applied sciences

MDPI

Article

A Method for the Calibration of a LiDAR and Fisheye Camera System

Álvaro Martínez [1,*], Antonio Santo [1], Monica Ballesta [1], Arturo Gil [1] and Luis Payá [1,2,*]

[1] Engineering Research Institute of Elche (I3E), Miguel Hernandez University, Avinguda de la Universidad d'Elx, s/n, 03202 Elche, Spain; a.santo@umh.es (A.S.); m.ballesta@umh.es (M.B.); arturo.gil@umh.es (A.G.)
[2] Valencian Graduate School and Research Network of Artificial Intelligence (ValgrAI), Camí de Vera s/n, Edificio 3Q, 46022 Valencia, Spain
* Correspondence: alvaro.martinezb@umh.es (Á.M.); lpaya@umh.es (L.P.)

Abstract: LiDAR and camera systems are frequently used together to gain a more complete understanding of the environment in different fields, such as mobile robotics, autonomous driving, or intelligent surveillance. Accurately calibrating the extrinsic parameters is crucial for the accurate fusion of the data captured by both systems, which is equivalent to finding the transformation between the reference systems of both sensors. Traditional calibration methods for LiDAR and camera systems are developed for pinhole cameras and are not directly applicable to fisheye cameras. This work proposes a target-based calibration method for LiDAR and fisheye camera systems that avoids the need to transform images to a pinhole camera model, reducing the computation time. Instead, the method uses the spherical projection of the image, obtained with the intrinsic calibration parameters and the corresponding point cloud for LiDAR–fisheye calibration. Thus, unlike a pinhole-camera-based system, a wider field of view is provided, adding more information, which will lead to a better understanding of the environment itself, as well as enabling using fewer image sensors to cover a wider area.

Keywords: autonomous robotic systems; calibration; fisheye image; information and sensor fusion; LiDAR; multi-sensor systems; perception and sensing

Academic Editor: Atsushi Mase

Received: 26 December 2024
Revised: 11 February 2025
Accepted: 12 February 2025
Published: 15 February 2025

Citation: Martínez, Á.; Santo, A.; Ballesta, M.; Gil, A.; Payá, L. A Method for the Calibration of a LiDAR and Fisheye Camera System. *Appl. Sci.* **2025**, *15*, 2044. https://doi.org/10.3390/app15042044

1. Introduction

In recent decades, LiDAR sensors have improved their performance, becoming lighter and more economical, and providing accurate distance measurements. While cameras were the most common sensors in the past for autonomous navigation or other applications in robotics, the use of LiDAR sensors has increased in recent years, and the combination of both kinds of sensors provides complementary information that has the potential to address autonomous navigation more safely and efficiently.

LiDAR sensors provide accurate distance measurements and are invariant to the lighting conditions of the scene since the scenario is illuminated by the laser sensor itself. However, laser sensors typically have limited resolution, mainly along the vertical axis. Meanwhile, cameras are economical and present low weight, and provide a large amount of information about the environment. Together, these systems enable a more complete and accurate representation of the surrounding environment, which is essential to ensure safety and efficiency in autonomous navigation. While cameras are sensitive to the lighting conditions or seasonal changes, LiDAR sensors are sensitive to the presence of dust or rain, which can affect the quality of the data.

The combination of LiDAR and camera data may improve the understanding of the environment in various applications, including the semantic segmentation of point clouds and images, thus facilitating the identification and classification of objects and surfaces in three-dimensional scenes. In addition, object detection may benefit from the fusion of three-dimensional and visual information, enabling more accurate localization and recognition of objects in real time and in challenging operations [1,2]. Some approaches developed in recent years for point cloud semantic segmentation [3] and object recognition [4] demonstrate that the successful design of methods that fuse information correctly provides clear improvements over single-sensor-based methods.

This paper addresses the estimation of the translation and rotation between a fisheye camera and a LiDAR sensor. The precise calibration of the transformation between the reference systems associated with each sensor is key to ensure that the data collected by both sensors are effectively fused, enabling accurate real-time interpretation of the environment.

The field of view of pinhole cameras is limited and typically below 90°, often between 40º and 75°, so the use of fisheye cameras, whose field of view could be up to 180° or more, is becoming more common. Fisheye cameras have an expanded field of view, which permits a more complete but distorted observation of the environment. However, the calibration of fisheye cameras is more complex than that of pinhole cameras since the fisheye lens introduces non-linear distortions in the image.

Pinhole cameras are the most widely used; thus, the methods for the calibration of LiDAR systems and this type of camera are abundant in the literature [5]. However, this article focuses on the use of a fisheye camera, which provides a much wider field of view than a standard camera, enabling a more complete observation of the environment. As described below, the calibration methods in the literature are not applicable to the proposed problem since most of them are designed for the calibration of pinhole cameras. Those methods, specifically designed for fisheye cameras, are not suitable for the calibration of a LiDAR–fisheye system since they require the transformation of the image to a pinhole model, which is not necessary in the proposed method.

The main contribution of this paper is the development of a method for the calibration of a LiDAR and fisheye camera system to obtain the translation and rotation parameters between the reference systems of both sensors. The algorithm is based on the use of a simple calibration pattern, which enables relating the 3D position of the pattern directly between the camera and LiDAR systems.

The methods have been tested in a simulated environment, where the ground truth of the transformation is known. Subsequently, the methods have also been tested in a real environment, where the transformation is not known, but metrics such as reprojection error can be used to evaluate the quality of the calibration.

The manuscript is organized as follows: Section 1.1 presents some works that are closely related to the methods proposed in this paper. Then, in Section 2, the proposed methods are described in detail alongside the design of the experiments. Section 3 presents the results of the experiments with the developed methods. Finally, a discussion of the results occurs in Section 4.

1.1. State of the Art

Calibration between camera and LiDAR systems is essential to ensure accurate and reliable perception of the environment in autonomous navigation and robotics applications. Thus, we are referring to methods that permit obtaining the following:

- The intrinsic calibration parameters of the fisheye camera that enable the projection of the world points (in the camera coordinate system) to the image plane.
- The relative transformation matrix T between the camera and LiDAR coordinate systems.

The first part is considered to be known since numerous methods can be found in the related literature to perform this calibration [6–8]. The proposed method focuses on the second part so that it permits obtaining the three-dimensional relationship between these sensors, which is the ultimate goal of performing the calibration. In the state of the art, calibration methods can be classified into two main categories: target-based methods and targetless methods. The subsequent subsections present some of the most relevant works in the literature.

1.1.1. Targetless Calibration

Targetless methods do not require a specific calibration object but use the natural features of the environment (corners or edges) to establish the relationship between the camera and LiDAR systems. These methods are more flexible than target-based methods but may be less accurate in some situations. All these methods have been developed for pinhole cameras, so they are not directly applicable to fisheye cameras [9–11].

The targetless calibration methods for LiDAR–camera systems can be classified into three main categories: motion-based, deep-learning-based, and traditional (image- and LiDAR feature-based) [5].

Motion-Based

These approaches take advantage of the motion of the camera–LiDAR system to estimate the transformation between the two systems. Optimization algorithms are used to estimate the transformation that best explains the observations in both systems. For instance, in Ishikawa et al. [9], a method based on ICP (Iterative Closest Point) is used to estimate the motion in the LiDAR reference system by using RANSAC (RANdom SAmple Consensus) [12] and the five-point algorithm. Also, in Nistér et al. [13], the camera motion between two frames is estimated, making use of descriptors obtained with the AKAZE algorithm [14] calculated from the point cloud.

