

Con este enfoque en mente, se han propuesto diversos algoritmos recogidos en cada uno de los capítulos de la Tesis Doctoral. En primer lugar, debido a la problemática de adquirir imágenes omnidireccionales con un sistema catadióptrico, se propone un programa de generación de imágenes omnidireccionales en entornos virtuales. Esto ayuda a la hora de configurar el sistema catadióptrico utilizado, ya que se podrá testar su funcionamiento a partir de las imágenes virtuales. Con ello se ahorra en costes y en tiempo ya que si se tuviera que capturar diferentes bases de datos reales con diferentes sistemas catadióptricos configurados de maneras diversas se necesitaría más tiempo y más dinero para la adquisición de los diferentes sistemas. En segundo lugar, se proponen dos algoritmos de localización en el plano del suelo que utilizan un descriptor basado en la transformada de Radon, partiendo de diferentes hipótesis iniciales y se comparan con métodos basados en la extracción de características y en descriptores de apariencia global más conocidos. En tercer lugar, se ha desarrollado un algoritmo de cálculo de altura relativa utilizando la apariencia global de las escenas y se compara con un método alternativo basado en la descripción de características locales de las escenas. Finalmente, se propone un método de graph-SLAM (Creación de mapas topológicos y localización de manera simultánea). Todos los algoritmos desarrollados han sido validados a través de experimentos que usan diversas bases de datos de imágenes omnidireccionales tanto virtuales como capturadas en entornos de trabajo reales.

Agradecimientos

En primer lugar, quiero dar las gracias a mis directores de tesis, Óscar Reinoso y Luis Payá, por su apoyo y confianza durante todo este tiempo. Gracias de corazón por todos vuestros consejos y revisiones que han ido dando forma a este trabajo.

También quiero dar las gracias a todos los miembros del grupo ARVC: Óscar, Luis Payá, Luis Miguel, David Valiente, Arturo, José María, David Úbeda y Mónica, por todo vuestro apoyo y colaboración siempre que os he necesitado. Especialmente, quiero dar las gracias a mis compañeros de laboratorio, quienes han compartido conmigo la mayor parte del tiempo durante este camino: Sergio Cebollada, gracias por compartir todos esos momentos dentro y fuera de la universidad, entre ellos existen multitud de anécdotas dignas de recordar para siempre. Adrián Peidró, gracias por todos tus consejos indispensables en el inicio de mi andadura como investigador. Cristóbal Parra, gracias por alegrar las mañanas con tu sentido del humor y por los momentos que hemos compartido dentro y fuera del laboratorio. Valerio, gracias por tu apoyo y tus ánimos, siempre has creído en mí, levantas el ánimo a cualquiera con el que hables. A todos vosotros os considero amigos más que compañeros de trabajo. Y también quiero dar las gracias a las personas que han pasado por el laboratorio aunque haya sido durante periodos breves: Vicente, Héctor, Amalia, Julio, Alba, Óscar,... Todos habéis conseguido crear un ambiente de trabajo perfecto, a pesar de las novatadas constantes.

Mi agradecimiento a mis responsables durante las estancias en Edmonton (Canadá) y en Bristol (Reino Unido), a la vez que a todas las personas que conocí lejos de mi hogar y con las que compartí innumerables experiencias.

Gracias Francisco Irles, por confiar en mí y recomendarme para realizar este trabajo. Sin tu apoyo esto no habría sido posible.

Fuera de la universidad, quiero dar las gracias a toda mi familia. Vuestro apoyo incondicional es indispensable para luchar por todo lo que me propongo. Gracias tías (José, Bienve, Mari, Noelia), por todos los días que hemos pasado juntos, por regalarme vuestro amor y cariño cada día que he pasado con vosotras. Gracias Chumari, la tata supo elegir a quién traer a la familia, tu generosidad es digna de admiración. Chiquitín, gracias por tus consejos y enseñanzas. Tío José, gracias por todo tu apoyo, aunque ahora estés mucho tiempo en Suiza, cuando vienes alegras la semana. Tío Fran, gracias por creer siempre en mí y compartir tu alegría conmigo. Tío Jorge, gracias por tus charlas. Tetes (Manolín, Carlos, Fran), gracias por vuestras bromas, no cambiéis. También quiero dar las gracias a todos mis primos por ayudarme y apoyarme siempre que los he necesitado.

Quiero dar las gracias a todos mis amigos que, durante todos estos años, han creído en mí y me han ayudado a llegar hasta aquí. También agradecer a mi amiga Amparo, quién es una persona muy importante para mí, pese a nuestras discusiones lingüísticas.

Doy las gracias a mis iaios, Paco y Encarna, quienes han supuesto una referencia a lo largo de mi vida. Sí iaia, me he sacado el doctorado, como tú dices. Gracias a mis abuelitos, Ventura y Catalina, por vuestro apoyo día a día. No puedo expresar con palabras todo lo que les debo a estas cuatro personas, quienes siempre supondrán una parte muy importante de mi persona. Gracias tía Lolín, tío Manolín, tía Pili, tío Pepe, tío Antonio y Maricarmen por dar siempre lo mejor de vosotros mismos. Vais siempre en mi corazón.

Hay personas que aunque no sean de mi propia sangre siempre serán mi familia. José Carlos y María José, gracias por acogerme y apoyarme durante todo este tiempo, siempre

habéis creído en mí. No me olvido de mi cuñadita Carmen, te he visto crecer y siempre te consideraré como mi hermana pequeña.

Cada uno de los logros que he conseguido hasta ahora se lo debo a mis tres fantásticos: mi padre, mi madre y mi hermano Noé. Vosotros me habéis moldeado y convertido en la persona que soy. Papá, mamá, gracias por dar absolutamente todo por mí. Tete, gracias por tantos innumerables momentos, de los cuales incluyen peleas, pero aun así todos los recuerdo con inmensa felicidad. Y gracias por tu apoyo, ya sabes que nos tendremos siempre para apoyarnos mutuamente. También quiero mencionar a la última incorporación, mi cuñada Rocío, gracias por regalar alegría en cualquier momento.

Por último, pero no menos importante, quiero darle las gracias a la persona que me hace levantarme con ilusión cada mañana. Alba, gracias por ser el trampolín en el fondo del agujero, levantándome el ánimo siempre que el pesimismo me invade. Sin tu apoyo todo esto habría sido imposible. Empieza nuestro futuro, y sé que juntos nada nos detendrá en nuestro camino. Te quiero.

En general, gracias a todas las personas que, durante este largo camino, han estado a mi lado apoyándome y regalándome su tiempo.

Gracias por ser como sois y por quererme tal como soy.

A mi familia

UNIVERSITAS
Miguel
Hernández

Índice general

UNIVERSITAS
Miguel
Hernández

Lista de Tablas

UNIVERSITAS
Miguel
Hernández

Lista de Figuras

1 Introducción

1.1 Estado del arte

Hoy en día, la presencia de robots en nuestra sociedad ha aumentado considerablemente. Inicialmente, se usaron para llevar a cabo algunas tareas que resultaban muy exigentes o peligrosas para las personas. Sin embargo, en la actualidad, se utilizan en otras innumerables tareas con diferentes propósitos, gracias a la evolución de los equipos y técnicas de percepción y computación empleadas. Actualmente, es posible diseñar robots más autónomos que no requieren la intervención humana para llevar a cabo sus tareas. Muchos de estos robots también son móviles y para cumplir con su tarea deben poder planificar una trayectoria para llegar a los puntos de destino y navegar hacia ellos mientras evitan los obstáculos presentes en el entorno. Para llevar a cabo la tarea de navegación de una manera eficiente, es necesario llevar a cabo dos tareas fundamentales. Por un lado, el robot debe crear una representación interna del entorno inicialmente desconocido (mapa) y, por otro, debe estimar su posición dentro del mapa. Por ello, el robot necesita uno o más sensores para extraer información del entorno. Hay muchos tipos de sensores que les brindan información útil, como sensores táctiles, encoders, sensores láser o de visión. Esta información puede usarse tanto para construir el modelo del entorno como para estimar la posición del robot.

Los sensores de visión se han convertido en una de las opciones más extendidas en robótica móvil gracias a la gran cantidad de información que proporcionan al robot. García et al. [21] presentan una recopilación de métodos de creación de mapas y localización utilizando sistemas de visión. Estos sistemas pueden tener diversas configuraciones, como cámaras individuales, cámaras estéreo, sistemas con arrays de cámaras, sistemas catadióptricos, etc. Los sistemas de visión catadióptricos consisten en una sola cámara

que apunta a un espejo convexo. Esta configuración permite tomar imágenes con un campo de visión de 360 grados alrededor del eje del espejo. La riqueza de la información que capturan es la razón por la cual se ha elegido este tipo de sistema como dispositivo que proporciona la información que se tendrá en cuenta para reconocer el entorno por el que se desplaza el robot móvil. Hay muchos trabajos previos que usan sistemas de visión catadióptricos en tareas de navegación. Por ejemplo, Winters et al. [70] describen un método para la navegación de robots equipados con sensores visuales omnidireccionales y demuestran que las imágenes capturadas por dichos equipos pueden ser utilizadas para realizar tareas de localización. A veces, la información visual se combina con otra fuente de información, como encoders, GPS (Global Positioning System) o IMU (Inertial Measurement Unit). Oriolo et al. [53] presentan un método para la localización de robots humanoides usando una cámara monocular, un IMU, encoders y sensores de presión. Satici et al. [61] presentan un sistema de navegación y control para robots móviles que utiliza un sensor de visión, un IMU y un encoder.

Sin embargo, las imágenes contienen mucha información redundante que puede cambiar en diversas circunstancias, como por ejemplo, el ruido y las oclusiones. Por esta razón, es necesario extraer información relevante de cada escena para crear el mapa. Esta información debe permitir estimar la posición del robot con robustez. Hay dos enfoques diferentes para llevar esto a cabo. Por un lado, la imagen se puede describir a través de la extracción y descripción de los puntos característicos locales de las escenas. Por ejemplo, Lowe et al. [40] llevan a cabo tareas de localización y creación de mapas utilizando SIFT (Scale-Invariant Feature Transform) y Bay et al. [6] presentan otro detector y descriptor de puntos característicos, denominado SURF (Speeded-Up Robust Features). Por otro lado, algunas obras usan la apariencia global de las escenas para crear solo un descriptor por imagen. En [54], se analizan y comparan varios métodos para obtener descriptores globales de escenas panorámicas para comprobar su validez en la creación y localización de mapas. Lamayoría de estos descriptores holísticos se pueden usar en tiempo real, porque el tiempo de cálculo para generarlos y manejarlos es bajo, y generalmente conducen a algoritmos de localización y creación de mapas más directos que los algoritmos basados en la extracción de características. En los últimos años, algunos trabajos se han centrado en el uso de imágenes omnidireccionales como la única fuente de información para resolver las tareas de creación de mapas y localización utilizando descriptores basados en la apariencia global de las escenas, como [55, 7, 18, 65, 41, 20, 22]. Payá et al. [55] presentan una evaluación comparativa de diferentes técnicas de apariencia global en tareas de creación de mapas utilizando imágenes omnidireccionales. Berenguer et al. [7] presentan diferentes métodos en tareas de localización y creación de mapas utilizando la apariencia global de imágenes omnidireccionales. Fernández et al. [18] presentan un enfoque de apariencia global para llevar a cabo tareas de localización y creación de mapas simultáneamente utilizando mapas métrico-topológicos híbridos.

El mapa del entorno se puede crear utilizando dos enfoques diferentes: métrico o topológico. Por un lado, los mapas métricos representan el entorno que define las ubicaciones de algunas de las características relevantes con respecto a un sistema de coordenadas. Esta configuración permite estimar la posición del robot con precisión geométrica. Munguía et al. [45] describen un sistema de localización y creación de mapas utilizando la

información métrica obtenida por diferentes sensores; un sensor de orientación, un sensor de posición (GPS) y una cámara monocular. Hay más trabajos que utilizan este enfoque métrico en las tareas de localización, como [19, 68, 11]. Forster et al. [19] emplea localización y creación de mapas utilizando múltiples vehículos micro aéreos (MAV) en entornos desconocidos. Weiss et al. [68] presenta un sistema para permitir vuelos autónomos de un micro vehículo aéreo (MAV) que solo dispone de mediciones lentas, ruidosas, retrasadas y posiblemente a escala arbitraria. Bunschoten et al. Por otro lado, los mapas topológicos tienden a modelar el entorno como un grafo con un conjunto de nodos que corresponden a diferentes ubicaciones y las relaciones de conectividad entre ellos. Existen trabajos que utilizan este enfoque de creación de mapas, como [31] donde se usa un marco topológico para llevar a cabo SLAM (Simultaneous Localization and Mapping) en un entorno subacuático utilizando sensores de visión. Más recientemente, algunos investigadores han combinado los conceptos métricos y topológicos para generar mapas híbridos, donde la información se organiza en varias capas con diferentes niveles de detalle. Kostavelis et al. [34] crean mapas compuestos por diversas capas utilizando el concepto de mapas híbridos para llevar a cabo tareas de navegación jerárquicas. Dayoub et al. [15] presentan un sistema de creación de mapas y navegación que permite al robot móvil planificar trayectorias y evitar obstáculos usando un mapa topométrico compuesto por un grafo de localización global el cual está formado por una nube de puntos 3D local unida a cada nodo.

1.2 Motivación

Esta tesis estudia la utilización de descriptores holísticos sobre imágenes omnidireccionales en tareas de localización y creación de mapas en el campo de la robótica móvil. La finalidad del uso de dichos descriptores radica en la descripción de imágenes captadas por un sistema catadióptrico, tratando de aprovechar las posibilidades de proyección de la información visual omnidireccional que proporcionan este tipo de sistemas.

Estos descriptores holísticos tratan la imagen en su conjunto sin realizar ninguna segmentación ni extracción de características. Esta es la diferencia fundamental con los descriptores basados en la extracción de características locales (*landmarks*) que son los usados tradicionalmente en las tareas de localización y creación de mapas usando información visual. Los descriptores holísticos han demostrado ser capaces de estimar la pose de un robot dentro de un mapa denso de imágenes. Sin embargo, la mayoría de descriptores basados en la apariencia global de las imágenes se aplican a transformaciones previas de la imagen omnidireccional proporcionada por el sistema catadióptrico, como por ejemplo la transformación de esta información en imágenes panorámicas.

Por este motivo, en la presente tesis, se propone la utilización de descriptores **aplicados directamente a la imagen omnidireccional** con el fin de realizar la localización del robot de forma más eficiente. Este tipo de sensores proporcionan una gran cantidad de información del entorno, con un coste relativamente bajo. Sin embargo, la cantidad de información proporcionada por dichos sistemas también complica la resolución de los problemas. Por este motivo, los algoritmos desarrollados deberán ser capaces de extraer la información más relevante de las imágenes omnidireccionales con el fin de realizar las tareas de localización y creación de mapas de forma eficiente.

A la hora de comprobar el funcionamiento de cualquier nuevo algoritmo se necesita una base de datos con la cual validar los resultados. En muchas ocasiones, los resultados de los experimentos con estas bases de datos capturadas muestran resultados poco satisfactorios. Para evitar este problema y poder realizar, de forma rápida, una comprobación inicial del desempeño de los algoritmos realizando además una sintonización gruesa de los parámetros de configuración del sistema, se ha creado un programa de adquisición de imágenes omnidireccionales simulando un entorno virtual. Esto proporcionará una base de datos virtual inicial con la cual comprobar el funcionamiento de los nuevos métodos y así elegir la correcta configuración a la hora de capturar la base de datos de imágenes reales acelerando el proceso de desarrollo de nuevos algoritmos de creación de mapas y localización.

La investigación realizada en la presente Tesis Doctoral pretende profundizar en el uso de descriptores holísticos para realizar tareas más complejas de localización y creación de modelos mediante el uso de robots móviles. Algunos trabajos previos han mostrado su buen desempeño en tareas de creación de mapas densos y localización, y lo que se pretende es ver si contienen información que permita realizar otras tareas más avanzadas.

Más concretamente, se abordan tres grandes líneas. La primera de ellas consiste en la localización en el plano del suelo. Por ello se han propuesto varios algoritmos de localización y creación de mapas en el plano del suelo partiendo de diferentes hipótesis iniciales. De igual forma tampoco se han encontrado referencias que traten el tema de la estimación de la altitud de un robot utilizando descriptores de apariencia global directamente sobre imágenes omnidireccionales. Y debido al creciente interés de los vehículos aéreos como plataformas de navegación y el uso de sensores visuales, la segunda línea propuesta se basa en un algoritmo de estimación de altura topológica del robot utilizando este tipo de descriptores. Y en tercer lugar, existen robots móviles que necesitan localizarse a la vez que van creando el mapa del entorno. Teniendo en cuenta este enfoque, tradicionalmente se utilizan varios sensores para realizar esta tarea simultánea de localización y creación de mapas denominada SLAM (*Symultaneous Localization And Mapping*). También existen trabajos que utilizan descriptores basados en la extracción de características con este propósito. Sin embargo, en esta tesis se propone el uso de imágenes omnidireccionales directamente como única fuente de información utilizando descriptores holísticos para describirlas y realizar tareas de SLAM en dos dimensiones (en el plano del suelo).

1.3 Objetivos

Esta tesis tiene como objetivos principales el desarrollo de diferentes algoritmos de localización y creación de mapas utilizando descriptores holísticos sobre imágenes omnidireccionales. Dichos objetivos pueden dividirse en los siguientes puntos:

- Creación de un programa capaz de generar imágenes omnidireccionales a partir de un entorno creado sintéticamente. En el Capítulo 3 se muestra el programa propuesto y varios ejemplos de bases de datos creadas.

- Propuesta de algoritmos de localización en dos dimensiones haciendo uso de imágenes omnidireccionales y descriptores holísticos. Para ello se parte de diferentes supuestos iniciales: (a) las posiciones de los nodos del mapa son conocidas y (b) las posiciones de los nodos del mapa son desconocidas, por lo que solo se poseen las imágenes omnidireccionales del mapa. En el Capítulo 4 se muestran los algoritmos propuestos y los resultados obtenidos tras su utilización.

- Desarrollo de un algoritmo de estimación de altura relativa haciendo uso de imágenes omnidireccionales y describiéndolas únicamente mediante descriptores de apariencia global. En el Capítulo 5 se presenta el algoritmo propuesto para calcular la altura relativa del robot y los resultados obtenidos.

- Creación de un algoritmo capaz de desempeñar tareas de graph-SLAM empleando como fuente de información únicamente imágenes omnidireccionales. Haciendo uso de descriptores holísticos para su descripción. En el Capítulo 6 se muestra el algoritmo de graph-SLAM propuesto junto con los resultados obtenidos.

1.4 Marco de la tesis

Esta tesis ha sido desarrollada bajo un marco sostenido por diferentes ramas relacionadas con la investigación, tales como subvenciones, proyectos y colaboraciones.

1.4.1 Becas

Este trabajo ha sido desarrollado con ayuda de una beca dentro del programa estatal de promoción del talento y su empleabilidad en I+D+i, del Ministerio de Economía y Competitividad, denominada “Ayuda para contratos predoctorales para la formación de doctores”, convocatoria de 2014, con referencia BES-2014-067845 y una duración de 3 años como periodo predoctoral.

1.4.2 Estancias de investigación en el extranjero

Durante el desarrollo de la tesis doctoral se han realizado dos estancias breves en centros de investigación en el extranjero. Las cuales se detallan a continuación:

- Una estancia breve financiada por el Ministerio de Economía y Competitividad del Gobierno Español a través de una beca, con referencia EEBB-I-16-10877. Esta beca hizo posible una estancia de 4 meses a finales del 2016 en el departamento de *Computer Science* de la Universidad de Alberta en Edmonton, Canadá.

- Una estancia breve financiada por el Ministerio de Economía y Competitividad del Gobierno Español a través de una beca, con referencia EEBB-I-17-11962 la cual hizo posible una estancia de 4 meses en 2017 en el departamento de *Computer Science* de la Universidad de Bristol en Bristol, Reino Unido.

1.4.3 Proyectos de investigación

La presente tesis se encuentra enmarcada dentro de tres proyectos del grupo de investigación ARVC del Departamento de Ingeniería de Sistemas y Automática de la Universidad Miguel Hernández de Elche. A continuación se describen brevemente dichos proyectos:

- Proyecto: *Localización y Creación de Mapas Visuales para Navegación de Robots con 6 GDL*.

  Financiado por: Generalitat Valenciana

  Duración: Enero 2015 - Dic 2016

  Resumen: La realización de tareas por parte de robots móviles que se desenvuelven por un entorno desconocido es una de las líneas de investigación abiertas en la actualidad y que previsiblemente tendrá mayor repercusión a medio plazo. Para ello es necesario poder tener como referencia una información lo más precisa y detallada posible con objeto de que el robot o los robots que se encuentran ejecutando una determinada tarea pueden localizarse dentro del entorno de trabajo. Durante los últimos años se ha trabajado de forma notable y con excelente resultados en esta línea de investigación de creación de mapas de entornos a través de los cuales se pudieran localizar los robots en un proceso conjunto (Simultaneous Localization And Mapping). El grupo de investigación proponente se ha centrado durante los últimos años en esta línea de investigación teniendo como datos de partida para la creación del mapa, la información visual de cada uno de los sistemas de visión con que cuenten los robots. A partir de esta información visual, se extraen mapas métricos con objeto de que los robots puedan localizarse y desarrollar tareas de navegación de una forma lo más precisa posible.

  Aportes: El aporte de la presente Tesis Doctoral al proyecto se basa en la creación de un algoritmo graph-SLAM basado en el uso de descriptores de apariencia global, disponiendo de imágenes omnidireccionales como única fuente de información del robot.

- Proyecto: *Navegación de Robots en Entornos Dinámicos Mediante Mapas Compactos con Información Visual de Apariencia Global*.

  Financiado por: CICYT Ministerio de Ciencia e Innovación

  Duración: 01/09/2014 al 31/05/2017

  Resumen: La realización de tareas por parte de robots móviles que se desenvuelven por un entorno desconocido es una de las líneas de investigación abiertas en la actualidad y que previsiblemente tendrá mayor repercusión a medio plazo. Para ello es necesario poder tener como referencia una información lo más precisa y detallada posible con objeto de que el robot o los robots que se encuentran ejecutando una determinada tarea pueden localizarse dentro del entorno de trabajo. Durante los últimos años se ha trabajado de forma notable y con excelente resultados en esta línea de investigación de creación de mapas de entornos a través de los cuales se pudieran localizar los robots en un proceso conjunto (Simultaneous Localization And Mapping). El grupo de investigación proponente se ha centrado durante los

últimos años en esta línea de investigación teniendo como datos de partida para la creación del mapa, la información visual de cada uno de los sistemas de visión con que cuenten los robots. A partir de esta información visual, se han configurado mapas métricos con objeto de que los robots puedan localizarse y desarrollar tareas de navegación de una forma lo más precisa posible. Dentro de esta propuesta de investigación, el grupo proponente ha obtenido notables resultados. Sin embargo, existen hoy en día incertidumbres y mejoras a satisfacer con objeto de poder utilizar dichos mapas de una forma más eficiente. Uno de los grandes problemas actualmente existentes es el tratamiento de la información visual que se ha modificado en los entornos a medida que los robots desarrollan sus tareas. Los mapas creados deben ser actualizados de forma paulatina teniendo en cuenta la dinámica realmente existente en los entornos de trabajo. Otra de las propuestas abiertas consiste en integrar en dichos mapas información de apariencia global a la vez que se compatibilizan la estructura métrica de los mismos, de manera que la localización del robot sea más efectiva en grandes entornos e incluso poder acometer una localización jerárquica dentro del mapa. Es en este ámbito donde se centra el presente proyecto de investigación, en el que a partir de los resultados alcanzados hasta el momento se plantea acometer nuevas líneas de investigación consistentes en desarrollar mapas visuales dinámicos teniendo en cuenta la información semántica y topológica aportada por los sistemas de visión todo ello en entornos de 6 grados de libertad.

Aportes: El aporte de la presente Tesis Doctoral al proyecto se basa diferentes desarrollos. Por un lado, la creación de un algoritmo de localización en 2D basado en la apariencia global de imágenes omnidireccionales almacenadas en un mapa tipo rejilla. Y por otro lado, la creación de un algoritmo de estimación de altura relativa basado en descriptores holísticos de imágenes omnidireccionales.

- Proyecto: *Creación de Mapas Mediante Métodos de Apariencia Visual para la Navegación de Robots*.

  Financiado por: CICYT Ministerio de Ciencia e Innovación

  Duración: 01/01/2017 al 31/12/2019

  Resumen: Para que un robot móvil pueda realizar una tarea de manera autónoma, debe ser capaz de moverse por cualquier tipo de entorno. Para ello, es necesario que dicho robot construya un modelo del entorno que le permita estimar su posición y navegar hacia los puntos destino. La creación de mapas y navegación es una línea de investigación muy activa, en la que se han centrado numerosos investigadores que han desarrollado muy diversos algoritmos usando diferente información sensorial. Hasta el momento la mayor parte de los esfuerzos se han centrado en establecer modelos de entornos a partir de información puntual y relevante del mismo sin considerar un estudio global de la escena en su conjunto.

  En respuesta a este desafío, este proyecto plantea la mejora y desarrollo de nuevos mecanismos que permitan un modelado eficiente, robusto y preciso de entornos, haciendo uso de sistemas de visión omnidireccional. El grupo de investigación proponente ha avanzado en estas áreas en los últimos años, desarrollando diversos algoritmos de creación de mapas, localización, SLAM y exploración a partir de la información proporcionada por diferentes tipos de sistemas de visión montados sobre

los propios robots. Para ello, ha explorado con profundidad las posibilidades que ofrecen los métodos de descripción de escenas basados en la extracción de características locales y los basados en la apariencia visual global, llegando a resultados notables en estas áreas.

Aportes: El aporte de la presente Tesis Doctoral al proyecto se basa en la creación de mapas locales utilizando descriptores de apariencia global y un algoritmo masa-muelle.

1.5 Publicaciones que soportan esta tesis

Las principales implementaciones y contribuciones realizadas en esta tesis están respaldadas por un conjunto de publicaciones en revistas clasificadas en JCR Science Edition. Los siguientes artículos de revista respaldan el trabajo realizado en este documento, que representan el resultado de la investigación bajo el alcance de esta tesis, con relación directa a la motivación y los objetivos ya establecidos:

- *Position Estimation and Local Mapping Using Omnidirectional Images and Global Appearance Descriptors*. [7]
  Y. Berenguer, L. Payá, M. Ballesta, O. Reinoso
  Sensors. Vol 15(10) (2015)
  ISSN:1424-8220. Ed. MDPI.
  JCR-SCI Impact Factor: 2.033, Cuartil **Q1**.
  Web: http://doi.org/10.3390/s151026368
  DOI: 10.3390/s151026368

- *Using Omnidirectional Vision to Create a Model of the Environment: A Comparative Evaluation of Global-Appearance Descriptors*. [55]
  L. Payá, O. Reinoso, Y. Berenguer, D. Úbeda
  Journal of Sensors. Vol 2016
  ISSN:1687-725X. Ed. Hindawi.
  JCR-SCI Factor de impacto: 1.704, Cuartil **Q2**.
  Web: http://doi.org/10.1155/2016/1209507
  DOI: 10.1155/2016/1209507

El primer artículo se presentan varios algoritmos para la localización y creación de mapas topológicos de un robot móvil, usando la información suministrada por un sistema de visión omnidireccional instalado sobre el propio robot. Para extraer la información más relevante de las escenas se utilizan descriptores holísticos. El trabajo es el resultado de los Capítulos 3 y 4. En el segundo de los artículos se comparan diversos algoritmos de descripción de escenas y se optimizan sus parámetros para ser usados de manera eficiente en procesos de construcción de mapas del entorno utilizando visión omnidireccional. Dicho trabajo es el resultado del estudio de descriptores holísticos utilizados a lo largo de todos los capítulos de la presente tesis.

Estos artículos han sido añadidos en el Apéndice 8.

1.6 Estructura

El documento se estructura de la siguiente manera:

- El Capítulo 2 muestra las herramientas matemáticas utilizadas a lo largo de los demás capítulos de la tesis. Dentro de estas herramientas se encuentran los diferentes descriptores holísticos utilizados, las diferentes técnicas para compararlos y un conjunto de algoritmos adicionales empleados en diferentes tareas dentro de los algoritmos presentados en otros capítulos.
- El Capítulo 3 presenta el concepto de visión omnidireccional junto con la descripción del sistema catadióptrico utilizado para la realización de los experimentos de la tesis. A su vez, en este capítulo, se propone un algoritmo de creación de imágenes virtuales y se muestran diferentes ejemplos de imágenes generadas en diferentes entornos.
- El Capítulo 4 propone diferentes algoritmos de localización y creación de mapas en 2D. Estos algoritmos se emplean para resolver problemas partiendo de diferentes supuestos iniciales. Uno de ellos sirve para localizar al robot dentro de un mapa previamente adquirido, y el otro se utiliza en el caso de que el robot tenga las imágenes del mapa pero no sepa en qué posición se han capturado.
- En el Capítulo 5 se desarrolla un algoritmo para calcular la altura relativa de un robot con respecto a una posición de referencia. En él se muestran los resultados obtenidos utilizando tanto bases de datos virtuales como reales y se compara con un método alternativo basado en la extracción de características locales.
- El Capítulo 6 presenta un algoritmo para localizar al robot a la vez que va creando el mapa del entorno. Se utilizan diversos conjuntos de imágenes, capturadas en entornos de interior, bajo condiciones de trabajo reales, mientras el robot seguía una trayectoria, para comprobar su funcionamiento.
- Por último, el Capítulo 7 recoge las principales conclusiones de la tesis y las futuras líneas de investigación propuestas.

1.7 Resumen de materiales, métodos y discusión de resultados

En este punto se muestra un resumen de los materiales y métodos utilizados en la realización de las investigaciones presentadas en la tesis doctoral. A su vez se exponen los resultados obtenidos y una discusión de los mismos.

1.7.1 Materiales

Para la realización de las investigaciones de la presente tesis se han utilizado los siguientes materiales:

- Diferentes modelos de cámaras CCD: DFK-21BF04 y DFK-41BF02, cuyas características vienen detalladas en el Capítulo 3.
- Un espejo hiperbólico: Eizho Wide 70, cuyas características vienen detalladas en el Capítulo 3.
- Bases de datos de imágenes omnidireccionales capturadas en la Universidad Miguel Hernández de Elche utilizando los sensores enumerados anteriormente.
- Diferentes bases de datos de imágenes omnidireccionales capturadas en la Universidad de Bielefeld.

1.7.2 Métodos

Los métodos empleados en la realización de las investigaciones de la presente tesis se enumeran a continuación:

- Herramientas matemáticas empleadas: Firma de Fourier, transformada de Radon, Descriptor *gist*, HOG, Distancia euclídea, POC, método masa-muelle y el análisis de Procrustes. Todas ellas detalladas en el Capítulo 2.
- Método de creación de imágenes omnidireccionales a partir de entornos virtuales. Dicho método es el propuesto en el Capítulo 3.
- Varios métodos de localización en el plano del suelo utilizando imágenes omnidireccionales y descriptores holísticos. Dichos métodos se detallan en el Capítulo 4.
- Dos métodos de estimación de altura relativa de un robot. Uno de ellos utiliza la apariencia global de las escenas y el otro emplea descriptores basados en la extracción de características locales. Dichos métodos han sido propuestos en el Capítulo 5.
- Un método de graph-SLAM utilizando únicamente información visual capturada por un sistema de visión omnidireccional. Dicho método se propone en el Capítulo 6.

1.7.3 Resultados y discusión

Como consecuencia del periodo de investigación bajo el marco de esta tesis, se han obtenido distintos resultados, los cuales han permitido realizar diversas aportaciones. Las más relevantes han sido publicadas en revistas y congresos, tanto nacionales como internacionales. En este sentido, a continuación se expone un resumen de los resultados obtenidos de los métodos propuestos en cada uno de los capítulos y una discusión de los mismos.

- Capítulo 3: Se ha consiguido realizar un programa de simulación capaz de crear imágenes omnidireccionales a partir de entornos virtuales creados por el usuario. Esto permitirá poder desarrollar y probar nuevos algoritmos de localización y creación de mapas más rápidamente, debido a la rapidez de adquisición de estas bases de datos de imágenes virtuales.

- Capítulo 4: Han sido resueltos varios problemas de localización partiendo de diferentes situaciones iniciales. Por un lado, se consigue que el robot consiga localizarse dentro de un mapa previamente creado, comparando únicamente información visual. Por otro lado, se consigue que el robot cree un mapa local utilizando las imágenes más cercanas a la posición donde se encuentra en ese momento y se localice dentro de él. Estos métodos han demostrado ser robustos frente ruido y cambios en el entorno.

- Capítulo 5: Se consigue calcular la altura relativa del robot con respecto a una posición anterior. Esta estimación de altura se obtiene utilizando únicamente información visual y descriptores holísticos. Dicho método ha demostrado ser robusto frente a ruido y oclusiones en las imágenes omnidireccionales. Además, se ha realizado una comparación con otro método basado en extracción de características locales en la cual se demuestra la robustez del algoritmo.

- Capítulo 6: El método propuesto en este capítulo es capaz de realizar tareas de graph-SLAM utilizando únicamente imágenes omnidireccionales. Los resultados obtenidos muestran la efectividad en la creación del mapa y en la localización.

UNIVERSITAS
Miguel
Hernández

2 Herramientas matemáticas empleadas

En el campo de la robótica y la visión por computador existen infinidad de herramientas matemáticas empleadas para llevar a cabo tareas de localización y creación de mapas. En este capítulo haremos un recorrido por las herramientas matemáticas utilizadas dentro de los algoritmos de esta tesis. En primer lugar se presentarán los fundamentos de los descriptores de apariencia global utilizados, los cuales son: La firma de Fourier, la transformada de Radon, el descriptor *gist* y, por último, el descriptor basado en HOG (Histograma de Orientación del Gradiente). También se presentan los dos métodos utilizados para calcular la diferencia entre dos descriptores de apariencia global: la distancia euclídea y la correlación por fase (POC). Por último se presentan dos métodos que se utilizarán para la creación de modelos del entorno y para conocer la bondad de estos modelos. Por un lado, se detalla el funcionamiento del método de posicionamiento de partículas denominado masa-muelle; y por otro lado, se presenta el algoritmo Procrustes necesario para comparar la forma de la distribución espacial de dos conjuntos de puntos, independientemente de su escala y rotación relativa.

2.1 Descriptores holísticos

Los métodos basados en la apariencia global de escenas constituyen una alternativa robusta en comparación con los métodos basados en la extracción de puntos característicos. La clave es que los descriptores de apariencia global representan el entorno a través de características de alto nivel que se pueden interpretar y manejar fácilmente, y con un coste computacional razonablemente bajo.

Cualquier nuevo método de descripción de aspecto global debería satisfacer algunas propiedades: (1) debería ejercer un efecto de compresión en la información de la

imagen; (2) debería existir una correspondencia entre la distancia entre dos descriptores y la distancia métrica entre las dos posiciones reales donde se capturaron las imágenes; (3) el coste computacional para calcularlos y compararlos debe ser bajo, de modo que puedan ser usados en tareas en tiempo real; (4) debería proporcionar robustez contra el ruido, cambios en las condiciones de iluminación, oclusiones y cambios en la posición de algunos objetos en el entorno; y, finalmente, (5) debería contener información de la orientación que tenía el robot cuando capturó la imagen.

En este capítulo se detallan los descriptores de apariencia global utilizados como herramientas a la hora de desarrollar los algoritmos de los posteriores capítulos de esta tesis.

2.1.1 Firma de Fourier

El análisis en el dominio de Fourier es una herramienta matemática muy utilizada en el campo del análisis de funciones periódicas a través de su descomposición en una suma infinita de funciones senoidales simples, las cuales forman la serie de Fourier:

$$y(x) = \frac{b_0}{2} + \sum_{n=1}^{\infty} [b_n \cos(nx) + c_n \text{sen}(nx)] \tag{2.1}$$

donde b_n y c_n son los coeficientes de la transformada de Fourier de $y(x)$.

La transformada de Fourier que nos permite representar la función en el dominio de la frecuencia es una extensión de la serie de Fourier para funciones en general, no periódicas. Está definida por la siguiente ecuación:

$$\mathcal{F}\{f(t)\} = F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt \tag{2.2}$$

Esta expresión sirve cuando las funciones son continuas, pero en el caso de los datos utilizados en esta tesis se tratan de datos discretos. Para transformar funciones discretas existe la Transformada de Fourier Discreta (DFT). La transformada de Fourier discreta nos permite transformar la secuencia de números $\{a_n\} = \{a_0, a_1, a_2, ..., a_{N-1}\}$ en la secuencia compleja $\{An\} = \{A_0, A_1, A_2, ..., A_{N-1}\}$ a través de la siguiente ecuación:

$$\{A_n\} = \mathcal{F}[\{a_n\}] = \int_{n=0}^{N-1} a_n e^{-j\frac{2\pi}{N}kn}; \;\; k = 0, 1, ..., N-1 \tag{2.3}$$

donde N es número total de elementos de la secuencia.

Por tanto toda señal puede representarse mediante su espectro en el dominio de la frecuencia. En el caso de las imágenes, se trata de señales discretas compuestas por píxeles. Ishiguro et al. [30] proponen la creación de mapas visuales aplicando la

Transformada de Fourier en imágenes panorámicas. Para ello pasan cada una de las filas de la imagen al dominio de la frecuencia. Menegatti et al. [42] desarrollan esta misma idea bajo el nombre de *Firma de Fourier* (FS).

De esta forma, si se parte de una imagen $im_j(x,y) \in \mathbb{R}^{N_x \times N_y}$, es posible extraer la información más relevante de la imagen a través de la transformada discreta de Fourier (DFT) de cada una de sus filas. Con este paso se obtiene una nueva matriz $d_j(u,v) \in (C)^{N_x \times N_y}$, del mismo tamaño que la imagen original. Dentro de esta matriz, la mayor parte de la información de la imagen se encuentra concentrada en los componentes de baja frecuencia de cada fila. De este modo, se puede trabajar únicamente con las k_1 primeras columnas de la matriz resultante. A esta nueva matriz de tamaño N_x filas y k_1 columnas se le llama *Firma de Fourier*. Esta firma, $d_j(u,v)$ puede descomponerse en una matriz de módulos $A_j(u,v) = |d_j(u,v)|$ y una matriz de argumentos $\phi_j(u,v)$, ambas con el mismo número de filas y se columnas que $d_j(u,v)$. La matriz de módulos es invariante ante cambios de orientación del robot en el plano del suelo, si se trabaja con imágenes omnidireccionales (y el sistema omnidireccional está montado en vertical sobre el robot, y, a su ve, el movimiento del robot está restringido al plano del suelo). De este modo, puede considerarse un descriptor de posición del robot. Asimismo, la matriz de argumentos permite calcular la orientación relativa del robot (haciendo uso de la propiedad de desplazamiento de la transformada de Fourier discreta):

$$\mathcal{F}[\{a_{n-q}\}] = A_k \exp(-j\frac{2\pi ql}{N_y}); l = 0, ..., N_y - 1 \tag{2.4}$$

donde q es el desplazamiento de las columnas de la imagen $q = \phi \cdot N_y / 2\pi$.

2.1.2 Transformada de Radon

La transformada de Radon fue descrita inicialmente en [57] y se ha utilizado tradicionalmente en algunas tareas de visión por computador, como la descripción de formas y en tareas de segmentación [29, 28].

La transformada de Radon en 2D consiste en la integral de una función 2D sobre líneas rectas (proyecciones integrales de línea). Esta transformada es invertible y la transformada inversa de Radon reconstruye una imagen a partir de sus proyecciones integrales de línea. Por este motivo, se utilizó inicialmente en imágenes médicas (como el TAC [72] y la resonancia magnética (IRM) [39]).

La transformación de Radon de una función 2D $f(i,j)$ se puede definir matemáticamente como:

$$\mathcal{R}\{f(i,j)\} = \lambda_f(p,\phi) = \iint_{-\infty}^{+\infty} f(i,j)\ \delta(p - \overrightarrow{r}\,\widehat{\overrightarrow{p}})\ di\ dj \tag{2.5}$$

donde δ es la función de la delta de Dirac ($\delta(x) = 1$ cuando $x = 0$, y $\delta(x) = 0$ en cualquier otro caso). La línea de integración está descrita por el vector $\overrightarrow{p}$ que se define como $\overrightarrow{p} =$

$\widehat{\overrightarrow{p}} \cdot p$, donde $\widehat{\overrightarrow{p}}$ es un vector unitario en la dirección de $\overrightarrow{p}$. p es el módulo de $\overrightarrow{p}$:

$$p = |\overrightarrow{p}| \tag{2.6}$$

Las proyecciones de las integrales de línea evaluadas para cada ángulo de acimut, ϕ, producen una función polar en 2D, λ_f, que depende de la distancia radial p y del ángulo de acimut ϕ. $\overrightarrow{r}$ es un conjunto de puntos perpendiculares a $\overrightarrow{p}$.

La transformada de Radon de una imagen $im(i,j)$ a lo largo de la línea $c_1(d,\phi)$ (Figura 2.1) puede expresarse de forma más clara mediante la siguiente expresión equivalente:

$$\mathcal{R}\{im(i,j)\} = \int_{\mathbb{R}} im(i'\cos(\phi) - j'\text{sen}(\phi), i'\text{sen}(\phi) + j'\cos(\phi))\, dj' \tag{2.7}$$

donde:

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \cos(\phi) & \text{sen}(\phi) \\ -\text{sen}(\phi) & \cos(\phi) \end{bmatrix} \cdot \begin{bmatrix} i \\ j \end{bmatrix} \tag{2.8}$$

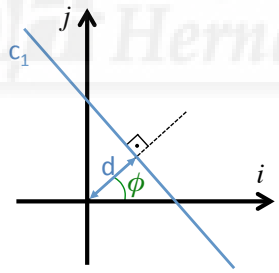
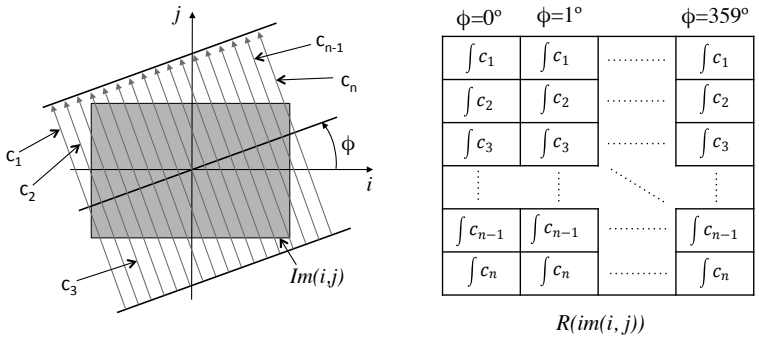


Figura 2.1: Parametrización de línea mediante la distancia al origen d y el ángulo entre una línea normal y el eje i, ϕ.

Cuando la transformada de Radon se aplica en imágenes, calcula las proyecciones de la imagen a lo largo de las direcciones especificadas a través de un conjunto de integrales de línea a lo largo de líneas paralelas en estas direcciones, entre las cuales suele haber una distancia de un píxel. La Figura 2.2(a) muestra las rutas de integración para calcular la transformada de Radon de una imagen en la dirección ϕ, y la Figura 2.2(b) muestra el valor de cada componente de la transformada de Radon en una notación simplificada.

La Figura 2.3 muestra una imagen de muestra en blanco y negro, a la izquierda, y su transformada de Radon, a la derecha. Además, muestra gráficamente el proceso para calcular la transformada de Radon.



(a) (b)

Figura 2.2: (**a**) rutas de integración para calcular la transformada de Radon de la imagen $im(i,j)$ en la dirección ϕ. (**b**) Matriz de la transformada de Radon de la imagen $im(i,j)$.

2.1.2.1 Propiedades de la transformada de Radon

La transformada de Radon tiene varias propiedades que la hacen útil en tareas de localización utilizando imágenes omnidireccionales. Estas propiedades son las siguientes:

- Linealidad: la transformada de Radon cumple con la propiedad de linealidad ya que la operación de integración es una función lineal del integrando:

$$\mathcal{R}\{\alpha f + \beta g\} = \alpha\mathcal{R}\{f\} + \beta\mathcal{R}\{g\} \tag{2.9}$$

- Traslación: La transformada de Radon es una operación que varía con la traslación. La traslación de una función de dos dimensiones, por un vector $\overrightarrow{r_0} = (x_0, y_0)$, tiene un efecto de traslación para cada proyección. Esta traslación viene dada por una distancia $\overrightarrow{r} \cdot (\cos\phi, \sin\phi)$.

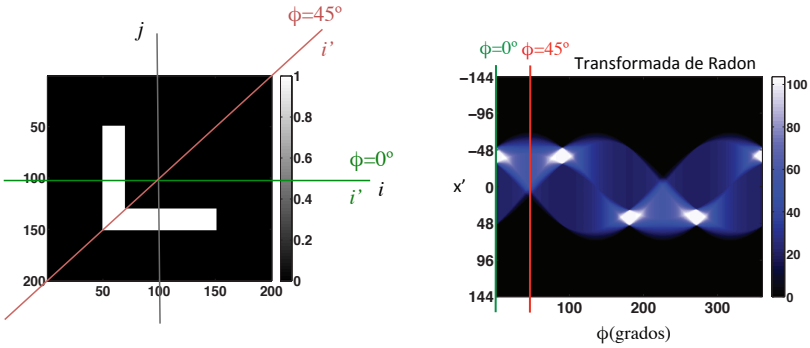
- Rotación: una rotación de la imagen en un ángulo ϕ_0, se traduce en un cambio ϕ_0 de la transformada de Radon a lo largo de la variable ϕ (las columnas se desplazan).

- Escalado: El escalado de f por un factor b implica un escalado de la coordenada d y la amplitud por un factor b, y supone un escalado de la transformada de Radon en un factor de $1/b$:

$$\mathcal{R}\left\{f\left(\frac{i}{b}, \frac{j}{b}\right)\right\} = |b|\lambda_f\left(\frac{d}{b}, \phi\right) \tag{2.10}$$

2.1.3 Descriptor *gist*

Los descriptores *gist* intentan imitar el sistema de percepción humana para describir las escenas identificando regiones con un color o textura prominente en relación con su entorno. El concepto *gist* fue introducido por Oliva y Torralba [51] con el nombre *holistic*



(a) (b)

Figura 2.3: (a) Imagen ejemplo. (b) Transformada de Radon de la imagen ejemplo.

representation of the spatial envelope. Los autores demostraron que este método de descripción crea un espacio multidimensional en el cual las escenas con la misma categoría semántica (por ejemplo, calles, cielo, edificios, *etc.*) se proyectan en puntos cercanos de dicho espacio.

Matemáticamente, el método trata de codificar la información espacial mediante la transformada 2D de Fourier en varias regiones de la imagen. Estas regiones se distribuyen en una cuadrícula espaciada regularmente y el conjunto de descriptores en cada región se reduce dimensionalmente mediante PCA (Principal Component Analysis) para generar un descriptor de imagen único con dimensión reducida. En trabajos recientes, se han utilizado pirámides de Wavelet en lugar de la transformada de Fourier, como en [64], donde los autores llevaron a cabo varios experimentos con diferentes grupos de imágenes naturales (costas, bosques, campos abiertos, desiertos, *etc.*) e imágenes artificiales (imágenes de edificios, puertas, interiores de edificios, carreteras, *etc.*). Los descriptores desarrollados funcionaron con éxito para clasificar estas imágenes de acuerdo con las propiedades anteriores.

La idea de categorizar una imagen a partir de estos términos viene de trabajos anteriores como [49, 50], donde se basan en la capacidad del ser humano para reconocer la información visual. Estos trabajos demuestran que se puede clasificar una escena con imágenes borrosas en las que únicamente se llega a apreciar la forma de la escena. Por lo tanto, es posible realizar una representación de las imágenes de manera holística, sin la necesidad de representar la forma de los objetos, sino tan solo su distribución u orientación en la escena.

Este tipo de representación es lo que se conoce como *gist* de una escena, que incluye diferentes niveles de procesamiento como características de bajo nivel (color de los

objetos), propiedades intermedias de la imagen (volumen, superficies) e incluso características de alto nivel (reconocimiento de objetos). Debido a que no existe una sola manera de extraer la 'esencia' de una imagen, pueden llegar a existir multitud de descriptores con el nombre de *gist*. Todos ellos tienen en común que obtienen características de la escena trabajando con el conjunto global de la imagen, pero trabajando de forma diferente. Algunos emplean solamente filtros para extraer la orientación de los objetos de la escena, mientras que otros añaden información para trabajar con diversas escalas de la imagen. En trabajos recientes se ha utilizado el concepto de prominencia junto con *gist*, que resalta las zonas que difieren más de sus vecinos [63]. Este descriptor está construido con la información de intensidad, orientación y color.

2.1.4 Histograma de Orientación del Gradiente (HOG)

HOG se ha utilizado tradicionalmente como un método de descripción en el campo de la detección de objetos. Fue descrito inicialmente por Dalal et al. [14] y lo usaron en tareas de detección de personas. Sin embargo, hay varias investigaciones en las que se ha mejorado este método de descripción, como [74], donde mejoran la precisión y el coste computacional.

La idea fundamental de los descriptores de Histograma de Orientación del Gradiente (HOG) es que la apariencia de un objeto y su forma dentro de imagen pueden ser descritas mediante la distribución de la intensidad de gradientes o dirección de bordes. La implementación básica consiste en dividir la imagen en pequeñas celdas conectadas y calcular el histograma de las orientaciones del gradiente de los píxeles en cada celda. Entonces, el descriptor se compone de estos histogramas dispuestos en un solo vector.

Para mejorar el descriptor, se puede normalizar el contraste de los histogramas locales calculando una medida de intensidad dentro de una región más grande de la imagen, a la que se le llama bloque. Este valor sirve para normalizar todas las celdas dentro del bloque. Este paso mejora los resultados ante variaciones de iluminación en la escena.

Fernández et al. [17] analizan este tipo de descriptor en tareas de localización al aire libre. Además, hacen un análisis comparativo entre varios métodos para describir imágenes panorámicas.

Todos estos algoritmos basados en el cálculo del Histograma de Orientación del Gradiente se dividen en tres pasos fundamentales:

1. Cálculo del Gradiente.
2. Cálculo de orientación en las celdas.
3. Creación y Normalización de Celdas.

El caso de la presente tesis se utiliza un descriptor HOG que se obtiene dividiendo la imagen de entrada en celdas horizontales, cuyo ancho sea el mismo que el de la imagen

a describir. La altura de estas celdas será un parámetro configurable. Por tanto el descriptor $\overrightarrow{h}_1$ construido de esta forma es invariante ante rotaciones del robot. Su tamaño será $1 \times k_2 \cdot b$ donde k_2 es el número de celdas horizontales y b es el número de canales en cada histograma de orientación. La Figura 2.4 muestra el procedimiento de división de celdas horizontales para obtener el descriptor HOG de una imagen.

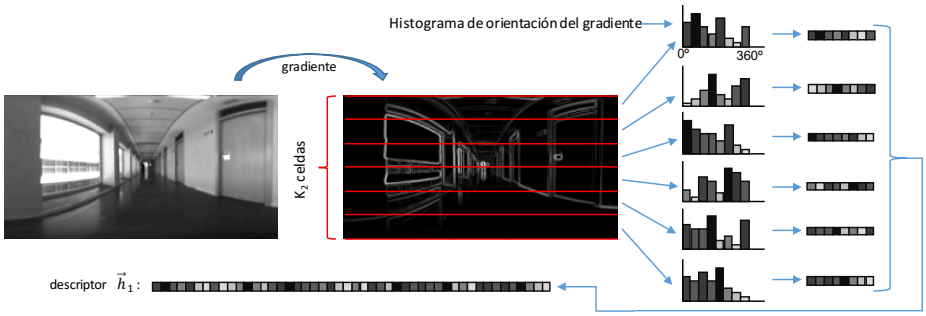


Figura 2.4: Procedimiento de obtención de un descriptor HOG de una imagen.

2.2 Comparación de descriptores

A la hora de comparar descriptores entre sí existen diferentes formas de realizar la comparación. Los dos métodos utilizados en las líneas de investigación desarrolladas de esta tesis son:

1. Distancia Euclídea
2. POC (Phase Only Correlation)

A continuación, en los siguientes apartados, se presentan los detalles de cada uno de ellos.

2.2.1 Distancia euclídea

La distancia euclídea entre dos vectores se calcula como la raíz de la suma del cuadrado de la resta de sus componentes entre sí, es decir:

$$dist_eu(V,U) = \sqrt{(v_1-u_1)^2+(v_2-u_2)^2+...+(v_n-u_n)^2} = \sqrt{\sum_{i=1}^{n}(v_i-u_i)^2} \qquad (2.11)$$

donde V es el vector de componentes $(v_1, v_2, ..., v_n)$ y U es el vector cuyas componentes son $(u_1, u_2, ..., u_n)$.

2.2.2 POC (Phase Only Correlation)

En general, una función en el dominio de la frecuencia se define por su módulo y su fase. A veces, solo se tiene en cuenta el módulo y la información de fase generalmente se descarta. Sin embargo, cuando el módulo y las características de fase se examinan en el dominio de Fourier, se deduce que las características de fase también contienen información importante porque reflejan las características de los patrones en las imágenes.

Oppenheim y Lim lo demostraron reconstruyendo imágenes usando la información completa de la fase con módulo unitario. Esto muestra que las imágenes se parecen a los originales, en contraste con la reconstrucción de imágenes usando la información completa del módulo con fase uniforme [52].

POC (Phase Only Correlation), propuesto en [36], es una operación realizada en el dominio de la frecuencia que proporciona un coeficiente de correlación entre dos imágenes [33]. La correspondencia entre dos imágenes $im_1(i,j)$ e $im_2(i,j)$ calculadas por POC viene dada por la siguiente ecuación:

$$C(i,j) = \mathcal{F}^{-1}\left\{\frac{\mathbf{IM}_1(u,v)\cdot\mathbf{IM}_2^*(u,v)}{\left|\mathbf{IM}_1(u,v)\cdot\mathbf{IM}_2^*(u,v)\right|}\right\} \tag{2.12}$$

Donde $\mathbf{IM}_1$ es la transformada de Fourier de la imagen 1 y $\mathbf{IM}_2^*$ es el conjugado de la transformada de Fourier de la imagen 2. $\mathcal{F}^{-1}$ es el operador inverso de la transformada de Fourier.

Para calcular la similitud entre las dos imágenes (im_1 e im_2) se puede utilizar la siguiente expresión:

$$sim_{POC}(im_1, im_2) = max\{C(i,j)\} \tag{2.13}$$

$Max\{(C(i,j)\}$ es un coeficiente que toma valores en el intervalo $[0,1]$ y mide la similitud entre las dos imágenes im_1 e im_2.

Esta operación es invariante frente a desplazamientos de las imágenes a lo largo de los ejes i y j. Además, es posible estimar estos desplazamientos Δ_x y Δ_y a lo largo de ambos ejes mediante:

$$(\Delta_x, \Delta_y) = argmax_{(i,j)}\{C(i,j)\} \tag{2.14}$$

2.3 Método masa-muelle

Cuando se dispone de diferentes imágenes adquiridas en posiciones desconocidas del entorno, y estas imágenes han sido caracterizadas por un adecuado descriptor de las mismas que permite extraer información acerca de su posición con respecto a las demás, en ocasiones resulta oportuno establecer una relación de vecindad y posición relativa entre ellas. Para determinar estas relaciones entre un conjunto de imágenes es posible utilizar el método masa-muelle.

