# Joint Intrinsic and Extrinsic Calibration of Perception Systems Utilizing a Calibration Environment

Louis Wiesmann Thomas Läbe Lucas Nunes Jens Behley Cyrill Stachniss

***Abstract*—Basically all multi-sensor systems must calibrate their sensors to exploit their full potential for state estimation such as mapping and localization. In this paper, we investigate the problem of extrinsic and intrinsic calibration of perception systems. Traditionally, targets in the form of checkerboards or uniquely identifiable tags are used to calibrate those systems. We propose to use a whole calibration environment as a target that supports the intrinsic and extrinsic calibration of different types of sensors. By doing so, we are able to calibrate multiple perception systems with different configurations, sensor types, and sensor modalities. Our approach does not rely on overlaps between sensors which is often otherwise required when using classical targets. The main idea is to relate the measurements for each sensor to a precise model of the calibration environment. For this, we can choose for each sensor a specific method that best suits its calibration. Then, we estimate all intrinsics and extrinsics jointly using least squares adjustment. For the final evaluation of a LiDAR-to-camera calibration of our system, we propose an evaluation method that is independent of the calibration. This allows for quantitative evaluation between different calibration methods. The experiments show that our proposed method is able to provide reliable calibration.**

***Index Terms*—Calibration and Identification; Mapping; Sensor Fusion**

## I. Introduction

CALIBRATING the sensors of any robotic system is key for obtaining reliable measurements and crucial for sensor fusion. This is a key prerequisite for tasks such as mapping, localization, or SLAM. Each sensor has typically its own coordinate system, therefore knowing the relative transformations (extrinsics) between those, allows for combining the measurements in a common frame. Intrinsic calibration, on the other hand, is important to have a correct model between the measurements of a sensor and the corresponding object properties in the physical world.

In this paper, we investigate the problem of extrinsic calibration, i.e., estimating the rigid body transformation between the sensors, as well as the intrinsics of each sensor.

Manuscript received: May 28, 2024; Revised: Aug. 7, 2024; Accepted: Aug. 28, 2024. This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – PhenoRob and under STA 1051/5-1 within the FOR 5351 (AID4Crops), by the European Union's Horizon Europe research and innovation programme under grant agreement No 101070405 (DigiForest), and by the German Federal Ministry of Education and Research (BMBF) in the project "Robotics Institute Germany", grant No. 16ME0999.

All authors are with the Center for Robotics, University of Bonn, Germany. Cyrill Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK, and with the Lamarr Institute for Machine Learning and Artificial Intelligence, Germany.

Digital Object Identifier (DOI): see top of this page.

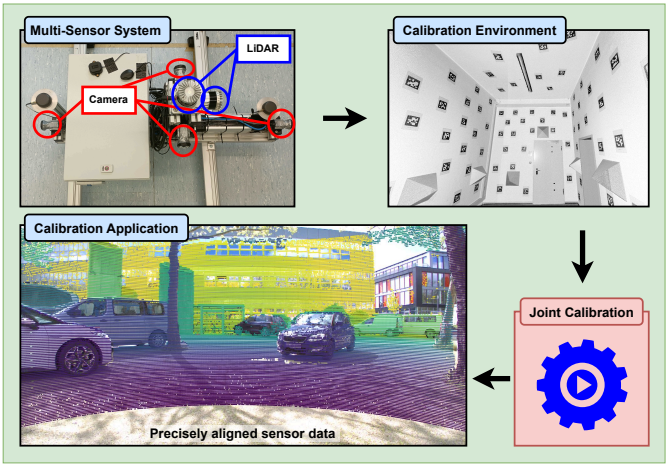



Fig. 1: Having a system that uses multiple sensors requires calibration. Our approach calibrates different perception sensors, like LiDAR sensors or cameras, by utilizing a calibration environment. Jointly optimizing the intrinsics and extrinsics of the sensors allows for different applications, such as sensor fusion. In this application, the scan of the horizontal LiDAR is projected into the left camera, where the colors denote the distance to the sensor.

Sensor calibration is a common problem in robotics [3], [20], [22] and crucial for nearly all multi-sensor systems. Depending on the sensor setup, i.e., the type of sensors, their arrangement, and the required accuracy, one can choose from different calibration procedures. Commonly used sensors for perception platforms are cameras and LiDAR. Having a system comprising multiple cameras allows using a checkerboard or AprilTags [17] for calibration. Depending on the configuration, however, the procedure needs to be adapted, e.g., for a stereo camera system one can exploit the overlapping field of view (FoV) [29], while a multi-camera configuration without overlapping FoVs is likely to require the use of bundle adjustment [23] where for at least some timestamps, multiple cameras see control points. When adding LiDAR sensors into the system, one has further possibilities: (1) operating on the spectral level exploiting the returned intensity values [13], or (2) exploiting the geometric information about the scene it provides [16]. Calibrating a system with a profile scanner [9] might look different from one with a multi-beam LiDAR.

One multi-sensor system that we aim to calibrate is shown in Fig. 1, which consists of four wide-angle cameras and two multi-beam LiDAR. Other robotic systems that we will look at have, e.g., four RGB-D cameras, one RGB camera and two 2D profile LiDARs.The main challenges of these systems are no or limited overlap between cameras and limited overlap between the LiDAR sensors. Furthermore, some of the cameras have a fisheye lens; therefore, we need the calibration able to work for different projection models (projection on a plane or a

sphere) as well. Additionally, we are interested in a calibration procedure that works with multiple sensor setups, is easy-to-extend, and requires minimal user input. We aim at millimeter accuracy and thus to be more precise than the sensors noise.