Deep Learning

These methods employ deep neural networks to learn the correspondences between image features and LiDAR point clouds. There are some methods that employ deep learning for LiDAR–camera calibration [10,15–17]. Deep learning has proven to be effective in calibrating LiDAR–camera systems, especially in complex and challenging environments.

Traditional

These approaches extract relevant pieces of information from images and point clouds, such as features of the environment (corners, edges, and flat surfaces) and intrinsic features of the camera and LiDAR (grayscale and reflectance, respectively), and establish correspondences between them. Then, they use optimization algorithms, such as RANSAC or ICP, to estimate the transformation that best aligns the corresponding features. These methods usually rely on automatic point feature detection algorithms, which are much more widespread for imagery than for LiDAR, so some proposed methods base their innovation on the development of feature detectors for point clouds.

Some of these methods use grayscale imaging and point cloud reflectivity [18]; others combine the extraction of Canny edges [19] for imaging and the computation of edges at discontinuities in depth for LiDAR [20,21]; and others [22,23] use lines detected in both the camera and the LiDAR.

Two of the most recent methods for the calibration of LiDAR–camera systems are presented in [11,24]. While the first is based on image segmentation supported by Segment Anything [25], the second uses keypoints in both the images and a LiDAR intensity image.

These methods are less sensitive to variations in the appearance and geometry of the environment than motion-based methods but can be more sensitive to noise and ambiguities in feature correspondences.

1.1.2. Target-Based Calibration

Target-based methods use a specific calibration object to establish a precise correspondence between the camera and the LiDAR point cloud (such as a standard calibration pattern or an ArUco mark-based pattern [26]). Some of these examples use the PnP algorithm [12] to calibrate the LiDAR camera system [27,28], while others employ ICP-based methods [29]. Other authors approach calibration through iterative random sampling and intersection-line-based quality evaluation [30] or through point-to-plane and point-to-line constraints [31]. These methods have been developed for pinhole cameras, so they are not directly applicable to fisheye cameras. Thanks to the use of a calibration object in the scene, these methods are generally accurate and reliable. Such calibration objects must be easily detectable in both the image and the point cloud and permit establishing a correspondence between the points in both sensor models.

Two different methods for the calibration of a LiDAR system and fisheye camera are presented in Verma et al. [32] and Zhou et al. [33]. Both methods propose an initial removal of image distortion from the fisheye model, converting the image to a pinhole model. As a result, the pinhole image facilitates the automatic detection of the calibration patterns. In these cases, the modification of the image model may not be perfectly accurate due to the non-linearity of fisheye lenses and the difficulty in obtaining the calibration parameters. The calibration of a fisheye lens is more complex than that of a pinhole camera since the fisheye lens introduces non-linear distortions in the image, and its projection model can obey different models, such as equidistant, equiangular, stereographic, or orthographic [34]. Moreover, an extra step with its corresponding added computational time would be included.

In this paper, a target-based method for the calibration of a LiDAR–fisheye system is proposed. By using a calibration pattern (a rectangular polygon with known measurements) the translation and rotation between the two systems is calculated. The target-based method facilitates the calculation of features and permits solving the problem in a simpler way than the other methods. In addition, the implementation of methods based on deep learning would require large sets of labeled data and knowing the ground truth, and in many cases such information is not available or the acquisition is costly.

The proposed calibration approach includes the following contributions:

- It provides a method to compute the transformation between a LiDAR and a fisheye camera.
- The points detected on the fisheye image are projected to the unit sphere without the need to reproject the points to a pinhole model.
- The Kabsch algorithm is used to calculate the rotation and translation between the two systems.

2. Materials and Methods

The methods developed in this article focus on the calculation of the relative transformation between systems. The Kabsch [35] algorithm, also known as the Kabsch–Umeyama [36] algorithm, is used to find the rotation and translation between two sets of points in the 3D space. The main idea is to use reference points in space with known positions relative to the reference systems of the camera and the LiDAR, respectively. A rectangular polygon with known dimensions is used to implement this idea.

The proposed methods are divided into the following parts:

a Calculation of the fisheye camera calibration parameters by means of the *toolbox* presented in [37].
b Obtaining a set of 3D points belonging to the calibration pattern in LiDAR coordinates (Section 2.1).
c Extraction of a set of 3D points belonging to the calibration pattern in camera coordinates (Section 2.2).
d Calculation of the transformation between both reference systems using the Kabsch algorithm.

Two different methods are used for the LiDAR and the camera in order to obtain the 3D coordinates of the corners of the rectangle. The 3D coordinates of the corners relative to the camera cannot be obtained in a direct way, unlike in the case of the LiDAR. The process to find the 3D coordinates of the points with respect to the camera is explained in Section 2.2.

2.1. Three-Dimensional LiDAR Coordinates

The LiDAR itself provides the three-dimensional coordinates of the points in space, but it is not known exactly which points belong to the calibration rectangle. The edges and corners of the rectangle have to be extracted from the LiDAR point cloud. The algorithm proposed to extract the edges and corners of the calibration rectangle based on the LiDAR point cloud follows the structure of Figure 1.

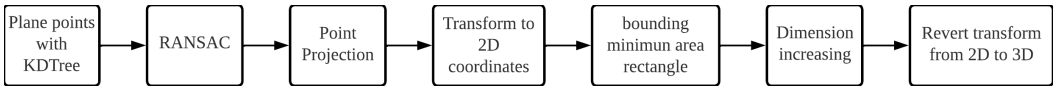


Figure 1. Block diagram with the steps to obtain the 3D LiDAR coordinates of the corners of the rectangular polygon.

The first step is to segment which points in the cloud correspond to the reference rectangle. The calibration rectangle is considered to produce a set of N points reflected in the LiDAR ${}^{L}P = \{{}^{L}\vec{p}_1, {}^{L}\vec{p}_2, \ldots, {}^{L}\vec{p}_N\}$. To this end, by visual inspection of a range image, such as the one in Figure 2, any point in the ${}^{L}\vec{p}_i$ plane is manually selected. By using a $kd-tree$ of the point cloud, points neighboring ${}^{L}\vec{p}_i$ that are within a radius r are found and added to the set of points ${}^{L}P$. The process is repeated for each added point ${}^{L}\vec{p}_j$ of ${}^{L}P$ until no neighboring points ${}^{L}\vec{p}_i$ are found within r. The process is described in detail in Algorithm 1. For this method to converge, it is necessary that the rectangle is not in contact with any other surface, so it has to be completely isolated since no features are considered to differentiate points on the calibration rectangle from other surrounding objects.

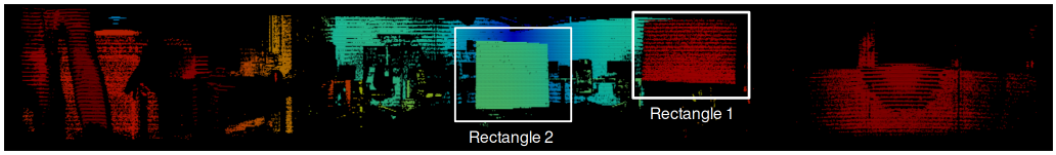


Figure 2. Equirectangular projection of a point cloud with two reference rectangles with a reflectivity filter applied. The black area denotes the absence of points. The color map ranges from red for closer points to blue for more distant points.