El método Masa-Muelle se basa en un sistema físico de fuerzas llamado *spring model* [30, 43]. Este modelo se basa en la combinación de dos principios: la ley de Hooke y la segunda ley de Newton. La Figura 2.5 muestra el esquema del modelo físico utilizando tres nodos. Cada nodo N_i corresponde a una imagen, y están vinculados con los conectores S_{ij} cuya longitud natural l_{ij0} es la distancia entre los descriptores de imágenes i y j.

El punto de partida son los descriptores de un conjunto de imágenes (cuyos puntos de captura se desconocen) y el objetivo es distribuir estas imágenes en un plano, de modo que la distribución resultante refleje la distribución inicial de los puntos de captura de las escenas. Para ello, a cada imagen se le asigna una partícula y cada partícula se inicializa en una posición aleatoria (dado que no se conocen las coordenadas de los puntos de captura).

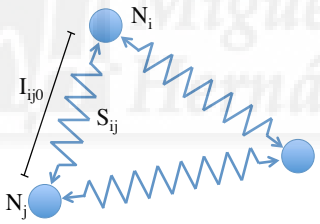


Figura 2.5: Modelo masa-muelle.

Todo ello se basa en el cálculo de las fuerzas elásticas ejercidas por los resortes en cada nodo y una vez que se ha calculado la fuerza resultante ejercida sobre cada nodo, procedemos a obtener la aceleración, la velocidad y la posición de cada uno de ellos. por tanto, el proceso de creación del mapa consiste en resolver en cada instante las siguientes ecuaciones:

$$\overrightarrow{F_i} = \sum_{S_{ij} \in S} \left(-K_{ij}\left(l_{ij0} - l_{ij}\right) - b_{ij}\left(v_i - v_j\right)\right) \cdot \frac{\overrightarrow{p}_i - \overrightarrow{p}_j}{|\overrightarrow{p}_i - \overrightarrow{p}_j|} \tag{2.15}$$

$$\overrightarrow{a_i} = \frac{\overrightarrow{F_i}}{m_i} \tag{2.16}$$

$$\overrightarrow{v_i}(t + \Delta T) = \overrightarrow{v_i}(t) + \overrightarrow{a_i}(t) \cdot \Delta T \tag{2.17}$$

$$\overrightarrow{r_i}(t+\Delta T) = \overrightarrow{r_i}(t) + \overrightarrow{v_i}(t) \cdot \Delta T, \tag{2.18}$$

donde la Ecuación (2.15) corresponde a la ley del oscilador armónico de Hooke. La fuerza resultante en el nodo i, $\overrightarrow{F_i}$, depende de la longitud de los conectores l_{ij}, de la constante elástica del muelle K_{ij} y de la posición de los nodos $\overrightarrow{p}_i$ y $\overrightarrow{p}_j$. También se ha agregado un atenuador con constante de amortiguación b_{ij} entre los nodos, ya que ayuda a mejorar la convergencia del algoritmo. La segunda ley de Newton nos permite obtener la aceleración $\overrightarrow{a_i}$ usando el la masa del nodo m_i. Finalmente, las últimas dos expresiones se usan para actualizar el cálculo de la velocidad $\overrightarrow{v_i}$ y la posición $\overrightarrow{r_i}$ de todos los nodos del sistema en cada iteración (para cada incremento de tiempo, ΔT).

El método masa muelle presentado se ha utilizado en la presente tesis con objeto de crear mapas locales de entornos cuando se desconocen las posiciones desde las cuales e han capturado las imágenes del mapa. En el Capítulo 4 se ofrecen más detalles de su aplicación.

2.4 Análisis de Procrustes

Cuando se tienen varios conjuntos de puntos en los que un conjunto puede estar rotado, escalado y trasladado respecto a otro. Lo que se quiere es obtener un coeficiente de similitud de forma entre ambos conjuntos, eliminando previamente estos efectos. Para ello existe un análisis que realiza esta conversión, llamado análisis Procrustes.

El análisis de Procrustes está basado en una serie de métodos estadísticos que aplican la teoría de grupos al análisis de un conjunto de datos. Este método de análisis combina datos históricos (ubicaciones geométricas que representan características significativas de una forma determinada) para calcular las mejores transformaciones euclídeas que preservan la forma. Estas transformaciones minimizan las diferencias de ubicación entre los datos de referencia comparados.

Primero, se organizan las coordenadas de los puntos donde se capturaron las imágenes en una matriz $A = [(\alpha_1, \beta_1)', (\alpha_2, \beta_2)', .., (\alpha_n, \beta_n)']'$ y organizamos las coordenadas de las partículas balanceadas del sistema de masa-muelle en una matriz $C = [(\gamma_1, \delta_1)', (\gamma_2, \delta_2)', ..., (\gamma_n, \delta_n)']'$. El análisis de Procrustes nos permite comparar la forma de estos dos conjuntos de puntos al determinar una transformación lineal (una traslación c, una reflexión, una rotación ortogonal T y un factor de escalado b) de los puntos en la matriz C de tal forma que los puntos en $bCT + c$ se ajusten mejor a los puntos en la matriz A.

Una vez que se han eliminado los efectos de traslación, rotación y escala, el criterio de bondad de ajuste es la suma de los errores al cuadrado. Gracias a este análisis podemos medir cuán preciso es el diseño de las partículas después del proceso de creación del mapa, en comparación con el diseño donde se tomaron las imágenes.

Por lo tanto lo que realiza es una comparación entre ambos conjuntos de datos X e Y y proporciona una transformación lineal formada por una traslación, una rotación y un

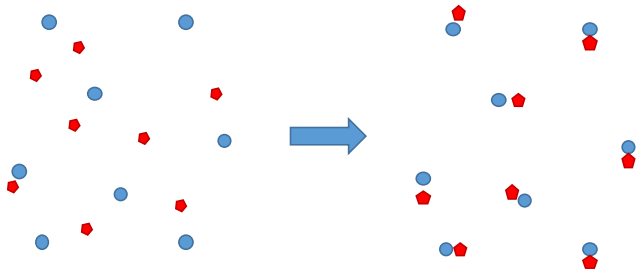


Figura 2.6: Aplicación del algoritmo de Procrustes.

escalado de los puntos de un conjunto de datos con respecto al otro, a la vez que un coeficiente de similitud entre ambos conjuntos después de realizar dichas transformaciones. El algoritmo se detalla de forma completa en [62]. En la Figura 2.6 se puede observar la transformación de dos conjuntos de puntos mediante el algoritmo de Procrustes.

El algoritmo Procrustes presentado se ha utilizado en la presente tesis con objeto comparar las posiciones de varios mapas creados utilizando el método masa-muelle. En el Capítulo 4 se ofrecen más detalles de su aplicación.

3 Creación de imágenes virtuales

3.1 Introducción

Actualmente, existen multitud de tipos de robots móviles que tienen que llevar a cabo diferentes tareas de forma autónoma en un entorno desconocido. Para ello, deben llevar a cabo dos pasos fundamentales. Por un lado, el robot debe crear una representación interna del entorno (es decir, un mapa) y por otra parte debe ser capaz de utilizar este mapa para estimar su pose actual (posición y orientación). El robot extrae información del entorno desconocido utilizando los diferentes sensores que lleve equipados. Esta información se compara con los datos de los mapas creados y se usa para estimar la pose del robot.

A lo largo de los últimos años se han utilizado diferentes tipos de sensores en tareas de creación de mapas y localización de robots. En muchas de estas investigaciones se han utilizado sensores visuales, ya que permiten muchas configuraciones posibles y le proporcionan al robot información muy completa del entorno que se puede utilizar en otras tareas de alto nivel (por ejemplo, detección de personas, identificación de semáforos, etc.) [13, 67]. Entre ellos, algunos trabajos se centran en las imágenes con información visual y métrica, como imágenes RGB-d. Dos ejemplos de esto se muestran en [60, 56] que utilizan este tipo de información en las tareas de creación de mapas y seguimiento.

Dentro de los sensores visuales existen unos sensores que captan información en todas las direcciones del entorno, son los conocidos como sensores de visión omnidireccional. Existen diferentes tipos, pero todos se basan en un mismo concepto, conseguir que la cámara o cámaras del sistema capten los rayos que provienen en cualquiera de las direcciones de alrededor del sensor. Estos sistemas pueden ser sistemas que usen

varias cámaras con el fin de enfocar cada una en una dirección diferente o pueden estar formados por diferentes elementos ya sean lentes especiales o espejos que lo que hacen es captar o reflejar los rayos de luz y dirigirlos hacia el foco del sensor óptico del sistema.

Podemos encontrar muchos trabajos previos que usan imágenes omnidireccionales en tareas de creación de mapas y localización. Por ejemplo, Valiente et al. [65] presentan una comparación entre dos métodos diferentes de SLAM (*Simultaneous Localization And Mapping*) visual mediante el uso de imágenes omnidireccionales y en [41] se propone un sistema de navegación topológico mediante visión omnidireccional. El sistema visual catadióptrico se compone de una cámara que apunta a un espejo hiperbólico.

También hay que tener en cuenta el hecho de que, hoy en día, vehículos aéreos no tripulados (UAVs (*Unmanned Aerial Vehicles*)) se han convertido en plataformas muy populares y versátiles que pueden realizar infinidad de tareas. Algunos investigadores se han enfrentado anteriormente el problema de la localización con este tipo de plataforma, como [44] donde se presenta una aplicación de control para el aterrizaje de vehículos aéreos no tripulados. Otro ejemplo es el trabajo presentado por An et al. [3] donde se presenta el diseño de un control para calcular la posición relativa de un vehículo aéreo no tripulado en el proceso de reabastecimiento de combustible en vuelo.

En todo trabajo en que se pretenda desarrollar o mejorar algún método o algoritmo de localización es necesario disponer de una extensa base de datos con la cual validar los algoritmos de creación de mapas y localización que utilizan información visual. Este paso supone un gran coste en cuanto a tiempo empleado en obtener cada una de las imágenes de la base de datos. Además es necesario saber qué configuración usar en el sistema de visión, incluso se debe tener ya disponible la cámara y el espejo, sin saber si en realidad será mejor usar una resolución diferente a la que posee esa cámara o si el espejo debería o no tener una curvatura que permitiese capturar un campo de visión diferente. Una mala decisión en cuanto a cualquiera de estos parámetros supondría tener que adquirir una nueva cámara o un nuevo espejo y tomar de nuevo toda la base de datos, con el riesgo de que vuelva a ocurrir el mismo problema.

Debido a ello, se hace interesante la idea de poder obtener una base de datos inicial con imágenes generadas artificialmente, en la cual se pueda cambiar la configuración de todos los elementos del sistema de visión como por ejemplo el tipo de espejo, la resolución de la cámara o incluso la posición y orientación del robot en cualquier punto de un entorno creado virtualmente. Con este propósito, en este capítulo se propone un programa capaz de generar *data sets* de imágenes para diseñar y mejorar los algoritmos que utilizan visión omnidireccional. Para llevar a cabo esta generación de bases de datos de imágenes, se ha desarrollado una plataforma para crear *data sets* de imágenes que cambian el tipo de cámara y todos los parámetros del sistema de visión omnidireccional [70].

Esta plataforma es útil para crear conjuntos de escenas sin ningún coste adicional. Estos conjuntos pueden utilizarse para validar cualquier algoritmo nuevo de creación de mapas y localización que utilice información visual omnidireccional. Estos mapas pueden ser creados usando diferentes números de imágenes y diferentes tipologías de mapas, como los mapas de trayectoria o de tipo rejilla. Por otra parte, estos mapas pueden

contener tantas imágenes como sean necesarias. El programa también permite cambiar diferentes características de los entornos. Esta plataforma ha sido desarrollada utilizando el lenguaje de programación utilizado por Matlab.

Además, para probar los algoritmos de localización en un mapa previamente construido, es posible generar imágenes de prueba desde cualquier posición en el mapa simulando cualquier rotación u orientación de la plataforma robótica en el entorno (6 grados de libertad), tarea que sería verdaderamente difícil de realizar utilizando UAVs debido a la complejidad de cambiar la orientación del robot en cualquier ángulo sin que pierda su estabilidad en el aire. Otra ventaja de esta plataforma es que las imágenes creadas no tienen ningún tipo de ruido o imperfección porque son generadas a partir de un entorno virtual definido previamente. Permite probar los diferentes algoritmos en condiciones ideales, lo que puede ser útil en las etapas iniciales del diseño y puesta a punto de un nuevo algoritmo. Además, el ruido y las oclusiones se pueden añadir después de probar la robustez de los algoritmos, una vez que ya se han ajustado con imágenes ideales.

De esta manera, esperamos que el uso de esta plataforma ahorre tiempo y dinero durante el desarrollo de nuevos algoritmos de creación de mapas y localización de robots. Gracias a ello, los experimentos iniciales pueden ser llevados a cabo rápidamente, en una gran variedad de entornos y con precisión.

En este capítulo se presenta en primer lugar el concepto de visión omnidireccional utilizado a lo largo de toda la tesis. A continuación se presenta un algoritmo de creación de imágenes omnidireccionales utilizando entornos virtuales creados sintéticamente, el cual se emplea en el testeo de los nuevos algoritmos de localización y creación de mapas de esta tesis con la finalidad de configurar los parámetros para realizar las pruebas con bases de datos reales. Finalmente se comentan las conclusiones del capítulo.

3.2 Visión omnidireccional

Los sistemas de visión omnidireccional se caracterizan por obtener información del entorno para todas las orientaciones posibles, es decir, si este sistema estuviese montado en un robot móvil, éste sería capaz de obtener información de la habitación o entorno en el que estuviese operando para cualquiera de las posibles orientaciones alrededor suyo.

Para conseguir este propósito, estos sistemas deben de captar todos los rayos de luz procedentes de cualquier orientación alrededor de ellos. Por lo tanto, para los diferentes experimentos realizados a lo largo de la tesis, se ha utilizado un sistema de visión formado por una cámara CCD (Charge-Coupled Device), la cual capta la información visual y la transforma en información digital, y un espejo hiperbólico, el cual capta los rayos de luz procedentes de todas las orientaciones y los proyecta hacia la lente de la cámara CCD. Dicha cámara y el espejo están anclados a través de mecanismos fijos (Figura 3.1).

En la actualidad existen muchos tipos de sensores catadióptricos. El primero del que se tiene constancia es el desarrollado por Rees, y publicado en 1970 [59]. Posteriormente se fueron desarrollando nuevos tipos de sensores catadióptricos como los siguientes: [10, 72, 35, 27].

Figura 3.1: Sistema de adquisición de imágenes omnidireccionales. Está compuesto por una cámara CCD apuntando hacia un espejo hiperbólico.

En la Figura 3.2 se muestra un ejemplo de una imagen omnidireccional capturada con un sistema de visión catadióptrico.

Estos sistemas se suelen conocer como cámaras omnidireccionales catadióptricas, que combinan cámaras convencionales con espejos convexos para obtener campos de visión de 360° alrededor del eje del sistema. Los espejos convexos pueden ser esféricos [48], cónicos [71], elípticos [38], hiperbólicos [12] o parabólicos [47], dependiendo de cuánto queramos ampliar el campo de visión que capture el sensor. En [73] se comparan las diferentes configuraciones de espejos catadióptricos en sistemas omnidireccionales. Cada una de estas configuraciones tiene sus ventajas e inconvenientes pero si se desean obtener proyecciones perspectivas de la imagen, los espejos parabólicos e hiperbólicos son los más recomendables. Los sistemas de visión formados por estos dos tipos de espejos son sistemas con un único centro de proyección, es decir, todos los rayos que llegan a la superficie del espejo tienen un único punto de intersección común, que es el centro óptico. Para que esta propiedad se cumpla y los rayos converjan en el foco de la cámara, es necesario el uso de un conjunto formado por un espejo hiperbólico con una cámara con lente de proyección perspectiva (lente convencional), o bien usando un sistema compuesto por un espejo parabólico y una lente de proyección ortográfica (aquella que hace converger los rayos paralelos al foco de la cámara).

La Ecuación 3.1 define el espejo hiperbólico. Este espejo es simétrico y sus dimen-

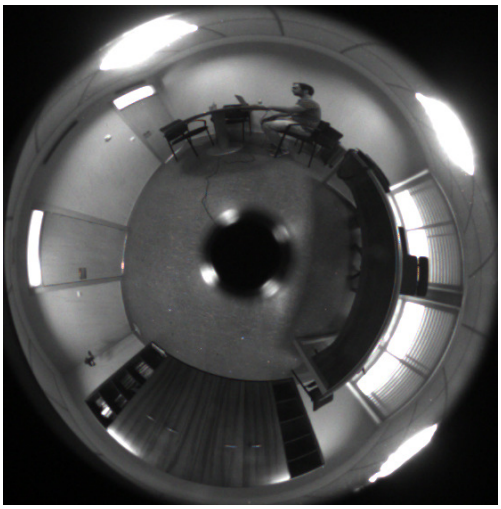


Figura 3.2: Imagen omnidireccional real.

siones se definen a partir de a y b. Estas variables también definen la distancia entre el foco del espejo hiperbólico y el origen del sistema de coordenadas del mundo, c (Ecuación 3.2).

$$\frac{x_c^2 + y_c^2}{a^2} - \frac{z_c^2}{b^2} = -1 \tag{3.1}$$

$$c = \sqrt{a^2 + b^2} \tag{3.2}$$

La Ecuación 3.3 define la ecuación de espejos parabólicos.

$$\frac{z_c}{c} = \frac{x_c^2}{a^2} + \frac{y_c^2}{b^2} \tag{3.3}$$

donde a, b, c definen la geometría del espejo parabólico con respecto al sistema de referencia (x_c, y_c, z_c).

Con las cámaras omnidireccionales, el modelo convencional de proyección difiere debido a que existe una transformación adicional provocada por la presencia del espejo entre la imagen capturada y el entorno:

$$\overrightarrow{p} = M \cdot F(\overrightarrow{P}) \tag{3.4}$$

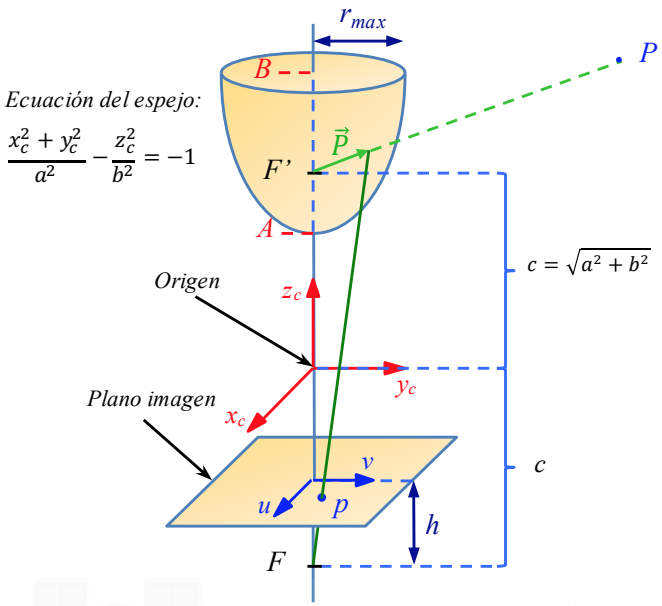


Figura 3.3: Modelo geométrico de proyección del sistema de visión omnidireccional con un espejo convexo.

donde $\overrightarrow{p}$ es la proyección en la imagen del punto $\overrightarrow{P}$ del mundo y M es la matriz de proyección que contiene los parámetros intrínsecos que realizan la transformación de las coordenadas del mundo real a las coordenadas de la imagen. Por otra parte F representa la función de reflexión de los rayos de luz sobre el espejo. El sistema de visión utilizado para la realización de esta tesis posee un centro de proyección único, gracias a una correcta configuración de la cámara y a la elección de un espejo adecuado (Figura 3.3), que en el caso del sistema utilizado se trata de un espejo hiperbólico. Gracias a esta propiedad podemos aplicar el modelo de proyección unificado [23, 24].

A partir de una imagen omnidireccional es posible obtener su proyección panorámica utilizando un programa sencillo para transformar las coordenadas polares de la imagen omnidireccional en coordenadas cartesianas de la imagen panorámica. La Figura 3.4 muestra la imagen panorámica obtenida a partir de la imagen omnidireccional mostrada en la Figura 3.2. La plataforma también permite obtener otras proyecciones diferentes, tales como vistas ortográficas, cilíndrica y proyecciones en esfera unitaria. Algunos algoritmos de creación de mapas y localización usan este tipo de información [2].

3.2.1 Sistemas catadióptricos empleados

Las cámaras empleadas para realizar los experimentos corresponden con los modelos DFK-41BF02 [26] y DMK-21BF04 [25] de la compañía *ImagingSource*. Se trata de cámaras a color de tipo CCD, con un puerto de conexión Firewire. En la tabla 3.1 podemos observar las características más importantes de ambas cámaras.

Figura 3.4: Imagen panorámica obtenida a partir de la imagen omnidireccional de la Figura 3.2.

Para poder obtener imágenes omnidireccionales, que capten toda la escena que rodea a la cámara, se ha montado un sistema compuesto por un espejo hiperbólico, un soporte para dicho espejo y elevadores que permiten ajustar la distancia entre la cámara y el espejo para conseguir que la superficie útil que se quiere capturar (es decir, el espejo) se ajuste al tamaño de la imagen capturada por la cámara. Concretamente, el espejo hiperbólico empleado para la realización de los experimentos ha sido el modelo Wide 70 de la compañía *Eizoh* [16] . Estos espejos ofrecen grandes ángulos de visión laterales que son especialmente útiles para captar partes más amplias de la escena. En la Tabla 3.2 podemos observar las características más importantes de dicho espejo.

Como hemos observado, el sistema catadióptrico es capaz de capturar una escena con un campo de visión de 360 $^\circ$ alrededor del eje z_c, con un ángulo lateral efectivo de 120° para ambos espejos, con la salvedad de que en el espejo Wide 70 el ángulo de visión superior (por encima del horizonte) e inferior (por debajo del horizonte) son iguales a 60° y en el caso del espejo Super-Wide el ángulo de visión inferior es de 65° y el superior de 55 $^\circ$. Este espejo puede observarse en la parte superior del sistema de la Figura 3.1.

La imagen obtenida en el sensor visual corresponde al reflejo de los rayos, procedentes de los objetos, en el espejo. Dicha imagen está representada en coordenadas polares, cuyo origen de coordenadas se corresponde con la proyección del foco del espejo en el sensor visual. La dirección de reflexión del rayo en el espejo se corresponde con el ángulo del pixel en la imagen omnidireccional debido a que el espejo tiene una geometría simétrica.

En la Figura 3.5 se muestra un ejemplo de una imagen omnidireccional capturada en un entorno de interior utilizando la cámara CCD DFK-41BF02 y el espejo hiperbólico Wide 70.

3.3 Imágenes omnidireccionales de entornos virtuales

Con el fin de poder crear imágenes omnidireccionales a partir de un entorno virtual, se ha modelado el sistema de visión catadióptrico mediante un algoritmo basado en la trayectoria que siguen los rayos de luz desde los objetos hasta que llegan al foco de la cámara, pasando por la reflexión del espejo hiperbólico. Dicho algoritmo ha sido desarrollado mediante el entorno de programación Matlab.

La creación de cada entorno es muy fácil de realizar a través de líneas de comando definiendo las características de cada uno de los nuevos objetos definiendo su forma,

Tabla 3.1: Características principales de las cámaras DFK-41BF02 y DFK-21BF04 utilizadas como sensores de visión para la realización de los experimentos.

Especificaciones de las cámaras CCD		
Características generales		
Modelo	**DFK-21BF04**	**DFK-41BF02**
Formato de imagen	640x480 (0.3 MP) UYVY @ 30 fps 640x480 (0.3 MP) BY8 @ 60 fps	1280x960 (1.2 MP) UYVY @ 7.5 fps 1280x960 (1.2 MP) BY8 @ 15 fps
Sensibilidad	0.1 lx	0.15 lx
Rango dinámico	8 bits	8 bits
Interfaz óptico		
Sensor	CCD Sony ICX098BQ	CCD Sony ICX205AK
Formato	1/4”	1/2”
Resolución	H: 640, V: 480	H: 1280, V: 960
Tamaño de píxel	H: 5.6 μm, V: 5.6 μm	H: 4.65 μm, V: 4.65μm
Montage de la lente	C/CS	C/CS
Interfaz eléctrica		
Interfaz	FireWire 400	FireWire 400
Voltaje de alimentación	De 8 VDC a 30 VDC	De 8 VDC a 30 VDC
Consumo de corriente	Aprox. 200 mA @ 12 VDC	Aprox. 200 mA @ 12 VDC
Interfaz mecánica		
Dimensiones	H: 50.6 mm, W: 50.6 mm, L: 56 mm	H: 50.6 mm, W: 50.6 mm, L: 56 mm
Masa	265 g	265 g
Ajustes		
Velocidad	De 1/10000 segundos a 30 segundos	De 1/10000 segundos a 30 segundos
Ganancia	De 0 dB a 36 dB	De 0 dB a 36 dB
Balance de blancos	de -2 dB a 6 dB	de -2 dB a 6 dB
Entorno de trabajo		
Temperatura trabajo	De -5 °C a 45 °C	De -5 °C a 45 °C
Temperatura almacenam.	De -20 °C a 60 °C	De -20 °C a 60 °C
Humedad trabajo	De 20 % a 80 %	De 20 % a 80 %
Humedad almacenam.	De 20 % a 95 %	De 20 % a 95 %

Tabla 3.2: Características principales del espejo hiperbólico utilizado para la realización de los experimentos.

Especificaciones del espejo hiperbólico	
Modelo	***Eizho* Wide 70**
Geometría	Hiperbólico
Diametro	70 mm
Altura	35 mm
Ángulo de visión superior	60°
Ángulo de visión inferior	60°
Distancia cámara-espejo	Variable - óptima: 165 mm
Peso	175 g

Figura 3.5: Ejemplo de imagen omnidireccional capturada utilizando la cámara CCD DFK-41BF02 y el espejo hiperbólico Wide 70.

posición, orientación y tamaño. Los entornos virtuales pueden ser definidos como entornos de interior o de exterior, dependiendo de la configuración de las paredes y de los objetos dentro de ellos.

Esto es muy útil para probar los algoritmos de localización en condiciones realistas. Por lo general, las imágenes de mapa son capturadas en un momento específico del día, pero la localización debe llevarse a cabo en diferentes momentos. Esto implica diferentes condiciones de iluminación y también otros cambios en la información visual, tales como oclusiones en escenas debido a la presencia de personas u otros objetos móviles alrededor del robot.

Teniendo en cuenta estos hechos, la plataforma tiene también otras opciones configurables en la generación de las imágenes, tales como la adición de puntos de luz, ruido y oclusiones (añadiendo objetos en el entorno virtual). También es posible cambiar el color de cada objeto en dicho entorno.

3.3.1 Definición de objetos

Algo indispensable para poder llevar a cabo la generación de imágenes omnidireccionales virtuales es la existencia de un entorno virtual. Dicho entorno debe tener definidas las limitaciones y los objetos presentes en él. Los objetos deben definirse de tal manera que la intersección entre los rayos de luz y los objetos dentro de este entorno se pueda simular de manera eficiente. Con este propósito, hemos definido estos objetos como grupos de caras, y cada una de estas caras está contenida en un plano diferente, por ejemplo, seis caras en seis planos diferentes definen un cubo. Desde este punto de vista, el entorno virtual está formado por muchas caras en el espacio que definen objetos con diferentes formas. La Figura 3.6 muestra los elementos que forman un paralelepípedo usando estas caras; el paralelepípedo se define por l_1, l_2, l_3, su posición en el entorno y el color de cada cara.

Todos estos objetos poseen propiedades que pueden ser modificadas cambiando así su aspecto o posición en el entorno. Las propiedades configurables son: su tamaño, el color de sus caras (pudiendo definir texturas simples) y su forma. En el Código 3.1 se muestra un ejemplo de definición de una cara de un objeto en el lenguaje de programación de Matlab.

Código 3.1: Código de definición de una cara e un objeto

```
1  %% DEFINICIÓN DE LA CARA FRONTAL DE LA PUERTA
2
3  % vector normal al plano de la cara de la puerta
4  uplano=[-1 0 0];
5  % producto vectorial del vector normal al plano y el vector desde ...
       el foco del espejo (para ver si son perpendiculares)
6  Paralel=cross(vectorP,uplano);
7  %Ecuaciones del plano de la cara
8  x3=-2998;
9  y3=(vectorP(2)*FespejoDesplazado(1) - ...
       vectorP(1)*FespejoDesplazado(2) - vectorP(2)*x3)/(-vectorP(1));
```

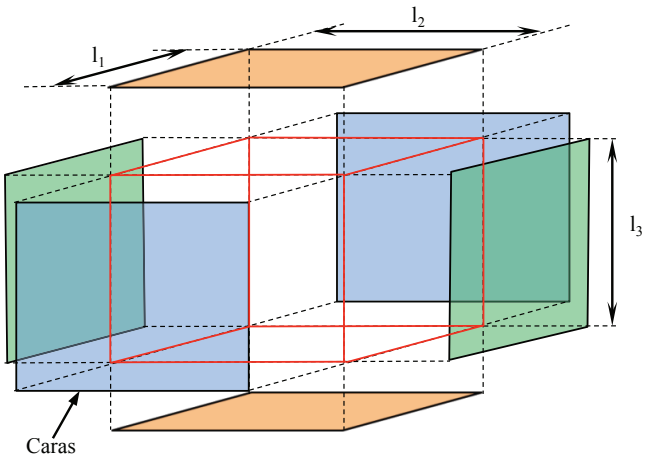


Figura 3.6: Caras que definen un objeto en el entorno virtual.

```
10  z3=(vectorP(3)*FespejoDesplazado(2) - ...
        vectorP(2)*FespejoDesplazado(3) - vectorP(3)*y3)/(-vectorP(2));
11  % ver si el objeto está en la misma dirección que el vector o por ...
        el contrario esta en dirección opuesta
12  lambda=(x3-FespejoDesplazado(1))/vectorP(1);
13  % si hay más de un objeto habría que guardar la distancia y ponerle ...
        el valor del color del objeto que está a menos distancia
14  d2=norm([x3 y3 z3]-FespejoDesplazado);
15  % ver si el punto de corte con el plano esta dentro del rectángulo ...
        que hemos definido
16  if(y3≥-1700 && y3≤-1000 && z3≥0 && z3≤1700 && lambda≥0 && d2<d1 && ...
        norm(Paralel)≠0)
17      % Color RGB de ese objeto
18      imagen(i,j,1)=204;
19      imagen(i,j,2)=102;
20      imagen(i,j,3)=0;
21      d1=d2;
22  end
```

3.3.2 Creación de imágenes

Una vez que se han generado todos los objetos, el programa simula la trayectoria seguida por el rayo partiendo del foco de la cámara. En la Figura 3.3 se puede ver la trayectoria de dicho rayo de luz, representado en color verde. El rayo parte del foco de la cámara, pasa por el plano de la imagen, rebota en el espejo hiperbólico y finalmente llega al objeto del entorno (Punto P).

Llegados a este punto existe un problema y es que existen infinitos rayos partiendo desde el foco de la cámara y hay que seleccionar cuales son los que se van a simular. Para ello, se define un rayo de luz para cada pixel del plano de la imágen. Por lo tanto lo

primero que hay que hacer es definir la resolución de la imagen omnidireccional (p_x x p_y) y la distancia h entre el plano imagen y el foco de la cámara. Por tanto, existe un vector $\overrightarrow{Fp_{ij}}$ por cada píxel del plano de la imagen (Ecuación 3.5). p_{ij} es el píxel seleccionado para trazar el rayo de luz. La Figura 3.7 muestra el plano de la imagen y la trayectoria del rayo para cada pixel p_{ij}.

$$\overrightarrow{Fp_{ij}} = \overrightarrow{p}_{ij} - \overrightarrow{F} \tag{3.5}$$

donde $\overrightarrow{p}_{ij}$ y $\overrightarrow{F}$ son los vectores cuyos componentes son las coordenadas de p_{ij} y F respectivamente, con respecto al sistema de referencia $\{x_c, y_c, z_c\}$.

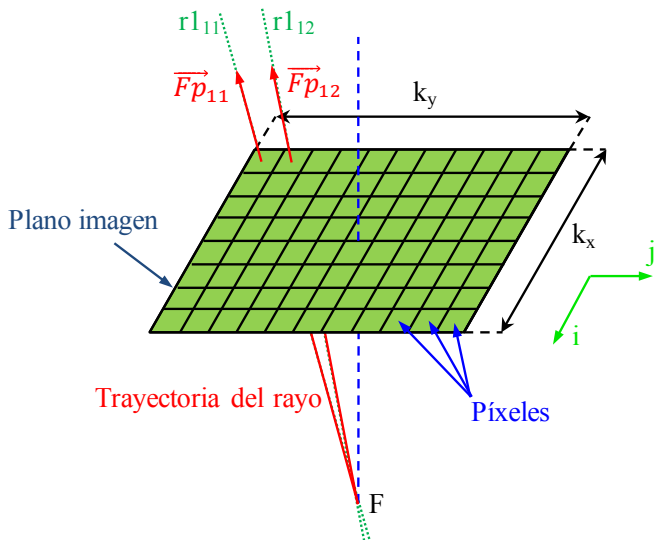


Figura 3.7: Trayectoria de los rayos para cada pixel en el plano imagen.

El rayo parte del foco de la cámara (F) y traza una línea recta $r1_{ij}$, definida por el vector $\overrightarrow{Fp_{ij}}$ y el punto F (como punto de esta línea), la cual se usa para calcular el punto de intersección (Q_{ij}) entre $r1_{ij}$ y el espejo hiperbólico (Ecuación 3.1). Este punto (Q_{ij}) se usa para calcular el vector $\overrightarrow{P}_{ij}$ por medio de esta ecuación:

$$\overrightarrow{P}_{ij} = \overrightarrow{Q}_{ij} - \overrightarrow{F'} \tag{3.6}$$

donde $\overrightarrow{Q}_{ij}$ y $\overrightarrow{F'}$ son los vectores cuyos componentes son las coordenadas de Q_{ij} y F' respectivamente, con respecto a la referencia del sistema $\{x_c, y_c, z_c\}$. F' es el foco del espejo hiperbólico (Figura 3.3). La Figura 3.8 muestra la trayectoria de dos rayos diferentes que rebotan en el espejo hiperbólico, procedentes del plano imagen.

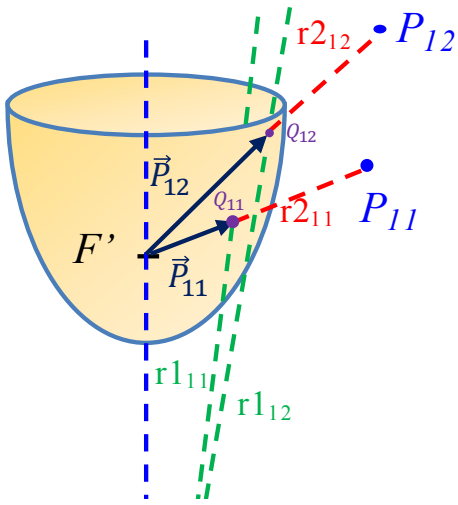


Figura 3.8: Trayectoria de los rayos en el espejo hiperbólico.

Finalmente, la línea recta $r2_{ij}$ definida por el vector $\overrightarrow{P}_{ij}$ y el punto F' se usa para calcular el punto de intersección (P_{ij}) entre ella y cualquier objeto en el entorno. Cuando $r2_{ij}$ atraviesa una cara de un objeto, el píxel del plano de la imagen utilizado en la Ecuación 3.5 toma el valor del color de la cara de ese objeto. $r2_{ij}$ puede atravesar varias caras, pero solo se considera la intersección más cercana (esto sucede en una situación real).

Por lo tanto, este proceso crea un conjunto de vectores compuesto por todos los vectores $\overrightarrow{P}_{ij}$, uno por cada píxel del plano imagen, y un punto (F'). Las intersecciones con los objetos del entorno se calculan utilizando las líneas definidas por estos vectores y el punto (F').

En el Código 3.2 se muestra el proceso de creación de los vectores $\overrightarrow{P}_{ij}$ cuya dirección viene definida por el foco del espejo y las rectas $r2_{ij}$. Tras la creación de estos vectores el programa ejecutaría el código de cada uno de los objetos (como el que se muestra en el Código 3.1) para cada uno de los píxeles de la imágen, obteniendo así el color de cada pixel.

Código 3.2: Código de definición de los vectores que salen desde el foco del espejo

```
1  %Script para crear vectores P omnidireccionales
2
3  %---Declaración de constantes
4  a=40;% término a de la ecuación de el espejo // influye en la ...
       anchura de la hiperboloide
5  b=160;% término b de la ecuación del espejo // influye en la altura ...
       de la hiperboloide
6  Alt=41;% Altura del espejo
7  px=250;% píxeles eje x (resolución)
```

```
8  py=250;% píxeles eje y (resolución)
9  l=200;% radio de la circunferencia de la imagen omnidireccional (en ...
       unidades del sistema de referencia del modelo 3D)
10 SR=[0 0 0];% punto del sistema de referencia
11
12 %----Matrices donde se guardarán las componentes x, y, z de los ...
       vectores omnidireccionales, el punto de origen de dichos ...
       vectores es Fespejo
13 Xomni=zeros(px,py);
14 Yomni=zeros(px,py);
15 Zomni=zeros(px,py);
16
17 c=sqrt(a^2+b^2);% distancia desde el origen al foco del espejo // ...
       coincide con la distancia desde el punto de proyección de la ...
       cámara y el origen
18
19 syms z;
20 A=solve(z-sqrt(b^2));% distancia desde el origen hasta el punto más ...
       bajo del espejo
21 A=double(A);% convertir de sym a double
22 B=Alt+A;% parte más alta del espejo, altura del espejo +A
23
24 imagen=zeros(px,py);% matriz donde guardaremos la imagen ...
       omnidireccional
25
26 syms y;
27 rmax=solve((y^2)/(a^2)-(B^2)/(b^2)+1);% radio máximo del espejo
28 rmax=double(rmax);
29 if(rmax(1)<0) % se coge solo la solución positiva
30     rmax=rmax(2);
31 else
32     rmax=rmax(1);
33 end
34
35 h=(l*(B+c))/rmax-c;% distancia desde el origen al plano de proyección
36
37 % factores de conversión de pixel a longitudes del modelo
38 factorConvX=l/px;
39 factorConvY=l/py;
40
41 Fproy=-[0 0 c];% punto del foco de proyección
42 Fespejo=[0 0 c];% punto del foco del espejo
43 FespejoDesplazado=Fespejo +SR;% punto del foco del espejo ...
       desplazado, se coge como origen en la recta que define el ...
       vector P
44
45 % inicializar variables
46 p1=[0 0 0];
47 v1=[0 0 0];
48
49 Pcopia=zeros(px,py);
50 %----Comienza la creación de la matriz de vectores P
51 tic;
52 for i=1:px
53   for j=1:py
54     x1=(i-(px/2))*factorConvX;% coordenada x del punto del plano de ...
         proyección
```

```
55      y1=(j-(py/2))*factorConvY;% coordenada y del punto del plano de ...
            proyección
56
57      if(x1==0)
58        x1=x1+factorConvX/2;% se hace para que no de errores con el 0 ...
              y se le suma algo que hará que se encuentre en el mismo ...
              pixel
59      end
60      if(y1==0)
61        y1=y1+factorConvY/2;
62      end
63      p1=[ x1 y1 h];% punto del plano de proyección
64      v1=p1-Fproy;% vector desde el foco de proyección hacia el punto ...
            en el plano de proyección
65
66      primer=(v1(1)/(a*v1(3)))^2 + (v1(2)/(b*v1(3)))^2-1/b^2;
67      segundo=2*(v1(1)*Fproy(1)/(a^2*v1(3))) - ...
            2*(v1(1)/(a*v1(3)))^2*Fproy(3) - ...
            2*(v1(2)/(a*v1(3)))^2*Fproy(3) + ...
            2*(v1(2)*Fproy(2)/(a^2*v1(3)));
68      tercer=1 + (v1(1)*Fproy(3))^2/(a*v1(3))^2 + ...
            Fproy(1)^2/a^2-2*(v1(1)*Fproy(1)*Fproy(3))/(a^2*v1(3)) + ...
            (v1(2)*Fproy(3))^2/(a*v1(3))^2 + Fproy(2)^2/a^2 - ...
            2*(v1(2)*Fproy(2)*Fproy(3))/(a^2*v1(3));
69
70      zpos=(-segundo + sqrt(segundo^2-4*primer*tercer)) / (2*primer);
71      zneg=(-segundo - sqrt(segundo^2-4*primer*tercer)) / (2*primer);
72
73      if (zpos < p1(3))
74        z2=zneg;
75      else
76        z2=zpos;
77      end
78
79      x2=v1(1)/v1(3)*(z2-Fproy(3))+Fproy(1);
80      y2=v1(2)/v1(3)*(z2-Fproy(3))+Fproy(2);
81
82      P=[x2 y2 z2] - Fespejo;% vector desde el foco del espejo hacia ...
            el espacio, es el que define la recta con la cual tenemos ...
            que ver si corta con algún objeto
83      % Xomni, Yomni y Zomni son las coordenadas X, Y y Z ...
            (respectivamente) de todos los vectores procedentes desde ...
            el foco del espejo.
84      Xomni(i,j)=P(1);
85      Yomni(i,j)=P(2);
86      Zomni(i,j)=P(3);
87    end
88  end
```

3.3.3 Traslaciones y giros

Para simular la translación del robot sólo es necesario trasladar el punto (F'). El conjunto de vectores no se modifica. El cambio en la elevación del robot se simula mediante una traslación en el eje z. La siguiente ecuación muestra la traslación del punto F':

$$\overrightarrow{F'_T} = \overrightarrow{F'} + \overrightarrow{T} \tag{3.7}$$

donde $\overrightarrow{F_T}$ es el vector cuyas componentes son las coordenadas de F_T, con respecto al sistema de referencia $\{x_c, y_c, z_c\}$. F'_T es el punto F' trasladado y T es el vector de traslación.

Por último, para tener en cuenta que el robot puede tener diferentes orientaciones en el espacio, es necesario el uso de una o más matrices de rotación para transformar cada vector $\overrightarrow{P}_{ij}$:

$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\text{sen}(\theta_x) \\ 0 & \text{sen}(\theta_x) & \cos(\theta_x) \end{bmatrix} \tag{3.8}$$

$$R_y(\theta_y) = \begin{bmatrix} \cos(\theta_y) & 0 & \text{sen}(\theta_y) \\ 0 & 1 & 0 \\ -\text{sen}(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix} \tag{3.9}$$

$$R_z(\theta_z) = \begin{bmatrix} \cos(\theta_z) & -\text{sen}(\theta_z) & 0 \\ \text{sen}(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

$$R_{total} = R_x(\theta_x) \cdot R_y(\theta_y) \cdot R_z(\theta_z) \tag{3.11}$$

$$\overrightarrow{PR}_{ij} = R_{total} \cdot \overrightarrow{P}_{ij} \tag{3.12}$$

donde R_x, R_y y R_z son la matriz de rotación con respecto a cada eje, R_{total} es la matriz de rotación resultante y $\overrightarrow{PR}_{ij}$ es el vector $\overrightarrow{P}_{ij}$ girado.

3.3.4 Opciones adicionales

El objetivo principal de este apartado es presentar una plataforma con el fin de crear imágenes omnidireccionales a partir de un entorno virtual como se presenta en el punto anterior. Además de esta función, el programa puede crear imágenes simulando diferentes tipos de sensores visuales, tales como cámaras simples, cámaras panorámicas y cámaras estéreo. Además, la plataforma puede generar y guardar los datos de nubes de puntos del entorno. Además de esto, la plataforma tiene también otras opciones configurables en la generación de las imágenes, tales como la adición de ruido y oclusiones (añadiendo nuevos objetos dentro del entorno).

En el Código 3.3 se muestra la definición de los vectores $\overrightarrow{P}_{ij}$ en el caso de cámara panorámica y en el Código 3.4 para el caso de cámara convencional. El Código 3.5 contiene el proceso para añadir ruido a la imagen una vez creada.

Código 3.3: Definición de vectores en cámara panorámica

```
% VECTORES CÁMARA PANORÁMICA

px=250;% pixels eje x
py=250;% pixels eje y
h=50;
z=2*h:-2*h/py:0;
r=100;
alfa=0:2*pi/(2*px):2*pi;
alfa2=0:2*pi/(2*px):2*pi;
x=r.*sin(alfa2);
y=r.*cos(alfa);
Fproy=[0 0 h/2];
factorConvX=1000/px;% factor de escala a modificar
factorConvY=1000/py;% factor de escala a modificar
for i=1:2*px+1
  for j=1:py+1
    P(i,j)={[x(i)-Fproy(1),y(i)-Fproy(2),z(j)-Fproy(3)]};
  end
 end
```

Código 3.4: Definición de vectores en cámara convencional

```
% VECTORES CÁMARA CONVENCIONAL

px=250;% pixels eje x
py=250;% pixels eje y
pk=px;
SR=[0 0 0];% punto del sistema de referencia
l=1000;% radio de la circunferencia de la imagen omnidireccional ...
    (en unidades del sistema de referencia del modelo 3d)
h=200; % distancia  del foco al plano imagen
disty=0;
distx=0;

%----------Matrices donde guardaremos las componentes x, y, z de ...
    los vectores P, el punto de origen de dichos vectores es Fespejo
Xsenc=zeros(px,py);
Ysenc=zeros(px,py);
Zsenc=zeros(px,py);

Fproy=[0+distx 0+disty h];% punto del foco de proyección

% inicializar variables
p1=[0 0 0];
v1=[0 0 0];

%factores de conversión de pixel a longitudes del modelo
factorConvX=l/px;% factor de escala a modificar
factorConvY=l/py;% factor de escala a modificar

%syms x y z x2 y2 z2 x3 y3 z3;

%--------Comienzo de la creación de los vectores
for i=1:px
```

```
31    for j=1:py
32      z1=(i-(px/2))*factorConvX;% coordenada x del punto del plano de ...
            proyección
33      y1=(j-(py/2))*factorConvY;% coordenada y del punto del plano de ...
            proyección
34      if(z1==0)
35        z1=z1+factorConvX/2;% se hace para que no de errores con el 0 ...
              y le sumo algo que hará que se encuentre en el mismo pixel
36      end
37      if(y1==0)
38        y1=y1+factorConvY/2;
39      end
40      p1=[h y1 z1];% punto del plano de proyección
41      v1= Fproy - p1; % vector desde el foco de proyección hacia el ...
            punto en el plano de proyección
42      P(i,j,:)=v1;% vector P
43      Xsenc(i,j)=P(i,j,1);
44      Ysenc(i,j)=P(i,j,2);
45      Zsenc(i,j)=P(i,j,3);
46    end
47  end
```

Código 3.5: Añadir ruido a una imagen

```
1  %  Añadir Ruido
2  nivel_ruido=10;
3  [size_ruido1,size_ruido2]=size(imagen);
4  imagen = double(imagen) +(nivel_ruido*((randn(size_ruido2)-0.5)*2));
5  imagen = uint8(imagen);
6  imagen(imagen>255)=255;
7  imagen(imagen<0)=0;
```

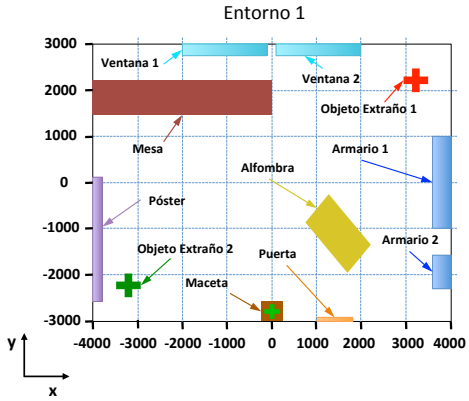
3.3.5 Ejemplos

Con el fin de comprobar el rendimiento del método propuesto, hemos creado dos entornos virtuales que representan dos habitaciones diferentes. En estos entornos, es posible crear una imagen utilizando diferentes configuraciones en el sistema de visión desde cualquier posición y con cualquier orientación.

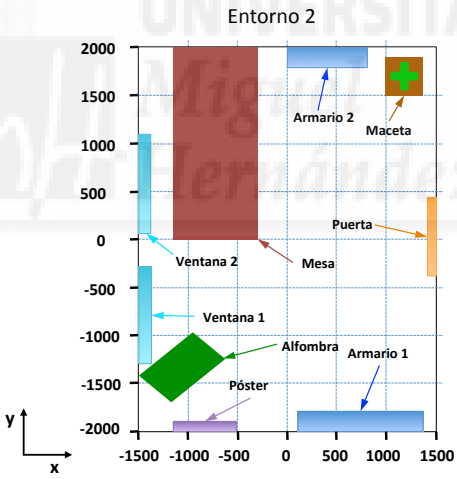
La Figura 4.11 muestra una vista cenital de ambos entornos.

La resolución de las imágenes se puede configurar sin limitaciones definiendo un conjunto de vectores mayor que parten del foco de la cámara. Para generar imágenes omnidireccionales, hemos optado por 250×250 píxeles para llevar a cabo los experimentos, lo cual conlleva la creación de $250 \times 250 = 62.500$ vectores que parten desde el foco de la cámara. Los parámetros utilizados en la ecuación del espejo (Ecuación 3.1) pueden ser configurados también, en este experimento hemos elegido $a = 40$ y $b = 160$.

Se han capturado varias imágenes en cada entorno, teniendo en cuenta cambios de elevación del robot, traslación, rotación e inclinación antes de capturar cada nueva



(a)



(b)

Figura 3.9: (a) Vista cenital del entorno 1. (b) Vista cenital del entorno 2. (Dimensiones en milímetros).

imagen. La Figura 3.10 muestra las coordenadas del foco del espejo F', la rotación en cada eje, y las imágenes generadas.

Como se mencionó anteriormente, la herramienta realizada también puede transformar estas imágenes omnidireccionales en proyecciones panorámicas. La Figura 3.11

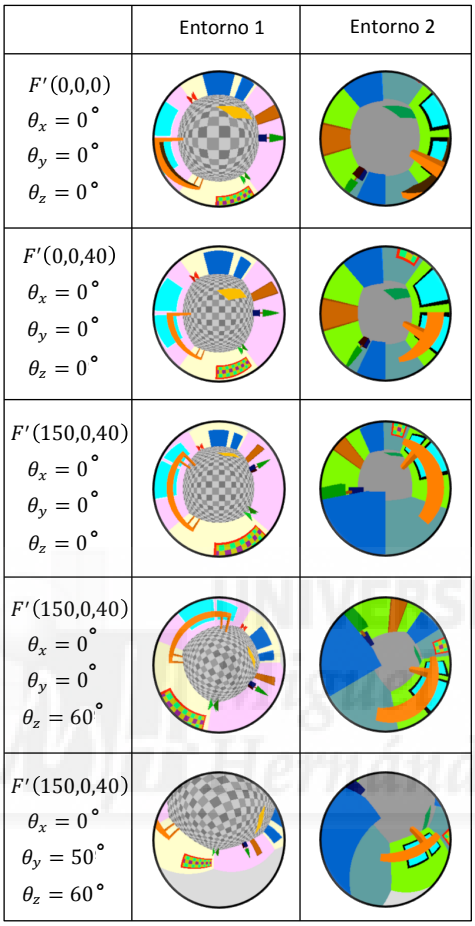


Figura 3.10: Imágenes virtuales generadas con la plataforma en ambos entornos, aplicando algunos cambios en la posición del robot y en la orientación. (dimensiones en centímetros).

muestra un ejemplo de la transformación de una imagen omnidireccional a una imagen panorámica del entorno 1.

También se han capturado diferentes imágenes de muestra utilizando varios tipos de sistemas visuales. La Figura 3.12 muestra tres tipos diferentes de imágenes tomadas con una cámara simple, una cámara panorámica y una cámara estéreo.

Por último, cabe destacar que hoy en día, algunos investigadores hacen uso de los datos de nubes de puntos en tareas de localización, tales como Andreasson et al. [4]. Por este motivo se ha añadido a la plataforma una opción adicional para generar una nube de puntos del entorno virtual. Este proceso consiste en guardar todos los puntos P (punto

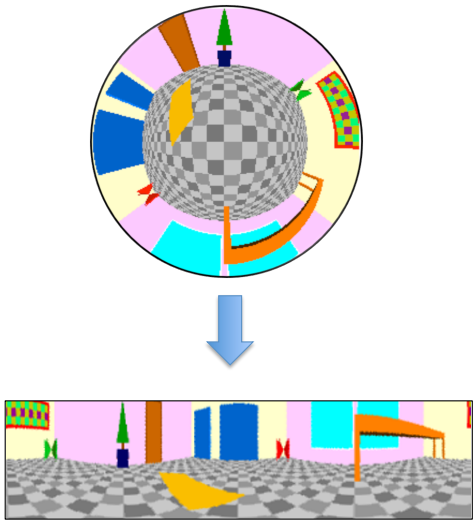


Figura 3.11: Ejemplo de una imagen panorámica generada a partir de una imagen omnidireccional del entorno virtual 1.

de intersección entre cada línea r_2 y los objetos en el entorno virtual). Estos datos de nube de puntos almacenan las coordenadas x, y y z de cada punto P y el color del objeto en este punto y emula la información capturada por una cámara RGB-d. La Figura 3.13 muestra un *data set* de nube de puntos de un entorno virtual usando una cámara virtual convencional.

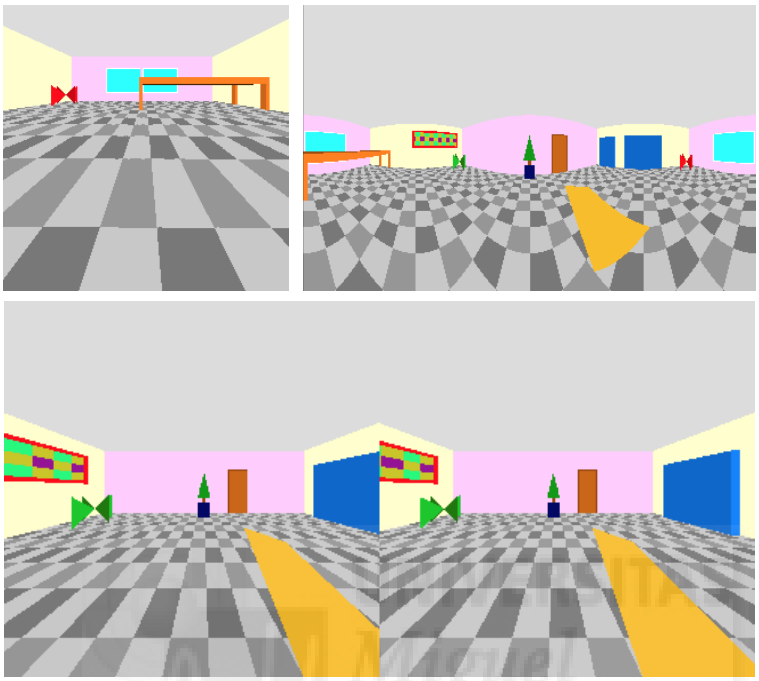


Figura 3.12: Diferentes tipos de sensores visuales. Arriba a la izquierda, cámara simple. Arriba a la derecha, cámara panorámica. Abajo, cámara estéreo.

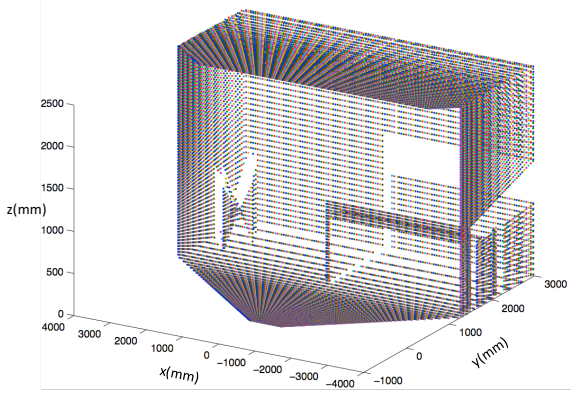


Figura 3.13: Ejemplo de una nube de puntos de un entorno virtual.