The estimation of the equation of the plane defined by the set of points of the set ${}^{L}P$ is carried out in order to continue the process. By using a RANSAC algorithm [12], an equation of the plane is obtained, in which the points ${}^{L}P$ are projected and from which its normal vector is extracted. Afterwards, by obtaining the normal vector of the plane, the reference system of the plane is changed to work on the set of projected points ${}^{L}P'$ in 2D

coordinates, removing the coordinate that would correspond to the depth, as shown in Figure 3.

Algorithm 1 Algorithm to obtain the LiDAR points belonging to the rectangle.

Require: ${}^{L}P$ (initial selection), *kdtree*, *r*
Ensure: ${}^{L}P$
1: ${}^{L}P_i$ = FindRadiusPoints($kdtree, r, {}^{L}P$)
2: **while** size of ${}^{L}P_i$ is not 0 **do**
3: **for** ${}^{L}p_i$ in ${}^{L}P_i$ **do**
4: ${}^{L}P_j$ = FindRadiusPoints($kdtree, r, {}^{L}p_i$)
5: ${}^{L}P'_i \leftarrow {}^{L}P'_i \cup {}^{L}P_j$
6: **end for**
7: DeleteRepeatedPoints(${}^{L}P'_i$)
8: ${}^{L}P_i \leftarrow {}^{L}P'_i$
9: ${}^{L}P \leftarrow {}^{L}P \cup {}^{L}P_i$
10: **end while**

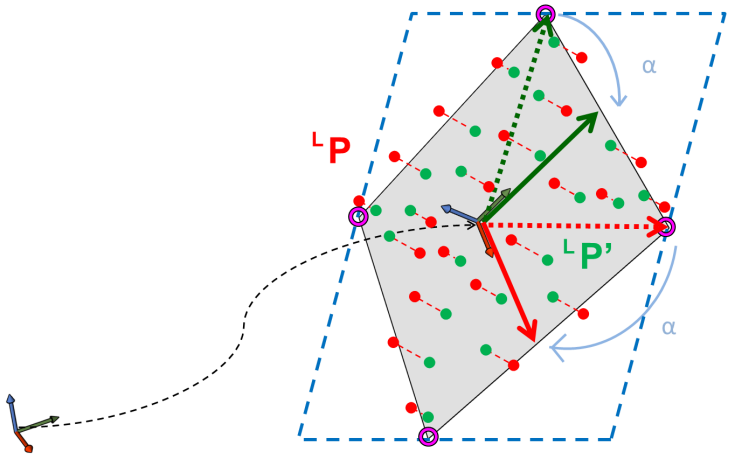


Figure 3. Representation of the calculation of the 3D coordinates of the corners of the rectangular polygon. The LiDAR points are projected onto the plane obtained with RANSAC, and then the corners of the rectangle are obtained.

Finally, the minimum area rectangle that surrounds this set of points ${}^{L}P'$ is calculated. For this purpose, ${}^{L}P'$ is repeatedly rotated by an angle $\alpha \in [0, \pi/2]$, obtaining the maximum and minimum coordinates of each axis that would determine a rectangular polygon containing the rotated set ${}^{L}P'$. The rectangle of minimum area will have an orientation $\rho = -\alpha_{Area_{\min}}$, where $\alpha_{Area_{\min}}$ corresponds to the rotation of ${}^{L}P'$ in which the rectangular polygon that bounds the points minimizes its area, this being area calculated from the maximum and minimum coordinates in x and y. Lastly, keeping the orientation of the bounding rectangle of minimum area and maintaining its center fixed, the width and height dimensions of the rectangle are increased, if necessary, by making them equal to those of the actual reference pattern. Once the 2D corner positions are known, the previously applied inverse transform is performed to obtain the 3D coordinates of the global system, finally obtaining the 3D coordinates of the corners of the rectangle.

2.2. Three-Dimensional Camera Coordinates

To find the coordinates of the corners of a rectangle in space from a fisheye image, a problem based on an overdetermined non-linear system of equations is stated.

The dimensions of the rectangle are known, and this information provides equations for the distance between the corners. In addition, the projection of the fisheye image to a

unit sphere (Figure 4) can be found from a function that requires the calibration parameters of the camera.

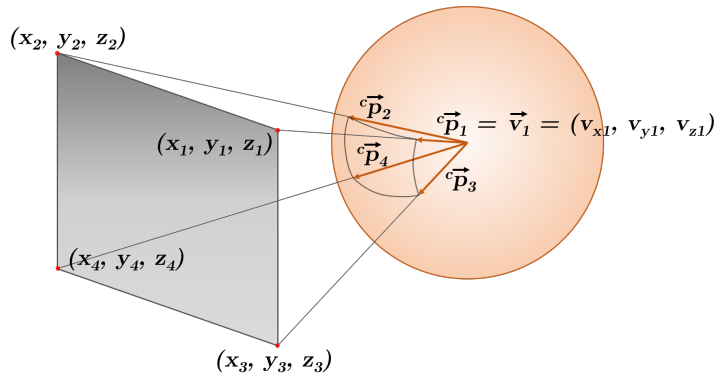


Figure 4. Representation of the rectangle projection on the unit sphere.

The process of obtaining the 3D coordinates of the points belonging to the calibration pattern is described in the next subsection.

2.2.1. Rectangle Detection on the Image

The first step is to find the rectangle corners in the image. To accomplish this, Segment Anything [25] is used to segment the rectangle. Once the rectangle mask is obtained, a small dilation is applied to this mask to effectively extract the entire rectangle. After that, the k-means algorithm is applied to divide the points into four groups in order to obtain the points of the mask contour farthest from the centroid. Via RANSAC, selecting for each corner one of the points of its group, the points that maximize the sum of the six possible distances between them are obtained. Finally, a small window around each point is selected, and the final characteristic point is the mean between the point extracted with RANSAC and the point from the contour of the mask inside the window, whose distance to the corner of the window closer to the center of the mask is maximum.

2.2.2. Point Projection on the Unit Sphere

The method described in Scaramuzza et al. [37] is used to obtain the calibration parameters and to transform the 2D coordinates of the image to 3D. Using several functions that require these parameters and the 2D coordinates of the image, the projection of each pixel to the unit sphere, which is a unit vector in the 3D direction of each point of the image, is determined.

$$ {}^c\vec{p}_i = (v_{x_i}, v_{y_i}, v_{z_i}) \qquad \forall\, i \in [1,2,3,4] \tag{1} $$

The equations of the line that passes through the origin and the corners of the rectangle are obtained by using the unit vector ${}^c\vec{p}_i$ and the origin point $p_0 = (x_0, y_0, z_0)$, where all values are 0. The parametric equations of the line are

$$ \begin{cases} x_i = x_0 + v_{x_i} t \\ y_i = y_0 + v_{y_i} t \\ z_i = z_0 + v_{z_i} t \end{cases} \qquad \forall\, i \in [1,2,3,4] \tag{2} $$

Thanks to the unit vector ${}^c\vec{p}_1$ (1), visible in Figure 4, the 3D coordinates of the corners of the rectangle lie on the line defined with the unit vector and origin point $p_0 = (x_0, y_0, z_0) =$

$(0,0,0)$, so they end up disappearing. After that, clearing t from the parametric equations of a line (2) leaves two linearly independent Equation (3):

$$\begin{cases} x_i v_{y_i} = y_i v_{x_i} \\ x_i v_{z_i} = z_i v_{x_i} \end{cases} \quad \forall\, i \in [1,2,3,4] \tag{3}$$

2.2.3. System of Equations

The final objective is to find the coordinates of the four points in space, corresponding to the corners of the reference rectangle, providing a total of twelve unknowns, three coordinates for each of the four points. The equations of the lines developed in Equation (2) provide a total of two equations per point (3), which makes a total of eight equations for the four corners of the rectangle.

Since it is a plane, a new equation can be created to restrict that all the points belong to the same plane. To achieve this, the following Equation (4) is proposed:

$$\begin{vmatrix} x_1 - x_2 & x_1 - x_3 & x_1 - x_4 \\ y_1 - y_2 & y_1 - y_3 & y_1 - y_4 \\ z_1 - z_2 & z_1 - z_3 & z_1 - z_4 \end{vmatrix} = 0 \tag{4}$$

As discussed above, the dimensions of the rectangular polygon (width w, height h, and diagonal d) are known, so a total of six equations can be stated (5) based on the distances between the points:

$$\begin{cases} w^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 \\ w^2 = (x_3 - x_4)^2 + (y_3 - y_4)^2 + (z_3 - z_4)^2 \\ h^2 = (x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2 \\ h^2 = (x_2 - x_4)^2 + (y_2 - y_4)^2 + (z_2 - z_4)^2 \\ d^2 = (x_1 - x_4)^2 + (y_1 - y_4)^2 + (z_1 - z_4)^2 \\ d^2 = (x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2 \end{cases} \tag{5}$$

Finally, a total of 15 equations (8 equations from (3), 1 from (4), and 6 from (5)) for 12 unknowns have been formulated, resulting in an overdetermined system. The system can be solved by least squares. In this case, the function `optimize.fsolve` from the Scipy Python library is used, which requires an initial estimation of the roots. The 3D coordinates of the corners obtained from the LiDAR are chosen as the initial estimation, thus eliminating negative solutions on the depth axis of the LiDAR reference system. Thus, the system of equations is solved, obtaining the 3D coordinates of the corners of the rectangle in the camera reference system, $(x_i, y_i, z_i) \forall i \in [1,2,3,4]$.

It should be pointed out that this initial estimation could not be used to solve the system if the LiDAR and the camera had a different orientation.

2.3. Kabsch Algorithm

The relative transformation between the two systems is calculated using the Kabsch algorithm [35] once the two sets of points in the 3D space are obtained (one expressed in the camera reference system and the other in the LiDAR reference system).

The Kabsch algorithm calculates the optimal rotation and translation between two systems by minimizing the mean square error deviation, returning three rotation angles (α, β, γ) and three translation values (x, y, z).

2.4. Experiments

In the next subsections, the software developed to test the proposed method is described, as well as the equipment used in the experiments and the simulation environment.

2.4.1. Developed Software

A Python-based application has been developed in order to test the proposed method. In Figure 2, a range image is presented, where each pixel has a value proportional to the distance from the point to the origin of the reference system of the LiDAR sensor. A color map has been used to better highlight the distances. The developed application enables the user to manually select the calibration rectangle (by using the GUI provided in the application). Then, Algorithm 1 is used to obtain the points of the rectangle. Subsequently, once the rectangular polygon is detected, there is an option to visualize the position of the rectangle corners with the LiDAR reference system next to the set of points of the rectangle (Figure 5).

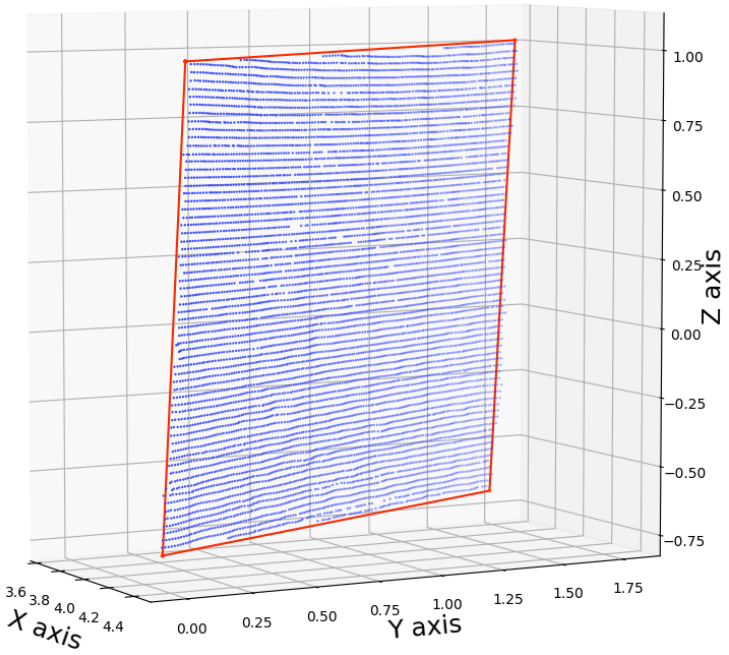


Figure 5. Set of points with LiDAR reference system belonging to the rectangular polygon in blue and estimation of the position of corners and edges in red.

Finally, the fisheye image is shown (Figure 6) to manually select the rectangle. In this step, the software developed in [25] is used and the position of the corners in the image is obtained (please refer to Section 2.2.1). Solving the system of equations proposed in Section 2.2.3, the 3D coordinates of the corners relative to the image are finally obtained. After calculating the transformation between camera and LiDAR in Section 2.3, the LiDAR information is projected onto the image to visually check that the calibration has been performed correctly.

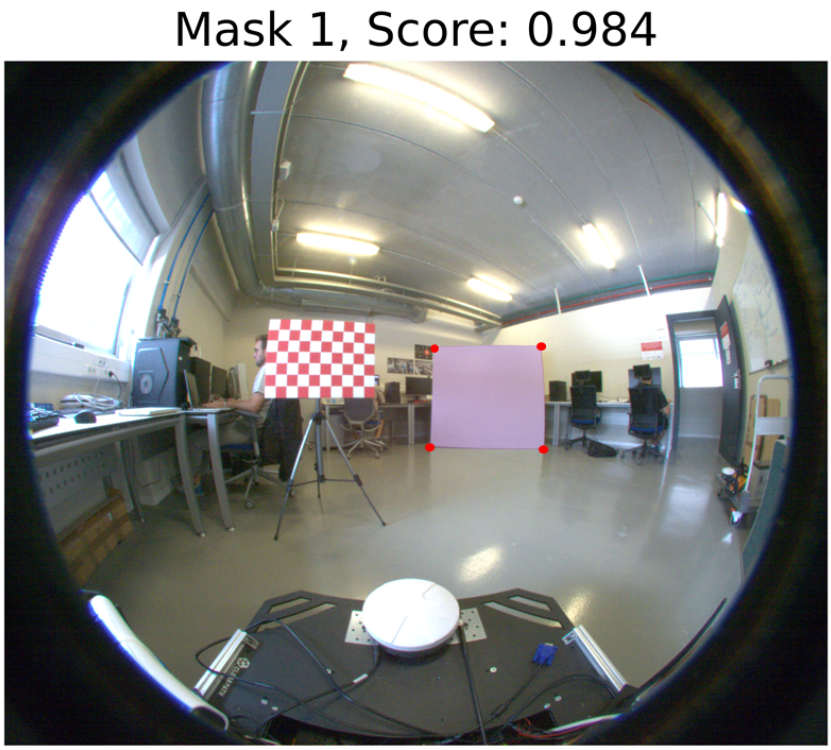


Figure 6. Image captured by the fisheye camera with the calibration rectangle segmented. The corners of the rectangular polygon are shown in red.