3.4 Conclusiones

Los resultados más relevantes del trabajo realizado en este capítulo han sido publicados en diversos congresos nacionales [8] e internacionales [9]. Entre las principales aportaciones realizadas en este trabajo, podemos destacar la creación de una plataforma para crear imágenes omnidireccionales utilizando entornos virtuales. El método se basa principalmente en la trayectoria de rayos y las intersecciones entre planos y rectas. Además, se pueden crear imágenes desde cualquier posición del entorno y con cualquier orientación del robot. Se espera que esta plataforma constituya una alternativa para generar conjuntos de imágenes de forma fácil, rápida y con flexibilidad, con el objetivo de probar y afinar cualquier nuevo algoritmo de creación de mapas y de localización. Esto puede ayudar a acelerar las etapas iniciales del diseño de algoritmos y a encontrar más rápidamente el valor óptimo para los parámetros del sistema visual y de las imágenes.

Los ejemplos de imágenes omnidireccionales han sido generados sintéticamente a partir de dos entornos virtuales diferentes. Los resultados demuestran que el método es capaz de crear conjuntos de imágenes con flexibilidad y eficiencia.

Los resultados de este trabajo nos animan a continuar con esta línea de investigación. Sería interesante para mejorar esta plataforma añadir más tipos de cambios en el entorno, como por ejemplo sombras. Además, este método permitirá diseñar algunos algoritmos para localizar el robot utilizando diferentes tipos de entornos, e incorporar estos algoritmos como opciones adicionales de la plataforma.

UNIVERSITAS
Miguel
Hernández

4 Localización 2D mediante el uso de descriptores holísticos

4.1 Introducción

En este capítulo, se propone una solución a los problemas de localización y creación de mapas utilizando sólo la información visual capturada por un sistema de visión omnidireccional montado en el robot. Este sistema se compone de una cámara que apunta a un espejo hiperbólico y captura imágenes omnidireccionales del entorno las cuales se describen con un solo descriptor de apariencia global.

Nuestro punto de partida es una base de datos de imágenes omnidireccionales capturadas en una rejilla de puntos en el entorno donde el robot debe navegar. Se estudiarán dos posibilidades iniciales, en la primera de ellas el robot conoce las posiciones del mapa y en la segunda de ellas no posee dicha información.

Los experimentos se han llevado a cabo con varias bases de datos de imágenes diferentes. La primera ha sido creada sintéticamente desde un entorno virtual, el cual se detalla en el Capítulo 3.3.5. Está formada por 4800 imágenes localizadas en un área de 8×6 metros. También usamos algunas bases de datos reales para probar la validez de los métodos propuestos, formadas por imágenes capturadas en áreas cuadradas de diferentes entornos de interior.

Por tanto, en este capítulo se presentan en primer lugar los descriptores de apariencia global utilizados y la forma en la cual se comparan dichos descriptores. A continuación se muestran los dos algoritmos propuestos para la localización del robot, partiendo de dos hipótesis iniciales diferentes. Finalmente, se detallan las bases de datos utilizadas para comprobar el funcionamiento de ambos algoritmos y los resultados obtenidos usando cada una de ellas, tanto usando bases de datos virtuales como reales.

4.2 Descriptores de apariencia global utilizados

Si bien en el Capítulo 2.1 se presentaron los descriptores de apariencia utilizados, sobre alguno de estos se ha realizado alguna variación con objeto de adecuar estos a las imágenes omnidireccionales empleadas. En este apartado se ofrecerá un detalle de estas variantes que se han propuesto en cada uno de estos.. Cada escena se representa a través de un descriptor que contiene información de apariencia global sin ninguna segmentación o extracción de características locales.

La complejidad de la descripción de imágenes omnidireccionales radica en que tienen una forma circular y debido a sus propiedades no pueden utilizarse los descriptores holísticos tradicionales directamente. Por este motivo, el método de descripción que ha sido empleado se basa principalmente en la transformada de Radon, ya que es una transformación que favorece la extracción de la información circularmente en las imágenes omnidireccionales, aunque no es el único que utilizamos. También se hace uso del descriptor *gist*, pero no aplicado a las imágenes omnidireccionales directamente, sino a la transformada de Radon. Este descriptor *gist* ha sido seleccionado debido a sus propiedades en las tareas de localización [54], que se detallarán más adelante.

4.2.1 Transformada de Radon

El descriptor holístico que se ha utilizado para describir imágenes omnidireccionales en este capítulo está basado en la transformada de Radon descrita en el Punto **??**. El método realiza la transformada de Radon considerando como entrada la imagen omnidireccional, lo que permite tener información sobre la forma de la imagen para cada uno de sus radios y almacenarla en las columnas de la transformada de Radon. En la Figura 4.2(a) podemos observar una imagen omnidireccional a la izquierda y su transformada de Radon a la derecha.

Llamaremos RT a la transformada de Radon de una imagen omnidireccional.

4.2.2 Descriptor *gist*

En el ámbito de esta Tesis se propone un descriptor basado en el concepto de *gist* de una imagen, descrito en el Capítulo 2.1.3. Se trabaja con un conjunto de imágenes interiores con muchas similitudes entre ellas y el descriptor debe poder trabajar con estas imágenes de tal forma que la distancia en el espacio del descriptor refleje la distancia geométrica entre los puntos donde se capturaron las imágenes. Sin embargo en el ámbito de esta Tesis se propone el uso del descriptor *gist*, no para describir directamente la imagen omnidireccional, sino para proporcionar un descriptor de la Transformada Radon de la imagen omnidireccional. Como esta transformada de Radon se puede interpretar como una imagen en escala de grises, solo se usa el concepto de prominencia (pero no la información de color) cuando se calcule el descriptor *gist*.

Esta variante del descriptor *gist* se emplea debido a que proporciona un rango de linealidad en la distancia entre ellos mayor que si simplemente obtenemos la diferencia de dos transformadas de Radon mediante POC (Ecuación 2.13). Los pasos que hemos

implementado para obtener el descriptor *gist* de la transformada de Radon de una imagen omnidireccional son:

1. Construir una pirámide a partir de la transformada Radon original: al principio, se crea una pirámide gaussiana de n transformadas de Radon para describir las características de la transformada de Radon en varias escalas. El primer nivel de la pirámide es la transformada original de Radon. Cada nuevo nivel se obtiene del anterior, aplicando un filtro de paso bajo gaussiano y haciendo un submuestreo para reducir su resolución. La Figura 4.1 muestra un ejemplo de una pirámide de cuatro niveles de una transformada de Radon.

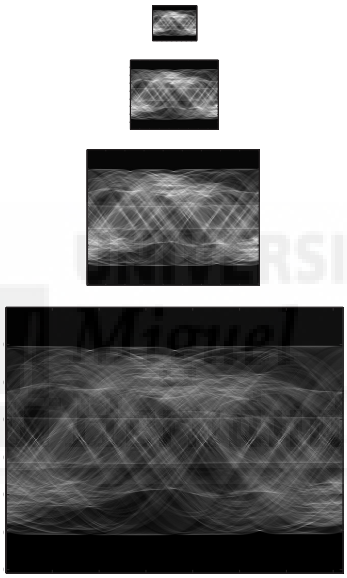


Figura 4.1: Ejemplo de una pirámide de 4 niveles de una transformada de Radon.

2. Filtrado de Gabor: Para incorporar la información de orientación en el descriptor, cada transformada de Radon de los dos niveles de la pirámide es filtrada por cuatro filtros Gabor con diferentes orientaciones $\theta = \{0, 45, 90, 135\}^\circ$. Como resultado, se obtienen cuatro matrices por nivel de pirámide, con información de orientación en las cuatro direcciones analizadas.

3. Agrupamiento en bloque. Finalmente, se necesita un proceso de reducción de la dimensionalidad. Con este objetivo, agrupamos los píxeles de cada matriz en bloques calculando el valor de intensidad promedio que tienen los píxeles en cada bloque. Hemos decidido usar bloques horizontales ya que este tipo de bloques es interesante debido a que la información en cada bloque es independiente de la orientación del robot. El descriptor final estará compuesto por b bloques horizontales por nivel de la pirámide y filtro Gabor, ya que se usará solo para fines de localización

(adicionalmente, podríamos haber definido bloques verticales si hubiéramos tenido que calcular la orientación del robot [54]).

La Figura 4.2 muestra el proceso que se propone para obtener el descriptor *gist* en este trabajo. En la Figura 4.2(a), podemos observar la transformada de Radon de una imagen omnidireccional. En la Figura 4.2(b) puede verse el uso de cada filtro de Gabor en la transformada de Radon y su agrupación en bloques. En este caso hemos usado $n = 2$ niveles y $b = 8$ bloques horizontales. Finalmente, en la Figura 4.2(c) se muestra la composición final del descriptor *gist*. El descriptor *gist* resultante estará formado por $4 \cdot n \cdot b$ componentes.

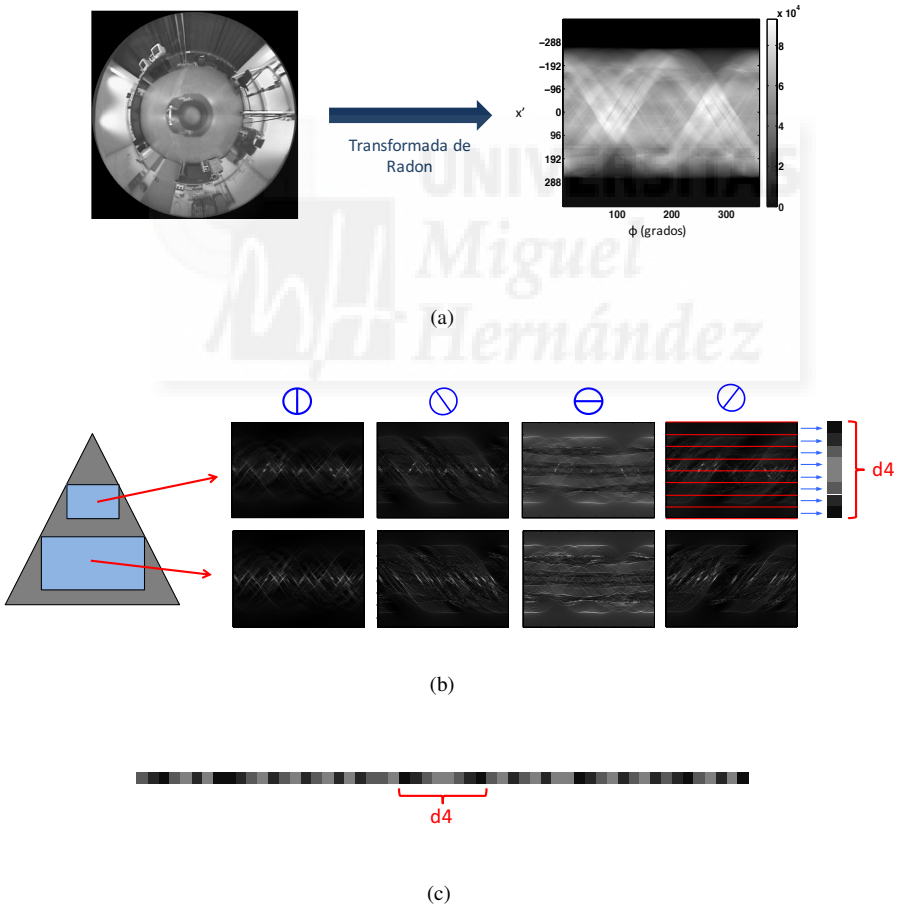


Figura 4.2: (**a**) Transformada de Radon de una imagen omnidireccional. (**b**) Matrices obtenidas aplicando cada uno de los cuatro filtros de Gabor a los dos niveles de la pirámide. (**c**) Composición final del descriptor *gist*.

4.3 Adaptación de POC (Phase Only Correlation)

En este punto se presenta la adaptación del algoritmo de comparación de matrices (POC) presentado en el Capítulo 2.2.2, el cual proporciona una medida de la diferencia entre la apariencia visual de dos localizaciones y la estimación del cambio de orientación del robot entre estas ubicaciones. El objetivo es conseguir una medida de distancia entre dos imágenes omnidireccionales diferentes que sea proporcional a la distancia métrica real entre ellas. Por lo tanto, será necesario algún algoritmo capaz de comparar los descriptores de las imágenes que nos proporcione la medida que estamos buscando. El más clásico es la distancia euclídea entre dos puntos pero en el caso de comparar dos transformadas de Radon resulta más útil emplear métodos de comparación más elaborados como POC.

En el caso abarcado en este capítulo, se comparan dos transformadas de Radon mediante el uso de POC (detallado en el Capítulo 2.2.2), pero esto no afecta al funcionamiento de POC porque la transformada de Radon se puede interpretar como una imagen. En general, permite obtener tanto la orientación relativa entre dos diferentes transformadas de Radon como un coeficiente de similitud entre ellas, como se muestra en [7].

Si se comparan las transformadas de Radon de dos imágenes omnidireccionales usando POC, el valor Δ_x (Ecuación 2.14) es proporcional a la orientación relativa α del robot al capturar las imágenes, de acuerdo con la Ecuación (4.1). La Figura 4.3 muestra las transformadas de Radon de dos imágenes omnidireccionales diferentes capturadas desde el mismo punto (x_w, y_w, z_w) pero con orientaciones del robot diferentes con respecto al eje z_w, α(Figura 4.3).

$$\alpha = \frac{\Delta_x \cdot 2\pi}{N} \tag{4.1}$$

De esta manera, POC es capaz de comparar dos transformadas de Radon independientemente de la orientación y también es capaz de estimar este cambio de orientación.

4.4 Estimación de distancia entre imágenes omnidireccionales

Para llevar a cabo el proceso de localización, necesitamos algún mecanismo para comparar descriptores y poder obtener una medida de la distancia relativa entre ellos. Con este propósito se proponen dos métodos diferentes para compararlos:

1. Comparar directamente las transformadas de Radon de las imágenes mediante POC (Capítulo 2.2.2).

2. Calcular lo descriptores *gist* de las transformadas de Radon de las imágenes omnidireccionales y compararlos entre sí utilizando la distancia euclídea (Capítulo 2.2.1).

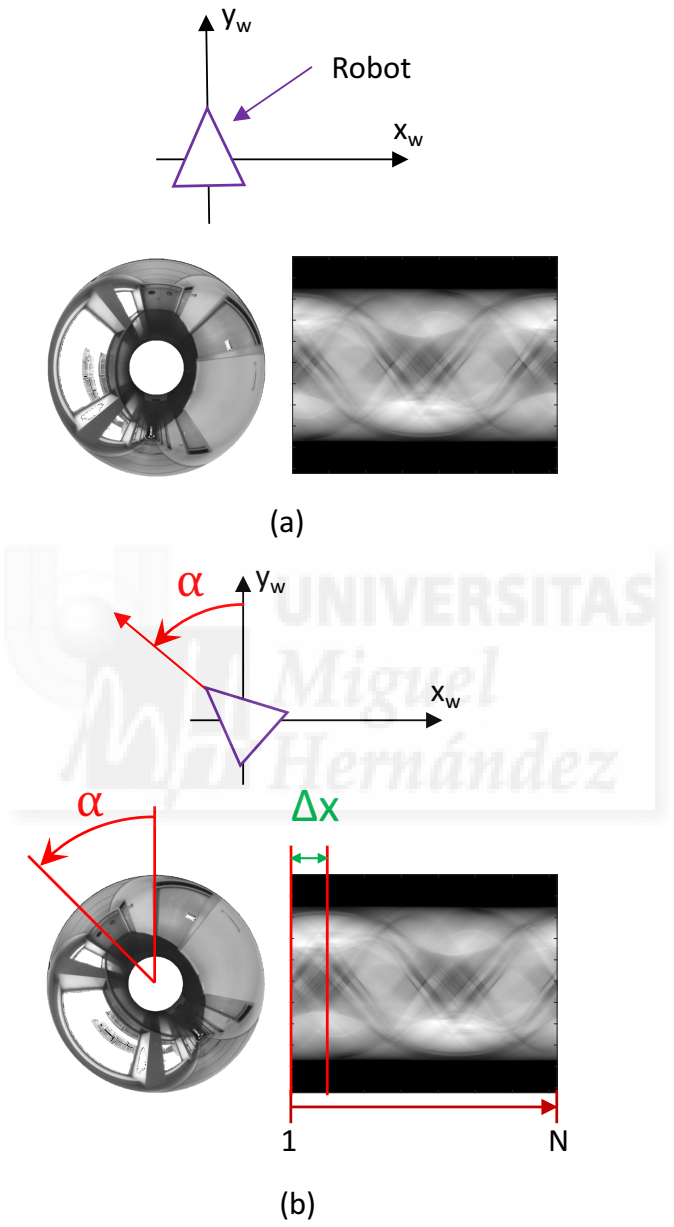


Figura 4.3: (a) Imagen omnidireccional capturada desde una posición específica de un entorno virtual y su transformada Radon. (b) Imagen omnidireccional tomada desde el mismo lugar cambiando solamente la orientación del robot alrededor del eje z_w, y su transformada Radon. Un cambio en la orientación del robot alrededor del eje z_w produce un desplazamiento en las columnas de la transformada Radon, Δ_x.

La primera de estas propuestas utiliza directamente las transformadas Radon (RT) de las dos imágenes para medir la distancia entre ellas por medio de la magnitud calculada usando POC (distancia POC) mediante la siguiente ecuación:

$$dist_{POC}(RT_1, RT_2) = 1 - sim_{POC}(RT_1, RT_2) \tag{4.2}$$

donde sim_{POC} se obtiene con la Ecuación 2.13.

La segunda de estas propuestas utiliza el descriptor *gist* para obtener una medida de distancia entre imágenes siguiendo los siguientes pasos:

1. En primer lugar, las dos imágenes omnidireccionales son transformadas mediante la transformada de Radon para crear dos descriptores RT_1 y RT_2.
2. En segundo lugar, se calcula el descriptor *gist* de RT_1 y RT_2 (Sección 4.2.2) para obtener $gist_1$ y $gist_2$.
3. Finalmente, se obtiene la distancia euclídea entre $gist_1$ y $gist_2$:

$$dist_{eu}(gist_1, gist_2) \tag{4.3}$$

Para comprobar el comportamiento de estas dos medidas de distancia se ha realizado un experimento considerando varios conjuntos de imágenes capturadas cada 50 mm a lo largo de varias rectas, y se muestra el resultado global (medias y varianzas) de la distancia desde la primera imagen hasta cada una de las siguientes para estos dos tipos de distancia. La Figura 4.4 muestra las distancias $dist_{POC}$ y $dist_{eu}$ frente a la distancia geométrica real entre los puntos donde se capturaron las imágenes. Según esta figura, la distancia POC muestra un comportamiento no lineal y muy abrupta, lo que nos lleva a pensar que esta distancia parece ser una opción prometedora para identificar la imagen más cercana, pero no es buena para estimar la distancia entre las imágenes debido a esa falta de linealidad. Esta distancia POC puede llegar a linealizarse utilizando la siguiente expresión:

$$dist_{POC_{linealizada}} = dist_{POC}^2 \tag{4.4}$$

donde $dist_{POC_{linealizada}}$ es la distancia POC linealizada y $dist_{POC}$ es la distancia POC original.

La Figura 4.5 muestra una comparación de la distancia POC y la distancia $POC_{linealizada}$ frente a la distancia geométrica real a lo largo de una trayectoria lineal en el entorno creado utilizando la base de datos virtual.

Por otro lado, la Figura 4.4 muestra que la distancia *gist* presenta un comportamiento bastante lineal, por lo que cabe esperar que sea una distancia útil para emplearla

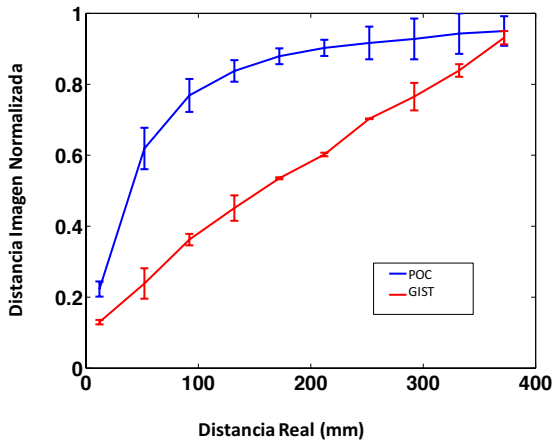


Figura 4.4: Distancia POC y distancia *gist vs.* distancia geométrica. Cada medida de distancia ha sido calculada a lo largo de diferentes caminos.

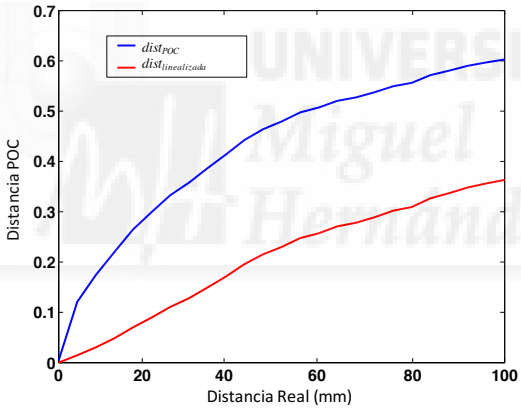


Figura 4.5: $dist_{POC}$ y $dist_{POC_{linealizada}}$ *vs.* distancia geométrica.

en tareas de creación de mapas. Esto se debe a que en el proceso de creación del mapa la linealidad de la distancia entre sus nodos es un factor determinante a la hora de posicionarlos.

Estas dos propuestas tienen características muy diferentes que las hacen útiles en diferentes tareas. Es decir, la primera propuesta ($dist_{POC}$), muestra un comportamiento muy selectivo en cuanto a distancia y su rango de linealidad se reduce bastante en comparación con la segunda propuesta. Por lo tanto esta distancia será útil en entornos pequeños o en los casos en que las imágenes estén muy cerca entre sí, a la vez que es mucho más restrictiva en cuanto a similitud entre imágenes. Por otro lado, la segunda propuesta ($dist_{eu}$) presenta un comportamiento lineal durante un rango de distancias mayor, por lo que será útil en entornos en los cuales las imágenes están más separadas.

La Figura 4.6 muestra un diagrama de flujo simplificado que muestra los pasos necesarios para calcular cada una de las dos distancias propuestas.

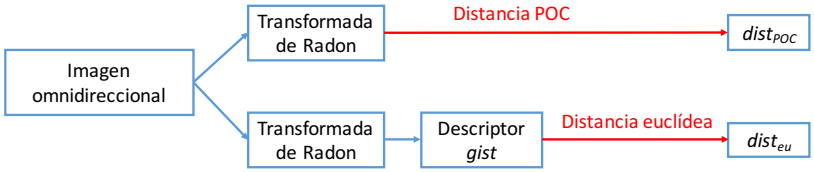


Figura 4.6: Diagrama de flujo del proceso de cálculo de distancias simplificado.

4.5 Algoritmos de localización

En esta sección, abordamos el problema de localización teniendo en cuenta que, inicialmente, el robot cuenta con un mapa del entorno que está compuesto por un conjunto de imágenes omnidireccionales de este entorno. A continuación, el robot captura una imagen desde una posición desconocida (imagen de prueba) y compara esta imagen con la información visual almacenada en el mapa. Realizando estos pasos el robot debe ser capaz de estimar su posición.

Con el fin de resolver el problema de localización en diferentes situaciones se han tomado en consideración dos puntos de partida diferentes: (1) las posiciones donde se capturaron las imágenes del mapa son conocidas (localización conociendo las posiciones de los nodos del mapa); y (2) estas posiciones son desconocidas, y deben estimarse antes del proceso de localización (construcción de mapas locales y localización en dichos mapas). Por lo tanto, hemos creado dos algoritmos diferentes para resolver cada una de estas dos situaciones:

1. conociendo las posiciones de los nodos del mapa.

2. Sin conocer las posiciones de los nodos del mapa.

El primer método consiste en calcular qué imagen del mapa es más similar a la imagen de prueba y, gracias a esta información, sabemos que el robot se encuentra en el entorno cercano de la imagen correspondiente. En el segundo método se calcula la distancia imagen entre la imagen de prueba y todas las del mapa y se retienen las imágenes más similares. El número de imágenes más cercanas retenidas es un parámetro que se puede modificar para optimizar el método. Con estas distancias, usamos el método de masa-muelle para estimar la posición de cada imagen. Como resultado, obtenemos el mapa local y la localización del robot dentro de ese mapa local. Estos dos métodos se detallan en los siguientes puntos.

4.5.1 Localización conociendo las posiciones de los nodos del mapa

Esta propuesta parte de la hipótesis en la cual el robot posee un mapa del entorno formado por imágenes omnidireccionales posicionadas en una distribución tipo rejilla en las posiciones en las cuales fueron capturadas en el entorno. Por lo tanto se parte con la información de las imágenes, y de su posición en el mapa.

El proceso de funcionamiento consta de los siguientes pasos:

1. El robot captura una imagen omnidireccional de su posición actual, la cual es desconocida (imagen de prueba). El objetivo del algoritmo es estimar esta posición.
2. Esta imagen se transforma utilizando la transformada de Radon.
3. La transformada de Radon de la imagen de prueba se compara con todas las transformadas de Radon del mapa utilizando la comparación POC y como resultado, obtenemos cuál es la imagen más similar, dentro del mapa, a la imagen de prueba.
4. La posición donde se tomó esta imagen omnidireccional es el vecino más cercano.

Después de este proceso, se supone que el robot está en esta posición. Este es un proceso de localización absoluto, ya que no se sabe la posición previa del robot ni su camino. La precisión depende principalmente de la distancia entre las imágenes del mapa. Esto quiere decir que el error será como máximo la mitad de la distancia entre posiciones del mapa, siempre y cuando la localización del vecino más cercano sea la correcta.

4.5.2 Localización y creación del mapa sin conocer las posiciones de los nodos del mapa

En esta propuesta se parte de la hipótesis de que el robot no posee ninguna información sobre las posiciones desde las cuales se capturaron las imágenes del mapa. Lo único que tiene son las imágenes del mapa, y por consiguiente, puede calcular las transformadas de Radon de cada imagen, pero no tiene información de las posiciones desde las cuales se capturaron las imágenes ni se sabe el orden en el que fueron capturadas. El objetivo del algoritmo es estimar estas posiciones en las regiones locales del entorno, es decir, tratando de crear un mapa local de los alrededores del punto de captura de la imagen de test, y estimando la posición de dicha imagen en este mapa.

El problema de la creación de mapas se resuelve mediante una propuesta en la cual se utiliza un modelo de masa-muelle, el cual viene detallado en el Capítulo 2.3, que representa el mapa final como un gráfico formado por nodos vinculados a través de conectores. Cada nodo contiene una imagen, y los conectores representan las relaciones vecinas entre dos nodos, de modo que se espera que las imágenes que se han capturado secuencialmente pertenezcan a los nodos que están conectados entre sí.

Independientemente de cual sea el método de descripción empleado así como la medida de distancia entre estos descriptores utilizados, la única información empleada

para la construcción del mapa a partir de un conjunto de imágenes es esta medida de distancia entre los descriptores de las escenas. Una vez que se construye el mapa, se hace uso del método de Procrustes [62], descrita en el Punto 2.4, para comparar el mapa resultante con el mapa real. Para simplificar el sistema de fuerzas, se ha utilizado una masa igual a uno para todos los nodos.

Por lo tanto, el algoritmo de localización propuesto en este apartado consta de los siguientes pasos:

1. Elegimos una cantidad de vecinos más cercanos para utilizarlos en el algoritmo (N_v). Para realizar esta selección se calcula la distancia POC entre el descriptor de la imagen de prueba y cada uno de los descriptores del mapa, y se retienen los N_v vecinos más cercanos. Estos vecinos más cercanos serán los descriptores Radon que presentan la distancia POC más baja a la transformada de Radon de la imagen de prueba. Se utiliza la distancia POC debido al comportamiento que muestra en la Figura 4.4, ya que cuanto más cercana es la imagen la distancia decrece exponencialmente. El número de descriptores seleccionados es un parámetro que se puede configurar dentro del algoritmo.

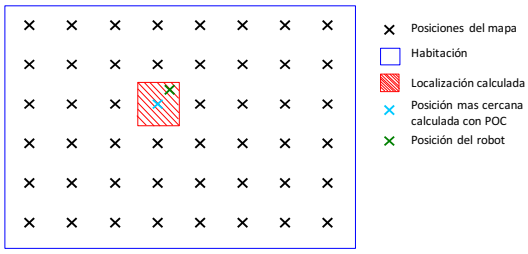
2. Se calcula el descriptor *gist* de la transformada Radon de cada uno de estos vecinos más cercanos y de la transformada Radon de la imagen de prueba.

3. Ahora se obtienen dos distancias diferentes, ya que, dependiendo del entorno y la forma de adquisición del mapa, será más útil una distancia u otra. Estas distancias son las descritas en el Punto 4.4. Son dos medidas de distancia diferentes: $dist_{POC}$ y $dist_{eu}$. En el caso de que sea más efectivo utilizar la distancia $dist_{POC}$, no será necesario calcular los descriptores *gist* de las imágenes. La elección de una distancia u otra dependerá del tipo de entorno en el cual se mueva el robot. Por ello se testearán ambas y se elegirá la más idónea en cada caso.

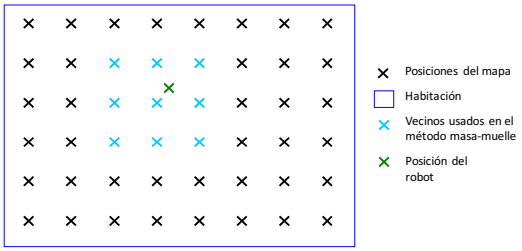
4. El método de masa-muelle se utiliza asignando una de estas distancias en los conectores. El experimento se repetirá usando cada una de las dos opciones y se analizarán los resultados obtenidos. Este paso proporciona una posición relativa entre cada descriptor *gist*. Como resultado, ahora se conoce la posición de las imágenes del mapa (mapa local) y la posición de la imagen de prueba, lo cual corresponde a la estimación de la posición del robot dentro del mapa local.

5. Para estimar el error de localización con respecto al mapa local, debemos tener en cuenta que este mapa puede presentar una rotación y un factor de escala en comparación con la posición real de las imágenes. Para considerar estos efectos, usamos el enfoque Procrustes [62] para estimar el error cometido en la estimación de la posición de la imagen de prueba. En el Punto 2.4 se detalla el funcionamiento de dicho algoritmo. Este método elimina la rotación y el factor de escala del mapa local para poder calcular el error final, que será la distancia entre la posición real del robot y la posición calculada después de eliminar ambos efectos.

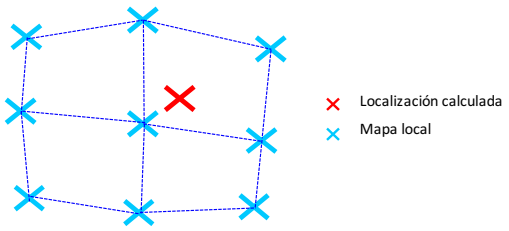
La Figura 4.7(a) muestra el área de localización calculada utilizando el primer método (posición más cercana del mapa). La Figura 4.7(b) muestra la ubicación calculada por el segundo método con los nueve vecinos más cercanos (creación de mapa y localización local). En esta figura, las posiciones del mapa se representan en los puntos de captura reales para fines de representación, pero el algoritmo no conoce estas posiciones. En la Figura 4.7(c), se muestra el mapa local creado y la localización del robot. Para hacer este mapa local, la posición inicial de cada nodo para iniciar el método de masa-muelle es aleatoria. Esta aleatorización puede tener un efecto en el resultado final. Por esta razón, ejecutamos el algoritmo varias veces para calcular el error. Los resultados de este método de masa-muelle son el mapa local y la localización del robot.



(a)



(b)



(c)

Figura 4.7: (**a**) Localización conociendo las posiciones de los nodos del mapa. (**b**) Posiciones de los vecinos más cercanos utilizados en el algoritmo masa-muelle. (**c**) Localización y creación del mapa sin conocer las posiciones del mapa.

4.6 Experimentos y resultados

En esta sección, primero, se presenta la base de datos virtual creada para probar los métodos y, en segundo lugar, se describe la base de datos de imágenes reales utilizada.

4.6.1 Base de datos virtual

Para comprobar el rendimiento de la técnica propuesta, hemos creado un entorno virtual que representa una sala interior. En este entorno, es posible crear imágenes omnidireccionales desde cualquier posición utilizando el sistema catadióptrico que se muestra en la Figura 4.8. La Figura 4.11 a muestra una vista en planta del entorno. Este método de creación de imágenes virtuales se encuentra detallado en el Capítulo 3 de la presente tesis.

Hemos optado por crear una base de datos con imágenes omnidireccionales que tienen una resolución de 250 $\times$ 250 píxeles, y se han creado utilizando un sistema catadióptrico compuesto por una cámara y un espejo hiperbólico cuya geometría se describe en la Figura 4.8. Los parámetros utilizados en la ecuación espejo son $a = 40$, $b = 160$ y $h = 41$.

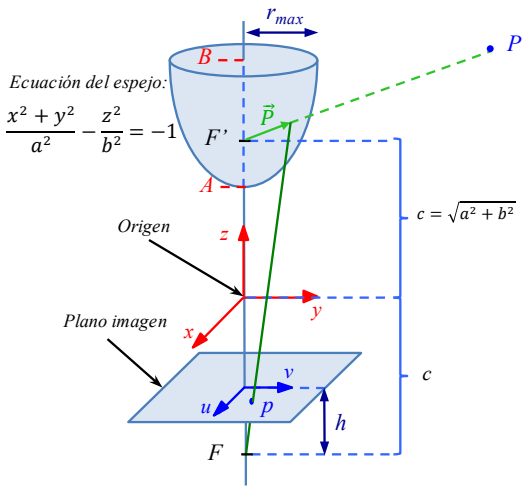


Figura 4.8: Esquema de sistema catadióptrico usado para calcular las imágenes sintéticas omnidirecconales.

Se han capturado varias imágenes en el entorno para crear el mapa desde diferentes posiciones en el plano del suelo. El mapa está compuesto por 4800 imágenes capturadas en una cuadrícula de 8 $\times$ 6 metros con un paso de 10 cm entre las imágenes. Para llevar a cabo los experimentos, podemos cambiar el paso entre las posiciones de la rejilla para probar la influencia de este parámetro. Debemos tener en cuenta que cuanto mayor sea la distancia entre posiciones del mapa, menos imágenes lo componen. La figura 4.11(b) muestra una imagen omnidireccional del entorno creado con nuestro programa como ejemplo.

4.6.2 Bases de datos Reales

En cuanto a bases de datos reales, se han utilizado dos diferentes dependiendo del algoritmo que se quiere probar.

La primera de ellas es la base de datos de Bielefeld [66], y se usa para la comprobación de los resultados del primer método. Esta base de datos contiene varios conjuntos de imágenes omnidireccionales capturadas en el mismo entorno con cambios en la posición de algunos objetos y en las condiciones de iluminación. Las imágenes se capturaron en una cuadrícula de 10 $\times$ 17 imágenes con un paso entre ellas de 30 cm. Por lo tanto, el área de la cuadrícula de captura fue de 2.7 m $\times$ 4.8 m, que cubría casi todo el espacio libre del suelo. Estos mapas de imágenes son:

- Original: la condición estándar o predeterminada de la habitación. Las imágenes se recogieron con barras fluorescentes suspendidas, cortinas y puertas cerradas y sin objetos extraños presentes.
- Chairs: tres sillas de oficina adicionales se colocaron dentro de la región de captura.
- Arboreal: se agregó una planta interior alta (3 m) al centro de la cuadrícula de captura.
- Twilight: cortinas y puerta abierta. La colección de imágenes duró desde el atardecer hasta la noche.
- Doorlit: la barra de luz cerca de la ventana se encuentra apagada.
- Winlit: la barra de luz cerca de la puerta estaba apagada.

La Figura 4.9 muestra un ejemplo de una imagen omnidireccional en cada una de estas bases de datos.

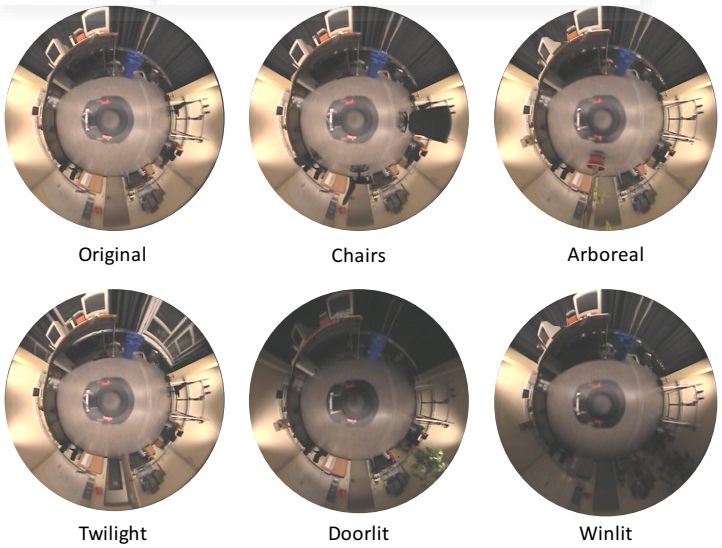


Figura 4.9: Ejemplo de una imagen omnidireccional de cada una de las bases de datos de Bielefeld.

Y la segunda base de datos consta de un conjunto de imágenes capturadas en dos habitaciones diferentes de la Universidad Miguel Hernández de Elche. Cubre dos salas diferentes, un laboratorio y un despacho. Las imágenes se han capturado en una cuadrícula con 8 $\times$ 8 imágenes con un paso de 20 cm, en ambos casos. La Figura 4.10 muestra un ejemplo de una imagen omnidireccional en cada una de estas bases de datos.

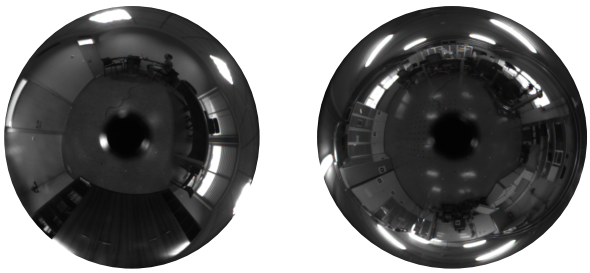


Figura 4.10: Ejemplo de una imagen omnidireccional de cada una de las bases de datos de la Universidad Miguel Hernández.

4.6.3 Resultados obtenidos con la base de datos virtual

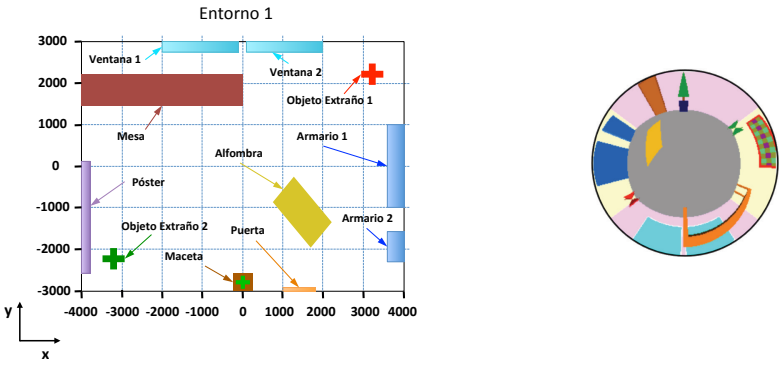
Para probar nuestros métodos, estudiaremos la influencia de algunos parámetros como la distancia entre imágenes de la cuadrícula del mapa y el número de vecinos utilizados en el método de masa-muelle. Por tanto, en esta subsección se comparan los resultados de las pruebas para tratar de optimizar cada uno de los métodos antes de probarlos con imágenes reales.

En primer lugar, analizamos el primer método (obteniendo la posición más cercana del mapa) y la influencia de la distancia entre imágenes del mapa. En segundo lugar, analizamos el segundo método (creación de mapa local y localización) y la influencia de la distancia entre imágenes del mapa, así como la influencia del número de vecinos más cercanos utilizados.

4.6.3.1 Resultados del primer método: localización conociendo las posiciones de los nodos del mapa

En este experimento, analizamos la influencia de cuatro configuraciones diferentes de distancias entre posiciones del mapa consecutivas. Las distancias que utilizaremos son 100, 200, 300 y 400 mm. El tamaño de la cuadrícula del mapa es de 8 m $\times$ 6 m en cada uno de los cuatro casos, por lo que el número de imágenes del mapa depende de la distancia entre imágenes. Esto tendrá una influencia importante en el coste computacional debido a tener que trabajar con un mapa más grande.

La Figura 4.12 muestra la distancia POC (Ecuación (5.7)) entre la transformada de Radon de una imagen de prueba y la transformada de Radon de cada imagen del mapa



(a) (b)

Figura 4.11: (**a**) Vista en planta del entorno virtual. (**b**) Ejemplo de una imagen omnidireccional capturada en el punto x = 0 e y = 0.

(200 × 200). La posición de la imagen de prueba es x = −2239 mm e y = −1653 mm. Además, la posición correspondiente de acuerdo con esta figura (el mínimo de la función 2D) es x = −2200 e y = −1600. Como podemos ver, la distancia disminuye bruscamente alrededor de esta posición.

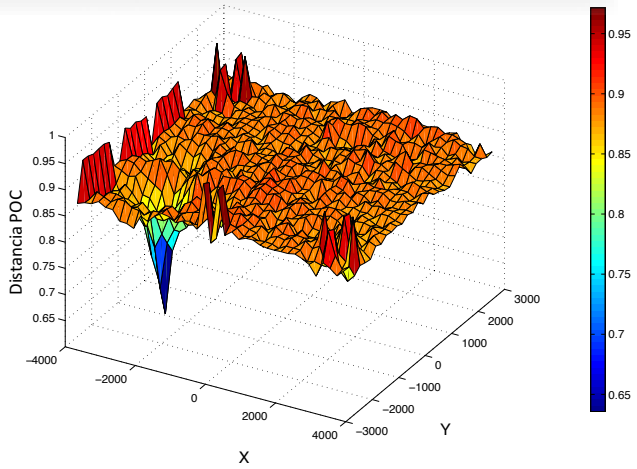


Figura 4.12: Distancia entre la transformada de Radon de una imagen de test y todas las transformadas de Radon de las imágenes del mapa. La posición de la imagen de test es x = −2239 mm e y = −1653 mm.

La Figura 4.13 muestra el resultado final de esta prueba. Hemos utilizado 3500

imágenes de prueba capturadas desde diferentes posiciones aleatorias del entorno. En cada barra, la parte azul representa la proporción de localizaciones correctas; la parte verde representa que el método ha localizado la segunda posición más cercana (*i*.e., la posición calculada no es la posición más cercana, sino la segunda posición más cercana del mapa); y la parte roja es la proporción de errores para cada configuración de distancia entre posiciones del mapa.

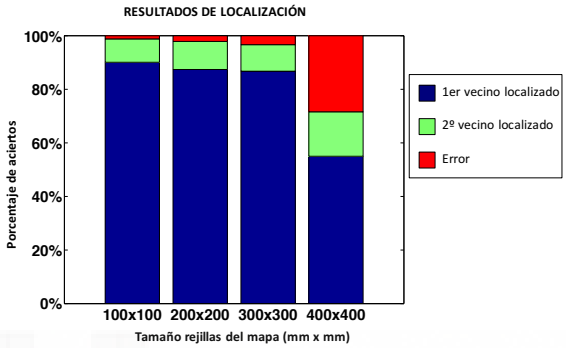


Figura 4.13: Resultados de la localización de la imagen más cercana: Ratio de acierto.

Tras realizar la localización el robot puede calcular su orientación con respecto a la imagen más cercana del mapa. La media del error de orientación frente a la distancia entre las posiciones del mapa (Ecuación (4.1)) se muestra en la Figura 4.14. Este error aumenta cuando la distancia entre las posiciones del mapa es mayor porque la imagen de prueba y la imagen del mapa correspondiente son menos similares.

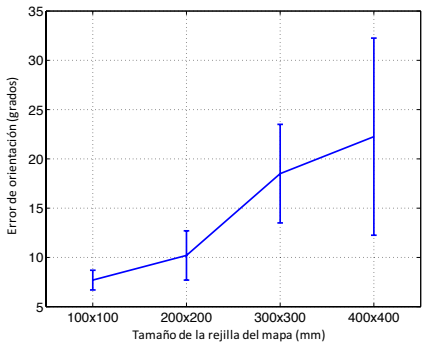


Figura 4.14: Media del error de orientación.

Podemos observar que el método funciona correctamente con una distancia entre posiciones del mapa que no exceda 300 $\times$ 300 milímetros, porque si este valor aumenta, la distancia de POC comienza a saturarse, como se indica en la Sección 4.4.

4.6.3.2 Resultados del segundo método: localización y creación de mapa local sin conocer las posiciones del Mapa

En esta subsección, se analiza el segundo método, y estudiamos la influencia de la distancia entre las posiciones del mapa y el número de vecinos más cercanos utilizados en el algoritmo de masa-muelle. Para hacer este experimento, hemos elegido las mismas distancias entre las posiciones del mapa que en la subsección anterior: 100, 200, 300 y 400 mm. Luego, elegiremos el valor que proporcione el mejor resultado, y después optimizaremos el número de vecinos más cercanos: probaremos las cuadrículas de 9, 25 y 49 vecinos más cercanos.

El algoritmo se ha lanzado con 500 imágenes de prueba capturadas desde posiciones aleatorias generadas con nuestro programa, repitiendo el experimento cuatro veces con cada imagen de prueba (ya que los resultados devueltos por el método masa-muelle pueden variar entre experimentos, debido a la fase inicial de inicialización de posiciones iniciales de forma aleatoria). La Figura 4.15 muestra la distribución de error de los experimentos con la base de datos virtual usando la distancia POC, dependiendo del número de vecinos más cercanos y la distancia entre imágenes del mapa. Este error es el error de localización de la imagen de prueba calculado de acuerdo con el Paso 5 (Sección 4.5.2). La Figura4.16 presenta la distribución del error de los mismos experimentos, pero con la distancia *gist*. En este entorno, la distancia *gist* presenta un error de localización menor que la distancia POC. Los parámetros utilizados en los descriptores *gist* para los experimentos son $n = 2$ niveles de la pirámide y $b = 16$ bloques horizontales.

Como podemos ver en las figuras, los mejores resultados se han logrado con una distancia ente imágenes igual a 100 mm y 200 mm, pero también debemos tener en cuenta el tiempo de cálculo. La Figura 4.17 muestra el tiempo promedio de cálculo por iteración en cada caso, en el supuesto de que hayamos elegido 25 vecinos más cercanos. La distancia entre las posiciones del mapa de 100 mm no es aconsejable, porque el tiempo de cálculo es relativamente alto. Por esta razón, elegimos la distancia de 200 mm para hacer las siguientes pruebas, con imágenes reales, porque teniendo en cuenta el error y el tiempo de cálculo, es la mejor opción. Podemos observar que el tiempo computacional del segundo método es el mismo para todos los casos, porque no depende del tamaño del mapa, sino del número de vecinos más cercanos.

4.6.4 Resultados con la base de datos real

Una vez que hemos analizado ambos métodos con una base de datos virtual, ahora se procede a probarlos con algunas bases de datos compuestas por imágenes capturadas en entornos reales de trabajo bajo condiciones reales de iluminación.

Primero, probamos el primer método usando una base de datos de terceros que tiene diferentes condiciones de iluminación (el conjunto de imágenes Bielefeld).

Segundo, probamos el segundo método con nuestras propias bases de datos (dos salas diferentes de la Universidad Miguel Hernández de Elche).

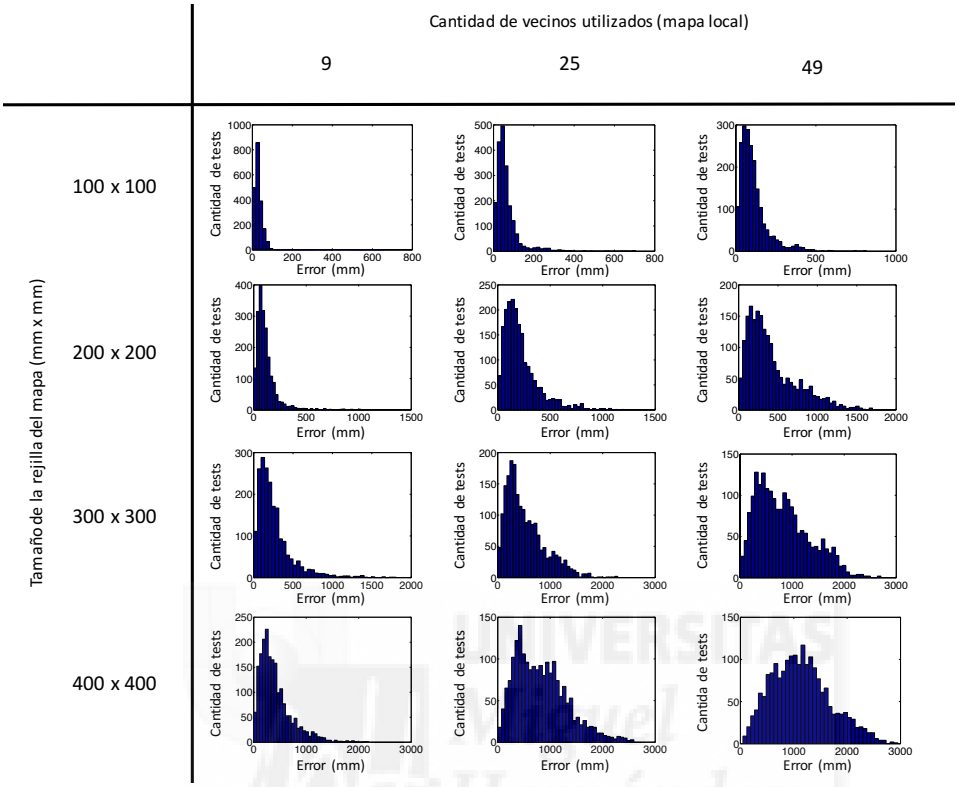


Figura 4.15: Distribución del error en los experimentos utilizando la base de datos virtual y la distancia POC.

4.6.4.1 Resultados del primer método: localización conociendo las posiciones de los nodos del mapa

A la hora de probar este método, hemos utilizado la base de datos de Bielefeld [66].

Para llevar a cabo este primer experimento, hemos utilizado el conjunto "original" como nuestro mapa de imágenes y las imágenes de los otros conjuntos como imágenes de prueba. La Figura 4.18 muestra la tasa de éxito de este experimento. Hemos utilizado 20 imágenes de prueba aleatorias de cada base de datos para obtener esta cifra. Los resultado demuestran que en todos los entornos la tasa de acierto en la localización del vecino más cercano es superior al 80% y en la localización de por lo menos el segundo más cercano es mayor al 90%.

Este es un experimento bastante interesante, ya que muestra la solidez del algoritmo, incluso cuando las imágenes de prueba presentan cambios con respecto al mapa.

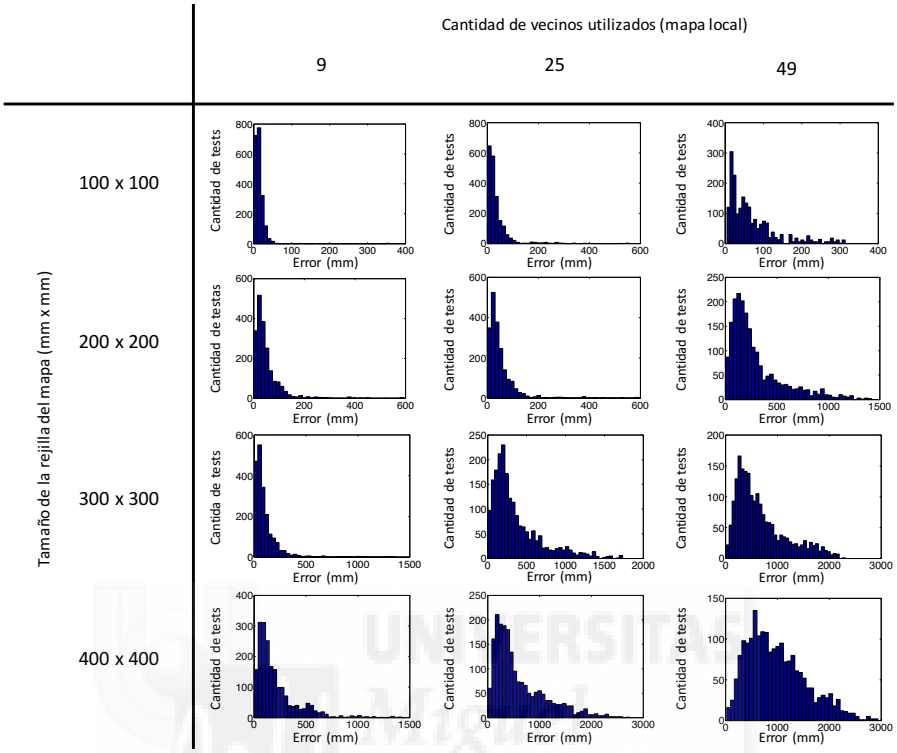


Figura 4.16: Distribución del error en los experimentos utilizando la base de datos virtual y la distancia *gist*.

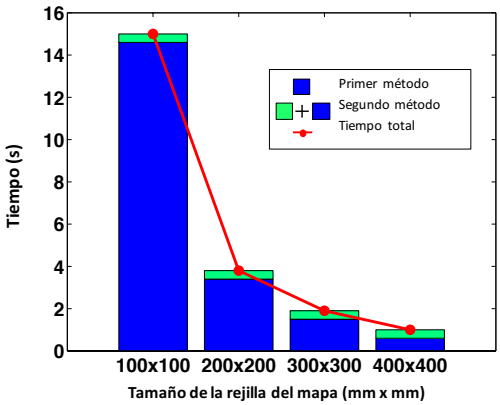


Figura 4.17: Tiempo computacional para cada iteración. Para obtener esta gráfica se consideran 25 vecinos más cercanos en el segundo método.

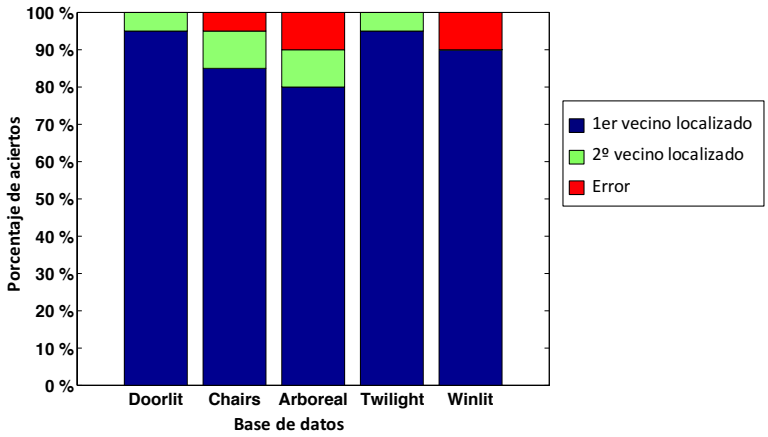


Figura 4.18: Ratio de acierto utilizando la base de datos de Bielefeld.

4.6.4.2 Resultados del segundo método: localización y creación de mapa local sin conocer las posiciones del Mapa

En este caso, hemos utilizado la base de datos de imágenes, capturada en la Universidad Miguel Hernández de Elche.