2.4.2. Equipment

To conduct the experiments, a Basler a2A2600-20gcBAS (https://docs.baslerweb.com/a2a2600-20gcbas, accessed on 16 January 2025) camera was used, with a FE185C057HA-1 fisheye lens with a field of view of 185° and an Ouster OS1-128 (https://data.ouster.io/downloads/datasheets/datasheet-revd-v2p0-os1.pdf, accessed on 16 January 2025) LiDAR with 128 channels, a range of view of 45°, and up to 90 m range. The setup used in the experiments is shown in Figure 7.

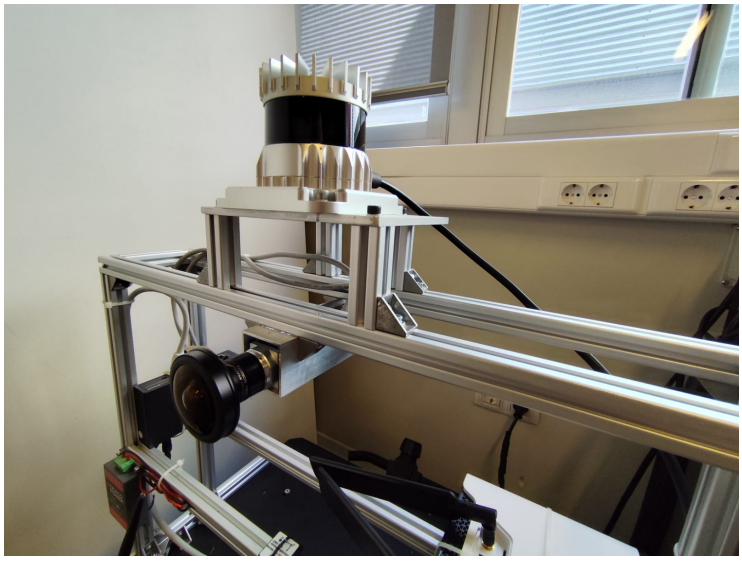


Figure 7. Setup used in the experiments. The LiDAR placed above the camera.

2.4.3. Simulation

A simulation was carried out using CoppeliaSim (https://www.coppeliarobotics.com/), as shown in Figure 8, in order to test the method in a controlled environment. The simulation consists of a LiDAR sensor, an ideal spherical image sensor with a horizontal field of view of 360° and vertical field of view equal to 180°, and two rectangular polygons with the same dimensions as the ones used in the real environment experiments.

This being a simulation, the images and point clouds obtained from the simulation are ideal, so exists no noise effects or unwanted image distortions.

In this case, the spherical projection is obtained directly from the image sensor, so the step to convert the image from a fisheye model to a spherical projection is not necessary.

The LiDAR and camera systems are placed at a constant distance from each other in order to recreate a similar environment to the real one, and, at the same time, the rectangles are placed at distances and orientations similar to those used in the real environment. The transformation between the LiDAR and the camera is known, so the results obtained can be compared with the ground truth.

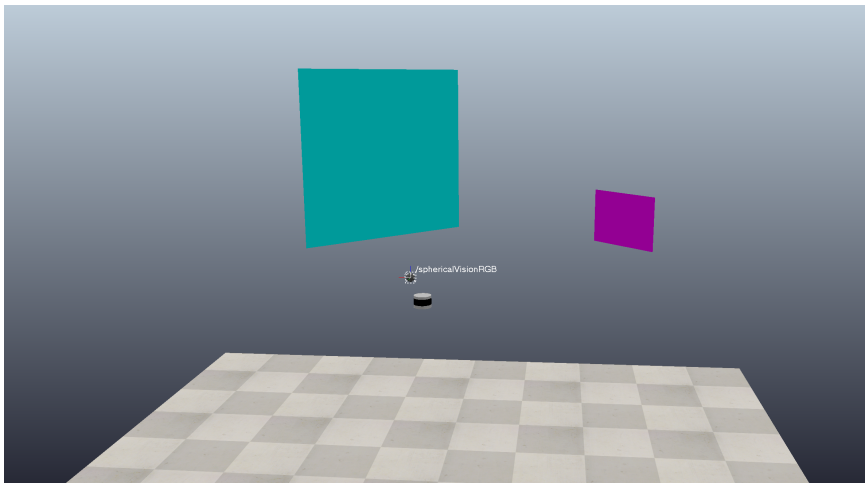


Figure 8. CoppeliaSim simulation environment.

The number of rectangular polygons and their dimensions are the same as the ones used in the real environment in order to imitate the real environment. With two rectangles of different dimensions, set up in different positions and orientations, the aim is to fill the maximum space in the images and point clouds so that the calibration can be performed with the maximum possible information.

3. Results

In this section, the results of the experiments are presented. The method was tested in a real environment and in a simulated environment. In the real environment, the transformation between the LiDAR and the camera is not known, so the results obtained cannot be compared with the actual transformation. However, the reprojection error can be used to evaluate the quality of the calibration. In the simulated environment, the transformation between the LiDAR and the camera is known, so the results obtained can be compared with the actual transformation.

3.1. Simulation

Ten pairs of images and point clouds were used in several experiments to obtain the simulation results. In these experiments, the position of the camera remained fixed, while the LiDAR was moved in the x, y, and z axes in a 3D grid with a step of 0.15 m from −0.45

m to 0.45 m with the objective of testing if the method is sensitive to the relative distance between both systems. The resolution of the images is 2160 × 1080 pixels. With the aim of speeding up the simulation, instead of using Segment Anything [25], the masks of the rectangles were obtained directly by selecting those pixels that are not black in the image, as shown in Figure 9, while the rest of the process is the same as in the real environment.

In this simulation, the proposed method is compared with a minimization method that minimizes the reprojection error based on the method proposed in [38]. For this approach, an initial solution is also provided, and the optimization equation is solved using the Scipy library in Python. The reprojection error is calculated as the sum of the distances between the projected LiDAR points and the corners of the rectangle in the camera image. The results of the simulation are shown in Table 1.

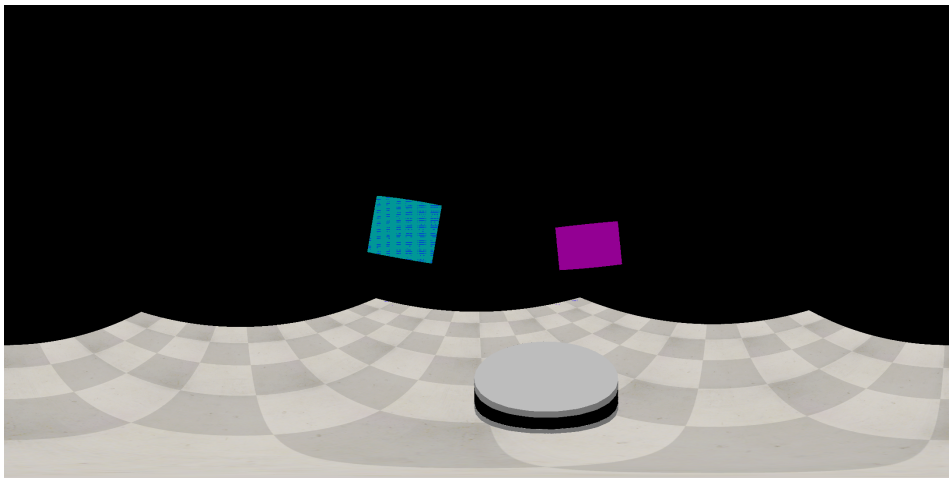


Figure 9. Spherical image obtained from the simulation.

Table 1. Results from the proposed method and the minimization method in the simulated environment. MPE is the mean pixel error of the selected corners of the rectangles in the point cloud once they are reprojected onto the equirectangular image, which corresponds to the spherical projection.

Method	Mean Rotation Error (Degrees)	Mean Translation Error (Centimeters)	MPE	Mean Time (Seconds)
Proposed method	0.0387	0.7135	0.6516	0.0444
Minimization method	0.0245	0.3233	0.5275	148.457

In Table 1, the results of the calibration in the simulation environment are shown. Despite the fact that the minimization method is directly based on the minimization of the mean pixel error, the results are not far from those of the minimization method. The maximum difference in the results appears in the translation error: the error produced by the proposed method is more than twice that of the minimization method. While the minimization method takes more time to solve the optimization equation, the proposed method is much faster and provides similar results, the mean pixel error being the most important metric to evaluate the quality of the calibration.

In Figure 10, the results for each distance seem to be more dependent on the quality of the extraction of characteristic points (both curves seem to change similarly for every distance). This could be due to the fact that both sensors in the simulation operate in the same manner, likewise obtaining the 3D coordinates of the simulated environment, while the image sensor reprojects them onto an equirectangular image (spherical projection). It

can be concluded that both methods are not very sensitive to the distance between the camera and the LiDAR.

Figure 10. Rotation, translation, and mean pixel error versus distance between camera and LiDAR.

3.2. Real Environment

For the real environment experiments, a set of 11 images and point clouds was used. The resolution of the fisheye images is 2600 × 2160 pixels, so the resolution of the spherical image, where the error in pixels is measured, is 4320 × 2160 pixels, which is double the size along each axis compared to the spherical image used in the simulation. The positions of the LiDAR and the camera during the experiments remained fixed, while the calibration rectangles were placed at different positions and orientations. The position of the LiDAR is close to the camera, and the rotation between both systems is assumed to be minimal but never 0, and, as stated before, the ground truth transformation is not known.

Two rectangular polygons with different dimensions are used in the experiments in such a way that they can cover the maximum space possible in the images and point clouds, making sure that the most distorted parts of the fisheye image are covered. The dimensions are exactly the same as the ones used in the simulation, one being 0.59 m × 0.41 m and the other 1.89 m × 1.70 m, and they are positioned at different distances from the camera and LiDAR, as can be seen in Figure 11b.

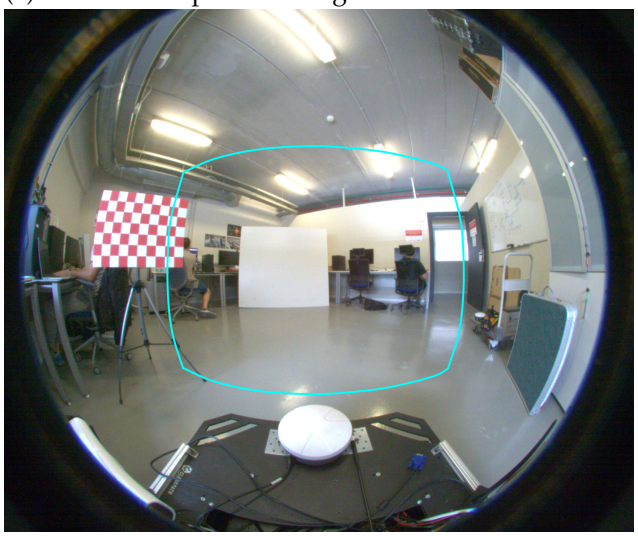
In these experiments, the results cannot be compared with the actual transformation, but the quality of the calibration can be evaluated by calculating the reprojection error, quantified as the MPE (mean pixel error).

Figure 11 shows sample images of the real environment experiment with the two rectangles of different sizes. Initially, Figure 11a is an undistorted pinhole image with a field of view of 90°, while Figure 11b is a fisheye-distorted image where the extra field of view is clearly visible. Figure 11a is shown just for comparative purposes. In the present work, only fisheye images are used.

Table 2 shows the results of the calibration in the real environment. The results of the calibration using the proposed method and the minimization method are shown. The MPE is the mean pixel error of the selected corners of the rectangles from every point cloud once

they are reprojected onto the equirectangular image, which corresponds to the spherical projection. The standard deviation is also shown. The mean pixel error and the standard deviation cannot be compared directly with the results from Table 1, where the image used to measure them was half the resolution in both axes.

(**a**) Undistorted pinhole image with a field of view of 90°.



(**b**) Fisheye-distorted image.

Figure 11. Images of the real environment experiment where the difference between a pinhole image and a fisheye-distorted image is shown.

Table 2. Results from the proposed method and the minimization method in the real environment.

Method		Axis			MPE (Pixels)	Standard Deviation (Pixels)	Time (s)
		x	y	z			
Proposed method	rotation (degrees)	0.012	−0.638	−0.131	4.421	3.754	0.3004
	translation (cm)	−5.97	0.93	14.40			
Minimization	rotation (degrees)	0.093	−0.735	−0.279	3.691	2.434	597.48
	translation (cm)	−7.318	1.337	13.526			

The first row from Table 2 shows the results from the proposed method where the Kabsch algorithm was used. The rotation for every axis is close to 0, which is expected because the LiDAR and the camera are placed in a similar orientation. The translation is

not very different from the minimization method in the second row, but the MPE from the proposed method is a bit higher, which means that the reprojection error is higher. The time required to calculate the transformation is much lower than for the minimization method, which is a plus for the proposed method.

Once again, as is evident in Table 2, the minimization method provides better results in terms of reprojection error, while the proposed method achieves a very close result in a small fraction of the time. The detection of the corners of the rectangular polygons in the image is not completely perfect and depends on the quality of both the image and the point cloud sensors and the quality of the fisheye lens calibration. This makes it harder to obtain a reprojection error as low as in the simulation. It is also noticeable that the transformation parameters obtained are not very different between the two methods, so the quality of the calibration is not very different between them. Even the rotation parameters are very close to each other.

Figure 12 presents qualitative results. In this figure, the point cloud is shown after being reprojected onto the fisheye image. The points are reprojected onto the image after translating and rotating them with the transformation obtained from the proposed method so that they are in the same reference system as the image. The point cloud is colored based on the distance from the camera, so the points closer to the camera are shown in red and the points further away are shown in blue. The reprojection of the point cloud onto the image is visually correct and enables good fusion of the information from both sensors with a large field of view thanks to the fisheye lens. It is also noticeable that, even though the mean reprojection error is around 1.8 pixels in the equirectangular image, the LiDAR points are correctly reprojected onto the image without much distortion because the MPE (mean pixel error) is measured in the equirectangular image, which is a projection of the fisheye image with a higher resolution, to be compared with the simulation results.