Seguimos dos pasos. En primer lugar, detectamos los vecinos más cercanos mediante la comparación POC de las transformadas de Radon de las imágenes del mapa. Segundo, creamos el mapa local con estas transformadas de Radon usando la distancia *gist* o POC. Finalmente, localizamos nuestro robot en este mapa local. La Figura 4.19 muestra la distribución de errores del segundo método utilizando las dos bases de datos ("laboratorio" y "despacho") y ambas medidas de distancia (distancia POC y *gist*). El experimento se llevó a cabo con 13 imágenes de prueba y 50 veces por imagen de prueba en el laboratorio y el mismo numero de imágenes de prueba y 50 veces por imagen de prueba en la oficina. La Figura 4.20 muestra la distribución del error del segundo método utilizando las bases de datos de la Universidad Miguel Hernández, añadiendo oclusiones a cada imagen de prueba. Dichas oclusiones son zonas rectangulares de la imagen de test en las cuales se cambia el valor de los píxeles a un mismo valor que se calcula como la media de todos los valores de los píxeles de la imagen; la zona total cubierta por dichas regiones es un porcentaje del total de los píxeles de la imagen. En la Figura 4.21 se muestran dos ejemplos de imágenes omnidireccionales con oclusiones. Teniendo en cuenta que el paso entre imágenes es de 20 cm, la distribución del error de localización obtenido en cada uno de los casos se centra en valores cercanos a la mitad de dicha distancia, exceptuando el caso en el cual se añaden oclusiones y se utiliza la distancia entre descriptores *gist*, en el cual la distribución del error cometido en la localización se centra en valores un poco mayores. Estos resultados se analizan más en profundidad en el Punto 4.6.5.

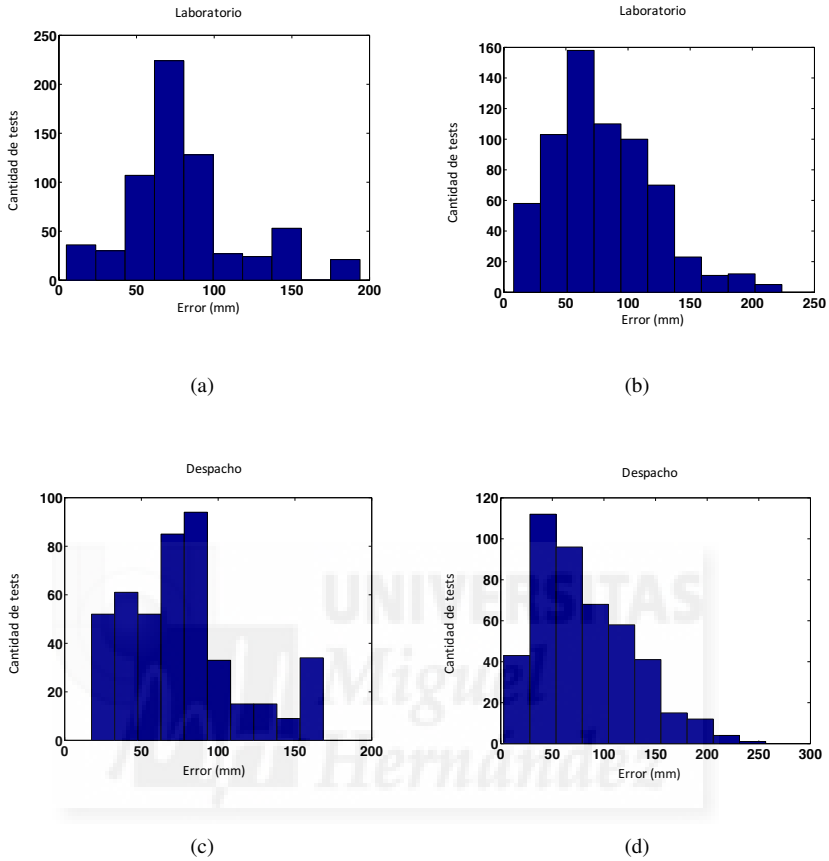


Figura 4.19: Distribución del error utilizando el segundo método. (**a**,**b**) Usando la base de datos del laboratorio y la distancia *gist* y la distancia POC, respectivamente. (**c**,**d**) Usando la base de datos del despacho y la distancia *gist* y la distancia POC, respectivamente. El tamaño de rejilla utilizado en estos cuatro experimentos ha sido de 200×200 mm.

4.6.5 Análisis de resultados

A la vista de los resultados obtenidos en este capítulo, el primer método (detección de la imagen más cercana del mapa) es un método muy bueno si el robot tiene información de las posiciones en las que se tomaron las imágenes del mapa. Permite llevar a cabo una localización absoluta, ya que el robot no necesita información de su ubicación anterior. Los resultados han demostrado que es un método muy fiable para estimar la posición más cercana del mapa, incluso cuando hay algunos cambios presentes (condiciones de iluminación, objetos nuevos, *etc.*). En cuanto a los valores de los parámetros, se ha estudiado la influencia del tamaño de la rejilla en el desempeño general del método. La mejor opción es 200×200 mm, ya que ofrece un buen equilibrio entre la precisión y el tiempo de

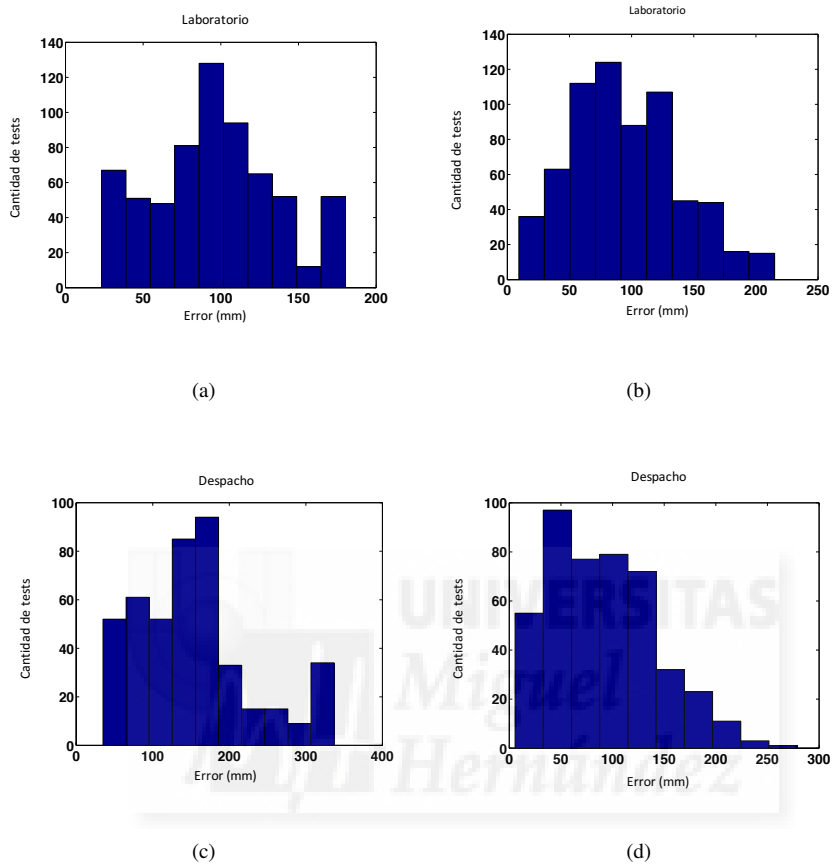
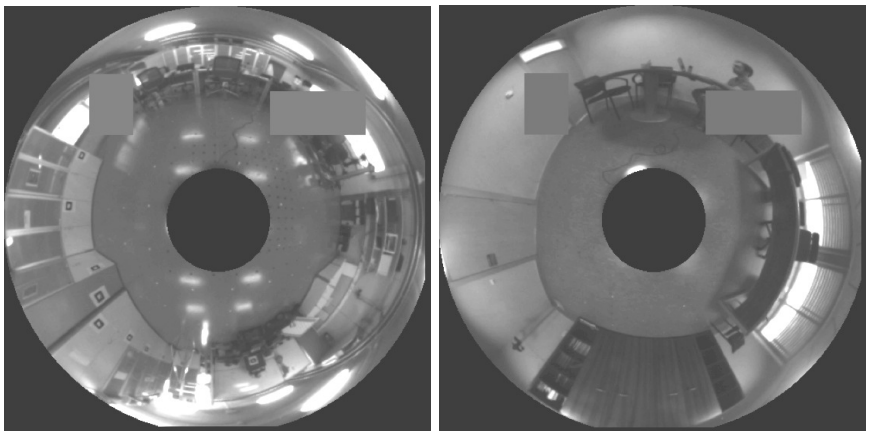


Figura 4.20: Esta figura representa los mismos experimentos que los mostrados en la Figura 4.19, añadiendo oclusiones aleatorias a la imagen de test. (**a**,**b**) Usando la base de datos del laboratorio y la distancia *gist* y la distancia POC, respectivamente. (**c**,**d**) Usando la base de datos del despacho y la distancia *gist* y la distancia POC, respectivamente. El tamaño de rejilla utilizado en estos cuatro experimentos ha sido de 200 $\times$ 200 mm.

cálculo.

El segundo método (creación de mapa local y localización) es un buen método si el robot no conoce las posiciones del mapa, ya que crea un mapa local. Este método también localiza las imágenes más cercanas y la imagen de prueba dentro de este mapa local. En cuanto a los valores de los parámetros, primero, la mejor opción es una distancia entre las posiciones del mapa de 200 $\times$ 200 mm, ya que ofrece un buen equilibrio entre la precisión y el tiempo de cálculo; segundo, el número de vecinos más cercanos que mejores resultados ofrece es nueve, porque conduce a un error menor; y finalmente, el tipo de distancia (distancia POC o *gist*) debe elegirse dependiendo de las características del entorno, porque con la base de datos virtual, la distancia *gist* es mejor, pero con la base de datos real de la Universidad Miguel Hernández, la distancia POC ha presentado



(a) (b)

Figura 4.21: (**a**) Ejemplo de una imagen omnidireccional capturada en nuestro laboratorio con dos oclusiones aleatorias. (**b**) Ejemplo de una imagen omnidireccional capturada en el despacho con dos oclusiones aleatorias.

resultados más precisos. En consecuencia, en relación con esta medida de distancia no podemos extraer resultados concluyentes a favor de una u otra.

Para finalizar, y con objeto de comparar los resultados obtenidos según los algoritmos propuestos, se ha realizado una evaluación comparativa entre el descriptor propuesto (descriptor Radón) y otros dos métodos de descripción clásicos en una tarea de localización. El objetivo es estudiar el rendimiento del descriptor propuesto, tanto en términos de precisión de localización como de costo computacional.

Los métodos de descripción que utilizamos para hacer esta comparación son la firma de Fourier (FS (Fourier Signature)) [54], un método clásico para describir la apariencia global de una escena panorámica, y SIFT [40], un método típico para extraer y describir características locales de una escena. La prueba consiste en determinar la imagen más cercana de un conjunto de imágenes omnidireccionales, y la base de datos utilizada es el laboratorio (de la base de datos real capturada en la Universidad Miguel Hernández).

En primer lugar, se realizó un experimento para optimizar el coste computacional del método de la transformada de Radón (RT). Con este objetivo, repetimos el experimento de localización varias veces, usando diferentes resoluciones, desde RT $\in \mathbb{R}^{709x360}$ a RT $\in \mathbb{R}^{22x6}$. Los resultados (tasa de éxito y costo computacional) se muestran en la Figura 4.22. Como se muestra, la resolución de la transformada de Radon puede reducirse a 173×45 manteniendo la tasa de éxito, lo que mejora claramente el costo computacional del proceso. De esta manera, elegimos esta resolución para llevar a cabo la comparación.

Al usar la firma de Fourier, cada escena omnidireccional se transforma al formato panorámico y se describe de acuerdo con [54]. La distancia euclídea se usa para comparar

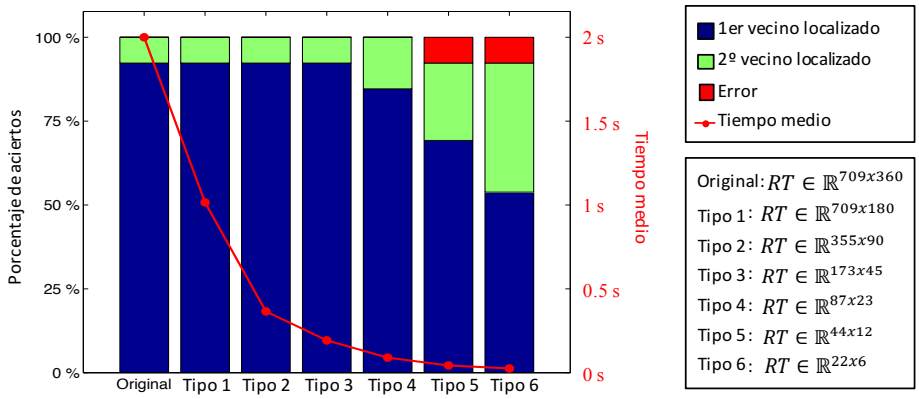


Figura 4.22: Ratio de acierto y tiempo medio de computación utilizando diferentes tipos de transformadas de Radon con la base de datos del laboratorio.

dos descriptores. Por otro lado, al usar SIFT, las características SIFT de las escenas omnidireccionales se extraen, se describen y se comparan. La distancia entre dos imágenes im_1 e im_2 utilizando este enfoque se define en la Ecuación 4.5.

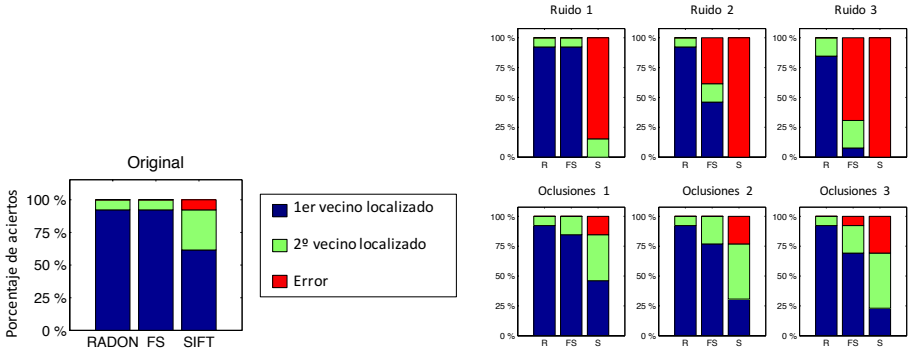
$$dist_{SIFT}(im_1, im_2) = \frac{1}{\sum_{i=1}^{n} \frac{1}{d_i}} \tag{4.5}$$

donde d_i es la distancia promedio entre cada característica SIFT en la primera imagen y su característica correspondiente en la segunda imagen, suponiendo que ambas imágenes estuviesen superpuestas. n es el número de características locales SIFT emparejadas.

La Figura 4.23(a) muestra la tasa de éxito del experimento comparativo utilizando las imágenes de prueba originales. Para obtener esta figura se han utilizado trece imágenes de prueba aleatorias. Más tarde, repetimos el experimento teniendo en cuenta la presencia de diferentes niveles de ruido y oclusiones en las imágenes de prueba, para probar la solidez de cada método en estas situaciones típicas. Los resultados se muestran en la Figura 4.23(b). Los ruidos 1, 2 y 3 son ruidos aleatorios con un valor máximo igual al 20%, 40% y 60% de la intensidad máxima. Las oclusiones 1, 2 y 3 implican que la imagen de prueba tiene una oclusión del 10%, 20% y 30%, respectivamente.

En cuanto al tiempo computacional utilizado para hacer este experimento, el método basado en la transformada de Radon necesita 0.17 s; SIFT emplea 12 s (usando el algoritmo de [40]) y el método de firma de Fourier necesita 0.06 s.

Teniendo en cuenta estos resultados, se ha demostrado que el descriptor basado en la transformada de Radon es un método efectivo. Muestra un coste computacional razonablemente bueno y su robustez en el proceso de localización es una característica



(a) (b)

Figura 4.23: Ratio de acierto usando la base de datos del laboratorio. (**a**) Usando las imágenes de test originales. (**b**) Añadiendo la presencia de diferentes niveles de ruido y oclusiones en las imágenes de test.

importante. Mientras que los métodos FS y SIFT tienen un comportamiento peor a medida que aumenta el ruido y las oclusiones, el método basado en la transformada de Radon tiene un comportamiento bueno y estable.

Para explorar el motivo de este comportamiento, representamos en la Figura 4.24 la distancia entre una imagen de prueba de muestra y cada imagen del mapa. La imagen de prueba está en la posición $(X,Y) = (3,3)$. En esta figura, el descriptor de Radon destaca por su capacidad de localizar el robot debido a que la distancia presenta un fuerte gradiente en el entorno de la posición correcta. Para verificar este rendimiento, lo que se quiere medir es si existe un mínimo claro en la función. Para ello se compara el valor medio de la función con el mínimo (Ecuación 4.6). Cuanto más claro sea ese mínimo, mayor será C.

$$C = \frac{mean(distance) - min(distance)}{max(distance) - min(distance)} \cdot 100(\%) \tag{4.6}$$

Usando esta ecuación con cada descriptor, SIFT, FS y RT y realizando 13 experimentos en cada caso, el valor promedio de C es: 60% usando SIFT, 61% usando FS y 79% usando RT.

4.7 Conclusiones

En este capítulo, se han presentado dos métodos diferentes para estimar la posición de un robot en un entorno en dos situaciones diferentes. En uno de ellos, el robot conoce las posiciones del mapa, y en el otro, solo conoce las imágenes del mapa. El primer

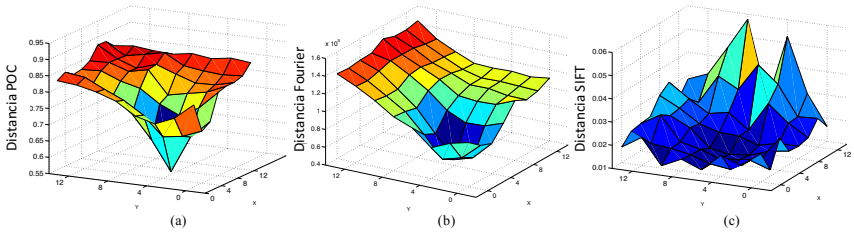


Figura 4.24: Distancia entre una imagen de test capturada en x = 3, y = 3 y todas las imágenes del mapa usando (**a**) el descriptor Radon y la distancia POC, (**b**) la firma de Fourier (FS) y (**c**) SIFT.

método calcula la posición más cercana del mapa con la imagen de prueba tomada por el robot, y el segundo método crea un mapa local con las imágenes más cercanas y la imagen de prueba tomada por el robot. Estos métodos usan imágenes omnidireccionales y descriptores de apariencia global.

También hemos desarrollado un método para construir descriptores de aspecto global a partir de imágenes omnidireccionales, basados en la transformación Radon.

Ambos métodos se prueban primero con la base de datos virtual para determinar los mejores parámetros, y luego se prueban con bases de datos reales. Uno de los métodos se ha analizado con una base de datos con cambios en las condiciones de iluminación y añadiendo objetos nuevos en el entorno. El otro método ha sido analizado con dos bases de datos diferentes capturadas en la Universidad Miguel Hernández.

Además, se ha llevado a cabo un análisis comparativo entre el descriptor propuesto y otros dos descriptores clásicos. Este análisis ha demostrado la solidez de nuestro método para resolver el problema de localización cuando las imágenes de prueba presentan ruido u oclusiones. Estos son fenómenos comunes en aplicaciones reales, y se ha demostrado que los métodos basados en la transformada de Radon es capaz de hacerles frente.

Los resultados presentados en este documento muestran la efectividad de descriptores de apariencia global de imágenes omnidireccionales para localizar el robot y crear un mapa.

Comparado con trabajos anteriores, las contribuciones de este capítulo son:

1. Se proponen descriptores de la apariencia global de un conjunto de imágenes: desarrollamos un método basado en la transformada de Radon [57] y el descriptor *gist* [51] para crear un modelo visual del entorno. No hemos encontrado ningún trabajo previo que utilice la transformada de Radon como un descriptor de apariencia global para resolver los problemas de localización y creación de mapas en robótica.

2. Se proponen métodos para resolver los problemas de creación de mapas y localización con estos descriptores: Esta contribución es doble:

a) Se resuelve el problema de localización con precisión, con respecto a un modelo visual (mapa) previamente creado: en este modelo, se conocen las posiciones donde se capturaron las imágenes.

b) Se resuelve el problema de localización con respecto a un modelo visual donde no se conocen las posiciones de captura de las imágenes: hemos implementado un algoritmo de creación de mapas locales y, posteriormente, resolvemos el problema de localización.

UNIVERSITAS
Miguel
Hernández

5 Cálculo de altura relativa utilizando apariencia global

5.1 Introducción

Recientemente, se han llevado a cabo algunos trabajos en el campo de la estimación de altura utilizando sensores de visión para resolver los problemas de creación de mapas y localización cuando el robot móvil puede cambiar su altitud durante la operación, como en el caso de los UAV (Unmanned Aerial Vehicles). Kim et al. [32] presentan un sistema de visión montado en dos vehículos aéreos no tripulados diferentes para ayudar a la planificación de la ruta de un vehículo terrestre, estimando su posición relativa y altitud. Usan una sola cámara montada en cada UAV para capturar las escenas. Otros autores usan una combinación de sensores para llevar a cabo la tarea de estimación de altitud, como Angelino et al. [5], que combinan la información visual de una cámara monocular y la información del GPS para estimar la posición y la altitud de un UAV. Estas obras utilizan una combinación de varios sensores, o el uso de dos o más sensores visuales, para llevar a cabo la estimación de altitud. En comparación con estos trabajos, el marco que proponemos utiliza solo la información capturada por un sensor de visión catadióptrico y los métodos de apariencia global para describir las escenas y estimar la altitud relativa del robot.

Los descriptores holísticos recopilan información sobre toda la escena. Comparando con métodos locales, no extraen ninguna información sobre objetos específicos o *landmarks*. Esta característica puede ser una ventaja porque los descriptores de apariencia global tienden a ser más compactos y se requiere menos tiempo de cálculo para crearlos y compararlos. Además, son una buena alternativa ya que los descriptores de apariencia global representan el entorno a través de características de alto nivel que se pueden interpretar y manejar fácilmente. Además, este tipo de descriptores tienden a ser

más robustos frente al ruido y las oclusiones parciales en las imágenes, en comparación con los descriptores basados en la extracción de características locales, como se muestra en capítulos anteriores, donde han sido utilizados en tareas de localización y creación de mapas. Además existen muchos más trabajos que han demostrado la validez de estas técnicas en la creación de mapas y localización de robots cuando el movimiento del robot está restringido al plano del suelo. Por ejemplo, en [7] se llevan a cabo diferentes tareas de localización y creación de mapas 2D usando descriptores de apariencia global y se comparan con algunos descriptores basados en la extracción de *landmarks* para comparar la efectividad y el coste computacional. Ranganathan et al. [58] presentan un método de creación de mapas topológico probabilístico que utiliza información de escenas panorámicas capturadas por un anillo de cámaras montadas en el robot, y se describen utilizando la firma de Fourier. También Menegatti et al. [43] muestran un método de localización de Monte Carlo con imágenes omnidireccionales en grandes entornos interiores utilizando la firma de Fourier como descriptor holístico. Sin embargo, pocos estudios se han llevado a cabo sobre la estimación de altitud utilizando enfoques de apariencia global. Teniendo en cuenta este punto de vista, en este capitulo se propone un método para estimar la altura relativa de un robot utilizando descriptores de apariencia global utilizados en capítulos anteriores para tareas de localización en 2D.

En anteriores capítulos se presentaron diferentes métodos con objeto de estimar la posición de un robot móvil a partir de la información suministrada únicamente por un sistema de visión, teniendo en cuenta tan sólo la apariencia global de las imágenes. En este capítulo, se propone un procedimiento que, a partir únicamente de dicha información y con procedimientos análogos, extrae información de la altura respecto a la cual se encuentra el robot. El algoritmo que se propone en este capítulo estima la altitud relativa del robot con respecto a la altitud que tenía cuando se creó el modelo, utilizando solo la información visual capturada por el robot desde su posición actual y la información visual almacenada en el mapa. Hay muchos robots móviles que cambian su altitud durante su funcionamiento, como los UAV. Muchos trabajos anteriores proponen diferentes soluciones al problema de localización utilizando UAV, como [44], donde estas plataformas se utilizan en tareas de navegación en entornos al aire libre utilizando imágenes omnidireccionales y otros sensores diferentes, como los giróscopos. Este trabajo se basa principalmente en la detección del horizonte para calcular la altitud y la rotación relativa del robot. Ashutosh et al. [46] muestran una combinación de cámaras omnidireccionales y de perspectiva para estimar la altitud del UAV extrayendo algunas características de las escenas.

En este capítulo, las imágenes se describen directamente (imágenes omnidireccionales), sin realizar ninguna proyección adicional, lo que supone un ahorro en el coste computacional. Con este objetivo, hacemos uso de la transformada de Radon [57], que describe la imagen en función de sus proyecciones integrales de línea a lo largo de algunos conjuntos de líneas paralelas. Como en el capítulo previo en el que se hacía uso de la transformada de Radon, proponemos seguir haciendo uso de la misma con objeto de poder obtener la altura relativa utilizando la misma información almacenada en el mapa, gracias a sus propiedades que fueron expuestas en el Capítulo 2.1.2. En líneas generales, el método consiste en comparar las transformadas de Radon de dos imágenes omnidireccionales capturadas desde diferentes altitudes. Esta comparación necesita un paso previo,

que consiste en calcular la diferencia entre las dos orientaciones que tenía el robot cuando capturó las imágenes omnidireccionales. Este paso se lleva a cabo usando POC (Phase Only Correlation) detallado en el Capítulo 2.2.2.

El método propuesto ha sido validado utilizando diferentes conjuntos de imágenes. En primer lugar, se ha probado usando nuestro propio conjunto de imágenes sintéticas, capturadas utilizando un sensor de visión catadióptrico virtual en dos salas virtuales diferentes. Este paso se llevó a cabo con el objetivo de realizar pruebas preliminares para mejorar el algoritmo antes de probarlo con imágenes reales. En segundo lugar, se ha probado utilizando algunos conjuntos de imágenes reales capturadas tanto en interiores como en exteriores, desde diferentes posiciones.

En resumen, la contribución de este capítulo es el uso de un único sistema de visión catadióptrico para estimar la altitud del robot utilizando un enfoque de apariencia global directamente sobre la imagen omnidireccional adquirida por el sistema de visión. El robot opera en un entorno y utiliza un sistema de visión omnidireccional como única fuente de información. También consideramos que el mapa del entorno se construyó a partir de un conjunto de imágenes capturadas mientras el robot tenía un movimiento plano y usaba técnicas de apariencia global. Teniendo en cuenta este hecho, proponemos dar un paso más en esta línea. El objetivo es intentar estimar, además de la posición en el plano del suelo donde se encuentra el robot (Capítulo 4), la altitud relativa donde se encuentra sin incorporar información adicional al mapa del entorno disponible y utilizando solo la información capturada por el sistema de visión omnidireccional.

En este capítulo se presenta en primer lugar el descriptor holístico utilizado para describir las imágenes omnidireccionales. A continuación se describe el algoritmo de estimación de altura relativa propuesto, las bases de datos empleadas para comprobar su funcionamiento y finalmente los resultados obtenidos y las conclusiones del capítulo.

5.2 Descriptor holístico utilizado

Esta sección presenta el método de descripción basado en la apariencia global que hemos implementado para describir las imágenes omnidireccionales. En trabajos anteriores se realizaron comparaciones entre diferentes métodos de descripción [55].

Este descriptor viene explicado en el Punto 2.1.2, donde se detallan las ecuaciones que lo definen y sus propiedades.

Como en el capítulo precedente, cuyo objetivo era localizar el robot en el plano del suelo, en esta propuesta se utiliza la Transformada Radon como descriptor para estimar la diferencia de altura, porque presenta algunas propiedades interesantes. Una de ellas es la propiedad de escalado que es la base de nuestro algoritmo de estimación de altura relativa: un escalado de la imagen $im(x,y)$ por un factor $1/b$ en las coordenadas x e y se traduce en un escalado de la transformada de Radon: la coordenada d se escala por un factor $1/b$ y la amplitud por un factor $|b|$:

$$\mathcal{R}\left\{im\left(\frac{x}{b},\frac{y}{b}\right)\right\} = |b|\lambda_f\left(\phi,\frac{d}{b}\right) \tag{5.1}$$

Otra ventaja es su robustez frente al ruido u oclusiones presentadas en las escenas, gracias al proceso de integración utilizado para construir el descriptor. Esta robustez ante el ruido y las oclusiones se demuestran en el capítulo anterior (Capítulo 4) al comparar la transformada de Radon con otros descriptores basados en las características locales (SIFT) [40] y en la apariencia global (Fourier Signature (FS)) [42].

5.3 Estimación de altura

En esta sección se describe y presenta el método propuesto para la estimación de la altitud, basado en la descripción de apariencia global mediante la transformada de Radon de la imagen omnidireccional adquirida. Además, se ha desarrollado e implementado un método basado en características locales. Gracias a ello, se puede llevar a cabo una evaluación comparativa para estudiar el rendimiento del método de apariencia global frente a un enfoque basado en la extracción y descripción más clásica de los *landmarks*.

Tanto el método propuesto como los métodos desarrollados e implementados con fines comparativos, proporcionan información sobre la dirección y el módulo del desplazamiento vertical del robot utilizando solo imágenes omnidireccionales capturadas por una cámara montada en el robot. Uno de los requisitos que se debe cumplir para el correcto funcionamiento del método propuesto es que la inclinación del robot con respecto al eje z_w del sistema de referencia (Figura 5.1) no cambie entre imágenes consecutivas del mapa. La Figura 5.1 muestra el sistema de referencia del robot y un cambio en sus ejes cuando se mueve hacia arriba desde p_0 hasta p_1. El método compara las imágenes capturadas de p_0 y p_1 y, como resultado, se obtiene una estimación topológica de la distancia entre estas posiciones (altitud relativa).

5.3.1 Estimación de altura utilizando un enfoque de apariencia global

A continuación se va a presentar el método de estimación de altura basado en la apariencia global. Primero, se muestran los fundamentos del método. En segundo lugar, se explica el método para comparar descriptores. Y finalmente, se presenta el funcionamiento del algoritmo de estimación de altura basado en la apariencia global.

5.3.1.1 Compresión-expansión de la transformada de Radon

La clave del método reside en las diferencias entre la transformada de Radon de dos escenas capturadas desde diferentes alturas cuando el robot se mueve verticalmente. Si el desplazamiento vertical es hacia arriba, los objetos en la escena omnidireccional tienden a aparecer más cerca del centro de la imagen. Esto hace que la información de estos objetos en las columnas de la transformada de Radon aparezca más cerca de la fila central. Y viceversa, si el desplazamiento es hacia abajo, la información en las columnas tiende a

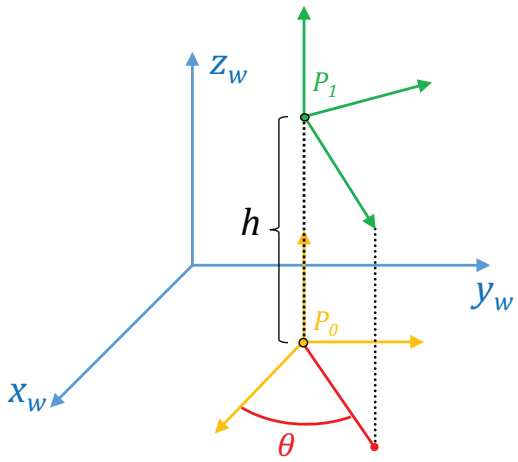


Figura 5.1: Sistema de referencia del robot. Se muestra un esquema de un cambio en la altitud del robot.

aparecer más lejos en la fila central. Este efecto está relacionado con la propiedad de escalado de la transformada de Radon.

Esta propiedad produce un cambio característico en la transformada de Radon. Cuando el robot se mueve hacia arriba, la información en las columnas de la transformada de Radon tiende a moverse hacia la fila central (efecto de compresión) y cuando el robot se mueve hacia abajo, la información en las columnas tiende a moverse hacia afuera de la fila central (efecto de expansión). El método propuesto hace uso de esta propiedad para estimar la altitud relativa del robot.

La Figura 5.2 muestra un ejemplo de este efecto. Se muestran dos imágenes omnidireccionales capturadas desde diferentes alturas ($1.25m$ y $2m$ respectivamente, con un movimiento puramente vertical) y sus correspondientes transformadas de Radon. En esta figura es posible observar que en la segunda imagen omnidireccional los objetos se han movido hacia el centro de la imagen omnidireccional en comparación con la primera imagen. Ambas transformadas de Radon contienen la misma información, pero la segunda presenta un efecto de "compresión" con respecto a la fila central.

Por lo tanto, es necesario diseñar un procedimiento que permita cuantificar estas compresiones/expansiones en la transformada de Radon y estudiar la correlación entre estos valores y las diferencias de altitud entre los puntos de captura de ambas imágenes omnidireccionales.

Se considera que el robot solo se mueve a lo largo del eje z_w y el objetivo es estimar h (Figura 5.1) . Sin embargo, antes de esto, es necesario detectar si el robot ha cambiado su orientación con respecto al eje z_w, porque introduciría un desplazamiento en las columnas de la transformada de Radon. Este es un paso fundamental para comparar dos transformadas de Radon diferentes. La estimación de la altura que se obtiene con

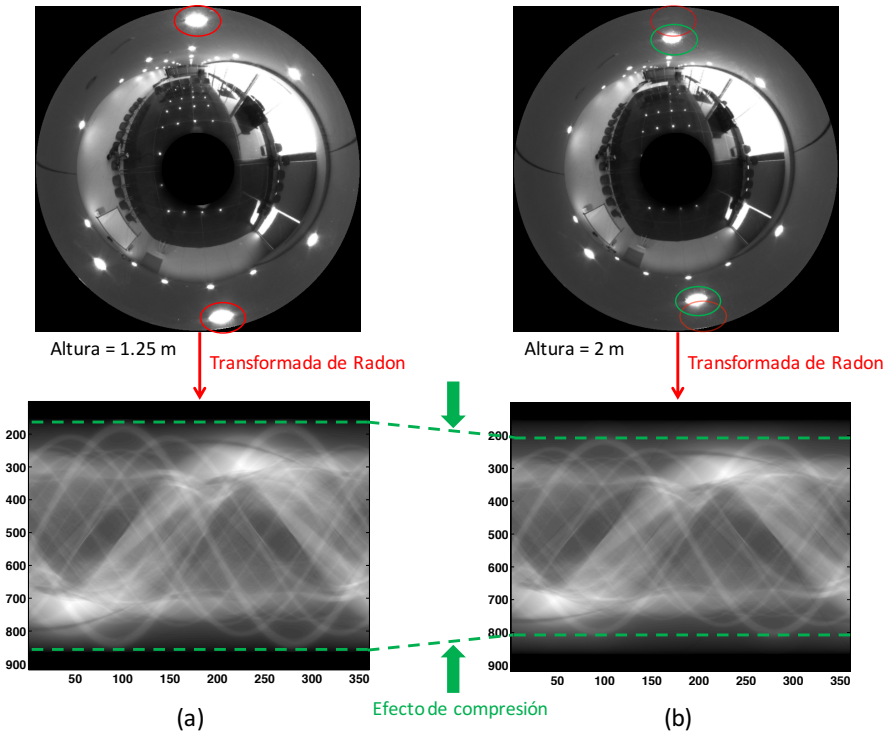


Figura 5.2: Ejemplo del efecto de compresión en la transformada de Radon. La figura muestra dos imágenes omnidireccionales y su transformada de Radon capturadas a una altura igual a: (a) h=1,25 m y (b) h=2 m. Comparando ambas transformadas de Radon, la segunda presenta un efecto de compresión con respecto a la primera.

el método propuesto una altura relativa, por lo que es una altura proporcional a la altura métrica real a falta de un factor de escala.

Para este propósito se utilizará el método basado en POC explicado en el Punto 2.2.2. Puesto que POC puede comparar dos imágenes de forma independiente a la orientación y también puede estimar este cambio de orientación.

5.3.1.2 Algoritmo de estimación de altura

El método de estimación de altura se basa en los conceptos descritos anteriormente. Es capaz de calcular los cambios en la orientación con respecto al eje z_w gracias al uso de POC para calcular esta rotación utilizando la Ecuación 4.1. La Figura 5.1 muestra el sistema de referencia global (x_w, y_w, z_w) y los sistemas de referencia del robot (x_r, y_r, z_r) cuando el robot se encuentra en los puntos P_0 y en P_1 con orientación relativa θ con respecto a z_w. El método se describe en los párrafos siguientes y se muestra un diagrama de flujo simplificado en la Figura 5.3.

El robot toma una imagen omnidireccional (imagen de referencia) desde su primera posición (P_0) y calcula su transformada de Radon. Luego, el robot se mueve hacia arriba o hacia abajo, captura una nueva imagen omnidireccional (imagen de test) y calcula su transformada de Radon. Después de esto, calcula el cambio de orientación θ entre ambas transformadas de Radon, usando POC, y lleva a cabo la corrección de desplazamiento angular de la segunda transformada de Radon, haciendo un desplazamiento en columnas igual a Δ_x (Ecuación 2.14). En la Figura 4.3 se puede observar un ejemplo de una imagen omnidireccional con diferentes orientaciones y el desplazamiento Δ_x en la transformada de Radon.

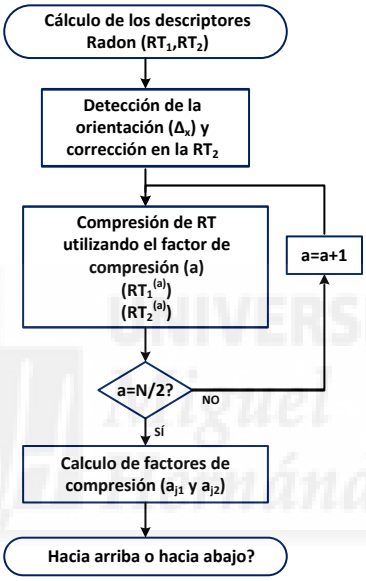


Figura 5.3: Diagrama de flujo simplificado del método de estimación de altura.

El siguiente paso consiste en estimar la diferencia de altitud entre ambas imágenes. Como un cambio de altitud produce un efecto de compresión en la transformada de Radon con respecto a su fila central, nuestro procedimiento consiste en aplicar un factor de escala a a cada columna de la transformada de Radon de la imagen de test, obteniendo $RT_2^{(a)}$ (el superíndice $^{(a)}$ indica que la transformada de Radon ha sido comprimida por un factor a utilizando la Ecuación (5.5)) y comparando el resultado con la transformada de Radon de la imagen de referencia (RT_1). Esta comparación se lleva a cabo usando la Ecuación 5.7 y el resultado es la distancia entre cada par de columnas. Esta ecuación calcula una distancia entre RT_1 y $RT_2^{(a)}$ considerando un factor de normalización por columna (están normalizados con respecto a su valor máximo).

La compresión se lleva a cabo interpolando los valores de las columnas de transformada de Radon teniendo en cuenta el factor de escala a (expresado como la mitad de la diferencia entre el número de píxeles de las columnas de ambas transformadas de Radon), el cual viene definido en la siguiente ecuación:

$$a = \frac{N - N^{(a)}}{2}. \tag{5.2}$$

donde $N^{(a)}$ es el número de píxeles de las columnas de $RT_2^{(a)}$.

$$RT_1 = \mathcal{R}\left\{im_{referencia}(x,y)\right\} \tag{5.3}$$
$$RT_2 = \mathcal{R}\{im_{test}(x,y)\}) \tag{5.4}$$

$$RT_k^{(a)}(x,y) = RT_k(x,\lfloor A \rfloor) + (y - \lfloor A \rfloor)\frac{RT_k(x,\lceil A \rceil) - RT_k(x,\lfloor A \rfloor)}{\lceil A \rceil - \lfloor A \rfloor} \tag{5.5}$$

donde $y = (1,2,...N-2a)$, $x = (1,2,...M)$, N es el tamaño de las columnas de la transformada de Radon original y A se calcula usando la siguiente ecuación:

$$A = \frac{y \cdot N}{N - 2a} \tag{5.6}$$

Hay que tener en cuenta que $\lfloor A \rfloor$ es el entero más grande menor o igual a A y $\lceil A \rceil$ es el entero más pequeño mayor que A.

Este paso se repite varias veces, considerando diferentes valores para el factor de compresión $a = \{a_1, a_2, ..., a_c\}$, hasta que la compresión no tenga sentido porque la transformada de Radon no tiene información relevante. En este momento, el robot tiene un vector de valores de distancia $\overrightarrow{Vd} = \{Vd_1, Vd_2, ..., Vd_c\}$ calculados usando la Ecuación 5.7.

$$dist(RT_1, RT_2^{(a)}) = \sum_{i=1}^{M}\sum_{j=1}^{N-2a}\left(\frac{\frac{RT_1(i,j+a)}{M_1(j+a)} - \frac{RT_2^{(a)}(i,j)}{M_2(j)}}{M \cdot (N-2a)}\right) \tag{5.7}$$

$$M_1(j+a) = \max(RT_{1a}(j+a)) \tag{5.8}$$
$$M_2(j) = \max(RT_{2a}(j)) \tag{5.9}$$

M es el número de filas de las transformadas de Radon y N es el número de columnas de RT_1. $M_1(j)$ es el valor máximo de la columna j de RT_1, y $M_2(j)$ es el valor máximo de la columna j de $RT_2^{(a)}$.

Cada elemento del vector Vd_i ha sido calculado considerando cada valor del factor de compresión a_i. El factor de compresión a_j que produce el mínimo del vector de distancias $\overrightarrow{Vd}$ (Ecuación 5.10), es una magnitud proporcional a la altura relativa.

$$j = \arg\min\{\overrightarrow{Vd}\} \tag{5.10}$$

En este punto, es necesario distinguir si la traslación del robot ha sido hacia arriba o hacia abajo. Si la traslación es hacia arriba, la transformada de Radon de la imagen de test sufre un efecto de compresión, pero si la traslación es hacia abajo, entonces la transformada de Radon de la imagen de referencia es la que sufre un efecto de compresión.

Para tener en cuenta este problema, primero, el robot comprime gradualmente la primera transformada de Radon y lleva a cabo el método descrito en los párrafos anteriores, para obtener el factor a_j, pero en este caso el robot también tiene que guardar la magnitud mínima d_{min} en el vector de distancias $\overrightarrow{Vd}$. Este caso sería el correcto si el robot se hubiera movido hacia arriba. En segundo lugar, el robot repite el proceso comprimiendo RT_1 en lugar de RT_2, obteniendo $RT_1^{(a)}$. Este caso sería el correcto si el robot se hubiera movido hacia abajo. Finalmente, el robot tiene dos factores: a_{j1} del primer caso y a_{j2} del segundo, y tiene dos distancias d_{min}: d_{min_1} (del vector de distancias del caso 1 ($\overrightarrow{Vd}_1$), Ecuación 5.11) y d_{min_2} (del vector de distancias del caso 2 ($\overrightarrow{Vd}_2$), Ecuación 5.12). El mínimo entre d_{min_1} y d_{min_2} determina cuál es el caso correcto, Ecuación 5.13. Al final del proceso, el robot tiene una magnitud a_{jk} proporcional al desplazamiento vertical y, dependiendo del caso correcto, el desplazamiento ha sido hacia arriba (caso 1) o hacia abajo (caso 2). La Figura 5.4 muestra un diagrama de flujo completo de este proceso.

$$d_{min_1} = \min\left(\overrightarrow{Vd}_1\right) \tag{5.11}$$

$$d_{min_2} = \min\left(\overrightarrow{Vd}_2\right) \tag{5.12}$$

$$k = \arg\min\{d_{min_1}, d_{min_2}\} \tag{5.13}$$

5.3.2 Estimación de altura utilizando un enfoque basado en la extracción de características locales

En este punto, se propone un método alternativo basado en características locales, con el objetivo de realizar un análisis comparativo para constatar las propiedades del método propuesto en el apartado anterior basado en un enfoque de apariencia global directa sobre la imagen omnidireccional.

Para ello se diseña un procedimiento basado en la extracción, descripción y correspondencia de características SURF (Speeded-Up Robust Features) [6], ya que es un descriptor clásico utilizado en tareas de localización de robots.

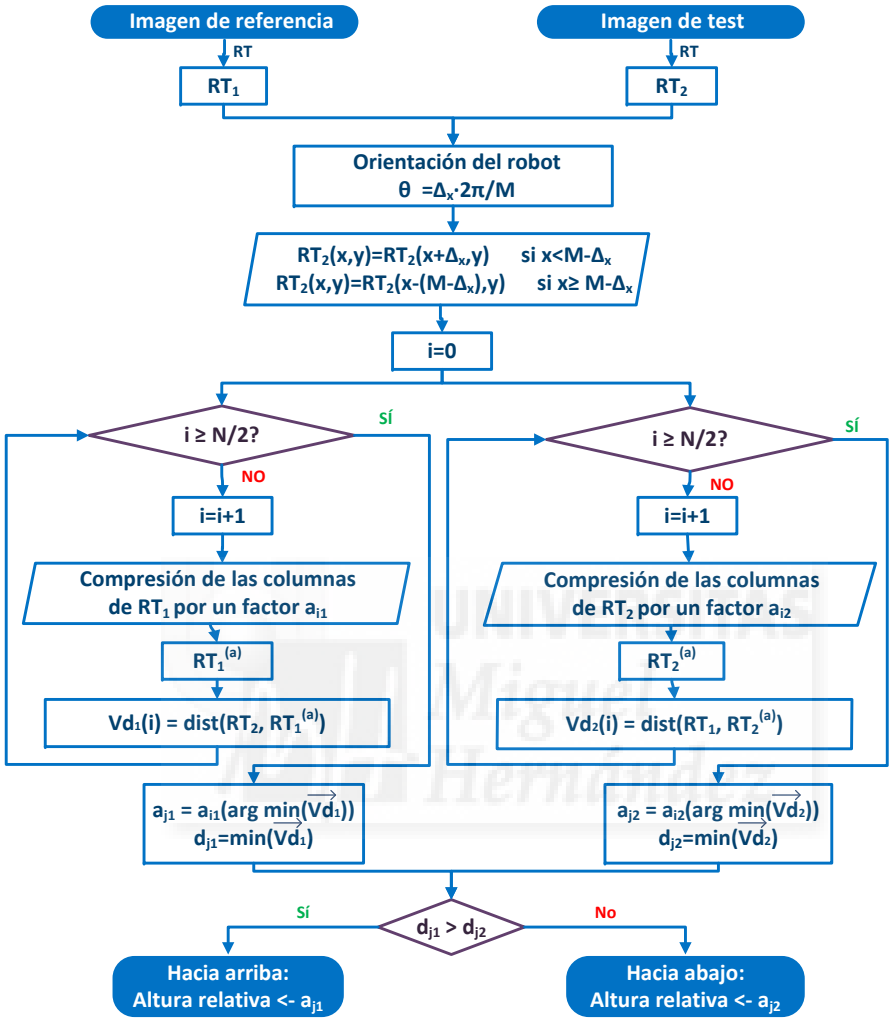
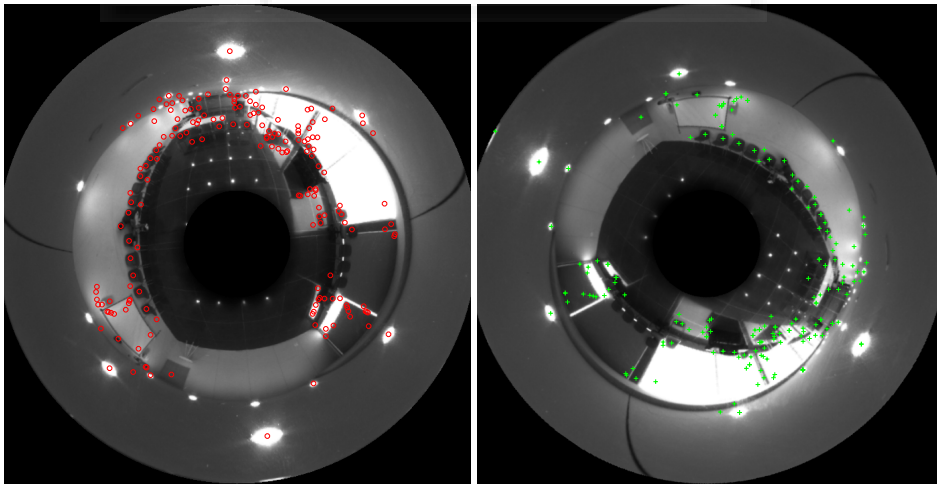


Figura 5.4: Diagrama de flujo del método de estimación de altura relativa.

El robot toma una imagen (imagen de referencia) desde su posición actual. Luego se mueve hacia arriba o hacia abajo y toma otra imagen omnidireccional (imagen de test). Los *landmarks* de cada imagen se calculan usando SURF. En este punto, el robot tiene las dos imágenes con los *landmarks* en cada imagen, por lo que debe hacer la correspondencia entre los *landmarks* en la imagen de referencia y los *landmarks* en la imagen de test. En la Figura 5.5 se muestran las correspondencias entre la imagen de referencia y la imagen de test en un entorno interior. Los puntos rojos son los *landmarks* correspondidos en la imagen de referencia y los puntos verdes son los *landmarks* correspondidos en la imagen de test. En la Figura 5.6 se muestran estas dos imágenes superpuestas, la

imagen de referencia coloreada de color rojo y la imagen de test coloreada de color azul; las líneas amarillas representan las conexiones entre *landmarks* correspondidos. Ahora, el robot debe determinar la diferencia de rotación entre ambas imágenes (rotación en el eje z_w). La diferencia angular entre ambas direcciones se calcula en cada par de puntos correspondidos. A continuación, todas las diferencias calculadas se comparan usando RANSAC (Random Sample Consensus) para obtener el valor de diferencia que maximiza el número de correspondencias con la misma dirección relativa (Figura 5.7). Este valor es la diferencia de rotación entre ambas imágenes, y las correspondencias con un valor de orientación diferente se descartan. Luego, la segunda imagen se rota por un valor igual a esta diferencia angular, y la estimación de la altura relativa se lleva a cabo calculando el promedio de distancia entre cada *landmark* de la imagen de referencia y el *landmark* correspondiente de la imagen de test, tras haber descartado los *landmarks* de las correspondencias erróneas usando RANSAC. Estas distancias vienen representadas en la Figura 5.8 con líneas amarillas.

La Figura 5.8 muestra un ejemplo de dos imágenes omnidireccionales comparadas para calcular la distancia entre las características correspondientes para obtener la diferencia de altitud entre ellas. Los puntos rojos son *landmarks* válidos en la imagen de referencia y los puntos verdes son los *landmarks* válidos en la imagen de test. Todas las direcciones entre *landmarks* correspondientes apuntan hacia el centro de la imagen.



(a) (b)

Figura 5.5: Correspondencias entre la imagen de referencia (a) y la imagen de test (b) en un entorno interior.

Figura 5.6: Correspondencias entre la imagen de referencia y la imagen de test en un entorno interior. Ambas imágenes son las de la Figura 5.5 superpuestas, la imagen coloreada en rojo es la imagen de referencia y la de color azul es la imagen de test.

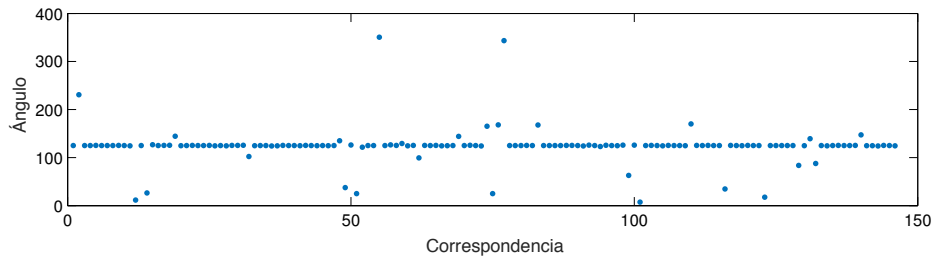


Figura 5.7: Diferencias angulares entre las direcciones desde las características en la imagen de referencia hacia las características de la imagen de test.

Figura 5.8: Paso final del método usado como referencia para comparar los resultados alcanzados por el método propuesto. La imagen coloreada en rojo es la imagen de referencia y la de color azul es la imagen de test.

5.4 Bases de datos utilizadas

Con el fin de verificar el funcionamiento del método propuesto para calcular la altura relativa, se han considerado diferentes conjuntos de imágenes omnidireccionales para realizar los experimentos. En primer lugar, se crearon dos entornos virtuales para tomar imágenes omnidireccionales fácilmente. Estas imágenes permiten probar la validez del método en condiciones ideales. Después de eso, se han capturado varios conjuntos de imágenes tanto en interiores como en exteriores bajo condiciones reales de trabajo para probar el algoritmo exhaustivamente con estas imágenes reales.

5.4.1 Base de datos de imágenes virtuales

Se han creado dos entornos virtuales que representan dos salas diferentes. En estos entornos, es posible crear imágenes omnidireccionales desde cualquier posición.

Las imágenes omnidireccionales utilizadas en los experimentos tienen $250x250$

píxeles y se han creado utilizando el espejo hiperbólico que se describe en la Figura 3.3. Los parámetros utilizados en la ecuación espejo son $a = 40$ mm y $b = 160$ mm.

Se han capturado varias imágenes en ambos entornos. En cada entorno, se han elegido varias posiciones en el plano del suelo y se han capturado un conjunto de imágenes verticalmente por encima de estas posiciones para llevar a cabo los experimentos. La altura mínima es de 100 mm, y la máxima de 2000 mm, con un paso de 100 mm. El total de imágenes en cada punto del suelo es de 20 imágenes, y como se han utilizado 14 puntos diferentes del suelo de los dos entornos, en total se han utilizado 280 imágenes. Los entornos empleados para la generación de estas imágenes son los presentados en el apartado 3.3.5. En la Figura 5.9 puede verse la vista cenital de ambos entornos y las posiciones desde las cuales se adquirieron el conjunto de imágenes variando la altura.

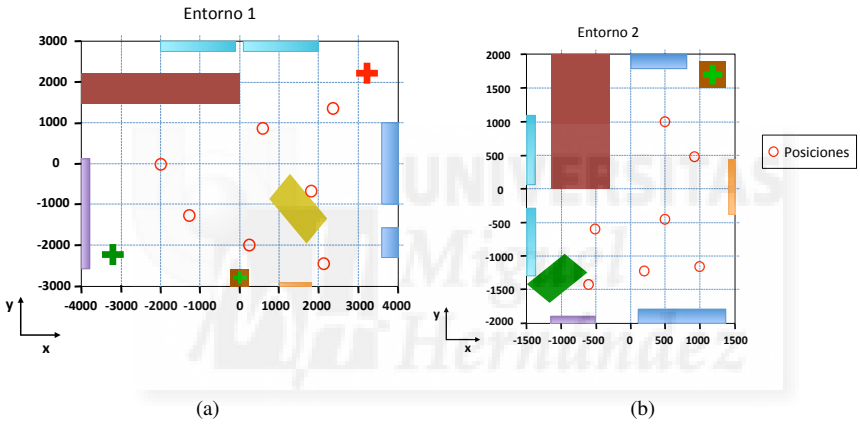


Figura 5.9: Posiciones desde las cuales se adquirieron las imágenes para realizar las pruebas.

5.4.2 Base de datos de imágenes reales

En el punto anterior, se ha presentado una base de datos de imágenes virtuales. Esta base de datos se usa para realizar una prueba preliminar del rendimiento del método propuesto y obtener la configuración adecuada para tomar la base de datos reales. Sin embargo, para probar la efectividad y la solidez del método, es necesario utilizar una base de datos real.

Esta base de datos real se compone de diferentes imágenes omnidireccionales tomadas en diferentes entornos interiores y exteriores. Estas imágenes tienen $717x717$ píxeles. Para crear esta base de datos, se han utilizado 10 entornos interiores diferentes y 10 entornos exteriores diferentes. En cada entorno, se han capturado varias imágenes desde diversas alturas. La altura mínima en entornos interiores es de 125 cm (h=1) y la altura máxima es de 230 cm (h=8), con un paso de 15 cm. En los entornos exteriores, la altura mínima es de 125 cm (denominada h=1 en los experimentos) y la altura máxima es

de 290 cm (h=12), con pasos de 15 cm. Esta base de datos es pública y puede encontrarse en [1].

Estas imágenes omnidireccionales se han capturado utilizando el sistema que se muestra en la Figura 3.1. Este sistema se compone de un espejo hiperbólico, una cámara y un trípode para cambiar la altura. Las coordenadas x_w, y_w del punto de captura en cada entorno son las mismas, solo cambia la coordenada z_w (altura).

Los entornos interiores se han elegido para cubrir diversas situaciones: áreas anchas y estrechas; entornos estructurados y no estructurados. Además, las coordenadas x_w, y_w se han elegido para cubrir una variedad de situaciones al aire libre: tanto cerca de edificios como en espacios abiertos.

(a) (b)

Figura 5.10: (a) Un ejemplo de imagen omnidireccional en cada uno de los 11 entornos de interior. (b) Imágenes capturadas en el entorno interior 3 (de la figura de la izquierda) desde la altura máxima (8) hasta la mínima (1).

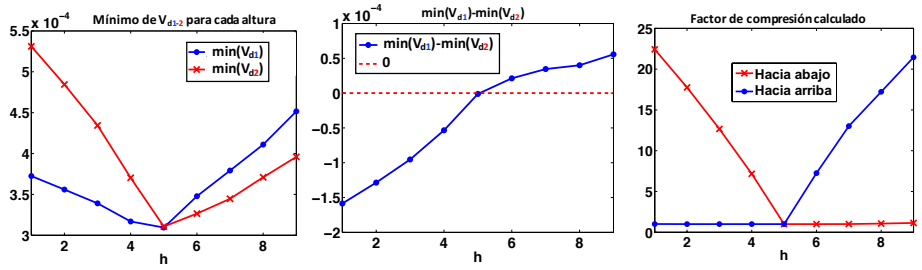
5.5 Resultados

En esta sección se muestran los resultados de los experimentos con nuestro método de estimación de altitud. Como se expone en la Sección 5.3, uno de los pasos necesarios

Figura 5.11: Una imagen omnidireccional de ejemplo por cada uno de los 10 entornos de exterior.

es diferenciar la dirección del movimiento (hacia arriba o hacia abajo). Para distinguir esto, es necesario calcular la diferencia entre ambos valores: $\min(\overrightarrow{Vd}_1) - \min(\overrightarrow{Vd}_2)$. Esta diferencia determina cuál es el mínimo absoluto y determina cuál es la dirección correcta.

La Figura 5.12 muestra un ejemplo de esto, utilizando el entorno virtual. En esta figura, tomamos la imagen capturada a 1,85 m (h=5) como referencia y todas las otras imágenes se consideran individualmente como imágenes de test y se comparan con la imagen de referencia. La Figura 5.12(a) muestra el mínimo de los vectores $\overrightarrow{Vd}_1$ (caso 1) y $\overrightarrow{Vd}_2$ (caso 2). La Figura 5.12(b) muestra la diferencia entre $\min(\overrightarrow{Vd}_1)$ y $\min(\overrightarrow{Vd}_2)$. Si la diferencia es negativa, el caso correcto es el caso 1 (hacia arriba) y si la diferencia es positiva, el caso correcto es el caso 2 (hacia abajo). En la Figura 5.12(c) se representa el factor a_j frente a la altura de la imagen de test en ambos casos (caso 1: hacia abajo y caso 2: hacia arriba). Este factor es proporcional a la altura relativa real entre cada imagen y la imagen de test. Como podemos observar en la Figura 5.12(b), el caso correcto para alturas inferiores a 1,85 m (h=5) es el caso 1 (hacia abajo) y para alturas superiores a 1,85 m es el caso 2 (hacia arriba), como se esperaba. Esto determina que en la Figura 5.12(c) la línea roja a la izquierda de h=5 indica la magnitud de la traslación hacia abajo de la imagen de referencia y la línea azul a la derecha de h=5 indica la magnitud de la traslación hacia arriba de la imagen de referencia. Podemos observar que las funciones presentan un comportamiento aproximadamente lineal.



(a) (b) (c)

Figura 5.12: (a) Mínimo de los vectores $\vec{V}d_1$(caso 1) y $\vec{V}d_2$(caso 2). (b) Diferencia entre $\min(\vec{V}d_1)$ y $\min(\vec{V}d_2)$. (c) a_j, factor el cual es proporcional a la altura relativa real entre cada imagen y la imagen de test. Este ejemplo se ha llevado a cabo utilizando un entorno interior de la base de datos de imágenes virtuales.

Uno de los parámetros relevantes que se debe configurar es el tamaño de la transformada de Radon. Un tamaño mayor en esta trasnformada podría dar mejores resultados por tener mayor resolución, pero a costa de ello el tiempo computacional aumentaría. Para ello se ha realizado un análisis mostrado en la Figura 5.13 donde se muestra el tiempo promedio que se emplea en cada estimación de altitud usando el algoritmo con diferentes tamaños de la transformada de Radon (MxN). Además, muestra una medida de incertidumbre calculada como el promedio entre la desviación típica de la altitud estimada usando 8 alturas diferentes en cada ambiente interior. Este promedio de la desviación típica viene dado como porcentaje de variación de la altura relativa calculada. Analizando los resultados, para hacer los experimentos posteriores, se ha elegido un tamaño de transformada de Radon de $MxN = 204.000$, ya que presenta un buen equilibrio entre el coste computacional y la incertidumbre resultante.

Para demostrar el rendimiento correcto del método presentado, se han realizado una serie de experimentos en ambos entornos virtuales. Se han seleccionado 14 posiciones al azar en el plano del suelo de estos entornos y se han capturado 20 imágenes por posición, cambiando solo la altitud con respecto a cada posición, con un espacio de altura de 10 cm entre imágenes consecutivas. En la Figura 5.14, se pueden observar los resultados globales de estos experimentos. El valor del estimador de altura topológica se representa frente a la altura métrica real de la imagen de test. La línea roja muestra la magnitud de la traslación hacia arriba y la línea azul muestra la magnitud de la traslación hacia abajo. Se puede observar que estos experimentos demuestran la linealidad del método para valores de altura relativa alrededor o por debajo de 1 metro.

Después de la validación del método utilizando entornos virtuales, es necesario probarlo utilizando la base de datos compuesta de imágenes omnidireccionales tomadas en entornos reales. Para hacer esto, se ha utilizado la base de datos descrita en el apartado 5.4.2. También se ha considerado la posible presencia de ruido y oclusiones en las imágenes de test, ya que son fenómenos habituales que un robot móvil tiene que

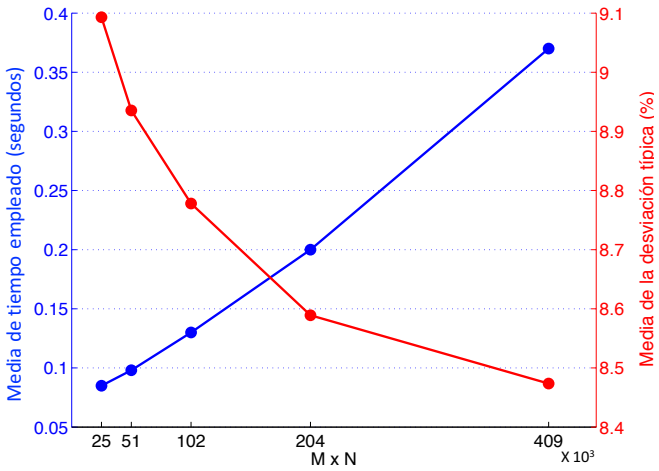


Figura 5.13: Media del tiempo de computo y media de la desviación típica utilizando diferentes tamaños de la transformada de Radon. Todos los entornos de interior han sido utilizados para llevar a cabo esta comparación, calculando 8 estimaciones de altura diferentes en cada uno de ellos.

enfrentar cuando se mueve de forma autónoma en un entorno de trabajo real. Para ello se ha considerado el ruido aleatorio con un valor máximo igual al 20% de la intensidad máxima de la imagen omnidireccional y las oclusiones que ocultan el 10% de la imagen omnidireccional.

Se han utilizado diferentes imágenes de referencia para demostrar el rendimiento correcto al estimar la altitud en ambas direcciones (hacia arriba y hacia abajo). Una de las imágenes de referencia se encuentra a 125 cm (h=1) y la otra a 185 cm (h=5).

La Figura 5.17 muestra los resultados de los experimentos utilizando la base de datos que contiene entornos interiores (Figura 5.10). Esta figura muestra el promedio y la desviación típica de todas las estimaciones de altura al agregar diferentes niveles de ruido (ruido aleatorio cuyo valor máximo depende del valor máximo de la intensidad de píxeles de la imagen de test, Figura 5.17 (a)), y con presencia de oclusiones diferentes (las oclusiones cubren un porcentaje de la imagen de test, Figura 5.17 (b)). En la Figura 5.15 se muestran los diferentes niveles de ruido empleados aplicados a una imagen omnidireccional y en la Figura 5.16 se muestran los diferentes niveles de oclusiones. Los factores de compresión a_j en cada experimento se han normalizado con respecto al máximo factor a_j en cada entorno. Por último, la Figura 5.18 muestra los mismos experimentos empleando la base de datos capturada en entornos al aire libre (Figura 5.11). Contiene más puntos de test que la base de datos interior porque no hay limitaciones de techo en entornos al aire libre. En estos experimentos en exteriores se observa menos robustez frente al ruido que puede ser debido a la mayor homogeneidad de las imágenes.

Los resultados muestran que el método propuesto es capaz de estimar la altitud topológica del robot utilizando un solo sensor de visión omnidireccional. Sin embargo, va

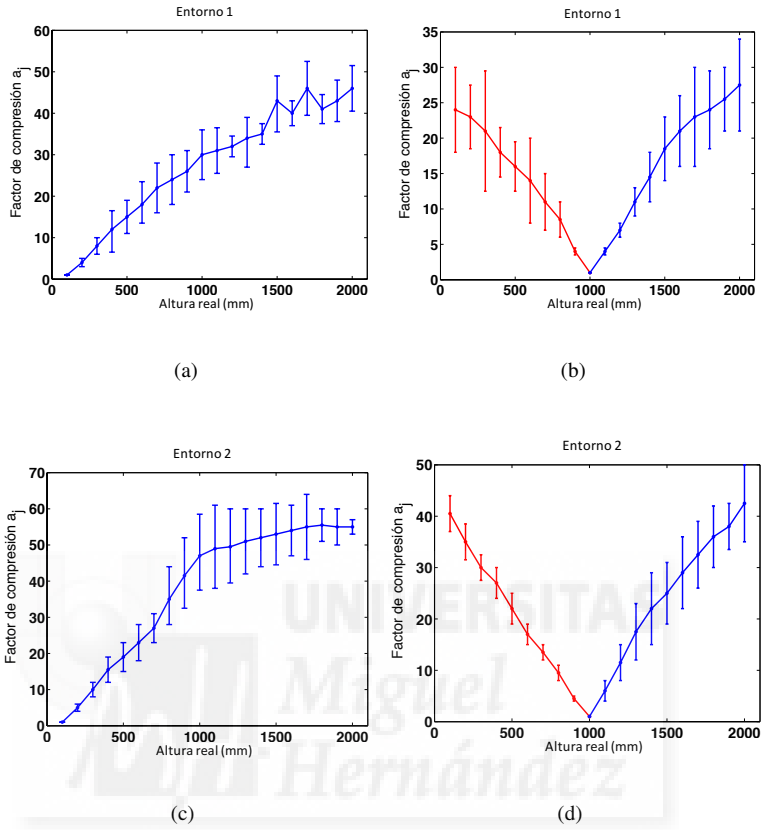


Figura 5.14: (a) Experimentos con el entorno virtual 1 usando como imagen de referencia la que se encuentra a una altura de 100 mm. (b) Experimentos con el entorno virtual 1 usando como imagen de referencia la que se encuentra a una altura de 1000 mm. (c) Experimentos con el entorno virtual 2 usando como imagen de referencia la que se encuentra a una altura de 100 mm. (d) Experimentos con el entorno virtual 2 usando como imagen de referencia la que se encuentra a una altura de 1000 mm.

más allá del concepto topológico de conectividad; el método proporciona una medida de altura que es proporcional a la altitud geométrica del robot (a excepción de un factor de escala). Además, dado que se utiliza la apariencia global de las imágenes y se considera un enfoque topológico, la estabilidad de los parámetros del sistema de visión no es crítica.

Por último, este método se ha comparado con el método alternativo descrito en la Sección 5.3.2. En primer lugar, en lo que respecta al tiempo de cálculo, el método basado en características locales toma, en promedio, 1.3 segundos. El método de apariencia global que se propone en este capítulo emplea 0.2 segundos en promedio, cuando la transformada de Radon tiene 204.000 componentes (Figura 5.13). En segundo lugar, se estudia la precisión del método en la estimación de la altura. La Figura 5.19 muestra los mismos experimentos que en la Figura 5.17 pero utilizando el método de características

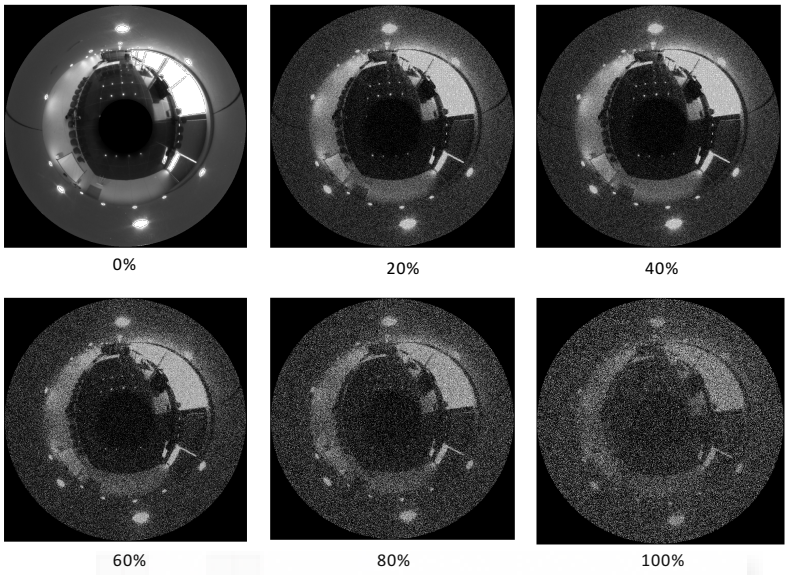


Figura 5.15: Diferentes niveles de ruido aplicados a una imagen omnidireccional

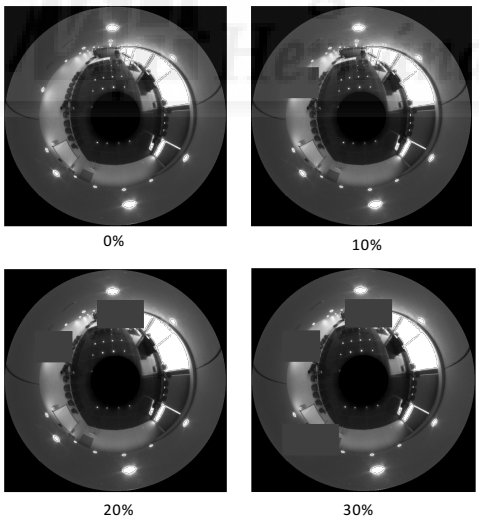


Figura 5.16: Diferentes niveles de oclusiones aplicados a una imagen omnidireccional.

locales; además difiere en el nivel máximo del ruido añadido que es el 30% del valor más alto de intensidad de todos los píxeles de la imagen de test y las oclusiones añadidas

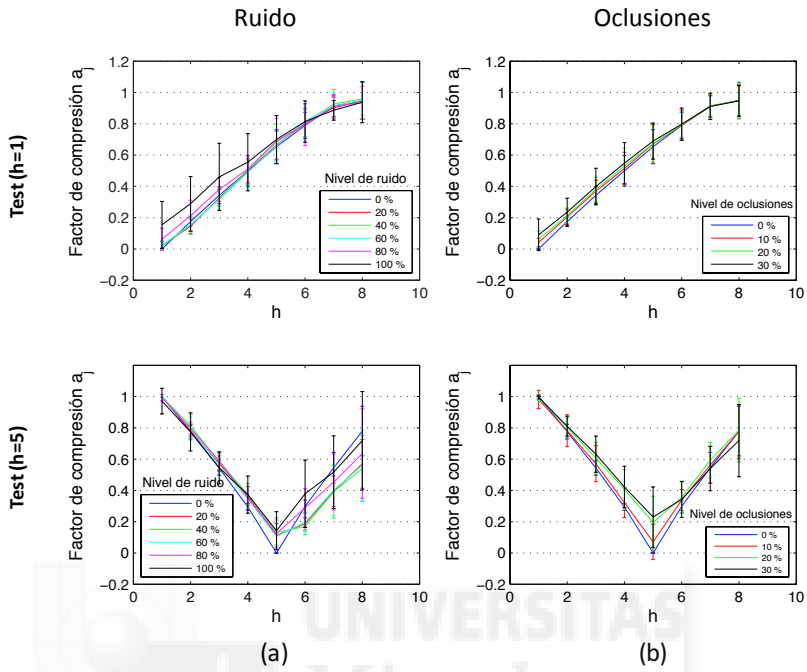


Figura 5.17: Estimación de altura utilizando la base de datos capturada en entornos interiores, usando la imagen de referencia en el suelo (h=1) y en una posición central (h=5). Media y desviación típica de todas las localizaciones considerando (a) ruido en las imágenes de test y (b) oclusiones añadidas a las imágenes de test.

cubren el 30% de la imagen de test. Como conclusión general, el método basado en la apariencia global presenta una evolución más lineal cuando el ruido y las oclusiones están presentes, y la desviación de los resultados tiende a ser menor. Cuando la imagen de test no presenta ruido ni oclusiones, el resultado de ambos métodos es bastante similar, en lo que respecta a la linealidad y la desviación. Sin embargo, el método de apariencia global presenta un coste computacional sustancialmente más bajo en todos los casos. La Figura 5.20 muestra los resultados de los mismos experimentos que en la Figura 5.19 pero utilizando los entornos exteriores (Figura 5.11).

5.6 Conclusiones

En este capítulo se ha presentado un método para estimar la altitud relativa de un robot móvil. Este método utiliza imágenes omnidireccionales y las transforma con la transformada de Radon para crear un descriptor holístico por imagen. Además, compara los descriptores y finalmente estima la altura relativa del robot, teniendo en cuenta los cambios que sufre la transformada de Radon de las escenas cuando el robot cambia su altura. Un aspecto notable es que el método puede detectar estos cambios de altura tanto en ambientes interiores como exteriores usando el mismo mapa que se usa en tareas de lo-

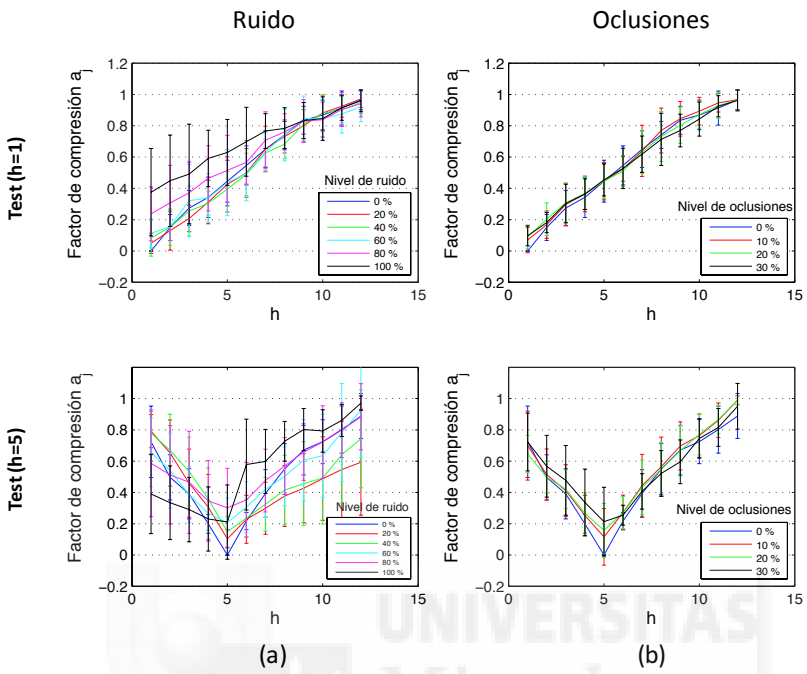


Figura 5.18: Estimación de altura utilizando la base de datos capturada en entornos exteriores, usando la imagen de referencia en el suelo (h=1) y en la mitad de la altura máxima (h=5). Media y desviación típica de todas las localizaciones considerando (a) ruido en las imágenes de test y (b) oclusiones añadidas a las imágenes de test.

calización. Además, este enfoque permite estimar la altura del robot incluso cuando tiene una rotación con respecto al plano del suelo porque la comparación POC permite estimar y corregir esta rotación.

Los experimentos incluidos en este documento utilizan conjuntos de imágenes creadas sintéticamente a partir de dos entornos virtuales diferentes. Además, se han llevado a cabo experimentos que utilizan bases de datos reales para probar la validez del método en entornos interiores y exteriores, incluso utilizando imágenes con ruido y oclusiones. Los resultados demuestran que el método es capaz de estimar la altitud relativa entre dos imágenes con robustez y linealidad incluso en presencia de ruido y oclusiones. Además, el método puede estimar la altitud relativa del robot en tiempo real.

El método se ha comparado con un método alternativo basado en el marco clásico de extracción, comparación y seguimiento de características locales. Los resultados han demostrado que el método de apariencia global que se propone supera al método de características locales y presenta un menor coste computacional.

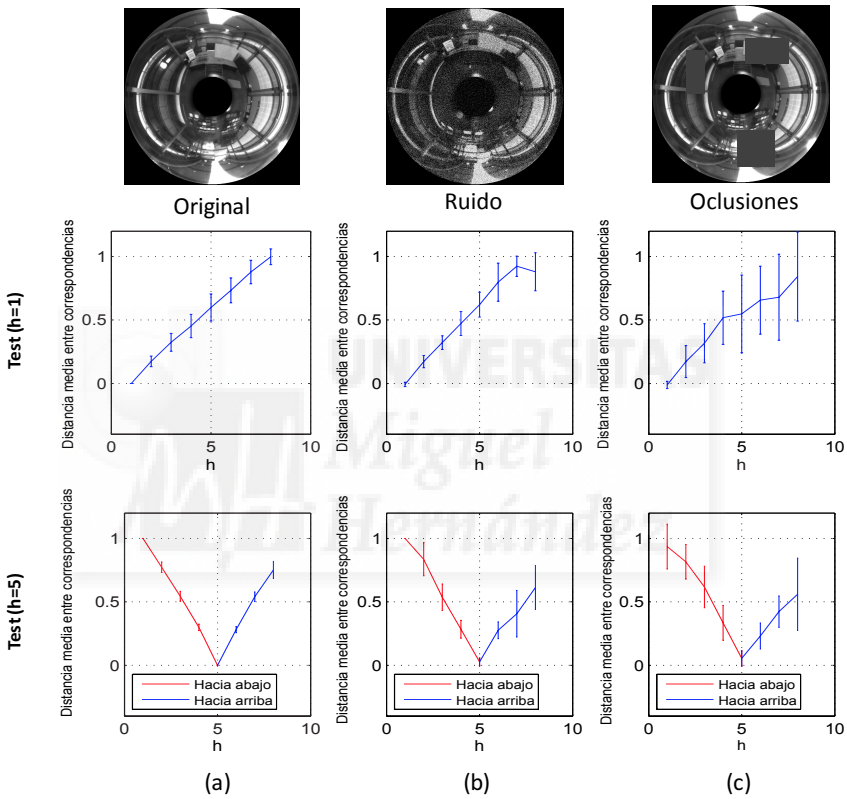


Figura 5.19: Estimación de altura usando el método basado en características locales y la base de datos capturada en entornos interiores, usando imágenes de referencia en la parte inferior (h = 1) y en una altura media (h = 5). Media y desviación típica de todas las localizaciones considerando (a) ruido en las imágenes de test y (b) oclusiones añadidas a las imágenes de test.

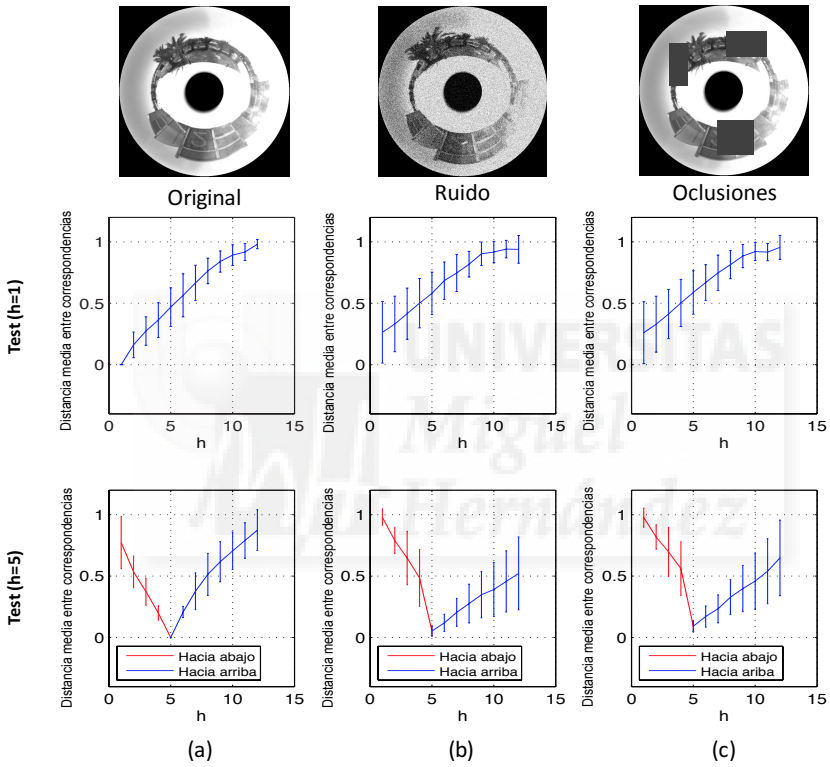


Figura 5.20: Estimación de altura usando el método basado en características locales y la base de datos capturada en entornos exteriores, usando imágenes de referencia en la parte inferior (h = 1) y en una altura media (h = 5). Media y desviación típica de todas las localizaciones considerando (a) ruido en las imágenes de test y (b) oclusiones añadidas a las imágenes de test.

6 Graph-SLAM utilizando descriptores holísticos

6.1 Introducción

El SLAM (Simultaneous Localization and Mapping) se ha considerado como uno de los procedimientos más importantes en la investigación de la robótica móvil durante los últimos años [45, 69]. Muchos de estos trabajos se centran en el uso de información visual para desarrollar algoritmos de SLAM, debido a la rica información proporcionada por los sensores visuales.

En el campo del SLAM, las imágenes omnidireccionales tienen multitud de ventajas gracias a su campo de visión, ya que una sola imagen contiene información de todos los alrededores del robot. Podemos encontrar muchos trabajos previos que usan imágenes omnidireccionales en tareas de creación de mapas y localización. Por ejemplo, Valiente et al. [65] presentan una comparación entre dos métodos visuales SLAM diferentes usando imágenes omnidireccionales y Garcia et al. [22] muestran una recopilación de métodos de creación de mapas y localización basados en visión por computador.

Habitualmente, el proceso de creación de mapas produce un error en cada posición del mapa debido al cálculo iterativo de nuevas poses (posición y orientación) del robot. Esto puede ser un gran problema en entornos extensos cuando el robot tiene que calcular muchas poses nuevas, ya que el error está aumentando en cada iteración. Esta incertidumbre puede reducirse mediante la detección de cierres de bucle y el uso de algoritmos de optimización para reubicar las poses. Este problema se estudia a fondo en este trabajo.

La contribución de este capítulo consiste en la creación de un método para llevar a cabo las tareas de SLAM utilizando sólo la información visual del entorno y los descriptores

de apariencia global al utilizar directamente la imagen omnidireccional adquirida por el sistema de visión. Cada escena omnidireccional adquirida por el robot se describe utilizando estos descriptores. El método consta de tres pasos diferentes: calcular la pose del robot (posición y orientación), detectar cierres de bucle (comparando descriptores de apariencia global) y optimizar el mapa (utilizando el algoritmo de optimización G2O). El algoritmo de optimización utilizado se denomina G2O y fue presentado por Kümerle et al. [37].

Los experimentos se han llevado a cabo con dos conjuntos diferentes de imágenes capturadas en dos entornos reales de trabajo. El primero se ha tomado siguiendo un camino rectangular en un entorno interior y el segundo ha sido capturado siguiendo un camino realista incluyendo varias habitaciones en un edificio.

En este capítulo se presentan en primer lugar las herramientas utilizadas en el algoritmo de localización. A continuación se presenta dicho algoritmo de graph-SLAM, detallando cada una de sus fases. Posteriormente se muestran los experimentos realizados para comprobar la validez del algoritmo. Y, finalmente, se presentan las conclusiones del capítulo.

6.2 Herramientas utilizadas

A lo largo de este capítulo, se utilizan dos métodos para describir la apariencia global de las escenas: la transformada de Radon y HOG (Histogram of Oriented Gradients), los cuales vienen descritos en el Capítulo 2.1.2 y 2.1.4. En el caso del descriptor HOG, se ha decidido calcular el descriptor HOG pasándole como entrada el descriptor Radon de las imágenes omnidireccionales. Los parámetros de configuración del descriptor que se han usado son $b = 8$ niveles en el histograma y $k_2 = 16$ como número de celdas horizontales.

Además, se utilizan los métodos descritos en el Punto 2.2 para comparar dichos descriptores. Por último, para recalcular las posiciones del mapa se utiliza un algoritmo de optimización llamado G2O, el cual se detalla a continuación.

6.2.1 Algoritmo de optimización: G2O

G2O es un algoritmo de optimización descrito en [37]. Este método fue creado para optimizar funciones de errores no lineales basadas en grafos.

En el campo del SLAM, el robot tiene que calcular su pose cuando toma cada nueva imagen con respecto a las poses anteriores almacenadas en el mapa. Esta operación tiene un error asociado que va incrementando con cada nuevo cálculo de pose, por lo que necesitamos corregir las poses almacenadas en el mapa para disminuir esta desviación. G2O puede volver a calcular cada pose del mapa usando nuevas restricciones. Una de estas restricciones se puede obtener cuando se producen cierres de bucle entre una pose del mapa existente y la nueva pose del robot. Entonces, G2O reubica cada nodo del mapa modificándolos gradualmente para cumplir con la restricción de cierre de bucle. Finalmente, el nuevo nodo se ubica en la misma posición que la pose equivalente almacenada en el mapa.

En general lo que realiza este algoritmo es una minimización de la siguiente función de error no lineal:

$$F(x) = \sum_{i,j \in \mathbb{C}} e(x_i, x_j, z_{ij})^T \Omega_{ij} e(x_i, x_j, z_{ij}) \tag{6.1}$$

donde $x = (x_1^T, ..., x_n^T)$ es un vector que representa las poses x_i. z_{ij} y Ω_{ij} son, respectivamente, la media y la matriz de información de las restricciones que relacionan x_i y x_j (en nuestro caso los cierres de bucle). $e(x_i, x_j, z_{ij})$ es un vector que contiene el error que cuantifica como los conjuntos x_i y x_j satisfacen las restricciones z_{ij}, este error es 0 cuando x_i y x_j satisfacen perfectamente las restricciones, es decir, las posiciones definidas en un cierre de bucle están posicionadas en el mismo lugar.

6.3 Algoritmo de graph-SLAM

En esta sección, presentamos el algoritmo de SLAM visual propuesto en este capítulo. El robot pasa por el entorno y captura imágenes de algunas posiciones. Cada vez que llega una nueva imagen, el robot incluye un nuevo nodo dentro del mapa, el cual está formado por nodos. A continuación, se resuelve el problema SLAM siguiendo estos tres pasos:

1. Primero, el robot calcula dos descriptores de la imagen: la transformada de Radon y el descriptor HOG y los almacena en el nodo. Entonces, el robot crea un nuevo nodo y lo ubica dentro del mapa calculando la posición y orientación del nuevo nodo con respecto al nodo previamente agregado y se crea un enlace entre ambos nodos. Este proceso de localización se realiza utilizando sólo información visual.
2. En segundo lugar, el robot comprueba la existencia de posibles cierres de bucle comparando la nueva escena con las escenas anteriores almacenadas en el mapa.
3. Finalmente, el mapa se optimiza utilizando el algoritmo G2O con los cierres de bucle detectados. Este proceso se repite en cada nueva ubicación.

6.3.1 Creación del mapa

Esta subsección presenta el método propuesto para calcular las coordenadas de las poses de los nuevos nodos. Estas poses se calculan obteniendo la distancia y el ángulo entre las escenas. Para cada nuevo nodo, el robot almacena el descriptor de Radon y el descriptor HOG de la nueva imagen omnidireccional para hacer posible su localización.

La Figura 6.1 muestra un esquema del proceso de creación de mapas. Consiste en el cálculo de las coordenadas (x_k, y_k) de cada nuevo nodo. Estas coordenadas se calculan a partir de la distancia y el ángulo respecto a la posición anterior.

La distancia entre ubicaciones se calcula usando la Ecuación 5.7. Es una distancia imagen y no es una distancia métrica, es decir, esta distancia no es una unidad de medida real, sin embargo en capítulos anteriores se ha comprobado que puede llegar a considerarse proporcional a la distancia real si se linealiza (Figura 4.5).

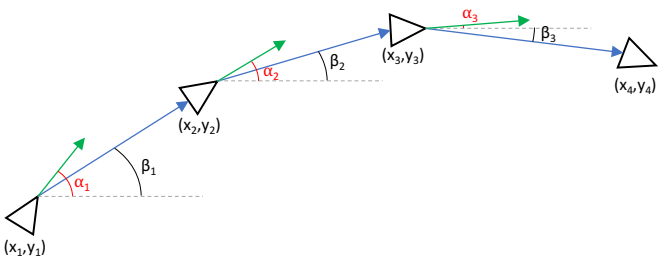


Figura 6.1: Esquema del proceso de creación de mapas.

La orientación del robot con respecto al nodo anterior α_j, se calcula usando la Ecuación 6.2.

$$\alpha = \frac{\Delta_x \cdot 2\pi}{N} \tag{6.2}$$

El cálculo del ángulo que forma el nodo actual j con el eje x (ángulo β_j que se puede observar en la Figura 6.1) se lleva a cabo siguiendo el siguiente procedimiento:

1. Primero, el robot calcula el cambio de orientación α_j, entre esta imagen y la imagen omnidireccional del nodo anterior, haciendo uso de la comparación POC de las transformadas de Radon de ambas imágenes. Teniendo en cuenta este cambio de orientación, la imagen anterior se gira para que ambas imágenes estén orientadas en la misma dirección.

2. En segundo lugar, ambas imágenes omnidireccionales se transforman en vistas panorámicas.

3. En tercer lugar, las características de SURF se extraen de ambas imágenes panorámicas y se calculan las correspondencias entre las características de ambas imágenes.

4. En cuarto lugar, se extrae la componente horizontal del desplazamiento entre las características correspondidas de ambas imágenes, es decir, la distancia en el eje x entre un *landmark* de una imagen y su correspondiente *landmark* de la otra imagen panorámica. Esta distancia se calcula superponiendo las imágenes panorámicas una encima de otra. Este 'desplazamiento horizontal' se considera positivo si apunta hacia la derecha de la imagen o negativo si apunta a la izquierda.

5. Finalmente, se define un conjunto de ventanas. Estas ventanas tienen M filas y $0.5N$ columnas, $W = \{W_1, W_2, ...\}$, cuya distancia horizontal es de un píxel. Se cuenta la cantidad de *landmarks* con desplazamiento negativo dentro de cada ventana y el número de *landmarks* con un desplazamiento positivo en el exterior de cada ventana, lo que da como resultado el vector $\vec{n} = [n_1, n_2, ...]$. Se extrae el máximo de este vector ($p_w = argmax(\vec{n})$). Luego, la posición p_w permite estimar el ángulo buscado utilizando la Ecuación 6.3.

$$\beta = \frac{p_w \cdot 2\pi}{N} \tag{6.3}$$

En la Figura 6.2 se muestra un diagrama de flujo de este proceso de cálculo de la dirección del desplazamiento del robot.

Cabe destacar que el cálculo de este ángulo no ha sido posible aplicando directamente apariencia global en la imagen omnidireccional pero el enfoque de uso de los *landmarks* utilizados es un enfoque global, ya que se mira el comportamiento de todo el conjunto en zonas de la imagen (ventanas W).

La Figura 6.3 presenta dos imágenes panorámicas diferentes superpuestas, que muestran las características SURF coincidentes entre ellas. Las líneas rojas son los límites de la ventana de 180 grados. En ese caso, la ventana está colocada en la posición en la cual se cumple la condición para determinar la dirección de movimiento del robot. En ella se ve que todos los puntos dentro de la ventana tienen un desplazamiento negativo a lo largo del eje x y los puntos fuera de esta ventana tienen un desplazamiento positivo a lo largo del mismo eje.

Las nuevas coordenadas de nodo (x_k, y_k) se calculan mediante estas ecuaciones:

$$x_k = dist_{POC}(im_{k-1}, im_k) \cdot \cos(\beta_k) \tag{6.4}$$

$$y_k = dist_{POC}(im_{k-1}, im_k) \cdot \text{sen}(\beta_k) \tag{6.5}$$

donde $dist_{POC}(im_{k-1}, im_k)$ es la distancia POC entre las transformadas de Radon de las dos imágenes consecutivas, calculada usando la Ecuación 5.7, y β_k es el ángulo de dirección del movimiento del nodo k calculado según la Ecuación 6.3.

6.3.2 Cierres de bucle

El siguiente paso del algoritmo consiste en detectar cierres de bucle. Para ello, se compara el descriptor HOG de la nueva imagen tomada por el robot con los descriptores HOG almacenados en el mapa. Para calcular la distancia entre los descriptores HOG se utiliza la similitud de coseno entre ellos para calcular la distancia:

$$dist(\overrightarrow{d_1}, \overrightarrow{d_2}) = 1 - \frac{\overrightarrow{d_1} \cdot \overrightarrow{d_2}^T}{\sqrt{(\overrightarrow{d_1} \cdot \overrightarrow{d_1}^T)(\overrightarrow{d_2} \cdot \overrightarrow{d_2}^T)}} \tag{6.6}$$

donde $\overrightarrow{d_1}$ y $\overrightarrow{d_2}$ son los descriptores HOG de dos imágenes diferentes.

Los cierres de bucle deben determinarse definiendo un umbral máximo de distancia, W (Ecuación 6.7). Este umbral se define como una constante en el inicio del proceso de SLAM. Si la distancia es inferior a este umbral, las dos poses comparadas se considerarán como la misma posición (x, y), pero la orientación del robot puede ser diferente.

$$if(dist(\overrightarrow{d_1}, \overrightarrow{d_2}) < W) \rightarrow cierre\ de\ bucle \tag{6.7}$$

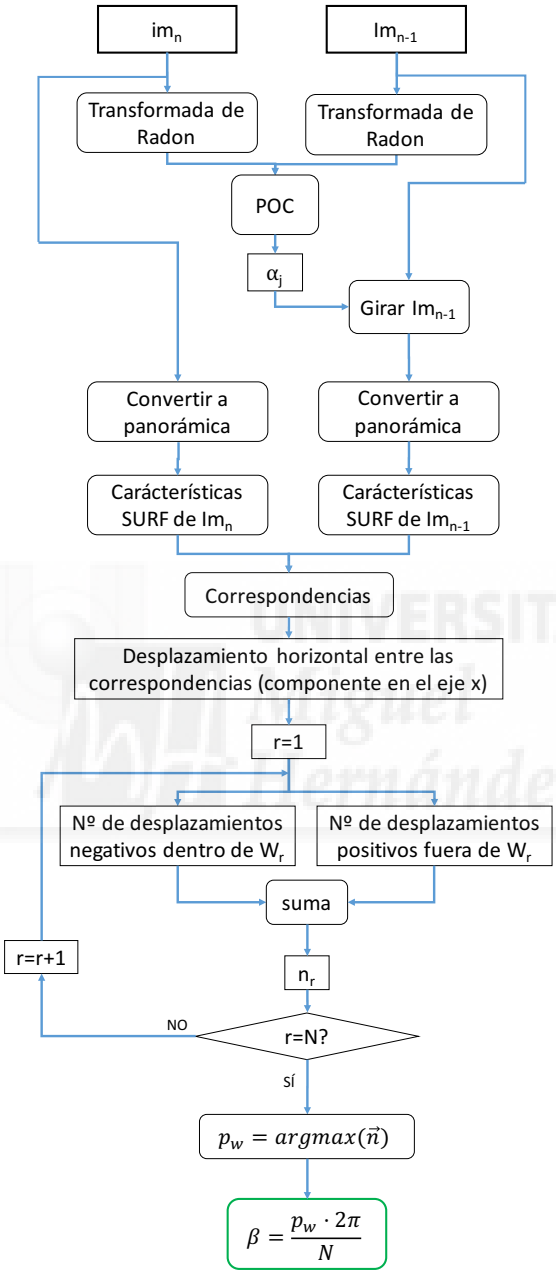


Figura 6.2: Diagrama de flujo del proceso del cálculo de la dirección de desplazamiento.

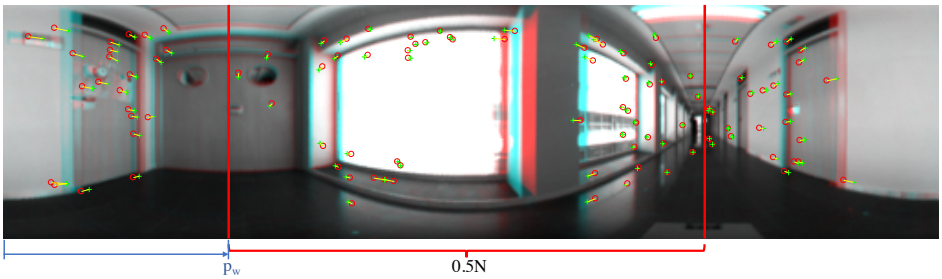


Figura 6.3: Correspondencias entre los descriptores SURF de dos imágenes panorámicas consecutivas de la base de datos.

6.3.3 Optimización del mapa

Teniendo en cuenta los cierres de bucle detectados, el robot utiliza esta información para optimizar el mapa almacenado. Esta optimización se realiza utilizando el algoritmo de optimización G2O, descrito en el Apartado 6.2.1.

Cuando el robot detecta un cierre de bucle, tiene que reubicar todos los nodos anteriores para reducir el error asociado en cada posición. Este proceso modifica todas las posiciones de los nodos en el mapa para tener en cuenta la nueva restricción calculada por la detección de cierre de bucle.

La modificación de la ubicación de los nodos se realiza mediante el algoritmo G2O. Dicho algoritmo recibe como entrada todas las posiciones de los nodos del mapa y la restricción de cierre de bucle. Entonces, G2O da como salida las nuevas posiciones de los nodos recalculadas.

Por lo tanto, los dos nodos del cierre del bucle se localizan en la misma posición y se modifican las coordenadas del resto de los nodos del mapa.

6.4 Experimentos

Esta subsección presenta los diferentes conjuntos de imágenes omnidireccionales utilizados para probar nuestro método y los resultados obtenidos en estos experimentos.

6.4.1 Bases de datos

Para comprobar el funcionamiento de la técnica propuesta, se utilizan dos conjuntos de imágenes capturadas. Para capturar el primer conjunto, el robot fue teleoperado para seguir un camino rectangular por el interior de un laboratorio. La trayectoria la forman 189 imágenes formando una trayectoria cuadrada de 3 $\times$ 3 metros. El segundo conjunto de imágenes fue capturado mientras el robot seguía un camino más complicado a través de varias habitaciones dentro de un edificio. Esta base de datos está formada por 783 imágenes omnidireccionales capturadas a lo largo de un entorno formado por un pasillo y

(a) (b)

Figura 6.4: (a) Muestra de una imagen omnidireccional de la trayectoria rectangular.(b) Imagen de muestra del segundo recorrido.

un salón de actos. Figura 6.4 muestra una imagen omnidireccional de muestra de cada entorno.

Estas dos bases de datos se han creado tomando una imagen omnidireccional cada 40 cm aproximadamente. La Figura 6.5 muestra el sistema catadióptrico utilizado para capturar las imágenes omnidireccionales, formado por la cámara (modelo: DFK-41BF02) y el espejo hiperbólico (modelo: Eizo Wide70).

6.4.2 Resultados

En esta sección se muestran los resultados de los experimentos realizados con nuestro algoritmo SLAM. Las dos bases de datos descritas en la Sección 6.4.1 se han utilizado para llevar a cabo estos experimentos.

El umbral máximo de distancia entre descriptores HOG es un parámetro importante a sintonizar. Para ello, hemos hecho algunas pruebas y elegido el mejor valor para detectar cierres de bucles. Después de estas pruebas se considera un umbral igual a 0,006 como un buen valor de la distancia entre descriptores HOG.

La Figura 6.6 muestra los resultados del algoritmo SLAM después de incorporar la posición final de la primera ruta. La línea azul es el mapa creado sin optimización y la línea verde es el mismo mapa optimizado. Esta optimización se realiza en cada iteración pero el mapa sin ninguna optimización se muestra a efectos comparativos. Como podemos ver, la línea verde es un camino con forma rectangular, la cual fue definida al tomar la base de datos.

La Figura 6.7 muestra los mismos resultados que en la Figura 6.6 pero utilizando la segunda ruta. La línea azul es el mapa creado optimizado y la línea amarilla es la trayectoria real.

Figura 6.5: Sistema de adquisición de imágenes omnidireccionales.

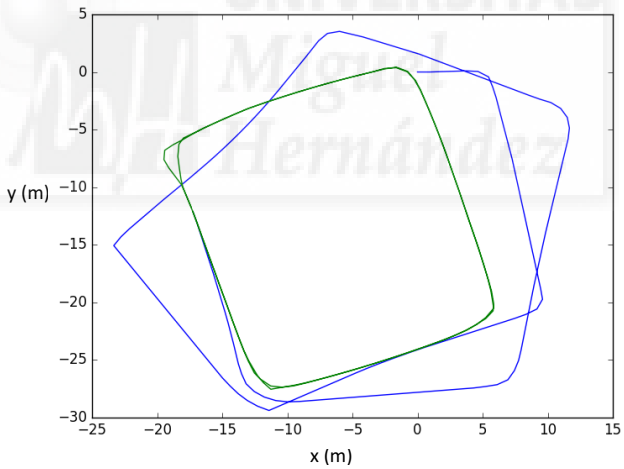


Figura 6.6: Mapa creado utilizando el primer recorrido. La línea azul es el mapa creado sin optimización y la línea verde es el mismo mapa optimizado.

En cuanto al tiempo computacional, el robot pasa un promedio de 0,65 segundos en cada iteración del proceso SLAM. Este tiempo aumenta en cada iteración porque el mapa está formado por mayor cantidad de nodos y la detección de cierre de bucle necesita comparar un mayor número de descriptores HOG. Cuando ejecutamos el algoritmo con la base de datos del pasillo y sólo utilizando las dos primeras imágenes se emplean 0,56 segundos, y en el último paso (con las dos últimas imágenes y todo el mapa almacenado) se emplean 0,70 segundos. El tiempo también varía entre iteraciones del programa debido

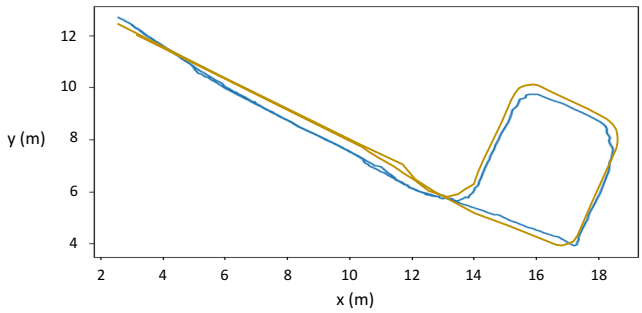


Figura 6.7: Mapa creado utilizando el segundo recorrido. La línea azul es el mapa creado optimizado y la línea amarilla es la trayectoria real.

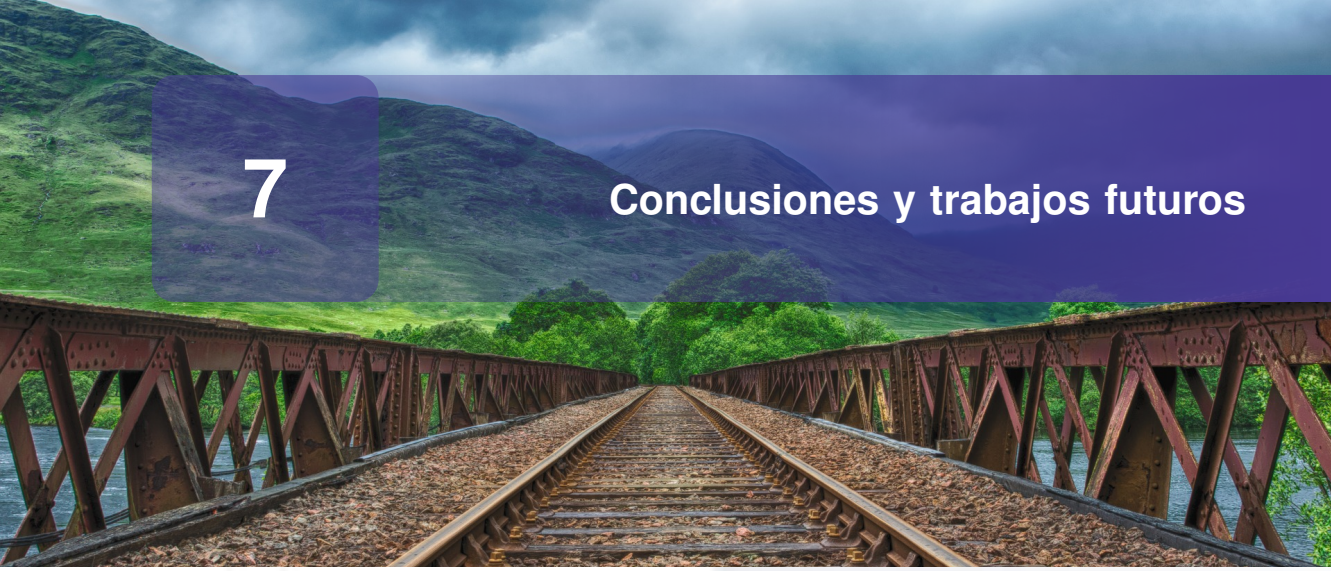
a que no siempre se detecta la misma cantidad de *landmarks*. Pero nunca llega a tardar un segundo en ninguno de nuestros experimentos, teniendo en cuenta que no se utiliza ningún algoritmo de procesamiento en paralelo.

6.5 Conclusiones

En este capítulo hemos presentado un método SLAM para estimar la posición y la orientación de un robot móvil en un entorno a la vez que el robot crea el mapa. Utilizamos dos descriptores de apariencia global diferentes para llevar a cabo el proceso SLAM y el mapa está formado por estos dos descriptores de cada imagen. Por último, el algoritmo ha sido probado con dos conjuntos de imágenes capturadas en dos entornos interiores diferentes.

Los resultados han demostrado la exactitud del método. En cuanto a los valores de los parámetros, el umbral máximo de distancia entre descriptores HOG es el parámetro de ajuste principal en este método.

Los resultados presentados en este trabajo muestran la efectividad de los descriptores de apariencia global de imágenes omnidireccionales para llevar a cabo tareas de SLAM gracias a la riqueza de la información que contienen.



7 Conclusiones y trabajos futuros

Habiendo detallado en los capítulos anteriores todo el trabajo realizado en el marco de esta tesis, ahora este capítulo resume las principales contribuciones que se pueden extraer de esta investigación. En consecuencia, algunos posibles trabajos futuros también se enumeran al final del capítulo.

7.1 Contribuciones

Hoy en día, la robótica móvil se encuentra presente en innumerables aplicaciones, lo cual ha llevado a la investigación enfocada a varios desafíos de suma importancia en este campo. Los robots móviles necesitan disponer de un mapa del entorno el cual debe ser construido de forma eficiente, de tal manera que pueda ser utilizado en tareas de localización. Muchas veces, ambas tareas deben ejecutarse en paralelo, lo cual se conoce como el problema de SLAM (Simultaneous Localization And Mapping). Los principales desafíos de este problema radican en la complejidad de los sistemas de adquisición de información del entorno y en los diferentes tipos de información proporcionados al robot. Normalmente se han utilizado las técnicas basadas en la extracción de características locales para describir las imágenes, y las técnicas basadas en la apariencia global de las escenas son técnicas más recientes que conducen a modelos más directos, y no han sido tan estudiadas como las primeras. En esta tesis se ha abordado el estudio de técnicas de este tipo para resolver estos problemas. En consecuencia, los principales objetivos se dirigieron al desarrollo de diferentes algoritmos de localización de robots utilizando imágenes omnidireccionales y descriptores de apariencia global. De acuerdo con estos objetivos, se derivaron diferentes avances y contribuciones, que se presentan por separado en cada capítulo de este documento. Esta sección incluye una síntesis con las contribuciones y los logros más relevantes logrados durante el período de investigación:

Capítulo 3

- Desarrollo de un programa que genera imágenes omnidireccionales desde un entorno virtual creado sintéticamente. Estas imágenes se pueden generar agregando ruido u oclusiones. Además, se puede simular el uso de diferentes tipos de sensores de visión para generar diferentes tipos de imágenes.
- Algunos ejemplos muestran las posibilidades del programa propuesto simulando diferentes ambientes interiores. Además, se muestran diferentes imágenes generadas simulando diferentes tipos de sensores visuales.

Capítulo 4

- Adaptación de diferentes descriptores de apariencia global para que puedan ser utilizados con imágenes omnidireccionales directamente, sin ninguna otra transformación.
- Desarrollo de dos algoritmos de localización diferentes:
  - Por un lado, un algoritmo para localizar el robot en un mapa previamente construido, calculando la posición más cercana del mapa, comparando una imagen omnidireccional capturada desde una posición desconocida con las imágenes almacenadas dentro de dicho mapa.
  - Por otro lado, un algoritmo para crear un mapa local del entorno cuando el robot tiene las imágenes del mapa pero no conoce las posiciones desde las cuales se capturaron estas imágenes. Este algoritmo crea un mapa local del entorno utilizando las imágenes más cercanas y calculando las distancias entre ellas.
- Resultados de rendimiento y precisión que validan los algoritmos de localización. Además, una evaluación del tiempo de cálculo empleado por el algoritmo muestra que se puede usar en aplicaciones en tiempo real.
- Estos algoritmos se han comparado con algoritmos basados en características SIFT y en la firma de Fourier. Esta comparación muestra la efectividad de los algoritmos propuestos.

Capítulo 5

- Creación de un algoritmo de estimación de altura relativa utilizando solo información visual. Se ha utilizado un único descriptor holístico basado en la transformada de Radon.
- Creación de un método alternativo basado en la extracción de características SURF para hacer una comparación con el método propuesto.

- La comparación de ambos métodos demuestra la efectividad del método propuesto basado en la apariencia global de las escenas frente al método alternativo basado en la extracción de características.