Figure 12. Point cloud projected onto the fisheye image after calculating the transformation between both systems with the minimization method. The color of the points is based on the distance from the camera, so the points closer to the camera are shown in blue and the points further away are shown in red.

The projection of the LiDAR points onto the image is not perfect but accurate enough for the fusion of the information from both sensors. It can be seen in Figure 13 that the points are correctly reprojected onto the image, but some points obtained by the LiDAR sensor are placed over surfaces that they do not belong to. This occurs because the reference system of the LiDAR points is changed for the projection, so, from the perspective of the camera reference system, there will be further points obtained by the LiDAR that will stay behind other closer points sharing the same surfaces.

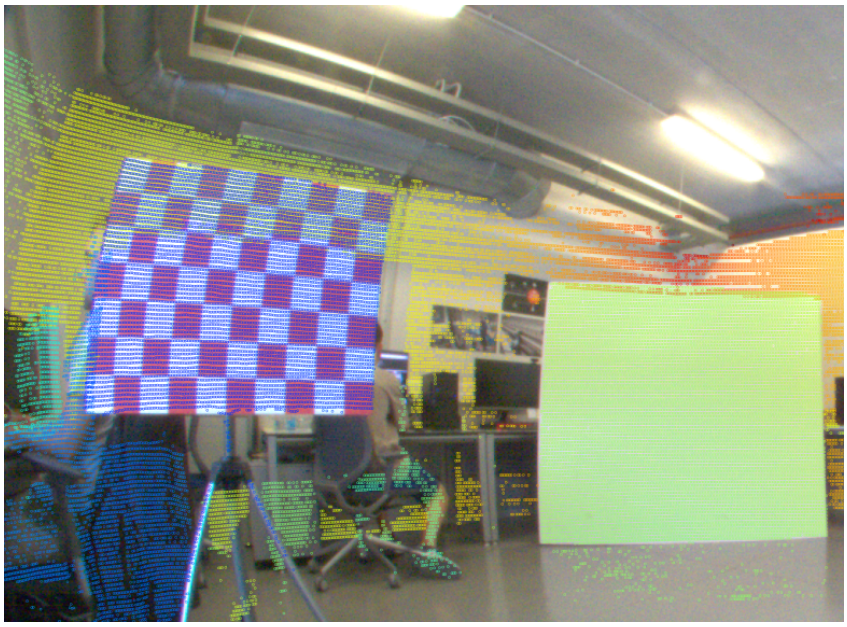


Figure 13. Image from Figure 12 zoomed in.

4. Discussion

The results of the experiments show that the proposed method is able to calibrate the system composed of a fisheye camera and a LiDAR sensor with a high degree of accuracy. The method has been tested both in a simulated environment and in a real environment. First, in the case of the simulated environment, the accuracy of the method has been measured through the mean rotation error, the mean translation error, and the MPE. Second, in the case of the real environment, the actual transformation between the camera and the LiDAR is not known, so the results obtained with the proposed method cannot be compared with this actual transformation. For this reason, we have used the reprojection error as a metric to evaluate the quality of the resulting calibration. In all the cases, these metrics indicate that the error is small enough to be used in applications where a wide field of view is needed. To clearly prove the validity of the approach, Figures 12 and 13 show a point cloud projected onto the corresponding fisheye image using the estimated calibration parameters to perform this projection.

The results obtained are of great importance for the development of autonomous vehicles or mobile robots where the priority is to obtain a wide field of view without the need for a great deal of image sensors since the proposed method permits the fusion of the information from both sensors with a high degree of accuracy. These fused data can subsequently be useful for processing by neural networks or any other algorithms that require a wide field of view to work properly and provide complete information regarding the surroundings of the sensors. Additionally, it is worth highlighting that the time needed to calculate the transformation with the proposed method is much lower than the minimization method, which is an advantage for the proposed method.

The limitations observed during the experiments are related to the quality of corner detection in the image, which depends on the specifications of the sensor and the calibration of the fisheye lens. This has an influence on the quality of the fusion of the information from both sensors, which is lower in the case of the real environment than under simulation, as expected.

5. Conclusions

In this paper, a method for the calibration of a system composed of a fisheye camera and a LiDAR sensor has been proposed. The proposed framework permits obtaining the calibration parameters (i.e., the transformation matrix between the reference frames of the fisheye camera and the LiDAR sensor) without the need to project the fisheye image to a pinhole model, leading to a computationally efficient algorithm.

The method has been tested in a simulated environment and in a real environment, where the results obtained show that the proposed method is able to calibrate the system with a high degree of accuracy. The reprojection error is used as a metric to evaluate the quality of the calibration, and the results obtained are of great importance for the development of autonomous systems that require a wide field of view. The time needed to calculate the transformation is substantially lower than the minimization method, which is a plus for the proposed method.

The method is a relevant contribution to the field of autonomous systems as it enables the fusion of the information from a fisheye camera and a LiDAR sensor. The experimental section has demonstrated that the results obtained by the proposed calibration method permit fusing the information from the camera and LiDAR accurately despite the great distortion present in the fisheye image, obtaining a combination of visual and distance information from the environment with a large field of view.

In the future, the method could be improved by using algorithms that can detect the corners of the rectangles in the image more accurately and in a completely automatic way so that the reprojection error can be reduced. The fusion of the information from both sensors will also enable us to perform additional developments, like semantic segmentation, object detection, object tracking, or even 3D reconstruction of the environment. Finally, we plan to compare the relative performance obtained in these tasks with this proposal compared to the case of fusing LiDAR and a pinhole camera.

Author Contributions: Conceptualization, Á.M.; methodology, Á.M.; software, Á.M. and A.S.; validation, Á.M.; formal analysis, Á.M.; investigation, Á.M.; resources, M.B., A.G. and L.P.; data curation, Á.M.; writing—original draft preparation, Á.M.; writing—review and editing, A.G. and L.P.; visualization, Á.M.; supervision, M.B., A.G. and L.P.; project administration, M.B., A.G. and L.P.; funding acquisition, M.B., A.G. and L.P. All authors have read and agreed to the published version of the manuscript.

Funding: MICIU/AEI/10.13039/501100011033: TED2021-130901B-I00; European Union NextGenerationEU/PRTR: TED2021-130901B-I00; MICIU/AEI/10.13039/501100011033: PID2023-149575OB-I00; FEDER, UE: PID2023-149575OB-I00.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study is publicly available on free repositories.