Capítulo 6

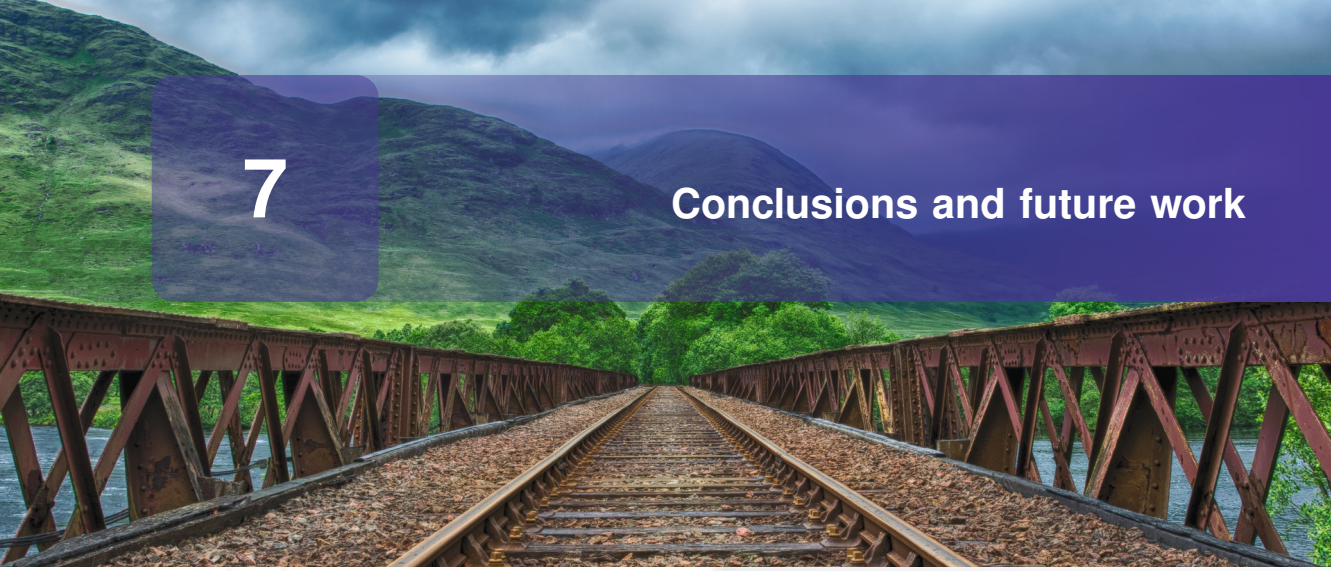
- Realización de tareas graph-SLAM usando solo un sensor visual, sin tener en cuenta datos de odometría ni de ningún otro sensor.
- Este método usa descriptores de apariencia global junto con la extracción de características. Sin embargo, el uso de puntos característicos no es el convencional, ya que se usa un comportamiento global de todos ellos para calcular la dirección del desplazamiento del robot.

7.2 Trabajos futuros

En las siguientes líneas, se describen algunas posibles líneas de investigación futuras que surgen como consecuencia de las contribuciones y resultados presentados en esta tesis:

- Estudio adicional sobre el uso del algoritmo graph-SLAM con bases de datos más extensas. Utilizando diferentes ambientes. Además, se realizará un estudio exhaustivo de la influencia de los diferentes parámetros en la precisión y el coste computacional.
- Agregar el modelo jerárquico dentro de los métodos de localización y creación de mapas para dar más eficiencia a los métodos propuestos. Permitiendo que el robot lleve a cabo su tarea en un entorno más grande debido a la reducción del tiempo de cálculo.
- Extensión del método de localización 2D y del método de estimación de altura relativa para hacer ambas localizaciones al mismo tiempo. Esto será posible ya ambos algoritmos utilizan la misma información del mapa, haciendo que el robot use el mismo mapa para ambos propósitos.
- Se estudiará el uso de un nuevo tipo de sensor omnidireccional. Este sensor proporciona una imagen equirrectangular del entorno, proporciona información de todos los puntos alrededor del robot. Este tipo de sensor proporciona información que se puede representar en una esfera unitaria, lo cual podría ser una ventaja para calcular los cambios de posición del robot.
- Según el último punto, este tipo de información permitirá al robot desarrollar tareas de localización en 6 grados de libertad.

UNIVERSITAS
Miguel
Hernández



7 Conclusions and future work

Having detailed in the previous chapters all the work conducted under the framework of this thesis, now this chapter summarizes the main contributions that can be extracted from this research. In consequence, some possible future works can be also listed.

7.1 Contributions

Nowadays, mobile robotics is present in innumerable applications, which has led to research focused on several challenges of great importance in this field. The mobile robots need to have a map of the environment which must be built efficiently, in such a way that it can be used in localization tasks. Many times, both tasks must be run in parallel, which is known as the SLAM problem (Simultaneous Localization And Mapping). The main challenges of this problem lie in the complexity of the information acquisition systems of the environment and in the different types of information provided to the robot. Typically, techniques based on the landmark extraction have been used, and the techniques based on the global appearance of the scenes are more recent techniques that lead to more direct models, and have not been as studied than the first ones. In this thesis has been addressed the study of techniques of this type to solve these problems. Consequently, the main objectives were directed to the development of different robot localization algorithms using omnidirectional images and descriptors of global appearance. According to these objectives, different advances and contributions were derived, as presented separately in each chapter of this document. This section includes a synthesis with the most relevant contributions and achievements accomplished during the research period:

Chapter 3

- Development of a program which generates omnidirectional images from a virtual environment created synthetically. These images can be generated adding noise or occlusions. Also, the use of different types of vision sensors can be simulated to generate different kinds of images.
- Some examples show the possibilities of the program proposed simulating different indoor environments. Moreover, different images generated simulating different visual sensors were shown.

Chapter 4

- Adaptation of different global appearance descriptors so that they can be used with omnidirectional images directly, without any transformation.
- Development of two different localization algorithms:
  - On the one hand, an algorithm to locate the robot in a previously built map, calculating the closest position of the map comparing an omnidirectional image captured from an unknown position with the images stored in the map.
  - On the other hand, an algorithm to create a local map of the environment when the robot has the images of the map but it does not know the positions where these images were captured. This algorithm creates a local map of the environment using the closest images and calculating the distances between them.
- Performance and accuracy results that validate the localization algorithms. Also, an evaluation of the computation time spent by the algorithm shows that it can be used in real time applications.
- These algorithms have been compared with algorithms based on the SIFT features and the Fourier Signature. It shows the effectiveness of the algorithms.

Chapter 5

- Creation of a relative height estimation algorithm using only visual information. A single global appearance descriptor based on the Radon transform has been used.
- Creation of an alternative method based on the extraction of SURF characteristics in order to make a comparison with the proposed method.
- The comparison of both methods demonstrates the effectiveness of the proposed method based on the global appearance of the scenes versus the alternative method based on the extraction of landmarks.

- Carry out graph-SLAM tasks using only a visual sensor, without taking any odometry data or any other sensor into account.
- This method uses global appearance descriptors together with feature extraction. However, the use of landmarks is not the conventional one, but a global behaviour of all of them is used to calculate the direction of displacement of the robot.

7.2 Future work

In the following lines, some possible future research lines which emerge as a consequence of the contributions and results presented in this work are described:

- Further study on the use of the graph-SLAM algorithm with more extensive databases. Using different environments. Also, an exhaustive study of the different parameter influence in the accuracy and computational cost will be made.
- Adding the hierarchical model inside the localization and mapping methods to give more efficiency to the proposed methods. It allows the robot to carry out its task in a bigger environment due to the reduction of the computational time.
- Extension of the 2D localization method and the relative height estimation method to do both localizations at the same time. It will be possible because both algorithms use the same map information so the robot will can use the same map fo both purposes.
- The study of a new type of omnidirectional sensor will be studied. This sensor provides an equirrectangular image of the environment, it provides information of all points around the robot. This kind of sensor provides information that can be represented in an unitary sphere. It could be an advantage to calculate the positions changes of the robot.
- According to the last point, this kind of information will enable the robot to develop localization tasks in 6 degrees of freedom.

UNIVERSITAS
Miguel
Hernández

Apéndice: conjunto de publicaciones

Las principales implementaciones y contribuciones realizadas en esta tesis están respaldadas por un conjunto de publicaciones en revistas clasificadas en JCR Science Edition. Los siguientes artículos de revistas respaldan el trabajo realizado en este documento:

Artículo de revista 1

Position Estimation and Local Mapping Using Omnidirectional Images and Global Appearance Descriptors. [7]
Y. Berenguer, L. Payá, M. Ballesta, O. Reinoso
Sensors. Vol 15(10) (2015)
ISSN:1424-8220. Ed. MDPI.
JCR-SCI Impact Factor: 2.033, Cuartil **Q1**.
Web: http://doi.org/10.3390/s151026368
DOI: 10.3390/s151026368

Artículo de revista 2

Using Omnidirectional Vision to Create a Model of the Environment: A Comparative Evaluation of Global-Appearance Descriptors. [55]
L. Payá, O. Reinoso, Y. Berenguer, D. Úbeda
Journal of Sensors. Vol 2016
ISSN:1687-725X. Ed. Hindawi.
JCR-SCI Factor de impacto: 1.704, Cuartil **Q2**.
Web: http://doi.org/10.1155/2016/1209507
DOI: 10.1155/2016/1209507

Las impresiones de los artículos se muestran a continuación:

Sensors **2015**, *15*, 26368-26395; doi:10.3390/s151026368

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

Position Estimation and Local Mapping Using Omnidirectional Images and Global Appearance Descriptors

Yerai Berenguer *, Luis Payá, Mónica Ballesta and Oscar Reinoso

Departamento de Ingeniería de Sistemas y Automática, Miguel Hernández University, Avda. de la Universidad s/n, Elche (Alicante) 03202, Spain; E-Mails: lpaya@umh.es (L.P.); m.ballesta@umh.es (M.B.); o.reinoso@umh.es (O.R.)

* Author to whom correspondence should be addressed; E-Mail: yberenguer@umh.es; Tel.: +34-96-522-2435; Fax: +34-96-665-8979.

Academic Editor: Kourosh Khoshelham

Received: 27 July 2015 / Accepted: 14 October 2015 / Published: 16 October 2015

Abstract: This work presents some methods to create local maps and to estimate the position of a mobile robot, using the global appearance of omnidirectional images. We use a robot that carries an omnidirectional vision system on it. Every omnidirectional image acquired by the robot is described only with one global appearance descriptor, based on the Radon transform. In the work presented in this paper, two different possibilities have been considered. In the first one, we assume the existence of a map previously built composed of omnidirectional images that have been captured from previously-known positions. The purpose in this case consists of estimating the nearest position of the map to the current position of the robot, making use of the visual information acquired by the robot from its current (unknown) position. In the second one, we assume that we have a model of the environment composed of omnidirectional images, but with no information about the location of where the images were acquired. The purpose in this case consists of building a local map and estimating the position of the robot within this map. Both methods are tested with different databases (including virtual and real images) taking into consideration the changes of the position of different objects in the environment, different lighting conditions and occlusions. The results show the effectiveness and the robustness of both methods.

Keywords: computer vision; position estimation; mapping; omnidirectional images; global appearance; Radon transform

1. Introduction

Nowadays, there are countless kinds of robots with innumerable configurations. Among them, mobile robots have been extended in recent years due to their flexibility, as they are able to change their position during operation. Usually, these robots have to solve a task in an unknown environment where the robot must estimate its position to be able to arrive at the target point, avoiding obstacles. There are two main approaches to solve the localization problem. First, the robot may have a map of the environment. In this case, the robot has to calculate its position within the map. Second, the robot may not have any *a priori* knowledge of the environment; thus, it has to create the map and then calculate its position. If these processes are carried out simultaneously, the problem is known as SLAM (simultaneous localization and mapping).

The robot can be equipped with many kinds of sensors to solve these problems, such as lasers, cameras, *etc*. They provide the robot with environmental information in different ways (e.g., lasers measure the distance to the nearest objects around the robot). This information is processed by the robot and permits building a map and estimating its position and orientation. With these data, the robot must be able to carry out its work autonomously.

Along the last few years, much research has been developed on robot mapping and localization using different kinds of sensors, and many algorithms have been proposed to solve these problems. Many these works use visual sensors, since they permit many possible configurations, and they provide the robot with very rich information from the environment. In this work, we consider a robot equipped with an omnidirectional vision system [1]. We can find many previous works that use omnidirectional images in mapping and localization tasks, such as [2–5]. Valiente *et al*. [2] present a comparison between two different visual SLAM methods using omnidirectional images. Maohai *et al*. [3] propose a topological navigation system using omnidirectional vision. Garcia *et al.* [4] present a map refinement framework that uses visual information to refine the topology of the environment. At last, Garcia *et al.* [5] make a survey of vision-based topological mapping and localization methods.

Traditionally, the developments in mobile robots using visual sensors are based on the extraction and description of some landmarks from the scenes, such as SIFT (scale-invariant feature transform) [6] and SURF (speeded-up robust features) [7] descriptors. This approach presents some disadvantages: the computational time to calculate and compare the descriptors is usually very high; thus, these descriptors may not be used in real time, and this leads to relatively complex mapping and localization algorithms. As an advantage, only a few positions need to be stored in the map to make the localization process possible.

More recently, some works proposed using the global information of the images to create the descriptors. These techniques have been demonstrated to be a good option to solve the localization and navigation problems in 2D. Chang *et al.* [8], Payá *et al.* [9] and Wu *et al.* [10] propose three examples of this approach. In [11], several methods to obtain global descriptors from panoramic scenes are analyzed and compared to prove their validity in map building and localization. The majority of these global appearance descriptors can be used in real time, because the computational time to calculate and handle them is low, and they usually lead to more straightforward mapping and localization algorithms.

In this work, we propose a solution to the mapping and localization problems using only the visual information captured by an omnidirectional vision system mounted on the robot. This system is composed of a camera pointing to a hyperbolic mirror and captures omnidirectional images of the environment. Each scene is described with a single global-appearance descriptor.

Our starting point is a database of omnidirectional images captured on a grid of points in the environment where the robot has to navigate.

Compared to previous works, the contributions of this paper are:

1. Describing the global appearance of a set of images: We develop a method based on the Radon transform [12] and the gist descriptor [13] to create a visual model of the environment. We have not found any previous work that uses the Radon transform as a global appearance descriptor to solve the mapping and localization problems in robotics.
2. Solving the mapping and localization problems with these descriptors: This contribution is two-fold:
   (a) Solving the localization problem with accuracy, with respect to a visual model (map) previously created: In this model, the positions where the images were captured are known.
   (b) Solving the localization problem with respect to a visual model where the capture positions of the images are not known: We have implemented a local mapping algorithm, and subsequently, we solve the localization problem.

The experiments have been carried out with several different image databases. The first one has been created synthetically from a virtual room. We also use some real databases to test the validity of our method.

The remainder of this paper is structured as follows. Section 2 introduces the concept of global appearance and the descriptors that we have developed. Section 3 presents the algorithm that we have implemented to solve the local mapping problem (spring method). Section 4 describes our localization method to estimate the position of the robot. In Section 5, the experiments and results are presented. At last, a discussion is made in Section 6, and Section 7 outlines the conclusions.

2. Global Appearance of Omnidirectional Images

Methods based on the global appearance of the scenes constitute a robust alternative compared to methods based on landmark extraction. The key is that the global appearance descriptors represent the environment through high level features that can be interpreted and handled easily, and with a reasonably low computational cost.

This section presents the transform that we have employed to describe the scenes (omnidirectional images). Each scene is represented through only one descriptor that contains information of the global appearance without any segmentation or local landmark extraction. We also present the distance measures we use to compare descriptors, and we study their properties.

Any new global appearance description method should satisfy some properties: (1) it should make a compression effect in the image information; (2) there should be a correspondence between the distance between two descriptors and the metric distance between the two positions where the images were

captured; (3) the computational cost to calculate and compare them should be low, so that the approach can be used in real time; (4) it should provide robustness against noise, changes in lighting conditions, occlusions and changes in the position of some objects in the environment; and, at last, (5) it should contain information of the orientation that the robot had when it captured the image.

The description method we have employed is mainly based on the Radon transform. We also make use of the gist descriptor, because of its properties in localization tasks [11].

2.1. Radon Transform

The Radon transform was initially described in [12]. It has been used traditionally in some computer vision tasks, such as shape description and segmentation ([14,15]).

The Radon transform in 2D consists of the integral of a 2D function over straight lines (line-integral projections). This transform is invertible. The inverse Radon transform reconstructs an image from its line-integral projections. For this reason, it was initially used in medical imaging (such as CAT scan and magnetic resonance imaging (MRI)).

The Radon transform of a 2D function $f(i,j)$ can be defined mathematically as:

$$\mathcal{R}\{f(i,j)\} = \lambda_f(p,\phi) = \iint_{-\infty}^{+\infty} f(i,j)\,\delta(p - \overrightarrow{r}\,\widehat{\overrightarrow{p}})\,di\,dj \tag{1}$$

where δ is the Dirac delta function ($\delta(x) = 1$ when $x = 0$, and $\delta(x) = 0$ elsewhere). The integration line is specified by the radial vector $\overrightarrow{p}$ that is defined by $\overrightarrow{p} = \widehat{\overrightarrow{p}} \cdot p$, where $\widehat{\overrightarrow{p}}$ is a unitary vector in the direction of $\overrightarrow{p}$. p is the $\overrightarrow{p}$ magnitude:

$$p = |\overrightarrow{p}| \tag{2}$$

The line-integral projections evaluated for each azimuth angle, ϕ, produce a 2D polar function, λ_f, that depends on the radial distance p and the azimuth angle ϕ. $\overrightarrow{r}$ is a cluster of points that is perpendicular to $\overrightarrow{p}$.

The Radon transform of an image $im(i,j)$ along the line $c_1(d,\phi)$ (Figure 1) can be expressed more clearly by the following equivalent expression:

$$\mathcal{R}\{im(i,j)\} = \int_{\mathbb{R}} im(i'\cos\phi - j'\sin\phi, i'\sin\phi + j'\cos\phi)\,ds \tag{3}$$

where:

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \cdot \begin{bmatrix} i \\ j \end{bmatrix} \tag{4}$$

When the Radon transform is applied to images, it calculates the image projections along the specified directions through a cluster of line integrals along parallel lines in these directions. The distance between the parallel lines is usually one pixel. Figure 2a shows the integration paths to calculate the Radon transform of an image in the ϕ direction, and Figure 2b shows the value of each component of the Radon transform in a simplified notation.

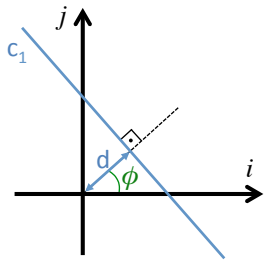
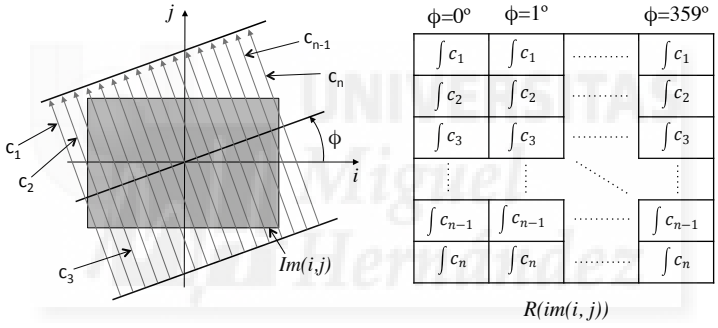
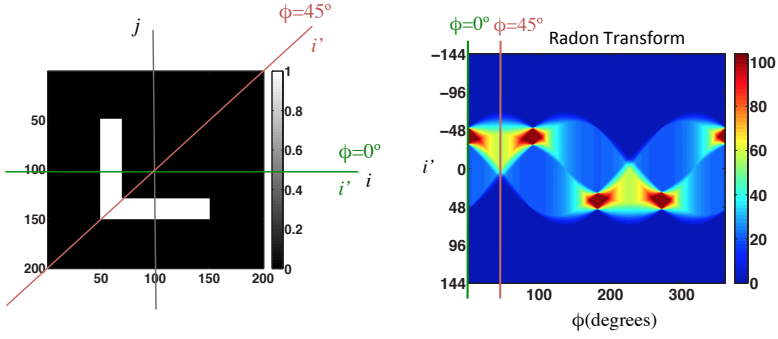


Figure 1. Line parametrization through the distance to the origin d and the angle between the normal line and the i axis, ϕ.



(**a**) (**b**)

Figure 2. (**a**) Integration paths to calculate the Radon transform of the image $im(i, j)$ in the ϕ direction. (**b**) Radon transform matrix of the image $im(i, j)$.



(**a**) (**b**)

Figure 3. (**a**) Sample image. (**b**) Radon transform of the sample image.

Figure 3 shows a sample black and white image, on the left, and its Radon transform, on the right. Furthermore, it shows graphically the process to calculate the Radon transform.

2.1.1. Radon Transform Properties

The Radon transform has several properties that make it useful in localization tasks using omnidirectional images. These properties are the following:

- Linearity: The Radon transform meets the linearity property as the integration operation is a linear function of the integrand:
$$\mathcal{R}\{\alpha f + \beta g\} = \alpha\mathcal{R}\{f\} + \beta\mathcal{R}\{g\} \tag{5}$$
- Shift: The Radon transform is a variant operation to translation. A translation of the two-dimensional function by a vector $\overrightarrow{r_0} = (i_0, j_0)$ has a translation effect on each projection. This translation is given by a distance $\overrightarrow{r} \cdot (\cos\phi, \sin\phi)$.
- Rotation: If the image is rotated an angle ϕ_0, it implies a shift ϕ_0 of the Radon transform along the variable ϕ (columns shift).
- Scaling: A scaling of f by a factor b implies a scaling of the d coordinate and amplitude of the Radon transform by a factor b:
$$\mathcal{R}\left\{f\left(\frac{i}{b}, \frac{j}{b}\right)\right\} = |b|\lambda_f\left(\frac{d}{b}, \phi\right) \tag{6}$$

2.2. Gist Descriptor

The gist descriptors try to imitate the human perception system to describe the scenes. They identify regions with a prominent color or texture in relation to the environment. The gist concept was introduced by Oliva and Torralba [13] with the name holistic representation of the spatial envelope. The authors proved that this description method creates a multidimensional space in which the scenes with the same semantic category (e.g., streets, sky, buildings, *etc.*) are projected in near points.

Mathematically, the method tries to codify the spatial information by means of 2D Fourier transform in several regions of the image. These regions are distributed in a regularly-spaced grid. The set of descriptors in each region is dimensionally reduced by PCA (principal component analysis) to generate a unique scene descriptor with reduced dimension. In recent works, wavelet pyramids have been used instead of the Fourier transform, as in [16], where the authors carried out several experiments with different groups of natural images (coasts, forests, open fields, deserts, *etc.*) and artificial images (images of buildings, doors, interior of buildings, roads, *etc.*). The developed descriptors worked successfully to classify these images according to the above properties.

Recent works use the prominence concept together with gist, which highlights the zones that differ more from their neighbors [17]. This descriptor is built with the information of intensity, orientation and color.

In this work, some changes are made to this descriptor, since it will be used for a different purpose. We work with a set of indoor images with many similarities between them. The descriptor must be able to work with these images in such a way that the distance in the descriptor space reflects the geometrical

distance between the points where the images were captured. Additionally, the Radon transform has been used instead of images. Since this transform can be interpreted as a grayscale image, only the concept of prominence is used (but not color information). The steps we have implemented to build the gist descriptor are:

1. Building a pyramid of Radon transforms: At first, a Gaussian pyramid of n Radon transforms is created to describe the Radon transform features in several scales. The first level of the pyramid is the original Radon transform. Each new level is obtained from the previous one, applying a Gaussian low-pass filter and subsampling it to reduce its resolution. Figure 4 shows an example of a Radon transform four-level pyramid.

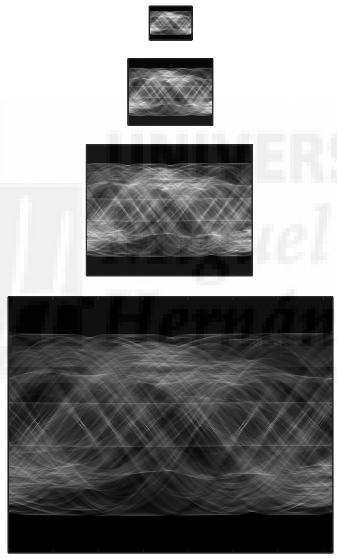


Figure 4. Example of a Radon transform four-level pyramid.

2. Gabor filtering: In order to incorporate the orientation information in the descriptor, each Radon transform of the two levels of the pyramid is filtered by four Gabor filters with different orientations $\theta = \{0, 45, 90, 135\}^{\circ}$. As a result, four matrices per pyramid level are obtained, with orientation information in the four analyzed directions.
3. Blockification. Finally, a dimensionality reduction process is needed. With this aim, we group the pixels of every matrix in blocks calculating the average intensity value that has the pixels in each block. We have decided to use horizontal blocks. This type of block is interesting, because the information in each block is independent of the robot orientation. The final descriptor will be composed of b horizontal blocks per pyramid level and Gabor filter, since it will be used only for localization purposes (additionally, we could have defined vertical blocks if we had needed to compute the robot orientation [11]).

Figure 5 shows the process to obtain the gist descriptor. In the Figure 5a, we can observe the Radon transform of an omnidirectional image. The use of each Gabor filter in the Radon transform and the blockification is shown in Figure 5b. In this case, $n = 2$ levels, and $b = 8$ horizontal blocks. Finally, in Figure 5c, the final composition of the gist descriptor is shown. The size of the descriptor is equal to $4 \cdot n \cdot b$ components.

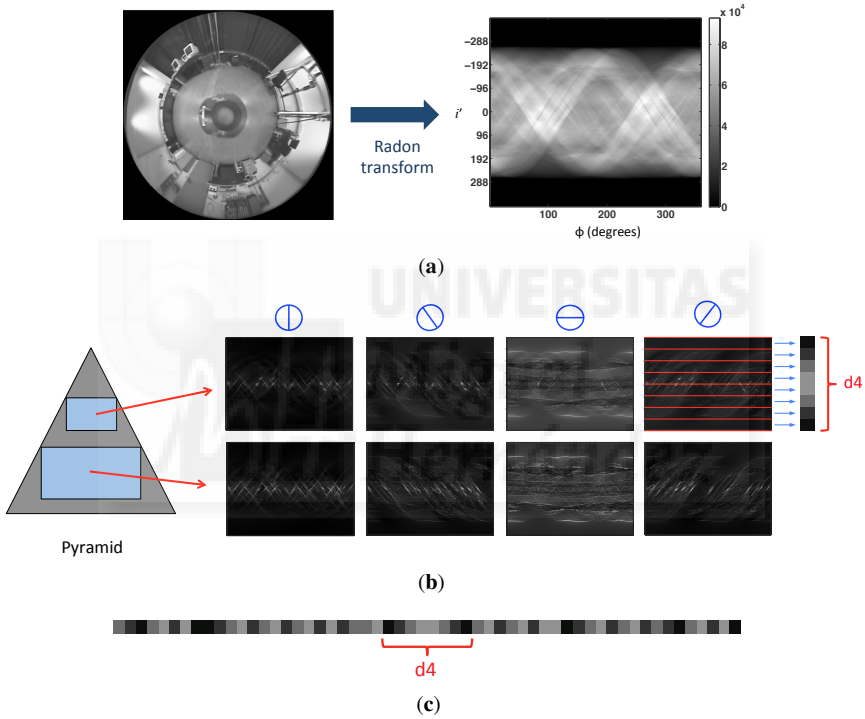


Figure 5. (**a**) The Radon transform of an omnidirectional image; (**b**) Matrices obtained after applying each of the four Gabor filters to the two levels of the pyramid; (**c**) Final composition of the gist descriptor.

2.3. Phase-Only Correlation

In this subsection, we present the method employed to compare the Radon transform of two images.

In general, a function in the frequency domain is defined by its magnitude and its phase. Sometimes, only the magnitude is taken into account, and the phase information is usually discarded. However, when the magnitude and the phase features are examined in the Fourier domain, it follows that the phase features contain also important information, because they reflect the characteristics of patterns in the images.

Oppenheim and Lim [18] have demonstrated this by reconstructing images using the full information from the phase with unit magnitude. This shows that the images resemble the originals, in contrast to reconstructing images using the full information from the magnitude with uniform phase.

Phase-only correlation (POC), proposed in [19], is an operation made in the frequency domain that provides a correlation coefficient between two images [20]. In our case, the objective is to compare the Radon transforms of two different images. This does not affect the POC performance, because the Radon transform can be interpreted as an image. Therefore, we explain POC using images.

The correlation between two images $im_1(i,j)$ and $im_2(i,j)$ calculated by POC is given by the following equation:

$$C(i,j) = \mathcal{F}^{-1}\left\{\frac{\mathbf{IM}_1(u,v)\cdot\mathbf{IM}_2^*(u,v)}{|\mathbf{IM}_1(u,v)\cdot\mathbf{IM}_2^*(u,v)|}\right\} \tag{7}$$

where $\mathbf{IM}_1$ is the Fourier transform of Image 1 and $\mathbf{IM}_2^*$ is the conjugate of the Fourier transform of Image 2. $\mathcal{F}^{-1}$ is the inverse Fourier transform operator.

To measure the similitude between the two images, the following coefficient is used:

$$sim_{POC}(im_1, im_2) = max\{(C(i,j)\} \tag{8}$$

This coefficient takes values in the interval $[0,1]$, and it is invariant under shifts in the i and j axes of the images. Furthermore, it is possible to estimate these shifts Δ_i and Δ_j along both axes by:

$$(\Delta_i, \Delta_j) = argmax_{(i,j)}\{C(i,j)\} \tag{9}$$

If we use Radon transforms instead of images, the value Δ_i allows us to estimate the change of the robot orientation when capturing the two images.

This way, POC is able to compare two images independently on the orientation, and it is also able to estimate this change in orientation.

2.4. Distance Measure

To carry out the localization process, we need any mechanism to compare descriptors (distance measure). We have tested two different methods.

First, we can measure the distance between the Radon transform (RT) of two scenes by means of the POC magnitude (POC distance) by the following equation:

$$dist_{POC}(RT_1, RT_2) = 1 - sim_{POC}(RT_1, RT_2) \tag{10}$$

Second, we can make use of the gist descriptor to obtain a distance measure between images following the next steps:

1. Firstly, the two omnidirectional images are transformed by the Radon transform to create two descriptors RT_1 and RT_2.
2. Secondly, RT_1 and RT_2 are transformed with the gist approach (Section 2.2) to obtain g_1 and g_2.
3. Finally, the Euclidean distance between g_1 and g_2 is obtained (gist distance).

Figure 6 shows the POC and the gist distance *vs.* the geometric distance between the points where the images were captured. According to this figure, the POC distance shows a nonlinear behavior. This method seems to be a promising option to identify the nearest image, but it is not good to estimate the distance between images. The POC distance can be linearized by the following expression:

$$dist_{linearized} = dist_{POC}^2 \tag{11}$$

where $dist_{linearized}$ is the final distance and $dist_{POC}$ the original POC distance. Figure 7 shows the POC distance and the linearized POC distance *vs.* the geometric distance of one aleatory linear path using our virtual database.

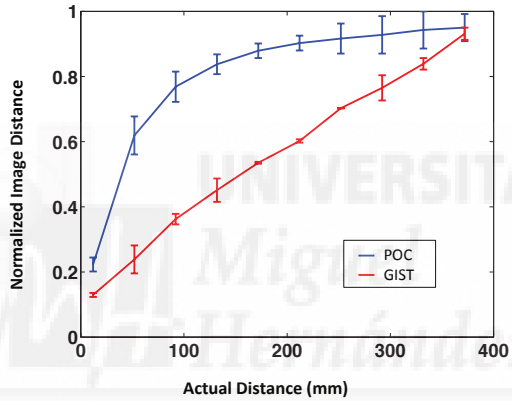


Figure 6. POC distance and gist distance *vs.* geometric distance. Each distance measure has been calculated along four different paths.

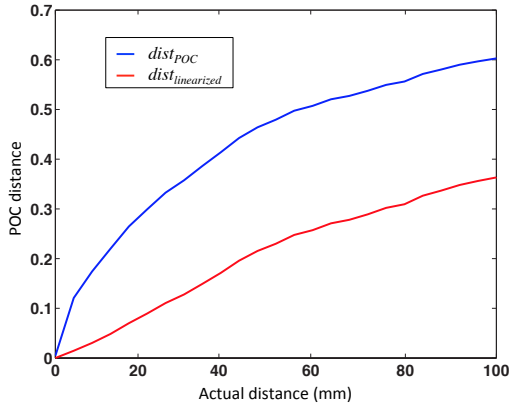


Figure 7. Phase-only correlation (POC) distance and POC distance linearized *vs.* geometric distance.

On the other hand, Figure 6 shows that the gist distance presents a quite linear behavior, so we expect it to be useful in mapping tasks.

3. Local Map Building (Spring Method)

In this section, we address the problem of building local maps of an unknown environment. We start from a set of omnidirectional images captured from different points of the environment to map, and we have no information of the capture positions nor of the order they were captured. Our objective is to estimate these positions in local regions of the environment. Thanks to this, we will be able to carry out a subsequent localization process (Section 4).

The mapping problem is solved using a mass-spring model, which represents the final map as a graph formed by nodes linked through connectors. Each node contains one image, and the connectors represent the neighbor relationships between two nodes, so that the images that have been captured sequentially are expected to belong to nodes that are connected together.

The only information we use to build the map is the distance among image descriptors. Once the map is built, we make use of the Procrustes transformation [21] to compare the resulting map with the actual map.

The method is based on a physical system of forces named the spring model [22,23]. The model is formed by the combination of two principles: Hooke's law and Newton's second law. Figure 8 shows the physical model scheme using three nodes. Each node N_i corresponds to an image, and they are linked with the springs S_{ij} whose natural length l_{ij0} is the distance between descriptors of images i and j. This method connects all of the nodes that have been previously chosen to create the local map and estimate the accuracy of the process.

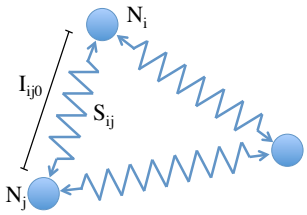


Figure 8. Spring model.

This method is based on the calculation of the elastic forces exerted by the springs on each node. Once the resulting force exerted on each node has been calculated, we proceed to obtain the acceleration, speed and position of each node.

The mapping process consists of solving at each time instant the next equations:

$$\overrightarrow{F_i} = \sum_{S_{ij} \in S} \left(-K_{ij} \left(l_{ij0} - l_{ij} \right) - b_{ij} \left(v_i - v_j \right) \right) \tag{12}$$

$$\overrightarrow{a_i} = \frac{\overrightarrow{F_i}}{m_i} \tag{13}$$

$$\overrightarrow{v_i}(t+\Delta T) = \overrightarrow{v_i}(t) + \overrightarrow{a_i}(t)\cdot\Delta T \tag{14}$$

$$\overrightarrow{r_i}(t+\Delta T) = \overrightarrow{r_i}(t) + \overrightarrow{v_i}(t)\cdot\Delta T, \tag{15}$$

where Equation (12) corresponds to Hooke's harmonic oscillator law. The resulting force on node i, $\overrightarrow{F_i}$, depends on the length of the springs l_{ij} and the elastic constant of the spring K_{ij}. A damper with damping constant b_{ij} has also been added between nodes, as it helps to improve the convergence of the algorithm. Newton's second law allows us to obtain the acceleration $\overrightarrow{a_i}$ using the node mass m_i. To simplify the system of forces, we used a mass equal to one for all nodes. Finally, the last two expressions are used to update the calculation of the speed $\overrightarrow{v_i}$ and the position $\overrightarrow{r_i}$ of all nodes in the system at each iteration (for each time increment, ΔT).

4. Localization Methods

In this section, we address the localization problem. Initially, the robot has a map of the environment. It is composed of a set of omnidirectional images of this environment. Then, the robot captures an image from an unknown position (test image). Comparing this image to the visual information stored in the map, the robot must be able to estimate its position.

To develop the algorithms, we take into consideration two different situations: (1) the positions where the map images were captured are known (pure localization); and (2) these positions are not known, and they must be estimated prior to the localization process (local map building and localization).

We have designed these algorithms to solve the problems:

1. Detecting the nearest image of the map.
2. Local mapping and localization.

The first method allows us to know the nearest image of the map (the most similar to the test image). Thanks to this information, we know that the robot is located in the surroundings of the corresponding image. The second method consists of calculating the image distance between the nearest images of the map and the test image. The number of nearest images is a parameter that can be modified to optimize the method. With these distances, we use the spring method to estimate the position of each image. As a result, we have the local map and the localization of the robot. These two methods are detailed in the following subsections.

4.1. Nearest Position of the Map

The operation consists of the following steps:

1. The robot captures an omnidirectional image from its current unknown position (test image). The objective is to estimate this position.
2. This image is transformed using the Radon transform.

3. The Radon transform of the test image is compared to all of the Radon transforms of the map using the POC comparison. As a result, we know which is the most similar image in the map to the test image.
4. The position where this omnidirectional image was taken is the nearest neighbor.

After this process, we assume that the robot is around this position. This is an absolute localization process, since we did not know the previous position of the robot nor its path. The accuracy depends mainly on the distance between the images of the map.

4.2. Local Mapping and Localization

In this subsection, we propose a method to localize the robot without any information of the position where the images of the map were captured. We have the Radon transforms of each image, but we do not have information about the position in which these images were captured. This localization method consists of the following steps:

1. We choose a number of nearest neighbors to use. These nearest neighbors are the descriptors that present the lowest POC distance to the Radon transform of the test image.
2. The gist descriptor of each Radon transform is calculated including the test image.
3. The distance between each pair of gist descriptors is obtained. We test two different distance measures: the Euclidean and the POC distance.
4. The spring method is used with these distances. This step provides a relative position between each gist descriptor. As a result, the position of the map images (local map) and the position of the test image are now known. This is our estimation of the robot position.
5. To estimate the localization error with respect to the local map, we must take into account that this map may present a rotation and a scale factor comparing to the actual position of the images. To consider these effects, we use a Procrustes approach [21] to estimate the error. This method removes the rotation and scale factor of the local map. The final error is the distance between the actual position of the robot and the calculated position after removing both effects.

Figure 9a shows the location area calculated by the first method (nearest position of the map). Figure 9b shows the location calculated by the second method with the nine nearest neighbors (local mapping and localization). In this figure, the map positions are represented at the actual capture points for representation purposes, but the algorithm does not know these positions. In Figure 9c, the local map created and the localization of the robot is shown. To make this local map, the initial position of each node to launch the spring method is random. This randomization may have an effect on the final result. For this reason, we run the algorithm several times. The results of this spring method are the local map and the robot localization.

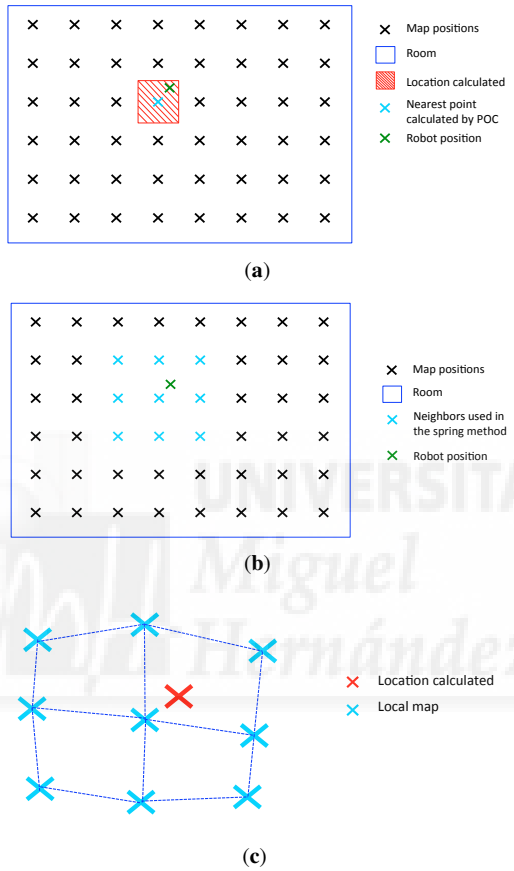


Figure 9. (**a**) Nearest position of the map; (**b**) Map positions and neighbors used in the spring method; (**c**) Local mapping and localization.

5. Experiments and Results

In this section, first, we present the virtual database created to test the methods and the results obtained with it. Second, we describe the database of real images that we have used and the results obtained with this second database.

5.1. Virtual Database

In order to check the performance of the proposed technique, we have created a virtual environment that represents an indoor room. In this environment, it is possible to create omnidirectional images from

any position using the catadioptric system showed in Figure 10. Figure 11a shows a bird's eye view of the environment.

The omnidirectional images have 250 × 250 pixels, and they have been created using a catadioptric system composed of a camera and a hyperbolic mirror whose geometry is described in Figure 10. The parameters used in the mirror equation are $a = 40$ and $b = 160$.

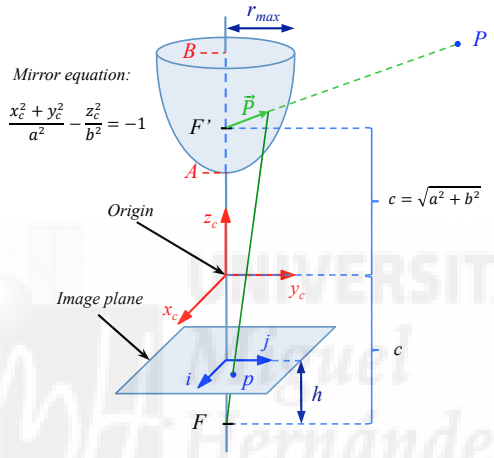
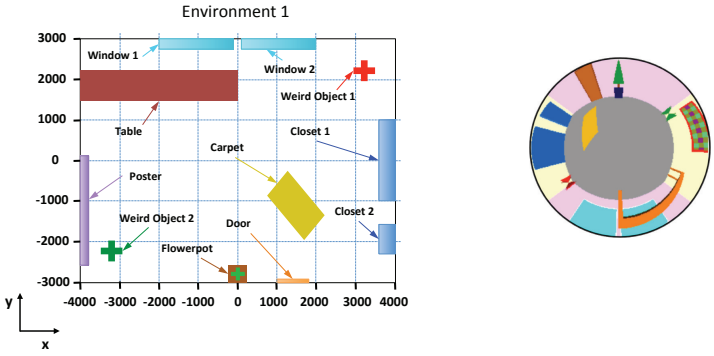


Figure 10. Catadioptric system used to capture the synthetic omnidirectional images.



(**a**) (**b**)

Figure 11. (**a**) Bird eye's view of the virtual environment; (**b**) Example of an omnidirectional image captured from the points x = 0 and y = 0.

Several images have been captured in the environment to create the map from several positions on the floor. The map is composed of 4800 images captured on a 8 × 6 meter grid with a step of 10 cm

between images. To carry out the experiments, we can change the step of the grid to test its influence. We must take into account that the higher the grid step, the fewer images compose the map. Figure 11b shows one sample omnidirectional image of the environment created with our program.

5.2. Results Obtained with the Virtual Database

To test our methods, we will study the influence of some parameters, such as the grid step of the map and the number of neighbors used in the spring method. In this subsection, we compare the results of the tests to try to optimize each method.

Firstly, we analyze the first method (obtaining the nearest position of the map) and the influence of the grid step of the map. Secondly, we analyze the second method (local mapping and localization) and the influence of the grid step of the map and the number of nearest neighbors.

5.2.1. First Method: Detecting the Nearest Image of the Map

In this experiment, we analyze four different step sizes between consecutive map positions. The distances that we will use are 100, 200, 300 and 400 mm. The size of the grid is 8 m $\times$ 6 m in all cases, so the number of map images depends on the step size. It will have an important influence on the computational cost.

Figure 12 shows the POC distance (Equation (10)) between the Radon transform of a test image and each image of the map (200 $\times$ 200). The position of the test image is x = -2239 mm and y = -1653 mm. Additionally, the corresponding position according to this figure (the minimum of the 2D function) is x = -2200 and y = -1600. As we can see, the distance decreases sharply around this position.

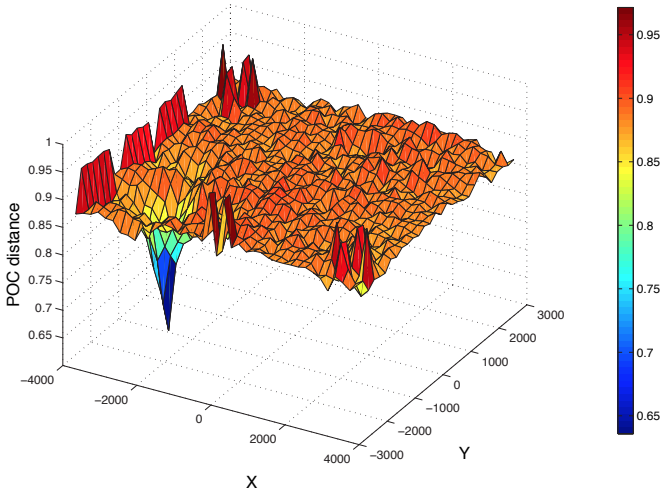


Figure 12. Distance between the Radon transform of a sample test image and all of the images of the map. The position of the test image is x = -2239 mm and y = -1653 mm.

Figure 13 shows the final result of this test. We have used 3500 test images captured from different random positions of the environment. In each bar, the blue part represents the proportion of correct localizations; the green part represents that the method has localized the second nearest position (*i.e.,* the position calculated is not the nearest position, but rather the second nearest position of the map); and the red part is the proportion of errors for each map size.

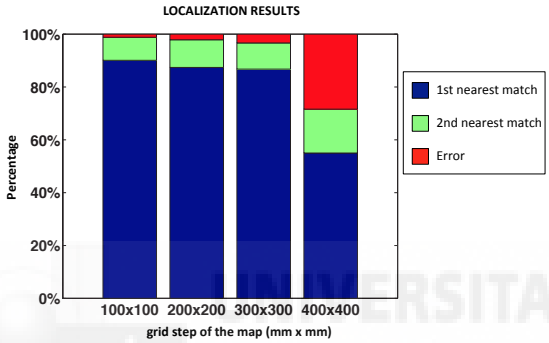


Figure 13. Results of the nearest image experiments: success rate.

We can observe that the method is working properly with a distance between map positions not exceeding 300 $\times$ 300 millimeters, because if this value increases, the POC distance begins to saturate, as stated in Section 2.4.

5.2.2. Second Method: Local Mapping and Localization

In this subsection, the second method is analyzed, and we study the influence of the distance between map positions and the number of nearest neighbors used in the spring algorithm. To do this experiment, we have chosen the same distances between map positions as in the previous subsection: 100, 200, 300 and 400 mm. Then, we will choose the value that provides the best result, and we will then optimize the number of nearest neighbors: we will test grids of 9, 25 and 49 nearest neighbors.

The algorithm has been launched with 500 test images captured from random positions generated with our program, repeating the experiment four times with each test image. Figure 14 shows the error distribution of the experiments with the virtual database using the POC distance, depending on the number of nearest neighbors and the grid step of the map. This error is the localization error of the test image calculated according to Step 5 (Section 4.2). Figure 15 presents the error distribution of the same experiments, but with the gist distance. In this environment, the gist distance presents a lower localization error than the POC distance. The parameters used in the gist descriptors for the experiments are $n = 2$ levels and $b = 16$ horizontal blocks.

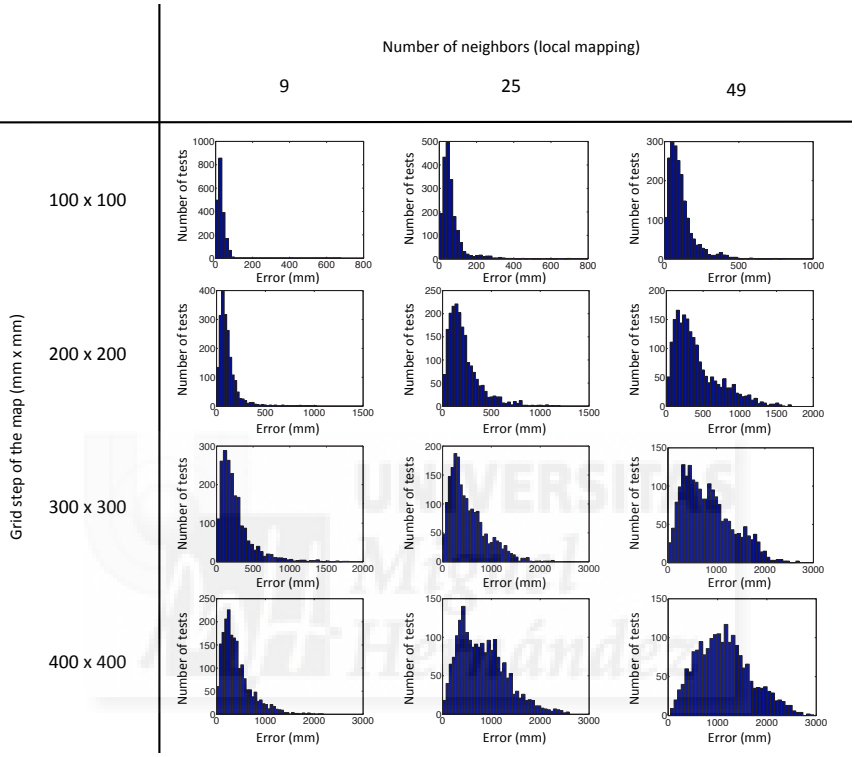


Figure 14. Error distribution of the experiments with our virtual database, using POC distance.

As we can see in the figures, the best results of the test have been achieved with a grid step equal to 100 mm and 200 mm, but we must also consider the computation time. Figure 16 shows the average computation time per iteration in each case, in the case that we have chosen 25 nearest neighbors. The distance between map positions of 100 mm is not advisable, because the computation time is relatively high. For this reason, we choose the distance of 200 mm to do the following tests, with actual images, because taking into account the error and the computation time, it is the best option. We can observe that the computational time of the second method is the same for all cases, because it does not depend on the size of the map, but rather on the number of nearest neighbors.

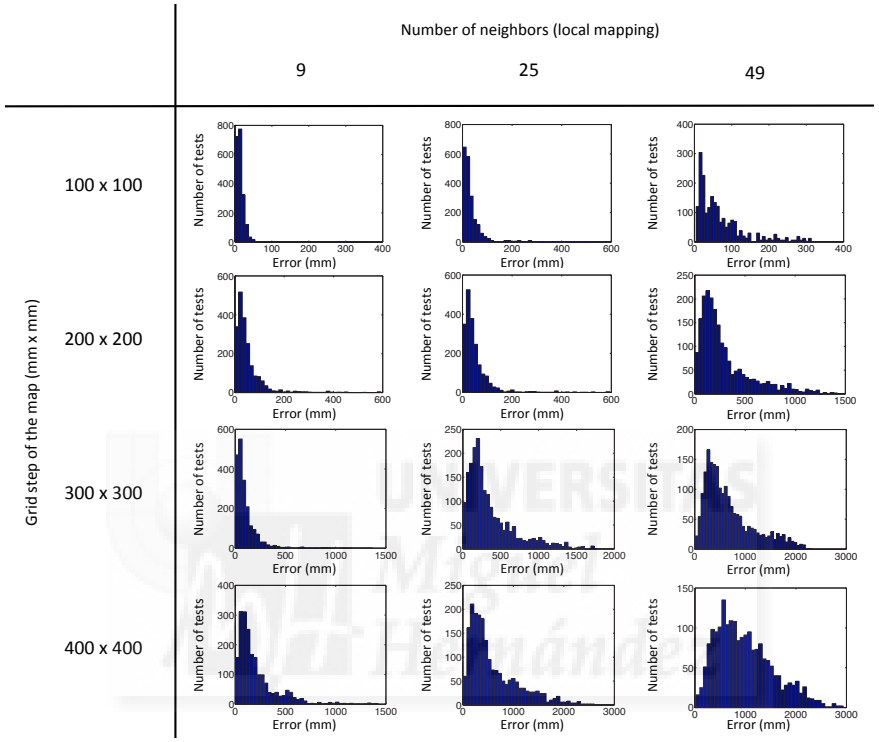


Figure 15. Error distribution of the experiments with our virtual database, using gist distance.

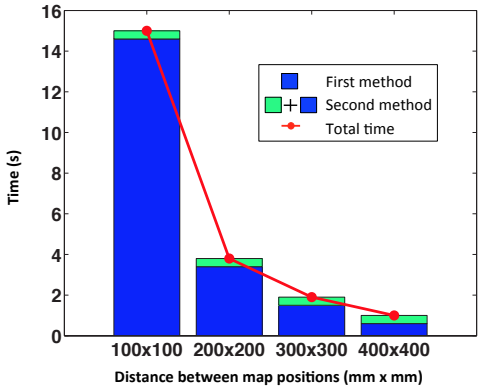


Figure 16. Computational time for each iteration. Using 25 nearest neighbors in the second method.

5.3. Results with Actual Databases

Once we have analyzed our methods with a virtual database, now we will test them with some databases composed of images captured in real working environments under real lighting conditions.

First, we test the first method using a third-party database that has different lighting conditions.

Second, we test the second method with our own databases (two different rooms of our university).

5.3.1. First Method: Detecting the Nearest Image of the Map

To test this method, we have used the Bielefeld database [24]. This database contains several image maps with changes in the position of some objects and in the lighting conditions. The images were captured on a 10 × 17 grid with a 30-cm step. Hence, the area of the capture grid was 2.7 m × 4.8 m, which covered nearly all of the floor's free space. These image maps are:

- Original: the standard or default condition of the room. Images were collected with both overhead fluorescent light bars on, the curtains and door closed and no extraneous objects present.
- Chairs: three additional office chairs were positioned within the capture grid.
- Arboreal: a tall (3 m) indoor plant was added to the center of the capture grid.
- Twilight: curtains and door open. The image collection lasted from sunset to the evening.
- Doorlit: the light bar near the window was switched off.
- Winlit: the light bar near the door was switched off.

To carry out this first experiment, we have used the "original" set as our map of images and the images in the other sets as test images. Figure 17 shows the success rate of this experiment. We have used 20 random test images of each database to obtain this figure.

This is quite an interesting experiment, as it shows the robustness of the algorithm, even when the test images present changes with respect to the map.

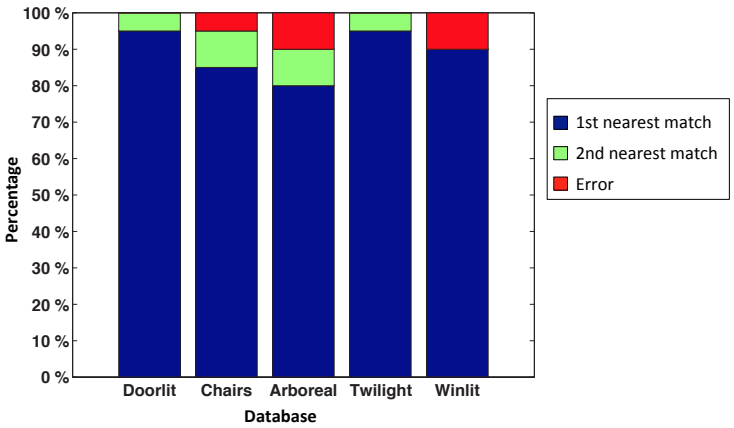


Figure 17. Success rate with the Bielefeld database.

5.3.2. Second Method: Local Mapping and Localization

In this case, we have used our database of images, captured at our university. It covers two different rooms, a laboratory and an office. The images have been captured on a grid with 8 × 8 images with a 20-cm step, in both cases.