Acknowledgments: This research work is part of the project TED2021-130901B-I00 funded by MICIU/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. It is also part of the project PID2023-149575OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by FEDER, UE.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MPE	Mean Pixel Error
LiDAR	Light Detection And Ranging
ICP	Iterative Closest Point
RANSAC	RANdom SAmple Consensus

References

1. Wang, Z.; Huang, Z.; Gao, Y.; Wang, N.; Liu, S. MV2DFusion: Leveraging Modality-Specific Object Semantics for Multi-Modal 3D Detection. *arXiv* **2024**, arXiv:2408.05945.
2. Zhang, H.; Liang, L.; Zeng, P.; Song, X.; Wang, Z. SparseLIF: High-performance sparse LiDAR–camera fusion for 3D object detection. In Proceedings of the European Conference on Computer Vision, Milan, Italy, 29 September–4 October 2024; pp. 109–128.
3. Abdelkader, A.; Moustafa, M. *Camera and LiDAR Fusion for Point Cloud Semantic Segmentation, Proceedings of the Seventh International Congress on Information and Communication Technology, London, UK, 21–24 February 2022* ; Yang, X.S., Sherratt, S., Dey, N., Joshi, A., Eds.; Springer Nature: Singapore, 2023; pp. 499–508.
4. Wu, H.; Wen, C.; Shi, S.; Li, X.; Wang, C. Virtual Sparse Convolution for Multimodal 3D Object Detection. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–23 June 2023; pp. 21653–21662. [CrossRef]
5. Tan, Z.; Zhang, X.; Teng, S.; Wang, L.; Gao, F. A Review of Deep Learning-Based LiDAR and Camera Extrinsic Calibration. *Sensors* **2024**, *24*, 3878. [CrossRef] [PubMed]
6. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]
7. Zhang, Z.; Tang, Q. Camera Self-Calibration Based on Multiple View Images. In Proceedings of the 2016 Nicograph International (NicoInt), Hanzhou, China, 6–8 July 2016; pp. 88–91. [CrossRef]
8. Zhao, Y.; Wang, H. Conic and circular points based camera linear calibration. *J. Inf. Comput. Sci.* **2010**, *7*, 2478–2485.
9. Ishikawa, R.; Oishi, T.; Ikeuchi, K. LiDAR and Camera Calibration Using Motions Estimated by Sensor Fusion Odometry. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 7342–7349. [CrossRef]
10. Lv, X.; Wang, B.; Dou, Z.; Ye, D.; Wang, S. LCCNet: LiDAR and Camera Self-Calibration Using Cost Volume Network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Nashville, TN, USA, 20–25 June 2021; pp. 2894–2901.
11. Luo, Z.; Yan, G.; Li, Y. Calib-Anything: Zero-training LiDAR-Camera Extrinsic Calibration Method Using Segment Anything. *arXiv* **2023**, arXiv:2306.02656.
12. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
13. Nistér, D.; Naroditsky, O.; Bergen, J. Visual odometry for ground vehicle applications. *J. Field Robotics* **2006**, *23*, 3–20. [CrossRef]
14. Alcantarilla, P.F.; Nuevo, J.; Bartoli, A. *Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces, Proceedings of the British Machine Vision Conference, BMVC 2013, Bristol, UK, 9–13 September 2013*; Burghardt, T.; Damen, D.; Mayol-Cuevas, W.W.; Mirmehdi, M., Eds. BMVA Press: Durham, UK, 2013. [CrossRef]
15. Schneider, N.; Piewak, F.; Stiller, C.; Franke, U. RegNet: Multimodal sensor registration using deep neural networks. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 1803–1810. [CrossRef]
16. Iyer, G.; Ram, R.K.; Murthy, J.K.; Krishna, K.M. CalibNet: Geometrically Supervised Extrinsic Calibration using 3D Spatial Transformer Networks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1110–1117. [CrossRef]
17. Ren, F.; Liu, H.; Wang, H. A LiDAR-Camera Joint Calibration Algorithm Based on Deep Learning. *Sensors* **2024**, *24*, 6033. [CrossRef]
18. Pandey, G.; McBride, J.; Savarese, S.; Eustice, R. Automatic Targetless Extrinsic Calibration of a 3D LiDAR and Camera by Maximizing Mutual Information. In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, Toronto, ON, Canada, 22–26 July 2012.
19. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. [CrossRef]

20. Yuan, C.; Liu, X.; Hong, X.; Zhang, F. Pixel-Level Extrinsic Self Calibration of High Resolution LiDAR and Camera in Targetless Environments. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7517–7524. [CrossRef]
21. Liu, X.; Yuan, C.; Zhang, F. Targetless Extrinsic Calibration of Multiple Small FoV LiDARs and Cameras Using Adaptive Voxelization. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–12. [CrossRef]
22. Ma, T.; Liu, Z.; Yan, G.; Li, Y. CRLF: Automatic Calibration and Refinement based on Line Feature for LiDAR and Camera in Road Scenes. *arXiv* **2021**, arXiv:2103.04558.
23. Bai, Z.; Jiang, G.; Xu, A. LiDAR-Camera Calibration Using Line Correspondences. *Sensors* **2020**, *20*, 6319. [CrossRef] [PubMed]
24. Koide, K.; Oishi, S.; Yokozuka, M.; Banno, A. General, Single-shot, Target-less, and Automatic LiDAR-Camera Extrinsic Calibration Toolbox. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 11301–11307. [CrossRef]
25. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment Anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 4–6 October 2023; pp. 4015–4026.
26. Romero-Ramirez, F.; Muñoz-Salinas, R.; Medina-Carnicer, R. Speeded Up Detection of Squared Fiducial Markers. *Image Vis. Comput.* **2018**, *76*, 38–47. [CrossRef]
27. Cui, J.; Niu, J.; Ouyang, Z.; He, Y.; Liu, D. ACSC: Automatic Calibration for Non-repetitive Scanning Solid-State LiDAR and Camera Systems. *arXiv* **2020**, arXiv:2011.08516.
28. Wang, W.; Sakurada, K.; Kawaguchi, N. Reflectance Intensity Assisted Automatic and Accurate Extrinsic Calibration of 3D LiDAR and Panoramic Camera Using a Printed Chessboard. *Remote Sensing* **2017**, *9*. [CrossRef]
29. Dhall, A.; Chelani, K.; Radhakrishnan, V.; Krishna, K.M. LiDAR-Camera Calibration using 3D-3D Point correspondences. *arXiv* **2017**, arXiv:1705.09785.
30. Yoo, J.H.; Jung, G.B.; Jung, H.G.; Suhr, J.K. Camera–LiDAR Calibration Using Iterative Random Sampling and Intersection Line-Based Quality Evaluation. *Electronics* **2024**, *13*, 249. [CrossRef]
31. Jiao, J.; Chen, F.; Wei, H.; Wu, J.; Liu, M. LCE-Calib: Automatic LiDAR-Frame/Event Camera Extrinsic Calibration With a Globally Optimal Solution. *IEEE/Asme Trans. Mechatron* **2023**, *28*, 2988–2999. [CrossRef]
32. Verma, S.; Berrio, J.S.; Worrall, S.; Nebot, E. Automatic extrinsic calibration between a camera and a 3D LiDAR using 3D point and plane correspondences. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 3906–3912.
33. Zhou, L.; Li, Z.; Kaess, M. Automatic Extrinsic Calibration of a Camera and a 3D LiDAR Using Line and Plane Correspondences. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 5562–5569.
34. Xu, J.; Han, D.W.; Li, K.; Li, J.J.; Ma, Z.Y. A Comprehensive Overview of Fish-Eye Camera Distortion Correction Methods. *arXiv* **2024**, arXiv:2401.00442.
35. Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. Sect.* **1976**, *32*, 922–923. [CrossRef]
36. Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 376–380. [CrossRef]
37. Scaramuzza, D.; Martinelli, A.; Siegwart, R. A Toolbox for Easily Calibrating Omnidirectional Cameras. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–16 October 2006, pp. 5695–5701.
38. Yan, G.; He, F.; Shi, C.; Wei, P.; Cai, X.; Li, Y. Joint Camera Intrinsic and LiDAR-Camera Extrinsic Calibration. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May– 2 June 2023; pp. 11446–11452. [CrossRef]