We follow two steps. First, we detect the nearest neighbors by the POC comparison of the Radon transforms of the map images. Second, we create the local map with these Radon transforms using the gist or POC distance. Finally, we localize our robot in this local map. Figure 18 shows the error distribution of the second method using our databases ("laboratory" and "office") and both distance measures (POC and gist distance). The experiment has been carried out with 13 test images and 50 times per test image in the laboratory and with mine test images and 50 times per test image in the office. Figure 19 shows the error distribution of the second method using our databases, adding occlusions to each test image. Two examples of omnidirectional images with occlusions are shown in Figure 20.

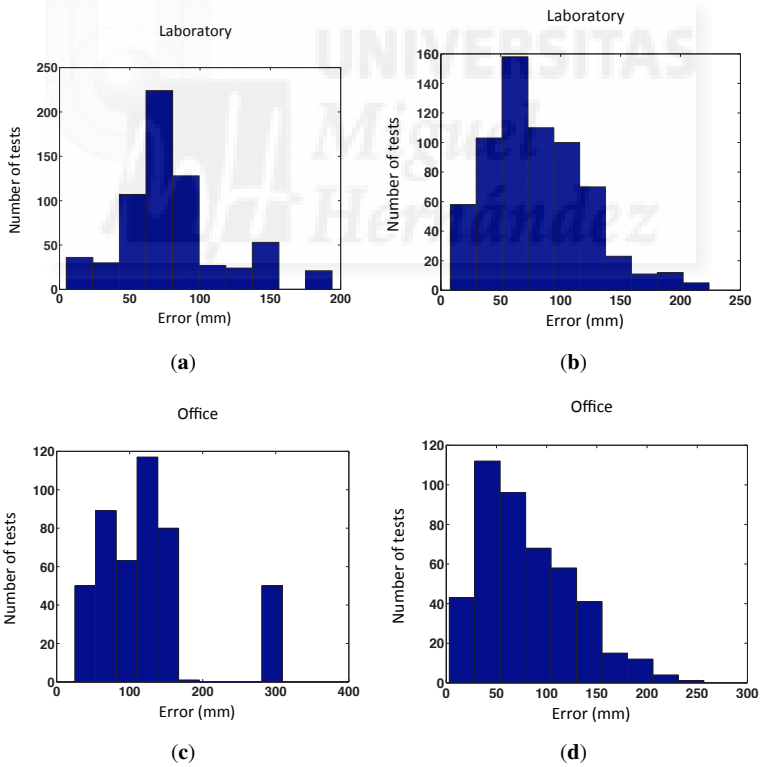


Figure 18. Error distribution of the second method. (**a**,**b**) Laboratory database and gist and POC distances, respectively; (**c**,**d**) Office database and gist and POC distances, respectively. The distance between map positions in the four experiments is 200 × 200 mm.

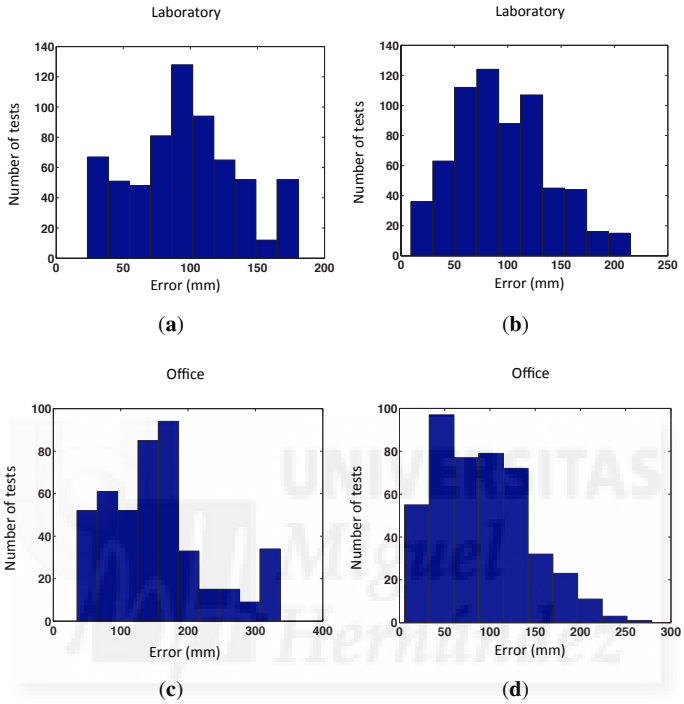


Figure 19. This figure represent the same experiments of Figure 18, adding random occlusions to the test image. (**a**,**b**) Laboratory database and gist and POC distances, respectively; (**c**,**d**) Office database and gist and POC distances, respectively. The distance between map positions in the four experiments is 200 × 200 mm.

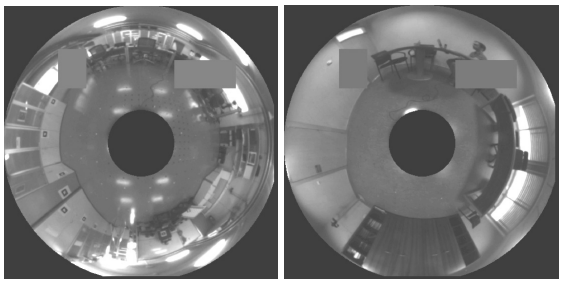


Figure 20. (**a**) Example of an omnidirectional image captured in our laboratory with two occlusions; (**b**) Example of an omnidirectional image captured in our office with two occlusions.

6. Discussion

Once we have presented the results, in this section, we provide a discussion of these results with both methods. We have arrived at some conclusions about the advantages and disadvantages of each method with different values of the variable parameters.

The first method (detecting the nearest image of the map) is a very good method if the robot has information of the positions in which the map images were taken. It permits carrying out an absolute localization, because the robot needs no information of its previous location. The results have demonstrated that it is a very reliable method to estimate the nearest position of the map, even when some changes are present (lighting conditions, new objects, *etc.*). As for the values of the parameters, the distance between map positions is the variable parameter in this method. The best choice is 200 $\times$ 200 mm, as it offers a good balance between accuracy and computational time.

The second method (local mapping and localization) is a good method if the robot does not know the map positions, because it creates a local map. This method also localizes the nearest images and the test image within this local map. As for the values of the parameters, first, the best choice is a distance between map positions of 200 $\times$ 200 mm, as it offers a good balance between accuracy and computational time; second, the best number of nearest neighbors is nine, because it leads to lower error; and finally, the kind of distance (POC or gist distance) should be chosen depending on the characteristics of the environment, because with the virtual database, the gist distance is better, but with the actual database of our university, the POC distance has presented more accurate results.

To finish, a comparative evaluation has been carried out between the proposed descriptor (Radon) and two other classical description methods in a localization task. The objective is to study the performance of the proposed descriptor, both in terms of localization accuracy and computational cost.

The description methods we use to make this comparison are the Fourier signature (FS) [11], a classical method to describe the global appearance of a panoramic scene, and SIFT [6], a typical method to extract and describe local features from a scene. The test consists of determining the nearest image from a set of omnidirectional images, and the database used is the laboratory (from the actual database captured at our university).

First, an experiment has been carried out to optimize the computational cost of the Radon transform (RT) method. With this aim, we repeat the localization experiment several times, using different resolutions, from RT $\in \mathbb{R}^{709x360}$ to RT $\in \mathbb{R}^{22x6}$. The results (success rate and computational cost) are shown on Figure 21. As shown, RT resolution can be reduced to 173×45 keeping the success rate, which clearly improves the computational cost of the process. This way, we choose this resolution to carry out the comparison.

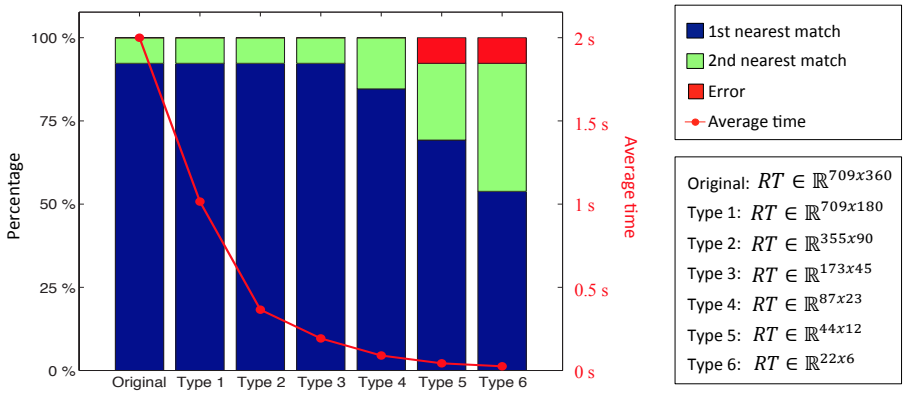


Figure 21. Success rate and average time using different types of Radon transforms with the laboratory database.

When using the Fourier signature, each omnidirectional scene is transformed to the panoramic format and described according to [11]. The Euclidean distance is used to compare two descriptors. On the other hand, when using SIFT, the SIFT features of the omnidirectional scenes are extracted, described and compared. The distance between two images im_1 and im_2 is defined in Equation (16).

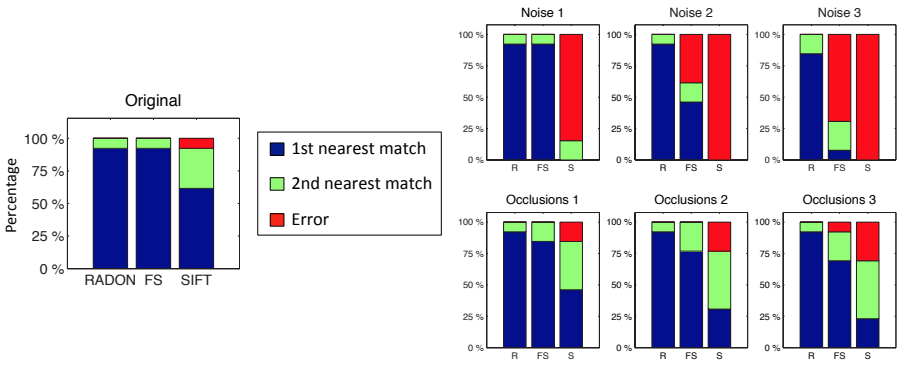
$$dist_{SIFT}(im_1, im_2) = \frac{1}{\sum_{i=1}^{n} \frac{1}{d_i}} \tag{16}$$

where d_i is the average distance between each SIFT feature in the first image and its corresponding feature in the second image. n is the number of matched SIFT features.

Figure 22a shows the success rate of the comparative experiment using the original test images. Thirteen random test images have been used to obtain this figure. Later, we repeated the experiment taking into consideration the presence of different levels of noise and occlusions on the test images, to test the robustness of each method under these typical situations. The results are shown in Figure 22b. Noises 1, 2 and 3 are random noises with maximum value equal to 20%, 40% and 60% of the maximum intensity. Occlusions 1, 2 and 3 imply that the test image has 10%, 20% and 30% occlusion, respectively.

As for the computational time used to do this experiment, the Radon transform spends 0.17 s; SIFT spends 12 s (using the algorithm of [6]). At last, the Fourier signature method needs 0.06 s.

Taking these results into account, the RT descriptor has been proven to be an effective method. It shows a reasonably good computational cost, and its robustness in the localization process is an important feature. While the FS and SIFT methods have worse behavior as noise and occlusions increase, RT has a good and stable behavior.



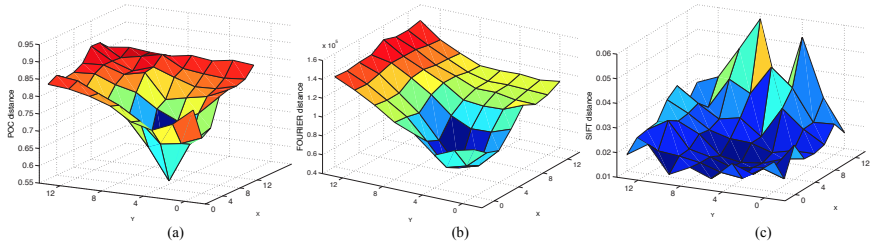
(**a**) (**b**)

Figure 22. Success rate with the laboratory database. (**a**) Using the original test images. (**b**) Taking into consideration the presence of different levels of noise and occlusions on the test images.

To explore the reason for this behavior, we depict in Figure 23 the distance between one sample test image and each image of the map. The test image is in the position $(X,Y) = (3,3)$. In this figure, the Radon descriptor stands out for its ability to localize the robot sharply since the distance increases quickly around the minimum. To check this performance, we have defined a parameter that measures the sharpness level of the minimum in each case (Equation (17)). The sharper the function, the higher is C.

$$C = \frac{mean(distance) - min(distance)}{max(distance) - min(distance)} \cdot 100(\%) \tag{17}$$

Using this equation with each descriptor, SIFT, FS and RT and carrying out 13 experiments in each case, the average value of C is: 60% using SIFT, 61% using FS and 79% using RT.



(a) (b) (c)

Figure 23. Distance between a sample test image captured at x = 3, y = 3 and all of the map images using (**a**) the Radon descriptor and POC distance, (**b**) Fourier signature (FS) and (**c**) SIFT.

7. Conclusions

In this paper we have presented two different methods to estimate the position of a robot in an environment in two different situations. In one of them, the robot knows the map positions, and in the other, it only knows the map images. The first method calculates the nearest position of the map with the test image taken by the robot, and the second method creates a local map with the nearest map images and the test image taken by the robot. These methods use omnidirectional images and global appearance descriptors.

We have also developed a method to build global appearance descriptors from omnidirectional images, based on Radon transform.

Both methods are first tested with our virtual database to determinate the best parameters, and they are then tested with actual databases. One of the methods has been analyzed with a database with changes in lighting conditions and adding objects. The other method has been analyzed with two different databases captured at our University.

Furthermore, a comparative analysis between the proposed descriptor and two other classical descriptors has been carried out. This analysis has shown the robustness of our method to solve the localization problem when the test images present noise or occlusions. These are common phenomena in real applications, and the RT method has been proven to be able to cope with them.

The results presented in this paper show the effectiveness of global appearance descriptors of omnidirectional images to localize the robot and to create a map. We are now working to improve these methods, and we are trying to carry out the localization in a map with more degrees of freedom. We are also working to include the color information of the images in the global appearance descriptors.

Acknowledgments

This work has been supported by the Spanish government through the project DPI 2013-41557-P: "*Navegación de Robots en Entornos Dinámicos Mediante Mapas Compactos con Información Visual de Apariencia Global*" and by the Generalitat Valenciana (GVa) through the project GV /2015/031: "*Creación de mapas topológicos a partir de la apariencia global de un conjunto de escenas*".

Author Contributions

The work presented in this paper is a collaborative development by all of the authors. Yerai Berenguer and Oscar Reinoso defined the research line. Mónica Ballesta and Yerai Berenguer carried out the preliminary experiments with descriptors and distances to tune the parameters. Yerai Berenguer, Luis Payá and Oscar Reinoso designed and implemented the mapping and localization algorithms. At last, Luis Payá, Yerai Berenguer and Mónica Ballesta carried out the mapping and localization experiments with the robotic platform.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Winters, N.; Gaspar, J.; Lacey, G.; Santos-Victor, J. Omni-directional vision for robot navigation. In Proceedings of the IEEE Workshop on Omnidirectional Vision, Hilton Head Island, SC, USA, 12 June 2000; pp. 21–28.
2. Valiente, D.; Gil, A.; Fernández, L.; Reinoso, O. A comparison of EKF and SGD applied to a view-based SLAM approach with omnidirectional images. *Robot. Auton. Syst.* **2014**, *62*, 108–119.
3. Maohai, L.; Han, W.; Lining, S.; Zesu, C. Robust omnidirectional mobile robot topological navigation system using omnidirectional vision. *Eng. Appl. Artif. Intell.* **2013**, *26*, 1942–1952.
4. Garcia-Fidalgo, E.; Ortiz, A. Vision-based topological mapping and localization by means of local invariant features and map refinement. *Robotica* **2015**, *33*, 1446–1470.
5. Garcia-Fidalgo, E.; Ortiz, A. Vision-based topological mapping and localization methods: A survey. *Robot. Auton. Syst.* **2015**, *64*, 1–20.
6. Lowe, D. Object Recognition from local scale-invariant features. In Proceedings of the International Conference on Computer Vision, Corfu, Greece, 20–27 September 1999; pp. 1150–1157.
7. Bay, H.; Tuytelaars, T.; Gool, L. SURF: Speeded up robust features. *Comput. Vis. ECCV* **2006**, *3951*, 404–417.
8. Chang, C.; Siagian, C.; Itti, L. Mobile robot vision navigation and localization using GIST and saliency. In Proceedings of the International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 4147–4154.
9. Payá, L.; Fernández, L.; Gil, L.; Reinoso, O. Map building and Monte Carlo localization using global appearance of omnidirectional images. *Sensors* **2010**, *10*, 11468–11497.
10. Wu, J.; Zhang, H.; Guan, Y. An efficient visual loop closure detection method in a map of 20 million key locations. In Proceedings of the IEEE International Conference on Robotics and Automation, Hong Kong, China, 31 May–7 June 2014; pp. 861–866.
11. 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.
12. Radon, J. Uber die bestimmung von funktionen durch ihre integralwerte langs gewisser mannigfaltigkeiten. *Ber. Verh. Sacks. Akad.* **1917**, *69*, 262–277.
13. Oliva, A.; Torralba, A. Building the gist of a scene: the role of global image features in recognition. In *Visual Perception Fundamentals of Awareness: Multi-Sensory Integration and High-Order Perception*; Elsevier: Amsterdam, The Netherlands, 2006; pp. 23–36.
14. Hoang, T.; Tabbone, S. A Geometric Invariant Shape Descriptor Based on the Radon, Fourier, and Mellin Transforms. In Proceedings of the 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 2085–2088.
15. Hasegawa, M.; Tabbone, S. A Shape Descriptor Combining Logarithmic-Scale Histogram of Radon Transform and Phase-Only Correlation Function. In Proceedings of the 2011 International Conference on Document Analysis and Recognition (ICDAR), Beijing, China, 18–21 September 2011; pp. 182–186.

16. Torralba, A.; Murphy, K.; Freeman, W.; Rubin, M. Context-based vision system for place and object recognition. In Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; pp. 273–280.
17. Siagian, C.; Itti, L. Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 300–312.
18. Oppenheim, A.; Lim, J. The importance of phase in signals. *IEEE Proc.* **1981**, *69*, 529–541.
19. Kuglin, C.; Hines, D. The phase correlation image alignment method. In Proceedings of the IEEE International Conference on Cybernetics and Society, San Francisco, CA, USA, 23–25 September 1975; pp. 163–165.
20. Kobayashi, K.; Aoki, T.; Ito, K.; Nakajima, H.; Higuchi, T. A fingerprint matching algorithm using phase-only correlation. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2004**, *87*, 682–691.
21. Seber, G.A.F. *Multivariate observations*; John Wiley & Sons, Inc: New York, NY, USA, 1984.
22. Ishiguro, H.; Tsuji, S. Image-based memory of environment. In Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems' 96 (IROS 96), Osaka, Japan, 4–8 November 1996; pp. 634–639.
23. Menegatti, E.; Maeda, T.; Ishiguro, H. Image-based memory for robot navigation using properties of omnidirectional images. *Robot. Auton. Syst.* **2004**, *47*, 251–267.
24. Vardy, A.; Möller, R. Biologically plausible visual homing methods based on optical flow techniques. *Connect. Sci.* **2005**, *17*, 47–89.

Hindawi Publishing Corporation
Journal of Sensors
Volume 2016, Article ID 1209507, 21 pages
http://dx.doi.org/10.1155/2016/1209507

Research Article

Using Omnidirectional Vision to Create a Model of the Environment: A Comparative Evaluation of Global-Appearance Descriptors

L. Payá, O. Reinoso, Y. Berenguer, and D. Úbeda

Department of Systems Engineering and Automation, Miguel Hernández University, Avenida de la Universidad s/n, Elche, 03202 Alicante, Spain

Correspondence should be addressed to L. Payá; lpaya@umh.es

Received 23 October 2015; Accepted 11 February 2016

Academic Editor: Yassine Ruichek

Nowadays, the design of fully autonomous mobile robots is a key discipline. Building a robust model of the unknown environment is an important ability the robot must develop. Using this model, this robot must be able to estimate its current position and to navigate to the target points. The use of omnidirectional vision sensors is usual to solve these tasks. When using this source of information, the robot must extract relevant information from the scenes both to build the model and to estimate its position. The possible frameworks include the classical approach of extracting and describing local features or working with the global appearance of the scenes, which has emerged as a conceptually simple and robust solution. While feature-based techniques have been extensively studied in the literature, appearance-based ones require a full comparative evaluation to reveal the performance of the existing methods and to tune correctly their parameters. This work carries out a comparative evaluation of four global-appearance techniques in map building tasks, using omnidirectional visual information as the only source of data from the environment.

1. Introduction

During the last years, the presence of mobile robots in both industrial and household environments has increased substantially since they are able to solve many different tasks. The expansion of robots into such environments and applications has been eased thanks to the development of their abilities in perception, computation, autonomy, and adaptability to different circumstances. As far as perception is concerned, the robots must be equipped with sensors that allow them to extract the necessary information from the environment to be able to carry out autonomously their tasks. Vision sensors have gained popularity because they present some interesting advantages such as providing a big quantity of information with a relatively low cost and low power consumption (comparing to other sensors, such as laser rangefinders) and stable data both outdoors and indoors (unlike GPS, whose signal is prone to degradation indoors). They also permit carrying out additional high level tasks, such as people detection and recognition. Among vision sensors, catadioptric systems have extended in recent years as they are able to capture images with a field of view of 360 deg. around the robot [1]. In our approach, the mobile robot is equipped with a catadioptric system on it, which captures images from the environment. Using this information, the objective is building a robust model of the environment. In general, these models can be represented as a metric, a topological, or a hybrid map [2]. First, metric maps define the position of some relevant features of the environment with respect to a coordinate system and permit robot localization with geometric accuracy (except for a relative error) [3]. Second, topological maps often represent the environment as a graph where nodes are distinctive localizations (e.g., rooms) of the environment and links are the connectivity relationships between localizations. Usually, such maps do not permit fine localization but they are enough to estimate the position of the robot and to navigate to the desired localizations [4]. At last, hybrid maps are hierarchical models where information is arranged in multiple levels. Usually, there is a high level of topological information that allows an

approximate localization (in an area of the environment) and several low levels of metric information that permit refining the localization (in the previously detected area) [5].

In all cases, to build a functional map, it is necessary to extract relevant information from the scenes. Traditionally, researchers have focused on methods that extract some outstanding landmarks or interest points from the scenes and describe them with any robust description method. These methods have become popular in map building and mobile robots localization. For example, Angeli et al. [6] make use of SIFT features [7] to solve the mapping and global localization problems simultaneously (SLAM) and Valgren and Lilienthal [8] and Murillo et al. [9] make use of SURF features [10] to solve the localization problem in a previously created model. Using feature-based approaches in combination with probabilistic techniques, it is possible to build metric maps [3]. However, these methods present some drawbacks; for example, it is necessary that the environment be rich in prominent details (otherwise, artificial landmarks can be inserted in the environment, but this is not always possible); also, the detection of such points is sometimes not robust against changes in the environment and their description is not always invariant to changes in robot position and orientation. Besides, camera calibration is crucial in order to incorporate new measures in the model correctly. This way, small deviations in either the intrinsic or the extrinsic parameters add some error to the measures. At last, extracting, describing, and comparing landmarks are computationally complex processes that often make building the model in real time unfeasible, as the robot explores the environment.

In contrast, global-appearance techniques have gained relevance in more recent works [11–13]. These techniques are useful when the robot moves within unstructured environments where extracting and describing robust points is difficult. These approaches lead to conceptually simpler algorithms since each scene is described by means of a unique descriptor. Map creation and localization can be achieved just storing and comparing pairwise these descriptors. As a drawback, extracting metric relationships from this information is difficult; thus, this family of techniques is usually employed to build topological maps (unless the visual information is combined with other sensory data, such as odometry). Despite their simplicity, several difficulties must be faced when using these techniques. Since no local information is extracted from the scenes, it is necessary to use any compression and description method that make the process computationally feasible. These descriptors do not present invariance to changes neither in the robot orientation nor in the lighting conditions or other changes in the environment (position of objects, doors, etc.). They will also suffer problems in environments where *visual aliasing* is present, which is a common phenomenon in indoor environments with repetitive visual structures.

Many algorithms can be found in the literature working both with local features and with global appearance of images. All these algorithms imply many parameters that have to be correctly tuned so that the mapping and localization processes are correct. Feature-based approaches have reached a relative maturity and some comparative evaluations have been carried out, such as [14]. These evaluations are useful to choose the most suitable extractor and descriptor to a specific application. However, global-appearance-based approaches are still a field that is worth deeper exploration. We have not found any work that makes a comparative evaluation of the performance of such descriptors in mapping tasks. This is the main objective we propose in this paper. We have selected four accepted global-appearance description methods, adapted them to be used with omnidirectional visual information, and studied their properties. Then, we have developed the necessary algorithms to create a model of the environment, tested their performance, and studied the influence of the most relevant parameters.

The remainder of the paper is structured as follows. Section 2 presents briefly the description approaches that will be evaluated along the paper. After that, Section 3 describes the kind of models of the environment we will build to test the performance of the approaches. Then, Section 4 details the set of experiments designed and the results obtained. To finish, a final discussion is carried out in Section 5.

2. Global-Appearance Descriptors: State of the Art

This section outlines some methods to describe the global appearance of images. Four families of methods are proposed to be analysed: methods based on the discrete Fourier transform (Section 2.1), on principal components analysis (Section 2.2), on orientation gradients (Section 2.3), and on the essence of the scenes (Section 2.4). These are the description methods whose performance will be evaluated along the paper.

2.1. Methods Based on the Discrete Fourier Transform (DFT). The Discrete Fourier Transform (DFT) is a classical method to describe scenes that presents some interesting features. When the two-dimensional DFT of a scene $\mathrm{im}(x, y)$ is calculated, the result is a complex function $\mathrm{IM}(u, v)$ in the frequency domain (u and v are the frequency variables) that can be decomposed into magnitude and argument matrices. The first matrix (also known as amplitude spectrum) represents the distribution of spatial frequencies within the image (i.e., it contains information on the overall structure of the image: edges orientation, smoothness, width, etc.). On the other hand, the argument matrix contains information about the local properties of the scene (shape and position of the objects). Taking these facts into account, the amplitude spectrum can be used as a global descriptor of the scene, since it contains information about the dominant structural patterns and it is invariant to the distribution of the objects. This information has proved to be relevant to solve simple classification tasks [15]. However, this kind of descriptors has no information about the spatial relationships between the structures in the image. To have a complete description of the scene, such information must be included.

Considering it, we have opted for a formulation of the DFT which contains complete information. This formulation is the Fourier Signature (FS), described first in [12]. It is defined as the matrix composed of the 1D DFT of each row

in the original image. When applied to panoramic scenes, it offers rotational invariance. When we calculate the FS of a panoramic image $\text{im}(x, y) \in \mathbb{R}^{N_x \times N_y}$, we arrive to a new matrix $\text{IM}(u, y) \in \mathbb{C}^{N_x \times N_y}$, where the main information is concentrated in the low frequency components of each row (so we can retain only the k first columns, having a compression effect). This new matrix with N_x rows and k columns can be decomposed in a magnitude matrix $A_j(u, y) = |d_j(u, y)|$ with N_x rows and k_1 columns and an argument matrix $\Phi_j(u, y)$, with N_x rows and k_2 columns.

Based on the shift property of the DFT, when two panoramic images have been captured from the same position but have the robot different orientations, both images have the same magnitude matrix and the arguments matrices permit obtaining the relative robot orientation. This property allows us to use the magnitude matrix to estimate the position of the robot (as it presents rotational invariance) and, then, the arguments matrix to estimate the relative orientation of the robot.

2.2. Methods Based on Principal Components Analysis (PCA). Panoramic images are data that fall in a space with a very high number of dimensions. However, the image pixels tend to be very correlated data, since they have been captured from a 3DOF process (robot pose on the ground plane). Taking this fact into account, a natural way to compress the information is principal components analysis (PCA), as shown in [16]. This kind of descriptors has evolved from the original formulation to adapt them to be used in mapping and localization tasks. The works of Leonardis and Bischof [17] show some examples of how this analysis can be used to mobile robots localization in a robust way.

When we have a set of n images $\text{im}_j(x, y) \in \mathbb{R}^{N_x \times N_y}$, $j = 1, \ldots, n$, each image can be considered a point in a space with $N_x \cdot N_y$ dimensions, $\vec{x}_j(i) \in \mathbb{R}^{N_x \cdot N_y \times 1}$, $j = 1, \ldots, n$, ($n \ll N_x \cdot N_y$). Using the classical formulation of PCA, it is possible to transform each point $\vec{x}_j(i)$, in a new data point, namely, *image projection* $\vec{p}_j(i) \in \mathbb{R}^{k_3 \times 1}$ $j = 1, \ldots, n$, where k_3 is the number of PCA features that contain the most relevant information, $k_3 \leq n$. Turk and Pentland [18] show how the necessary transformation matrix $\mathbf{V}$ can be obtained in an efficient way. They make use of the SVD of the data matrix covariance, retaining only the eigenvectors with higher eigenvalues. If the number of eigenvectors is equal to n, then there is no loss of information during the compression process [16]. Thus, after applying PCA techniques, images can be handled efficiently, with a low computational cost. However, depending on the images' size, the process to obtain $\mathbf{V}$ may be substantially slow.

The use of PCA in mapping and localization tasks is limited since the image projections depend on the robot orientation. Independently, on using omnidirectional scenes, the images projections contain only information of the position and orientation the robot had when capturing the images. This is the reason why Jogan and Leonardis developed the concept of *eigenspace of spinning images* [19]. This model uses specific properties of panoramic images to obtain, in an efficient way, an optimal subspace that takes into account the different orientations a robot may have when capturing each image. The method takes profit of the symmetry properties the data matrix presents when we add the rotations information. This method has the advantage of permitting the estimation of the robot orientation, but the computational cost to obtain the transformation matrix $\mathbf{V}$ is extremely high. By this reason, it has been only used with small environments, with a limited number of images.

2.3. Methods Based on the Histogram of Oriented Gradients (HOG). HOG is a description method used traditionally in object detection. This technique considers the gradient orientation in localized parts of a scene. The method outstands by its simplicity, good computational cost, and relatively good results in object recognition tasks. It was initially described by Dalal and Triggs [20], who used it in people detection tasks. Later on, some researchers developed an improved version of the algorithm both in detection accuracy and in computational cost [21].

However, the experience with HOG descriptors in the mobile robotics field is limited to simple and small environments. Few previous works have made use of HOG in robot mapping and localization. Hofmeister et al. [22] use this descriptor in small robots localization tasks, with low resolution images and small environments not prone to *visual aliasing*. Under these limited conditions, the algorithm works well.

HOG is not defined as a global-appearance descriptor because the basic implementation consists in dividing the scene in a set of cells and obtaining a histogram of gradient orientation using the pixels information in each cell. The combination of all these histograms is the image descriptor. We have redefined the algorithm to obtain a unique descriptor per image that contains information of the global appearance of this image. The version of HOG we consider is described in [23], where a global version of HOG is used to carry out mapping and Monte-Carlo localization in large environments. Anyway, it is necessary to make an evaluation of the performance of this algorithm and systematize it in map creation tasks.

2.4. Methods Based on Gist and Prominence. The *gist* concept was first introduced by Oliva and Torralba [24], with the idea of creating a low-dimension scene descriptor, and avoiding segmentation and processing of points, objects, or individual regions. They inspired by some works that suggested that humans recognize scenes by codifying the global configuration and just ignoring most of the details and individual objects [25].

More recently, some works make use of the *prominence* concept together with *gist*. It refers to regions of pixels that stand out with respect to the neighbor regions, in contrast to *gist*, which implies the accumulation of statistical data from the whole image. Siagian and Itti [26] try to establish a synergy between the two concepts and they design a unique descriptor that takes both into account. This descriptor is built using the intensity, orientation and color information.

The experience with this kind of descriptors in mobile robots applications is limited. For example, Chang et al. [27] present a localization and navigation system based on *gist* and prominence and Murillo et al. [28] make use of *gist*

descriptors in a localization problem. However, they obtain these descriptors using specific regions in a set of panoramic images.

Like HOG, *gist* is not primarily defined as a global-appearance descriptor and we have redefined the algorithm to obtain a unique descriptor per image. The version of *gist* we consider in this evaluation is described in [23] and is built from orientation information, analysed in some resolution levels.

3. Creating a Visual Topological Map of the Environment

In this section we focus on the map creation problem. The robot, which is equipped with a catadioptric vision system on its top, explores the environment to map to cover it completely. During this process, the robot captures a set of omnidirectional scenes from several positions. Only this visual information will be used to build the map (neither odometry nor laser or other sensory data will be used). This way, the final model will be a topological map since it contains some localizations (represented as panoramic scenes) and connectivity relations, but no metric data. In Section 3.1, we describe how the nodes of the map are represented with each description method and, in Section 3.2, the process to add connections between the nodes is outlined.

3.1. Using Global-Appearance Descriptors to Create a Model of the Environment. Let us suppose that the mobile robot has gone across the environment to map (either in a teleoperated way or autonomously, following any exploration algorithm) and has captured a set of omnidirectional images $I = \{\text{im}_1, \text{im}_2, \ldots, \text{im}_n\}$, where $\text{im}_j \in \mathbb{R}^{N_x \times N_y}$.

From this set of images, a set of descriptors, one per original scene, is calculated. As a result, the nodes of the map will be a set of descriptors $D = \{d_1, d_2, \ldots, d_n\}$ where, in general, $d_j \in \mathbb{C}^{M_x \times M_y}$. With the objective that these nodes are functional, it is necessary that d_i contains information that permits estimating the position of the robot when capturing im_i (taking into account that the robot may have any orientation in this position). In the next subsections, we detail the kind of information each d_i should contain when using each description method.

3.1.1. DFT Descriptor. Each node d_i contains two matrices: the magnitudes one $A_i(u, y) \in \mathbb{R}^{N_x \times k_1}$ and the arguments matrix $\Phi_i(u, y) \in \mathbb{R}^{N_x \times k_2}$. k_1 is the number of columns we retain in the localization descriptor and k_2 is the number of columns retained in the orientation descriptor. The higher k_1 and k_2, the more information the descriptor contains. However, we must take into account that the main information is concentrated in the low frequency columns, and if noise is present on the image, it will affect high frequency components mostly; thus, removing these components may imply an additional benefit. The effect of both parameters in a mapping process will be evaluated.

3.1.2. PCA Descriptor. The PCA descriptor we use is proposed in the works of Jogan and Leonardis [19]. This model uses the specific properties of panoramic images to create a set of N_R spinning images from each of the n original panoramic images, so we get N_R data vectors per original image. To obtain the transformation matrix $\mathbf{V}$, the similarities among the rotated versions of each image are taken into account. This permits decomposing the original problem (which is computationally very heavy) in a set of lower order problems.

As a result of the process, the map will be composed of (a) a set of descriptors $\vec{p}_j(i) \in \mathbb{R}^{k_3 \times 1}$ $j = 1, \ldots, n$, which are the projections of the original panoramic images and contain information on the robot position, (b) a set of phase vectors, $\vec{\phi}_j \in \mathbb{R}^{k_3 \times 1}$, one per image, which contain information of the robot orientation, and (c) a unique transformation matrix $\mathbf{V} \in \mathbb{C}^{k_3 \times N_x \cdot N_y}$. k_3 is the number of eigenvector chosen. The higher the k_3, the more the information that the map contains. If $k_3 = n$, there is no loss of information.

3.1.3. HOG Descriptor. Each image will be described through two HOG descriptors. The first one, $\vec{h}_1$, is the position descriptor and is invariant against rotations of the robot. To obtain it, the panoramic image is divided into horizontal cells, whose width is equal to N_y (number of columns in the image) and whose height can be configured freely. The size of $\vec{h}_1$ is $1 \times k_4 \cdot b$, where k_4 is the number of horizontal cells and b is the number of bins in each orientation histogram. The second one, $\vec{h}_2$, is the orientation descriptor. To obtain it, the panoramic image is divided into vertical cells whose height is equal to N_x. Some overlap between these cells may exist. If the width of the cells is l_1 and the distance between consecutive cells d_1, then the number of vertical cells is $k_5 = N_y/d_1$. The size of the orientation descriptor $\vec{h}_2$ is then $1 \times k_5 \cdot b$. In the experiments, the influence of k_4, k_5, and b will be evaluated.

Figure 1 shows, from a panoramic image whose gradient has been calculated, the process to obtain both descriptors: (a) $\vec{h}_1$ and (b) $\vec{h}_2$.

3.1.4. Gist and Prominence Descriptor. The information of the orientation of the edges in the image is used to build the descriptor. First, two versions of each image are considered: the original one and a new version after applying a Gaussian low-pass filter and subsampling to a new size $0.5N_x \times 0.5N_y$. Second, both images are filtered with a bank of m Gabor filters with orientations evenly distributed between 0 and 180 deg. Third, to reduce the amount of information, the pixels in each resulting image are grouped into blocks. The block division is carried out in a similar fashion as in HOG: a position descriptor $\vec{g}_1$ is obtained by defining k_6 horizontal blocks and an orientation descriptor $\vec{g}_2$ is calculated with k_7 vertical blocks (with overlapping). In the experiments, the influence of k_6, k_7, and m will be evaluated. Figure 2 shows, from a panoramic image, the process to obtain $\vec{g}_1$.

To sum up, Table 1 shows the parameters to be tuned in each description method included in the evaluation. On the other hand, Table 2 gives details of the contents of the map when we consider each description method.

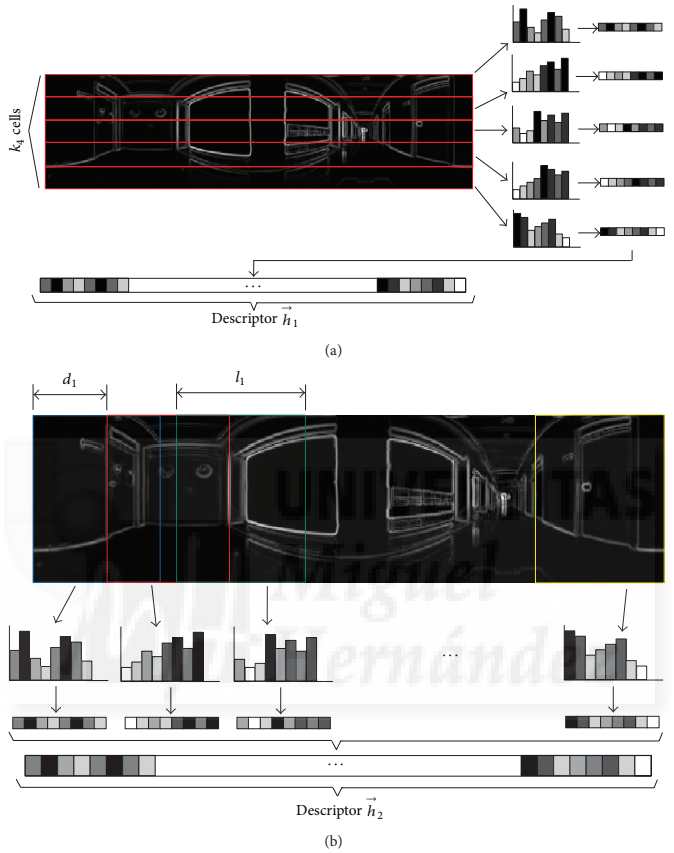


FIGURE 1: Process to obtain (a) the HOG position descriptor $\vec{h}_1$ and (b) the HOG orientation descriptor $\vec{h}_2$.

3.2. Adding Topological Relations. Our starting point is a set of images captured from unknown positions. The objective of this section consists in designing an algorithm that allows us to establish adjacency relations among them, with the goal of creating a topological map. Apart from this, we expect the distribution of the nodes in this map to be similar to the distribution of the points where the images were captured. It goes beyond the classical concept of topological map since besides adjacency it also introduces the concepts of closeness and farness. Thanks to this kind of maps, the robot will be able to plan its trajectory more accurately.

To create such a map, a method based on a mechanical system of forces is used. This kind of methods has been used often to simulate the movement of flexible bodies, as in [29], where the body is discretized into a set of particles, and the interaction among them is modelled with a set of springs. Our framework also includes a set of dampers in parallel with the springs, since the dampers can help to achieve an overdamped behaviour that facilitates reaching the steady state.

The idea we develop consists in considering each image a particle which is linked to the rest of images (particles) through a pair spring-damper, where the natural length of each spring is equal to the distance between the descriptors of the two images linked by this spring. The particles start their evolution from random positions. If we let the forces produced by springs and dampers move freely in the system until it tends to a minimum energy position, we expect the distribution of particles to be similar to the distribution

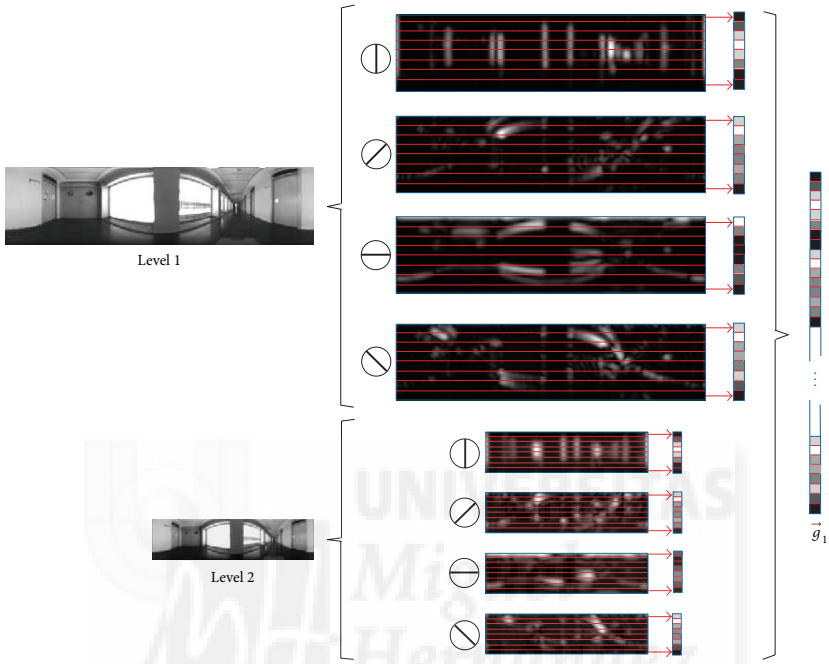


FIGURE 2: Process to build the *gist* position descriptor $\vec{g}_1$ ($m = 4$ orientations, $k_6 = 8$ blocks).

of capture points. The algorithm we use is inspired by the algorithm presented by Menegatti et al. [12], who used it in small environments.

3.2.1. Mass-Spring-Damper Method. Each image is considered a particle P_j, $j = 1, \ldots, n$, with mass m_j, where n is the number of images to include in the map. No information about the coordinates of the capture points is available.

Each pair of particles P_k and P_l is linked with a spring S_{kl} with elastic constant k_{kl} and a damper with damping constant κ_{kl}. The natural length of each spring l^0_{kl} is equal to the distance between the descriptors of the images associated with the particles P_k and P_l.

The initial positions of the particles are randomly initialised. After that, the system is allowed to evolve freely until it reaches a steady state. At this state, the distribution of the particles is expected to be similar to the distribution of capture points (except for a scale factor and a rotation). This way, the result is a scaled model of the real distribution. We consider the value of the elastic constants to be proportional to the distance between images and, from a threshold distance, the images are not linked by any spring.

Under these circumstances, the spring and damper linking each pair of particles P_i and P_j make on these particles the force:

$$\vec{f}_{ij} = -\left[-k_{ij} \cdot \left(\left|\vec{p}_i - \vec{p}_j\right| - l^0_{ij}\right) + \kappa_{ij} \cdot \left(\frac{\left(\vec{v}_i - \vec{v}_j\right) \cdot \left(\vec{p}_i - \vec{p}_j\right)}{\left|\vec{p}_i - \vec{p}_j\right|}\right)\right] \cdot \frac{\left(\vec{p}_i - \vec{p}_j\right)}{\left|\vec{p}_i - \vec{p}_j\right|}, \quad (1)$$

$$\vec{f}_{ji} = -\vec{f}_{ij},$$

where $\vec{p}_i, \vec{v}_i$ are the position and speed of the ith particle, respectively. Then, the resulting force on each particle is obtained:

$$\vec{F}_i = \sum_{j=\{1,\ldots,n\}, j \neq i} \vec{f}_{ij}. \quad (2)$$

Table 1: Parameters to be tuned in each description method.

Descriptor	Parameters
FS	$k_1 \Rightarrow$ components per row, localization descriptor $\mathbf{A}_j$ $k_2 \Rightarrow$ componentes per row, orientation descriptor Φ_j
r-PCA	$k_3 \Rightarrow$ number of eigenvectors $N_R \Rightarrow$ number of rotations per panoramic image
HOG	$b \Rightarrow$ number of bins per histogram $k_4 \Rightarrow$ number of horizontal cells, localization descriptor $\vec{h}_1$ $l_1 \Rightarrow$ width of vertical cells, orientation descriptor $\vec{h}_2$ $d_1 \Rightarrow$ distance between vertical cells, orientation descriptor $\vec{h}_2$ $k_5 = N_y/d_1 \Rightarrow$ number vertical cells, orientation descriptor $\vec{h}_2$
Gist	$m \Rightarrow$ number of orientations (Gabor filters), localization descriptor $\vec{g}_1$ $k_6 \Rightarrow$ number of horizontal blocks, localization descriptor $\vec{g}_1$ $l_2 \Rightarrow$ width of vertical blocks, orientation descriptor $\vec{g}_2$ $d_2 \Rightarrow$ distance between vertical blocks, orientation descriptor $\vec{g}_2$ $k_7 = N_y/d_2 \Rightarrow$ number of vertical blocks, orientation descriptor $\vec{g}_2$

Table 2: Contents of the map, relative to localization and orientation estimation, per image included in the model im_j, $j = 1, \ldots, n$.

Descriptor	Localization	Orientation
FS	$\mathbf{A}_j \in \mathbb{R}^{N_x \times k_1}$	$\Phi_j \in \mathbb{R}^{N_x \times k_2}$
r-PCA	$\mathbf{V} \in \mathbb{C}^{k_3 \times N_x \cdot N_y}$ $\vec{p}_j \in \mathbb{C}^{k_3 \times 1}$	$\vec{\phi}_j \in \mathbb{R}^{k_3 \times 1}$
HOG	$\vec{h}_{1j} \in \mathbb{R}^{k_4 \cdot b \times 1}$	$\vec{h}_{2j} \in \mathbb{R}^{k_5 \cdot b \times 1}$
Gist	$\vec{g}_{1j} \in \mathbb{R}^{2 \cdot k_6 \cdot m \times 1}$	$\vec{g}_{2j} \in \mathbb{R}^{k_7 \cdot m \times 1}$

From this resulting force, the acceleration of the particle is obtained from the 2nd Newton's law:

$$\vec{a}_i(t) = \frac{\vec{F}_i}{m_i}, \tag{3}$$

where $\vec{a}_i(t)$ is the ith particle acceleration at time instant t, $\vec{F}_i$ is the resulting force on particle i, and m_i is the ith particle mass. From this acceleration, the speed and position of particle i once it passed a period of time Δt can be calculated:

$$\begin{aligned} \vec{v}_i(t + \Delta t) &= \vec{v}_i(t) + \vec{a}_i(t) \cdot \Delta t, \\ \vec{p}_i(t + \Delta t) &= \vec{p}_i(t) + \vec{v}_i(t) \cdot \Delta t. \end{aligned} \tag{4}$$

This method, known as Euler integration, may not be stable if the step time is not low enough, which would increase the computational cost of the process. This is the reason why the Verlet integration is sometimes suggested. In this integration method, the position and speed are updated at each iteration with the following expressions.

At the time instant $t = \Delta t$,

$$\begin{aligned} \vec{p}_i(\Delta t) &= \vec{p}_i(0) + \vec{v}_i(0) \cdot \Delta t + 0.5 \cdot \vec{a}_i(0) \cdot \Delta t^2, \\ \vec{v}_i(\Delta t) &= \vec{v}_i(0) + \vec{a}_i(0) \cdot \Delta t. \end{aligned} \tag{5}$$

From this time instant,

$$\begin{aligned} \vec{p}_i(t + \Delta t) &= 2 \cdot \vec{p}_i(t) - \vec{p}_i(t - \Delta t) + \vec{a}_i(t) \cdot \Delta t^2, \\ \vec{v}_i(t + \Delta t) &= \frac{\vec{p}_i(t + \Delta t) - \vec{p}_i(t)}{2 \cdot \Delta t}. \end{aligned} \tag{6}$$

4. Experiments

In this section, we compare the performance of the four description methods. First, we describe the sets of images we have used to carry out the experiments. Then, the evaluation is carried out from several points of view to fully uncover the goodness of each method in mapping tasks. We analyse the computational cost of the mapping process, the relationship between the image distance and the geometric distance, and the performance in topological map building.

4.1. Sets of Images. To carry out the experiments, we make use of two sets of images, captured with two different catadioptric systems. First, set 1 has been captured by us in a building of Miguel Hernández University (Spain). The images were captured along 6 different rooms in an office-like environment. Figure 3(a) shows a bird's eye view of this environment. The database is composed of 873 panoramic 64 × 256-color images which have been captured on a dense 40 × 40 cm grid of points (red points in Figure 3(a)). Set 1 [30] is a challenging database due to the tendency to visual aliasing that presents the environment. There are many zones which, despite being geometrically far, present a similar visual appearance. Also, the images were captured in different times of day (changing lighting conditions) and the positions of some objects in the scenes are modified (e.g., changes in the state of doors). All the images were captured with an *Imaging Source DFK 21BF04* camera mounted on a *Pioneer P3-AT* robotic platform. The camera takes pictures of a hyperbolic mirror (*Eizoh Wide 70*) which is mounted on it with its axis aligned with the camera optic axis. The resulting omnidirectional images are transformed with a cylindrical projection to obtain their panoramic versions. The *P3-AT* robot has 4 drive wheels. Its maximum linear speed is equal to 0.7 m/s, its maximum turning speed is equal to 140 deg/s, and the minimum turning radius is null. The robot can move freely on the floor so the image capture process has 3 degrees of freedom: position on the ground floor with respect to a world coordinate system (x, y) and orientation with respect the z-axis (θ). Figure 3(b) shows the robot, the catadioptric system, and a sample image (omnidirectional and panoramic formats).

The second set of images has been captured by a third party [31]. It is composed of a set of panoramic grayscale images, captured in several rooms of a university and a

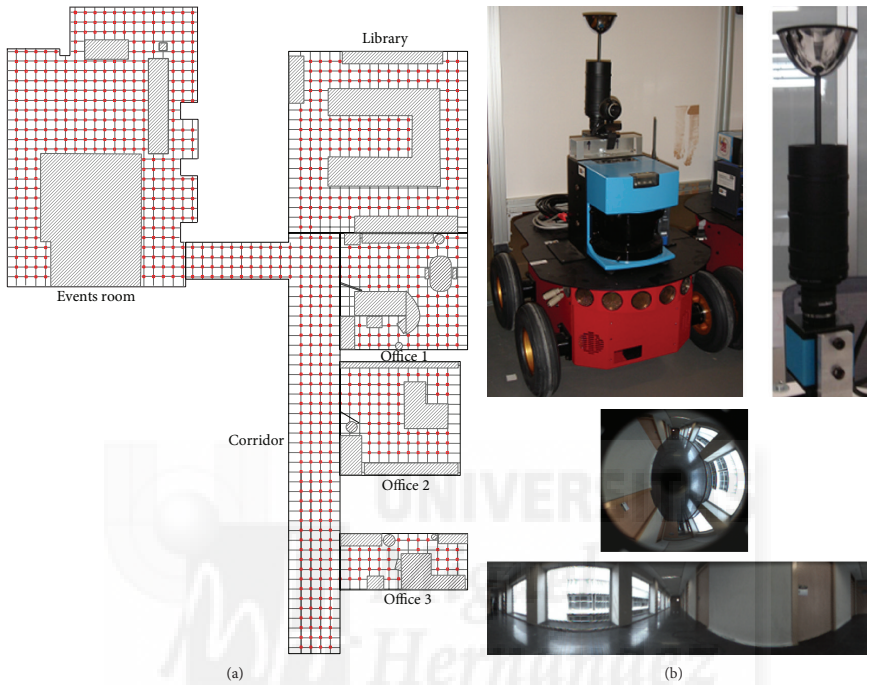


Figure 3: (a) Bird's eye view of the environment where set 1 was captured. (b) Catadioptric system mounted on the robot and sample scene captured in the corridor (omnidirectional and panoramic formats).

Table 3: Images set 2: rooms considered in the experiments and main parameters.

Room	Number of images	Grid	Size	Resolution
Laboratory	170	30 × 30 cm.	3 × 5 m	81 × 561 pixels
Corridor	200	50 × 50 cm.	5 × 10 m	81 × 561 pixels
Kitchen	108	10 × 10 cm.	1.2 × 1.0 m	81 × 583 pixels
Hall	242	10 × 10 cm.	2.2 × 1.2 m	81 × 583 pixels

flat. They were captured with a camera *ImagingSource DFK 4303* mounted on the robot *ActivMedia Pioneer 3-DX*. The hyperbolic mirror is the model *Accowle Wide View*. This is an interesting database because it presents different grid sizes in each room. It permits testing how this parameter influences the performance of the methods. Table 3 shows the rooms we have used and the main features of the images.

4.2. Computational Cost. Previously, Section 3.1 has outlined the contents of the map nodes. Now, the objective of this section consists in making a comparative evaluation of the computational cost of the four description methods during the creation of the map nodes. This study will be carried out depending on the value of the most relevant parameters of each description method. Data set 1 is used to carry out this comparative evaluation. This is an interesting study as it allows us to know which algorithms could work in real time.

First, Figure 4 shows the computation time using (a) FS versus k_1 and k_2, and (b) rotational PCA versus N_R. Second, Figure 5 shows the time when using HOG versus k_4, l_1, and d_1. At last, Figure 6 shows *gist* with $m = 8$ versus k_6, l_2, and d_2. In all cases, the time per image is depicted. The total time to build the map can be obtained by multiplying by 873 (number of images in set 1).

In the case of FS, as k_1 and k_2 increase, the time increases slightly. The cost to obtain the DFT of each row is the same, the difference is in the need of computing the magnitude and argument of a different number of components, which implies a low computational cost. In any case, the computational cost of FS is very low.

As far as rotational PCA is concerned, Figure 4 shows how the time increases exponentially as N_R does, arriving at up to 110 seconds per image (27 hours to build the whole

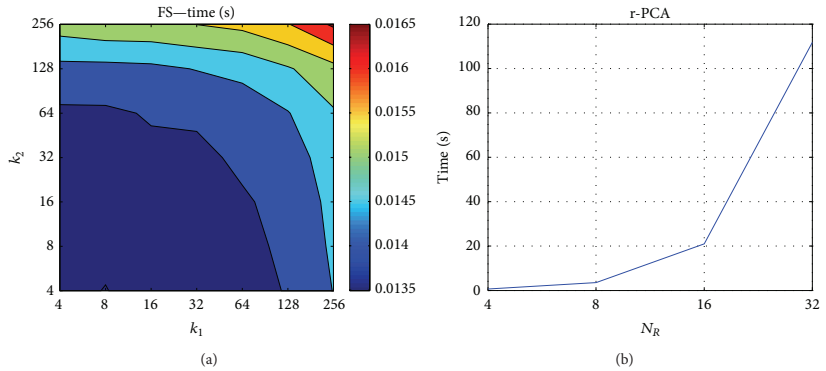


Figure 4: Computational cost to obtain the nodes' descriptors using (a) Fourier Signature and (b) rotational PCA.

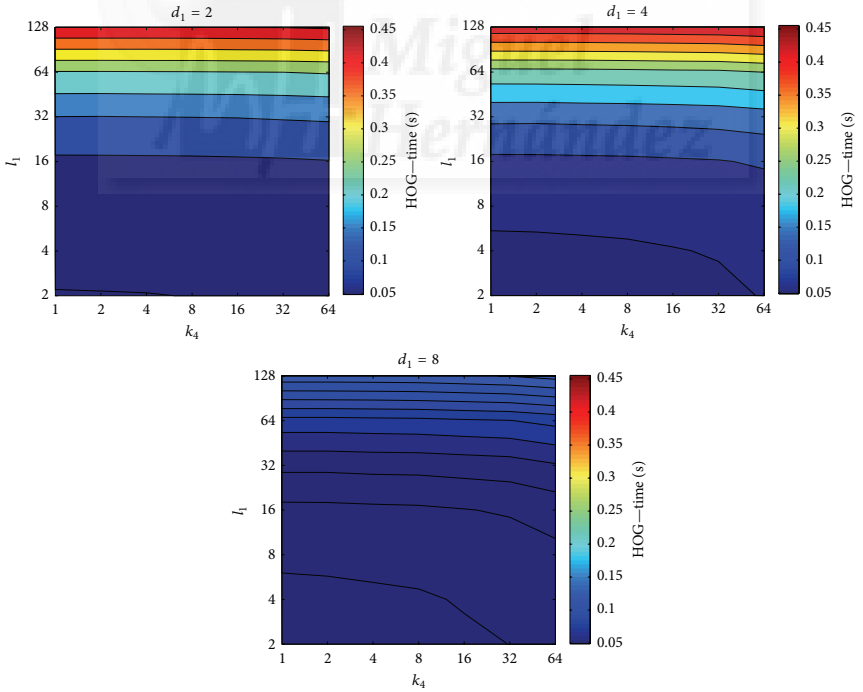


Figure 5: Computational cost to obtain the nodes' descriptors using HOG.

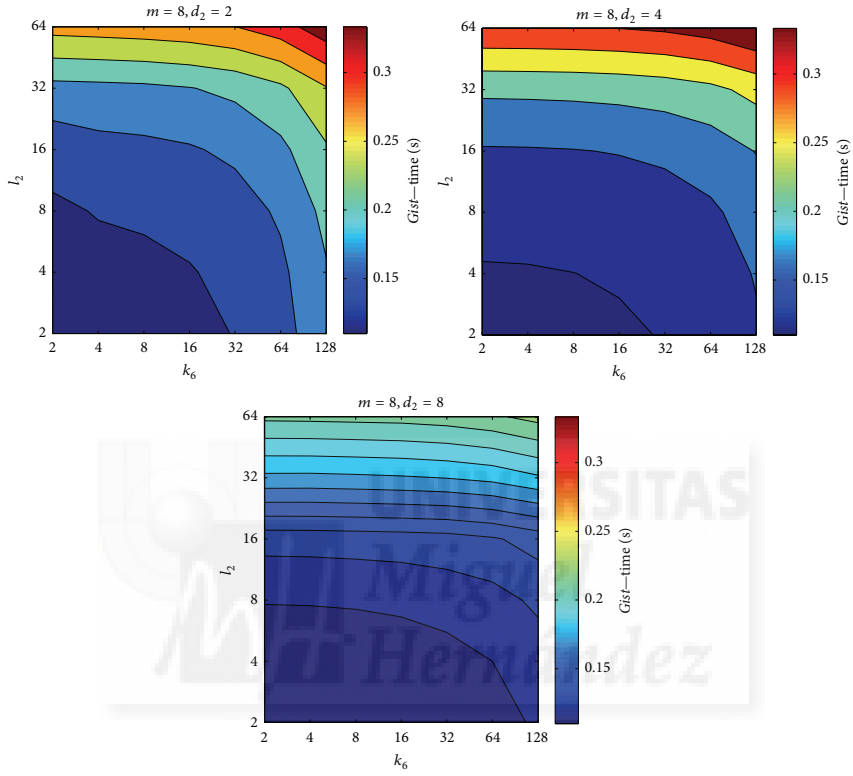


Figure 6: Computational cost to obtain the nodes' descriptors using *gist*.

map), when $N_R = 32$ rotations. It has been impossible to consider a higher number of rotations due to the enormous requirements of memory during the process.

If we analyse now HOG, on the one hand, the influence of k_4 is low and, on the other hand, time increases linearly when l_1 does. At last, when d_1 increases, the time decreases as fewer vertical cells are considered. In general, HOG presents a substantially higher computational cost compared to FS; despite it, the algorithm is quick enough to permit carrying out the mapping process in real time, as the robot explores the unknown environment.

At last, the computational cost of *gist* is, in general, approximately 10 times the cost of FS and similar to HOG. All of FS, HOG, and *gist* are computationally feasible algorithms. Nevertheless, rotational PCA could only be used if the mapping process is allowed to be done offline. Also, the maximum number of rotations included in the map has been $N_R = 32$. This means that the resolution in orientation estimation will be low. Anyway, even though the computational cost of rotational PCA had been low enough, this algorithm would not have permitted building maps online since all the training images must be available to start the process (unless any incremental PCA algorithm is used [32], which would add more computational cost to the process and make it unbearable in real time). The other three algorithms do not present this disadvantage since they are inherently incremental methods (each image is described independently on the rest of images so the robot can build the map as it is exploring the unknown environment).

4.3. Image Distance versus Geometric Distance. Once we know the computational cost of the description methods, the objective of this section consists in carrying out several experiments to test the applicability of these methods to the creation of topological maps.

The first experiment consists in studying the relationship between the geometrical distance between the positions where two images have been captured and the distance between the descriptors of these two images. The behaviour of this distance should be monotonically increasing and

linear, at least in a close interval around the point where the reference image was captured.

To carry out this study, several distance measures are taken into consideration. First, these distances are formalized. If we have two descriptors $\vec{r} \in \mathbb{R}^{l\times 1}$ and $\vec{s} \in \mathbb{R}^{l\times 1}$, where r_i and s_i are the ith components of $\vec{r}$ and $\vec{s}$, with $i = 1, \ldots, l$. The distance between these descriptors can be defined as follows.

(a) Weighted Metric Distance. Consider

$$\text{dist}_p(\vec{r}, \vec{s}) = \left(\sum_{i=1}^{l} \omega_i \cdot |r_i - s_i|^p \right)^{1/p}. \quad (7)$$

If we consider $\omega_i = 1$, $i = 1, \ldots, l$, the Minkowski distance is obtained. Two particular cases will be considered: dist_1 (*cityblock* distance), which is defined from the Minkowski distance with $p = 1$, and dist_2 (Euclidean distance), doing $p = 2$.

(b) Pearson Correlation Coefficient. It is a similitude coefficient that can be obtained as follows:

$$\text{sim}_{\text{Pea}}(\vec{r}, \vec{s}) = \frac{\vec{r}_d^{\,T} \cdot \vec{s}_d}{|\vec{r}_d|\,|\vec{s}_d|}, \quad (8)$$

where $\vec{r}_d = [r_1 - \bar{r}, \ldots, r_l - \bar{r}]$ and $\vec{s}_d = [s_1 - \bar{s}, \ldots, s_l - \bar{s}]$, $\bar{r} = (1/l)\sum_j r_j$, $\bar{s} = (1/l)\sum_j s_j$. It takes values in the range $[-1, +1]$. From this similitude coefficient, a distance measure can be defined as follows:

$$\text{dist}_{\text{Pea}}(\vec{r}, \vec{s}) = 1 - \text{sim}_{\text{Pea}}(\vec{r}, \vec{s}). \quad (9)$$

(c) Inner Product. It is also a similitude coefficient that can be calculated as the scalar product between the two vectors to compare

$$\text{sim}_{\cos}(\vec{r}, \vec{s}) = \frac{\vec{r}^{\,T} \cdot \vec{s}}{|\vec{r}|\,|\vec{s}|}. \quad (10)$$

As shown in the equation, $\vec{r}$ and $\vec{s}$ are usually normalized. In this case, this measure is known as *cosine similitude* and takes values in the range $[-1, +1]$. The corresponding distance value is

$$\text{dist}_{\cos}(\vec{r}, \vec{s}) = 1 - \text{sim}_{\text{in}}(\vec{r}, \vec{s}). \quad (11)$$

(d) Other Distance Measures. Other distance measures have been considered in the study, as they have provided good results when applied to very-high dimensional data in clustering tasks [33]. We name them log and root distances:

$$\text{dist}_{\log}(\vec{r}, \vec{s}) = -\log_{10}\left(1 - \frac{1}{l}\sum_{i=1}^{l} \frac{|r_i - s_i|}{\max(r_i) - \min(r_i)}\right), \quad (12)$$

where $\max(r_j)$ and $\min(r_j)$ are, respectively, the maximum and minimum value among the j components of the n vectors in $\mathbf{R} = \{\vec{r}_1, \ldots, \vec{r}_n\}$. This way, the distance does not only depend on $\vec{r}$ and $\vec{s}$, but also on the set of vectors in $\mathbf{R}$:

$$\text{dist}_{\text{root}}(\vec{r}, \vec{s}) = \sqrt{\frac{1}{l}\sum_{i=1}^{l} \left(\frac{r_i - s_i}{r_i + s_i}\right)^2}. \quad (13)$$

To study the relation between the image distance (distance between the descriptors of two images) and the geometric distance (Euclidean distance between the points where these images were captured) the rooms *kitchen* and *hall* of data set 2 have been used, since these are the two rooms whose grid presents a higher resolution (10×10 cm). In both cases, from a reference point, some sets of scenes both horizontally and vertically have been taken and the distance between the reference image and all of them has been obtained. The next figures show the results obtained (average distance and variance) after this set of experiments.

First, Figure 7 shows the distance results when using FS. This figures show how, in this case, the different distance measures present quite similar results. In a close interval to the reference image, the image distance increases (quite linearly in the case of the correlation and cosine distances). However, they present a nondesirable behaviour since they reach a maximum and then they begin to decrease. The cosine distance is not shown as it provides a very similar result to the correlation.

Next, Figure 8 shows the results when using rotational PCA to describe scenes. In all cases, $k_3 = 200$ components have been used. The result obtained with the distance *cityblock* is remarkable because, despite being the simplest measure, it behaves quite linearly when the number of rotations is high enough (but it presents a local minimum in the middle). Logarithm and root distances present also relatively good results. The data in Figure 9 allow us to analyse the influence of the number of PCA components. In all cases, including a low number of components (very compact descriptors), the behaviour is quite linear and monotonous with some distance measures.

Thirdly, Figure 10 shows the results obtained when the images are described through HOG. In all cases, the results are quite similar to the FS. However, the local maximum is reached in a closer point to the reference image. This fact limits the validity range of the computed distance.

To finish the distance results, we show the results obtained with *gist* in Figure 11. In this case, thanks to the linearity and monotony, the results obtained with correlation (and cosine) must be highlighted.

As a final conclusion, the FS and HOG descriptors present a limited utility to estimate the topological distance between images, provided that the behaviour of the distances is not monotonous (FS presents a larger useful interval). Rotational PCA presents a relatively good behaviour when using *cityblock*, Euclidean, and Minkowski distance. At last, the excellent performance of *gist* with the *cityblock* and correlation distances must be highlighted, due to their monotony and linearity. The goodness of this configuration suggests

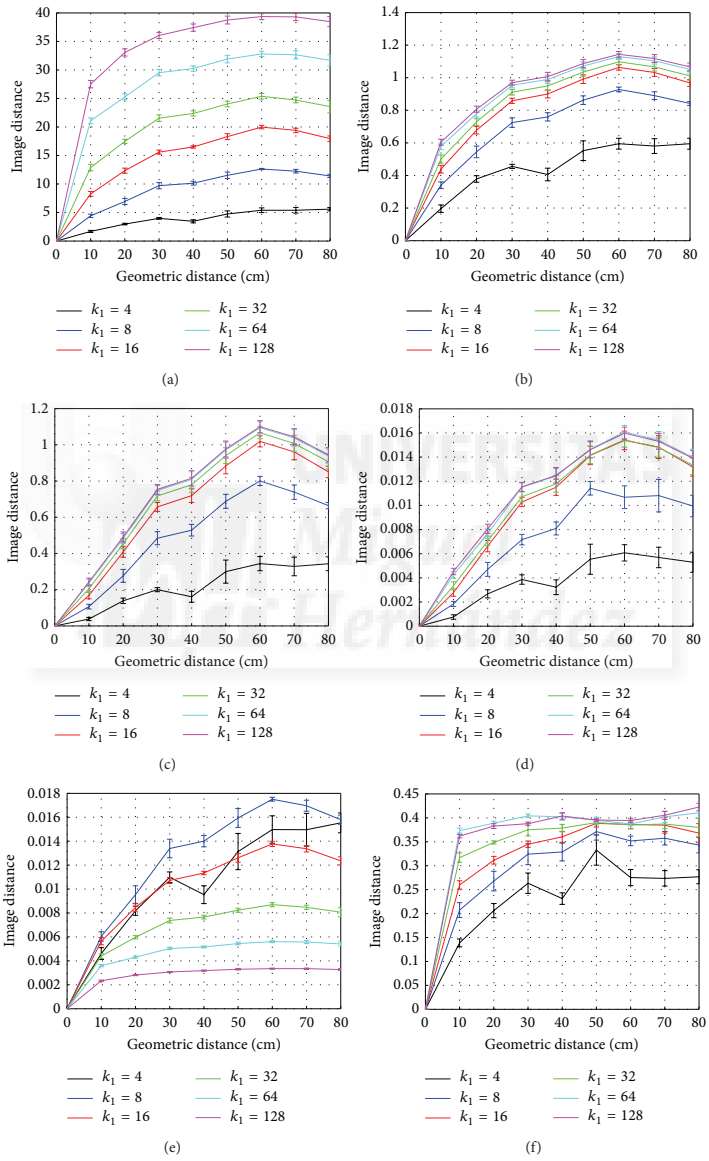


FIGURE 7: FS. Image distance versus geometric distance, depending on k_1. Distance measure: (a) *cityblock*, (b) Euclidean, (c) weighted $p = 3$, $\omega_i = 0.95^i$, (d) correlation, (e) logarithm, and (f) root.

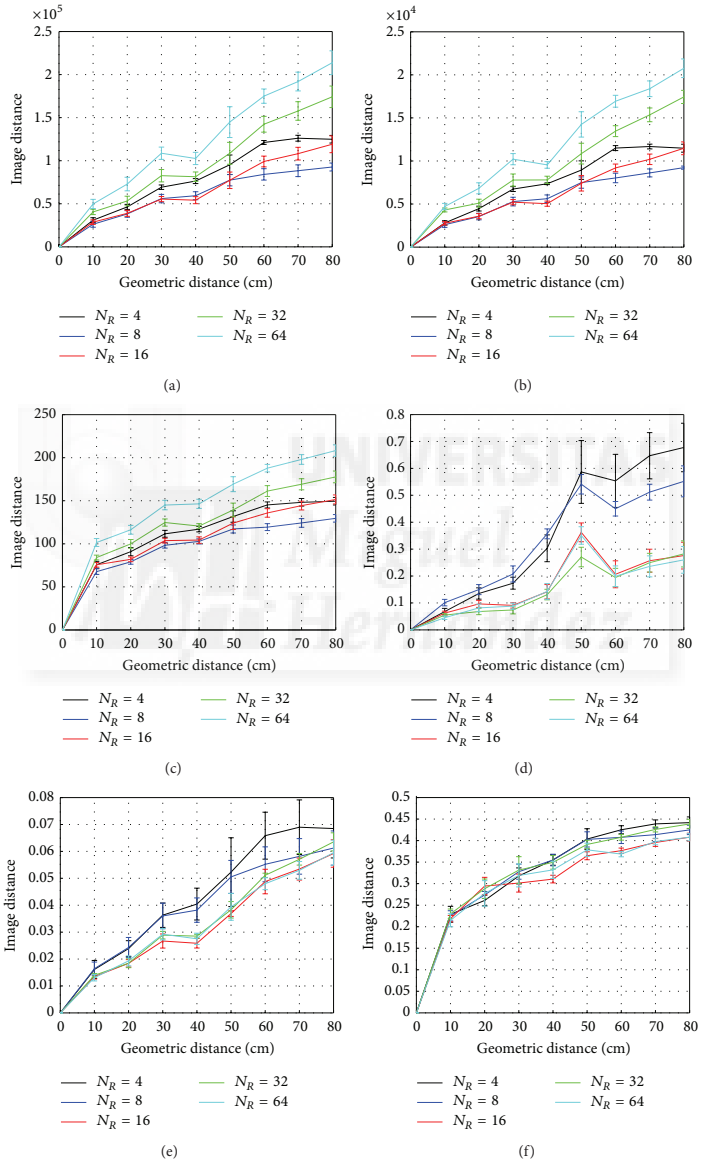


Figure 8: Rotational PCA. Image distance versus geometric distance, depending on N_R. Distance measure: (a) *cityblock*, (b) Euclidean, (c) weighted $p = 2$, $\omega_i = 0.95^i$, (d) correlation, (e) logarithm, and (f) root.

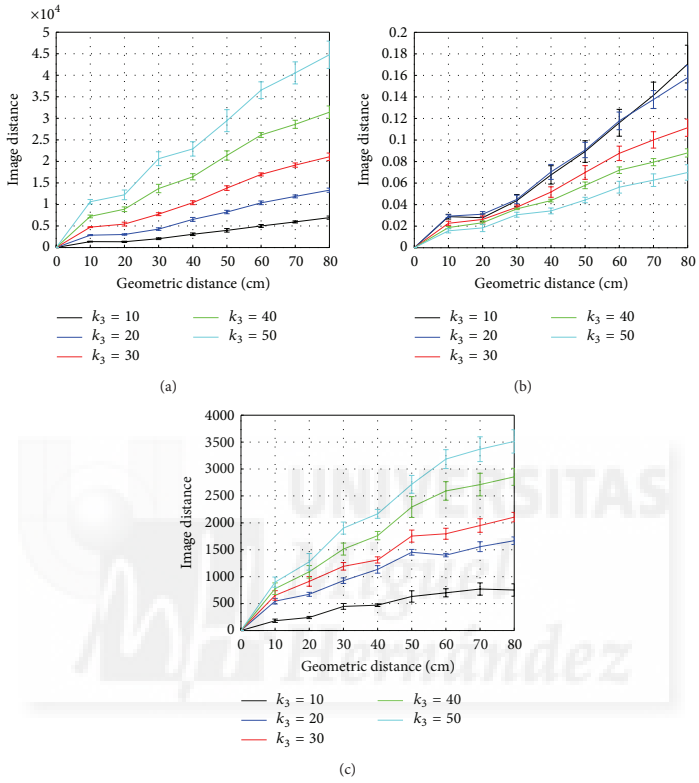


Figure 9: Rotational PCA. Image distance versus geometric distance, depending on k_3. (a) *cityblock* distance and $N_R = 64$, (b) logarithm distance and $N_R = 64$, and (c) Euclidean distance and $N_R = 4$.

that it could be the first option to implement a topological mapping algorithm.

4.4. Topological Model. This section reflects the last experiment carried out. The algorithm presented in Section 3.2 has been used to build several topological maps using the data set 2. This data set presents different grid steps, depending on the room considered. This way, it allows us to study the influence of this important parameter.

As far as the configuration of the mass-spring-damper algorithm is concerned, the most critical parameter is the spring constant. If we consider that all the springs have the same elastic constant, the results are not consistent, because the presence of *visual aliasing* in the environment introduces nondesired forces in the system. To avoid this effect, each elastic constant is calculated depending on the distances between the descriptors of the two particles i and j linked by this spring, according to the following expression:

$$k_{ij} = \min\left(\frac{c}{\text{dist}\left(\vec{d}_i, \vec{d}_j\right)^2}, 100\right), \quad i, j = 1, \ldots, n,\ i \neq j, \tag{14}$$

where c is the average slope measured on Figures 7–11, depending on the selected descriptor and parameters. The value of k_{ij} has been limited to 100 to avoid the presence of very high efforts.

At last, the natural length of each spring is equal to the distance between the descriptors of the particles linked by the spring:

$$l_{ij}^0 = \text{dist}\left(\vec{d}_i, \vec{d}_j\right). \tag{15}$$

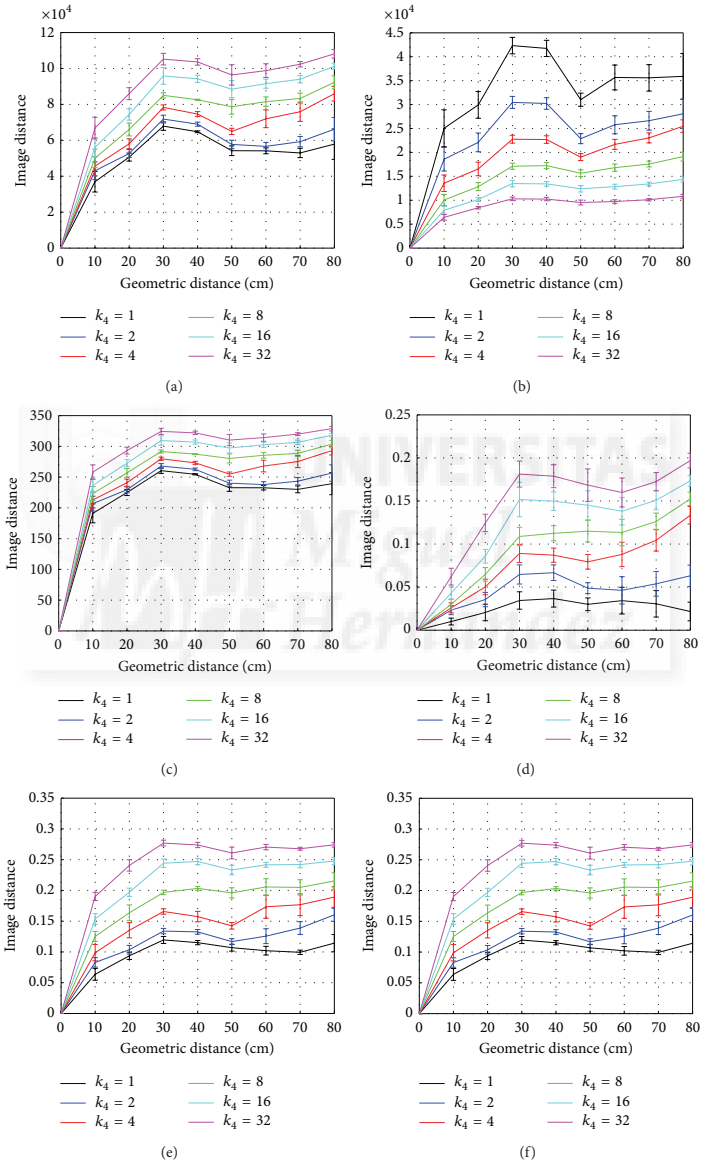


Figure 10: HOG. Image distance versus geometric distance, depending on k_4. Distance measure: (a) *cityblock*, (b) Euclidean, (c) weighted $p = 3$, $\omega_i = 1$, (d) correlation, (e) logarithm, and (f) root.

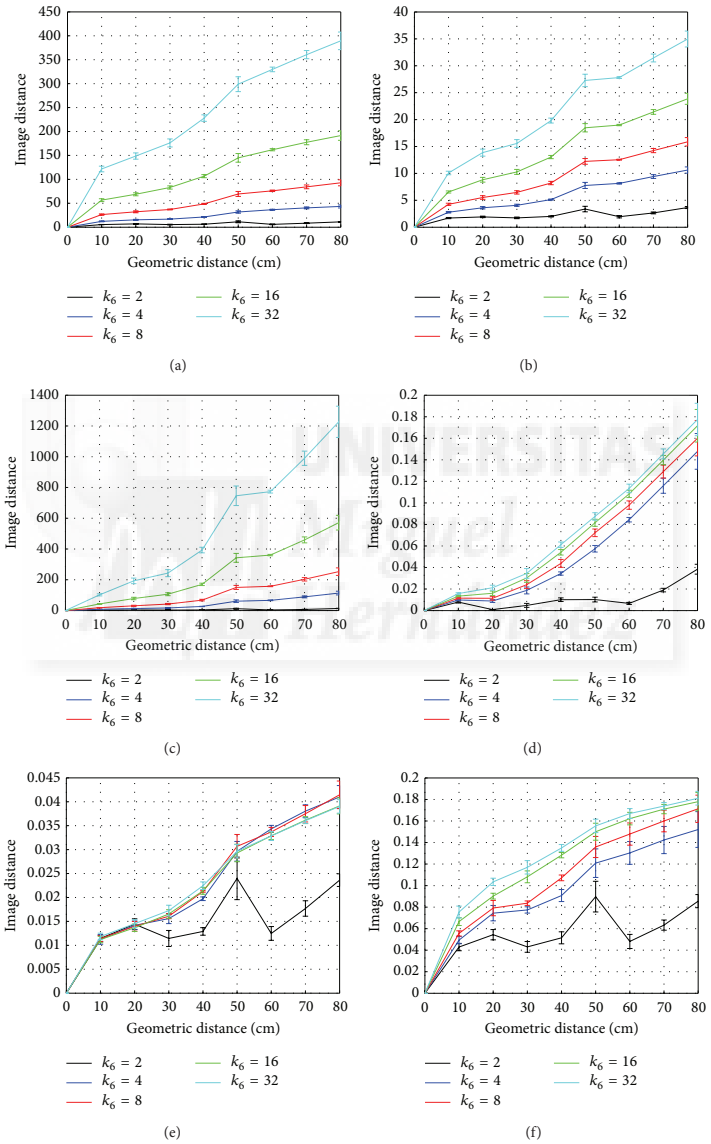


Figure 11: *Gist*. Image distance versus geometric distance, depending on k_6. Distance measure: (a) *cityblock*, (b) Euclidean, (c) weighted $p = 3$, $\omega_i = 1$, (d) correlation, (e) logarithm, and (f) root.

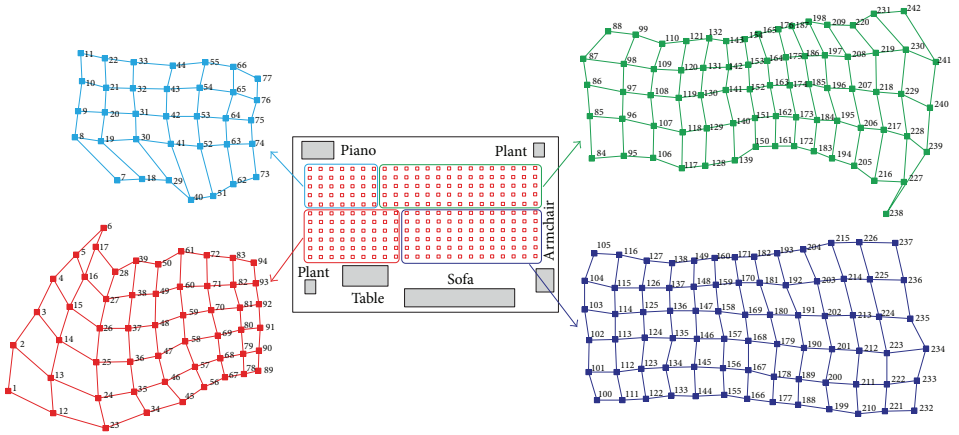


Figure 12: Topological maps created in the *hall*, data set 2. The grid size is 10×10 cm.

To finish, all the particles are considered to have the same mass $m_i = 1$, $i = 1, \ldots, n$, since our experiments have shown that it is not a relevant parameter. The damping constant of all the dampers is set to $\kappa_{ij} = 0.6$. Thanks to this dynamic friction, the behaviour of the system tends to be overdamped and more stable, permitting a gradual evolution from the initial position to the steady state, without large oscillations. To finish, we have defined the time step $\Delta t = 0.03$ s. It is an important parameter that influences both the settling time and the stability of the resulting system. A low value supposes a high settling time and a high value makes this time lower but the movement between two consecutive iterations may be so high that the system could destabilize.

After a complete bank of experiments, the best results have been obtained with the *gist* descriptor with $k_6 = 16$ blocks, $m = 16$ orientations, and correlation distance and with the FS descriptor with $k_1 = 32$ blocks and correlation distance. These results are in line with Section 4.3. HOG has not provided good results, as Figure 10 suggested.

Figures 12, 13, and 14 show some of the topological maps created in three different rooms of data set 2 (hall, laboratory, and corridor, resp.). We show the results of these rooms because they have different grid size. In each room, several sets of images, with different size and distribution along all the space of the room, have been chosen. Then, the mapping algorithm has been applied. The final distribution of each map is shown. In these maps, the lines are drawn just with representative purposes (when the algorithm starts, it has no information about the initial positions nor about the vicinity relations).

The figures show that, despite the different grid size, relatively good results are achieved in all cases. This way, global-appearance descriptors prove to be a good choice for the creation of topological maps where the concepts of closeness and farness are included.

Comparing to feature-based techniques, in a previous work [34], a new global-appearance description method was proposed and a preliminary comparison with a classical global-appearance method (the Fourier Signature) and a feature-based method (SIFT features) was carried out. The results showed that global-appearance descriptors are robust to solve the localization process and their computational cost is relatively low, improving the performance of local feature descriptors.

To finish, Table 4 shows a final comparison of the performance of the four methods in mapping tasks. First, to compare the computational cost, the table shows the minimum and the maximum necessary time ($t_{\min}$ and $t_{\max}$, resp.) to include each image in the model. Second, to study the relation between the image distance and the geometric distance, a least squares linear fit has been carried out with all the curves in Figures 7–11. In all cases, the origin has been weighted to ensure that the resulting line passes through it. The table shows the results of the best fit: the slope m, the coefficient r^2, and the values of the parameters.

5. Conclusion and Future Works

This paper has focused on the study of the mapping problem. It has been addressed from a topological point of view, using the information provided by an omnidirectional vision sensor to build the model, and methods based on global appearance to extract relevant information from the scenes. The work has carried out a comparative evaluation between some renowned description methods in map building tasks.

The main contributions of the paper include an exhaustive study of visual appearance techniques (FS, PCA, HOG, and *gist*) and the adaptation of some of these algorithms to store position and orientation information from panoramic scenes. Also, the computational cost to build the nodes

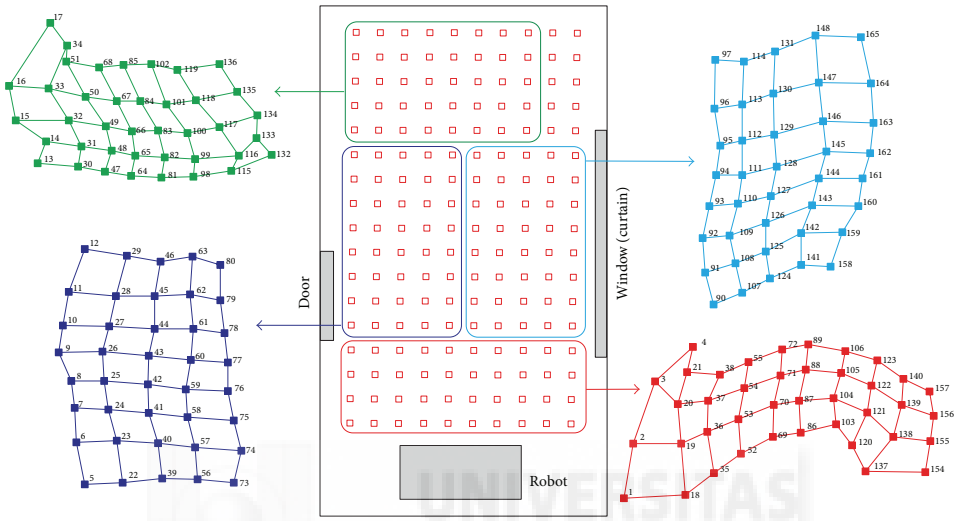


Figure 13: Topological maps created in the *laboratory*, data set 2. The grid size is 30 × 30 cm.

Table 4: Performance of the description methods: computational cost per image to build the model and best linear fit of the image distance versus the geometric distance.

Descriptor	Time		Best linear fit			
	t_{min} (ms)	t_{max} (ms)	m	r^2	Parameters	Distance
FS	13	17	0.50	0.9734	$k_1 = 4$	Weighted
r-PCA	675	111740	$2.7 \cdot 10^5$	0.9856	$k_3 = 50$, $N_R = 64$	*Cityblock*
HOG	49	456	0.17	0.9576	$k_4 = 16$	Correlation
Gist	110	333	14	0.9902	$k_6 = 4$	Correlation

of the map has been studied, including the influence of the most relevant parameters. This study has revealed that FS, HOG, and *gist* present a reasonable computational cost and, from this point of view, their use could be feasible in real time applications. Besides this, the performance of the descriptors has been tested in mapping tasks. First, we have focused on the relation between the image distance and the geometric distance, which allows us to know the descriptors that best reflect an idea of closeness and farness, since they are two important concepts to reflect in the map. All the description methods have been tested along with several distance measures, and the results have shown that *gist* and FS descriptors with certain distance measures present positive results. Second, a mass-spring-damper method has been implemented to build topological maps, their parameters have been tuned and several experiments have been carried out. To finish, several topological maps have been built, including not only connectivity but also closeness and farness concepts. The results have shown the goodness of the mapping approach and of the parameters tuning.

These results have demonstrated that global-appearance methods are a feasible approach to solve the mapping task. Thanks to them, the robot can build a model of the environment that goes beyond the classical *topological maps* since the model is a version of the original grid except for a scale factor. This suggests that the model could be used to estimate with accuracy the position and orientation of the robot in the environment, with computational efficiency. This fact may have interesting implications in future developments in the field of mobile robotics. As an example, this concept can be used to build hybrid maps that arrange the information in several layers, with different accuracy: a high level layer that permits carrying out a rough and quick localization and a lower layer that contains information with geometric accuracy and allows the robot to refine the estimation of its position. Global-appearance methods can be used on their own or in conjunction with feature-based techniques to develop algorithms that face these problems efficiently.

All these facts encourage us to go into this framework in depth. To build a fully autonomous mapping and localization

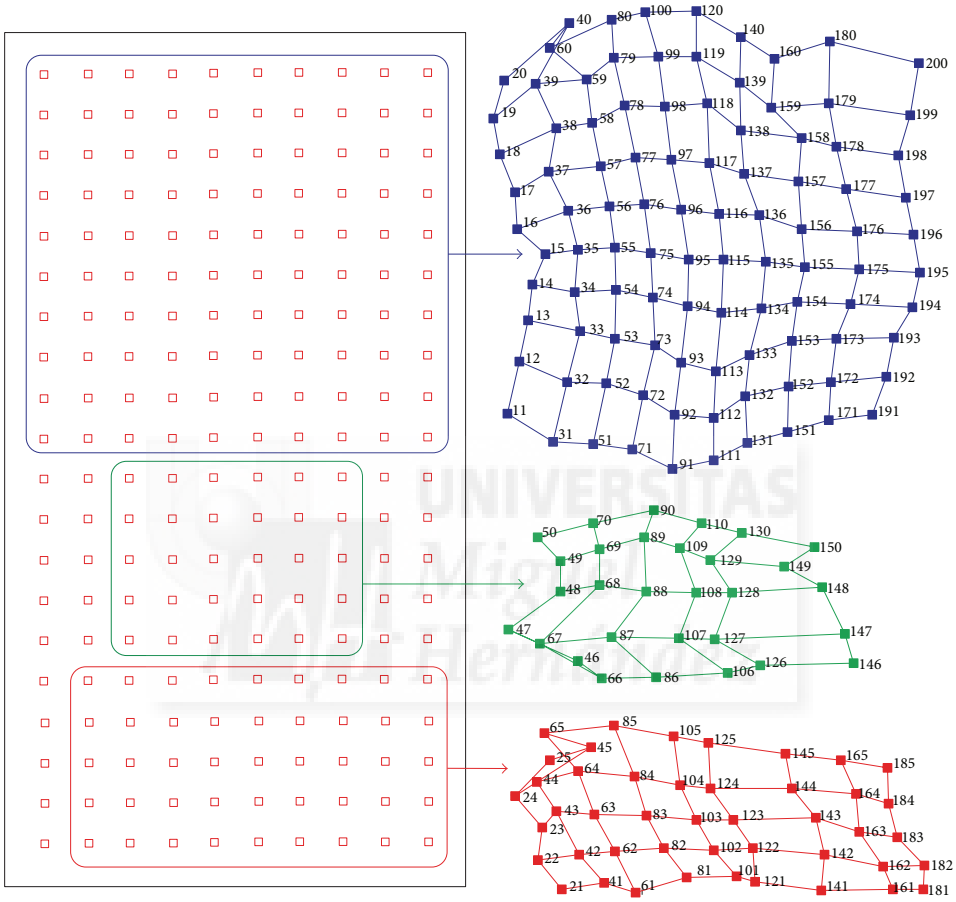


FIGURE 14: Topological maps created in the *corridor*, data set 2. The grid size is 50 × 50 cm.

system several future works should be considered. First, the image collection process could be automated to obtain an optimal representation of the environment. Second, this model must be used to estimate the current position and orientation of the robot taking into account typical situations such as changes in lighting conditions or visual occlusions. At last, both processes could be integrated in a topological SLAM system that carries out both the model creation and the localization from the scratch. To optimize these algorithms we also consider carrying out a complete comparison between global-appearance and feature-based techniques as a future work.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work has been supported by the Spanish Government through the project DPI 2013-41557-P, *Navegación de Robots en Entornos Dinámicos Mediante Mapas Compactos con Información Visual de Apariencia Global*, and by the *Generalitat Valenciana* through the project *GV*/2015/031: *Creación de*

Mapas Topológicos a Partir de la Apariencia Global de un Conjunto de Escenas.

References

[1] J. Gaspar, N. Winters, and J. Santos-Victor, "Vision-based navigation and environmental representations with an omnidirectional camera," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 890–898, 2000.

[2] S. Thrun, "Robotic mapping: a survey, in exploring artificial intelligence," in *The New Milenium*, pp. 1–35, Morgan Kaufmann, San Francisco, Calif, USA, 2003.

[3] A. Gil, Ó. Reinoso, M. Ballesta, M. Juliá, and L. Payá, "Estimation of visual maps with a robot network equipped with vision sensors," *Sensors*, vol. 10, no. 5, pp. 5209–5232, 2010.

[4] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: a survey," *Robotics and Autonomous Systems*, vol. 64, pp. 1–20, 2015.

[5] K. Konolige, E. Marder-Eppstein, and B. Marthi, "Navigation in hybrid metric-topological maps," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '11)*, pp. 3041–3047, Shanghai, China, May 2011.

[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 '09)*, pp. 4300–4305, IEEE, Kobe, Japan, May 2009.

[7] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[8] C. Valgren and A. J. Lilienthal, "SIFT, SURF & seasons: appearance-based long-term localization in outdoor environments," *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 149–156, 2010.

[9] A. C. Murillo, J. J. Guerrero, and C. Sagüés, "SURF features for efficient robot localization with omnidirectional images," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 3901–3907, IEEE, Rome, Italy, April 2007.

[10] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: speeded up robust features," in *Computer Vision—ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7–13, 2006. Proceedings, Part I*, A. Leonardis, H. Bischof, and A. Pinz, Eds., vol. 3951 of *Lecture Notes in Computer Science*, pp. 404–417, Springer, Berlin, Germany, 2006.

[11] J. Kosecka, L. Zhou, P. Barber, and Z. Duric, "Qualitative image based localization in indoors environments," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. II-3–II-8, Madison, Wis, USA, June 2003.

[12] E. Menegatti, T. Maeda, and H. Ishiguro, "Image-based memory for robot navigation using properties of omnidirectional images," *Robotics and Autonomous Systems*, vol. 47, no. 4, pp. 251–267, 2004.

[13] F. Rossi, A. Ranganathan, F. Dellaert, and E. Menegatti, "Toward topological localization with spherical fourier transform and uncalibrated camera in," in *Proceedings of the International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR '08)*, pp. 319–350, Springer, Venice, Italy, 2008.

[14] A. Gil, O. M. Mozos, M. Ballesta, and O. Reinoso, "A comparative evaluation of interest point detectors and local descriptors for visual SLAM," *Machine Vision and Applications*, vol. 21, no. 6, pp. 905–920, 2010.

[15] A. Guérin-Dugué and A. Oliva, "Classification of scene photographs from local orientations features," *Pattern Recognition Letters*, vol. 21, no. 13-14, pp. 1135–1140, 2000.

[16] M. Kirby, *Geometric Data Analysis: An Empirical Approach to Dimensionality Reduction and the Study of Patterns*, Wiley-Interscience, 2001, http://books.google.es/books?id=nRmFQgAACAAJ.

[17] A. Leonardis and H. Bischof, "Robust recognition using eigenimages," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 99–118, 2000.

[18] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[19] M. Jogan and A. Leonardis, "Robust localization using eigenspace of spinning-images," in *Proceedings of the IEEE Workshop on Omnidirectional Vision*, pp. 37–44, IEEE, Hilton Head Island, SC, USA, June 2000.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 886–893, San Diego, Calif, USA, June 2005.

[21] Q. Zhu, S. Avidan, M.-C. Yeh, and K.-T. Cheng, "Fast human detection using a cascade of histograms of oriented gradients," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, vol. 2, pp. 1491–1498, IEEE, June 2006.

[22] M. Hofmeister, M. Liebsch, and A. Zell, "Visual self-localization for small mobile robots with weighted gradient orientation histograms," in *Proceedings of the 40th International Symposium on Robotics (ISR '09)*, pp. 87–91, IFR, Barcelona, Spain, March 2009.

[23] L. Payá, F. Amorós, L. Fernández, and O. Reinoso, "Performance of global-appearance descriptors in map building and localization using omnidirectional vision," *Sensors*, vol. 14, no. 2, pp. 3033–3064, 2014.

[24] A. Oliva and A. Torralba, "Building the gist of ascene: the role of global image features in recognition," *Progress in Brain Reasearch*, vol. 155, pp. 23–36, 2006.

[25] I. Biederman, "Aspects and extension of a theory of human image understanding," in *Computational Processes in Human Vision: An Interdisciplinay Perspective*, Ablex, Norwood, NJ, USA, 1988.

[26] C. Siagian and L. Itti, "Biologically inspired mobile robot vision localization," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 861–873, 2009.

[27] C.-K. Chang, C. Siagian, and L. Itti, "Mobile robot vision navigation & localization using gist and saliency," in *Proceedings of the 23rd IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems (IROS '10)*, pp. 4147–4154, IEEE, Taipei, Taiwan, October 2010.

[28] A. C. Murillo, G. Singh, J. Kosecká, and J. J. Guerrero, "Localization in urban environments using a panoramic gist descriptor," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 146–160, 2013.

[29] A. Selle, M. Lentine, and R. Fedkiw, "A mass spring model for hair simulation," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 64:1–64:11, 2008.

[30] Automation-Robotics and Computer Vision Research Group (ARVC), "Quorum 5 set of images," Miguel Hernández University, Elche, Spain, http://arvc.umh.es/db/images/quorumv/.

[31] R. Möller, A. Vardy, S. Kreft, and S. Ruwisch, "Visual homing in environments with anisotropic landmark distribution," *Autonomous Robots*, vol. 23, no. 3, pp. 231–245, 2007.

[32] M. Artač, M. Jogan, and A. Leonardis, "Mobile robot localization using an incremental eigenspace model," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1025–1030, IEEE, May 2002.

[33] H. Spat, *Clustering Analysis Algorithms for Data Reduction and Classification of Objects*, Ellis Horwood, New York, NY, USA, 1982.

[34] 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.

UNIVERSITAS
Miguel
Hernández

Bibliografía

[1] Omnidirectional images data set 3: Altitude estimation. url: https://arvc.umh.es/db/images/altitude/. 93

[2] F. Amorós, L. Payá, O. Reinoso, and D. Valiente. Towards relative altitude estimation in topological navigation tasks using the global appearance of visual information. In *VISAPP 2014, International Conference on Computer Vision Theory and Applications*, volume 1, pages 194–201, 2014. 30

[3] S. An and S. Yuan. Relative position control design of receiver uav in flying-boom aerial refueling phase. *ISA Transactions*, 2017. 26

[4] H. Andreasson and A. Lilienthal. 6d scan registration using depth-interpolated local image features. *Robotics and Autonomous Systems*, 58(2), 2010. 44

[5] C. V. Angelino, V. R. Baraniello, and L. Cicala. High altitude uav navigation using imu, gps and camera. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 647–654, July 2013. 79

[6] H. Bay, T. Tuytelaars, and L. Gool. Surf: Speeded up robust features. *Computer Vision at ECCV*, 3951:404–417, 2006. 2, 87

[7] 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, 2015. iii, 2, 8, 53, 80, 121

[8] Y. Berenguer, L. Payá, L. Jiménez, M. Ballesta, and O. Reinoso. Generación de data sets simulando diferentes tipos de cámaras en entornos virtuales. In *XXXVII Jornadas de Automática*, pages 49–56, 09 2016. 47

[9] Y. Berenguer, L. Payá, O. Reinoso, A. Peidro, and L. Jiménez. Generation of data sets simulating different kinds of cameras in virtual environments. In *13th International Conference on Informatics in Control, Automation and Robotics ICINCO2016*, pages 352–359, 01 2016. 47

[10] S. L. Bogner. An introduction to panospheric imaging. In *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, volume 4, pages 3099–3106. IEEE, 1995. 27

[11] R. Bunschoten and B. Krose. Robust scene reconstruction from an omnidirectional vision system. *IEEE Transactions on Robotics and Automation*, 19(2):351–357, Apr 2003. 3

[12] E. L. Cabral, J. De Souza, and M. C. Hunold. Omnidirectional stereo vision with a hyperbolic double lobed mirror. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 1, pages 1–9. IEEE, 2004. 28

[13] J. Cui, H. Zha, H. Zhao, and R. Shibasaki. Multi-modal tracking of people using laser scanners and video camera. *Image and Vision Computing*, 26(2):240 – 252, 2008. 25

[14] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005. 19

[15] F. Dayoub, T. Morris, B. Upcroft, and P. Corke. Vision-only autonomous navigation using topometric maps. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1923–1929, Nov 2013. 3

[16] L. Eizoh Company. Eizoh Omnidirectional Vision Sensor. url: http://www.eizoh.co.jp/mirror/wide70.html. 31

[17] L. Fernández, L. Payá, O. Reinoso, L. Jiménez, and M. Ballesta. A study of visual descriptors for outdoor navigation using google street view images. *Journal of Sensors*, 2016, 2016. 19

[18] L. Fernández, L. Payá, O. Reinoso, and L. M. Jimenez. Appearance-based approach to hybrid metric-topological simultaneous localisation and mapping. *IET Intelligent Transport Systems*, 8(8):688–699, 2014. 2

[19] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza. Collaborative monocular slam with multiple micro aerial vehicles. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3962–3970, Nov 2013. 3

[20] E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization by means of local invariant features and map refinement. *Robotica*, 33:1446–1470, 8 2015. 2

[21] E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, 64:1 – 20, 2015. 1

[22] E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, 64:1 – 20, 2015. 2, 103

[23] J. Gaspar. Omnidirectional vision for mobile robot navigation. *IST-Universidade Tecnica de Lisboa*, page 150, 2002. 30

[24] J. Gaspar, N. Winters, E. Grossmann, and J. Santos-Victor. Toward robot perception through omnidirectional vision. In *Innovations in Intelligent Machines-1*, pages 223–270. Springer, 2007. 30

[25] T. I. S. E. GmbH. Color CCD camera DFK 21BF04. url: http://www.theimagingsource.com/products/industrial-cameras/firewire-400-color/dfk21bf04h/. 30

[26] T. I. S. E. GmbH. Color CCD camera DFK 41BF02. url: http://www.theimagingsource.com/products/industrial-cameras/firewire-400-color/dfk41bf02h/. 30

[27] V. Grassi Junior and J. Okamoto Junior. Development of an omnidirectional vision system. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 28(1):58–68, 2006. 27

[28] M. Hasegawa and S. Tabbone. A shape descriptor combining logarithmic-scale histogram of radon transform and phase-only correlation function. In *2011 International Conference on Document Analysis and Recognition (ICDAR)*, pages 182–186, Sept 2011. 15

[29] T. Hoang and S. Tabbone. A geometric invariant shape descriptor based on the radon, fourier, and mellin transforms. In *20th International Conference on Pattern Recognition (ICPR)*, pages 2085–2088, Aug 2010. 15

[30] H. Ishiguro and S. Tsuji. Image-based memory of environment. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems '96 (IROS 96)*, volume 2, pages 634–639 vol.2, Nov 1996. 14, 22

[31] P. D. Jr, S. Botelho, and S. Gomes. Slam in underwater environment using sift and topologic maps. In *Robotic Symposium, 2008. LARS '08. IEEE Latin American*, pages 91–96, Oct 2008. 3

[32] J. H. Kim, J. w. Kwon, and J. Seo. Multi-uav-based stereo vision system without gps for ground obstacle mapping to assist path planning of ugv. *Electronics Letters*, 50(20):1431–1432, September 2014. 79

[33] K. Kobayashi, T. Aoki, K. Ito, H. Nakajima, and T. Higuchi. A fingerprint matching algorithm using phase-only correlation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, pages 682–691, 2004. 21

[34] I. Kostavelis, K. Charalampous, A. Gasteratos, and J. K. Tsotsos. Robot navigation via spatial and temporal coherent semantic maps. *Engineering Applications of Artificial Intelligence*, 48:173 – 187, 2016. 3

[35] P. J. Kostelec and D. N. Rockmore. Ffts on the rotation group. *Journal of Fourier analysis and applications*, 14(2):145–179, 2008. 27

[36] C. Kuglin and D. Hines. The phase correlation image alignment method. In *In Proceedings of the IEEE, International Conference on Cybernetics and Society*, pages 163–165, 1975. 21

[37] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613, May 2011. 104

[38] Q. Li, N. Zheng, L. Ma, and H. Cheng. True single view point multi-resolution catadioptric system for intelligent vehicle. In *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, pages 155–160. IEEE, 2004. 28

[39] Z.-P. Liang and P. C. Lauterbur. *Principles of magnetic resonance imaging: a signal processing perspective*. SPIE Optical Engineering Press, 2000. 15

[40] D. Lowe. Object recognition from local scale-invariant features. In *ICCV 1999, International Conference on Computer Vision*, volume 2, pages 1150–1157, 1999. 2, 73, 74, 82

[41] L. Maohai, W. Han, S. Lining, and C. Zesu. Robust omnidirectional mobile robot topological navigation system using omnidirectional vision. *Engineering Applications of Artificial Intelligence*, 26(8):1942 – 1952, 2013. 2, 26

[42] 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. cited By 92. 15, 82

[43] 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. 22, 80

[44] I. Mondragon, M. Olivares-Ménded, P. Campoy, C. Martínez, and L. Mejias. Unmanned aerial vehicles uavs attitude, height, motion estimation and control using visual systems. *Autonomous Robots*, 29:17–34, 2010. 26, 80

[45] R. Munguia, S. Urzua, and A. Grau. Delayed monocular slam approach applied to unmanned aerial vehicles. *PLOS ONE*, 11(12):1–24, 12 2016. 2, 103

[46] A. Natraj, D. S. Ly, D. Eynard, C. Demonceaux, and P. Vasseur. Omnidirectional vision for uav: Applications to attitude, motion and altitude estimation for day and night conditions. *Journal of Intelligent & Robotic Systems*, 69(1):459–473, 2012. 80

[47] S. K. Nayar. Omnidirectional video camera. In *Proc. DARPA Image Understanding Workshop*, volume 1, pages 235–241. Citeseer, 1997. 28

[48] A. Ohte, O. Tsuzuki, and K. Mori. A practical spherical mirror omnidirectional camera. In *Robotic Sensors: Robotic and Sensor Environments, 2005. International Workshop on*, pages 8–13. IEEE, 2005. 28

[49] A. Oliva and P. G. Schyns. Coarse blobs or fine edges? evidence that information diagnosticity changes the perception of complex visual stimuli. *Cognitive psychology*, 34(1):72–107, 1997. 18

[50] A. Oliva and P. G. Schyns. Diagnostic colors mediate scene recognition. *Cognitive psychology*, 41(2):176–210, 2000. 18

[51] A. Oliva and A. Torralba. Chapter 2 building the gist of a scene: the role of global image features in recognition. In *Visual Perception Fundamentals of Awareness: Multi-Sensory Integration and High-Order Perception*, volume 155, Part B of *Progress in Brain Research*, pages 23 – 36. Elsevier, 2006. 17, 76

[52] A. Oppenheim and J. Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, May 1981. 21

[53] G. Oriolo, A. Paolillo, L. Rosa, and M. Vendittelli. Humanoid odometric localization integrating kinematic, inertial and visual information. *Autonomous Robots*, 40(5):867–879, 2016. 2

[54] 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. 2, 50, 52, 73

[55] 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. iii, 2, 8, 81, 121

[56] B. Peasley and S. Birchfield. Rgbd point cloud alignment using lucas-kanade data association and automatic error metric selection. *IEEE Transactions on Robotics*, 31(6):1548–1554, Dec 2015. 25

[57] J. Radon. Uber die bestimmung von funktionen durch ihre integralwerte langs gewisser mannigfaltigkeiten. *Berichte Sachsische Akademie der Wissenschaften*, 69(1):262–277, 1917. 15, 76, 80

[58] A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, 22(1):92–107, Feb 2006. 80

[59] D. W. Rees. Panoramic television viewing system, Apr. 7 1970. US Patent 3,505,465. 27

[60] L. Riazuelo, J. Civera, and J. Montiel. C2tam: A cloud framework for cooperative tracking and mapping. *Robotics and Autonomous Systems*, 62(4):401 – 413, 2014. 25

[61] A. Satici, D. Tick, J. Shen, and N. Gans. Path-following control for mobile robots localized via sensor-fused visual homography. In *2013 American Control Conference*, pages 6287–6293. IEEE, 2013. 2

[62] G. A. F. Seber. *Multivariate observations*. John Wiley & Sons, Inc, New York, 1984. 24, 59

[63] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(2):300–312, Feb 2007. 19

[64] A. Torralba, K. Murphy, W. Freeman, and M. Rubin. Context-based vision system for place and object recognition. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 273–280 vol.1, Oct 2003. 18

[65] D. Valiente, A. Gil, L. Fernández, and O. Reinoso. A comparison of ekf and sgd applied to a view-based slam approach with omnidirectional images. *Robotics and Autonomous Systems*, 62(2):108 – 119, 2014. 2, 26, 103

[66] A. Vardy and R. Möller. Biologically plausible visual homing methods based on optical flow techniques. *Connection Science, Special Issue: Navigation*, 17(1-2):47–89, 2005. 63, 68

[67] X. Wang, J. Tang, J. Niu, and X. Zhao. Vision-based two-step brake detection method for vehicle collision avoidance. *Neurocomputing*, 173, Part 2:450 – 461, 2016. 25

[68] S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart. Versatile distributed pose estimation and sensor self-calibration for an autonomous mav. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 31–38, May 2012. 3

[69] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016. 103

[70] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor. Omni-directional vision for robot navigation. *IEEE Workshop on Omnidirectional Vision*, pages 21–28, 2000. 2, 26

[71] Y. Yagi and S. Kawato. Panorama scene analysis with conic projection. In *Intelligent Robots and Systems' 90.'Towards a New Frontier of Applications', Proceedings. IROS'90. IEEE International Workshop on*, pages 181–187. IEEE, 1990. 28

[72] K. Yamazawa, Y. Yagi, and M. Yachida. Obstacle detection with omnidirectional image sensor hyperomni vision. In *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, volume 1, pages 1062–1067. IEEE, 1995. 15, 27

[73] K. Yoshida, H. Nagahara, and M. Yachida. An omnidirectional vision sensor with single viewpoint and constant resolution. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4792–4797. IEEE, 2006. 28

[74] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1491–1498, 2006. 19