The main contribution of this paper is a flexible calibration procedure that allows to estimate the intrinsics and extrinsics of a combination of different sensors. Instead of directly estimating the relative transformation between the sensors, we use an external high-accuracy sensor, a terrestrial laser scanner, as a target, we do this once, as a preprocessing step, to measure our calibration environment. For the actual calibration, the multi-sensor setup only needs to be brought into the calibration environment. This allows us to reliably estimate the poses of each sensor in an efficient manner. By doing so, we can exploit the strengths of each sensor individually, making it suitable across varying sensors with different configurations without the need for overlap between the sensors' field of views.

In sum, we make three key claims: Our approach is able to (i) reliably estimate the intrinsics and extrinsics of perception systems; (ii) operate with different sensor types (e.g., LiDAR, and camera) and modalities (e.g., profile scanner vs. multi-beam LiDAR; wide-angle vs. fisheye cameras); (iii) calibrate different perception systems with varying sensor configurations. The implementation of our approach is available at: `https://github.com/PRBonn/ipb_calibration`

## II. Related Work

Approaches for calibration can be divided into two categories: Methods using natural scenes, mostly outdoor, and methods using scenes with specific targets.

Early target-less approaches measured the correspondence between LiDAR and camera image points manually [22]. Recently, LiDAR intensity was used and compared with the camera image intensity for calibration [12], [13], [19]. We aim for an automated approach without manual measurements and do not want to rely on the quality of the LiDAR intensity. Geometrical approaches [11], [16] for natural scene-based calibration utilize the onboard sensors to first estimate the vehicle's trajectory or build a map, which can then be taken for localizing the other sensors. We believe that the calibration using natural scenes require a lot of user input and experience to make it work for different sensors and perception platforms.

Target-based approaches can be characterized by their targets: Planes with black rings [20], trihedrons [8], two planar triangles [5], and spheres [14] are among the less often-used objects. Some works [1], [31] define special objects that have different properties to support the different sensor characteristics. One of the most commonly used calibration targets is the checkerboard. While for the detection in the image standard tools exist [3], for the detection in the LiDAR, the approaches usually vary. Some approaches [10], [28], [34] try to estimate the full geometry of the board while others [18], [25], [27], [32] only use the plane normal and the center point to estimate the transformation since the edges themselves are too inaccurate. Verma et al. [27] use a genetic algorithm for the estimation, while Tsai et al. [25] introduce a quality measure to select a subset of frames used in the estimation procedure. All these approaches have the restriction that the board must be seen completely in both, the LiDAR and the camera, which reduces the possible positions and angles of the board and limits the calibration to systems that have overlap between those sensors. Only a small fraction of the LiDAR point cloud, usually significantly below 25%, can be used to estimate the transformation between the LiDAR and the target. As our target is a 3D terrestrial laser scan of a whole room (cf. Fig. 1), we are able to use nearly *all* points (except for some very small amount of outliers) of a LiDAR scan to estimate the transformation between the LiDAR sensor and the target.

Some approaches [6], [26], to which we would also associate our approach, do not only rely on targets that need to be moved, but rather use an infrastructure-based calibration where the whole environment is the target. The most similar approach compared to ours is the work by Xie et al. [26]. Like us, they use a room with coded targets (AprilTags) on the walls and perform camera-to-camera and camera-to-LiDAR calibration, but they only estimate extrinsic calibration parameters and assume known intrinsics. We estimate all calibration parameters in a joint adjustment instead of separating the estimation for the sensors and estimate the intrinsics. Our calibration room is additionally equipped with pyramids to have additional planes at different angles (c.f., Fig. 1) for a reliable estimation of all 6 extrinsic parameters between point clouds.

## III. Multi-Sensor Calibration using a precise Calibration Environment

In this section, we explain our multi-sensor calibration approach for perception platforms. The main idea for the calibration is to relate the measurements of each sensor to a once created, precise reference map of the calibration environment. The advantage of this is, that we can exploit the strengths of each sensor, do not rely on high FoV overlap between the sensors, and it is applicable to different sensor configurations and types. It substantially simplifies obtaining a high-quality calibration, especially when using multiple different robots or perception platforms. In this work, we look at robots and sensor systems consisting of a combination of different cameras and/or LiDAR sensors.

Our calibration procedure can be summarized in five steps:

1) Generating a reference map of the calibration environment.
2) Defining for each sensor an error function between the observations and the reference map based on the intrinsics and extrinsics of the respective sensor.
3) Collecting measurements from the sensors in the calibration environment.
4) Estimating initial values for the parameters as needed.
5) Performing a joint optimization to obtain the extrinsics and intrinsics of the whole sensor system.

Note that step 1 needs to be done only once for every calibration environment, and step 2 once for each sensor. In the following, we will describe the steps in more detail.

### A. Generating a Reference Map

Calibrating sensors with respect to a reference leads to some requirements on the sensor(s) and the target(s). In our case

the reference is a map of the calibration environment. The reference map should cover most of the scene that will be seen by each individual sensor in the calibration process. We do not rely on overlap in the FoV between the sensors of the multi-sensor system, but between the sensors and the reference map (a complete room in our case) instead. Additionally, the reference map should be as accurate as possible, since errors in the map could propagate into the parameters of the calibration.

Our idea is to rely on transformation estimation between the sensors and the reference map. For the LiDAR, we utilize ICP to the reference map and for the camera images rely on automatically extracted coded targets with given 3D coordinates, i.e., AprilTags [17]. A dense 3D map with the possibility to extract the AprilTag positions is required. Therefore, we propose to use a terrestrial laser scanner (TLS) as sensor to obtain the reference point cloud map. The TLS can produce point clouds with millimeter accuracy, 360° FoV, and with a high density. The 3D coordinates of the coded targets must be in the same reference frame. Thus, we extract them from the reference point cloud map.

For calibrating the LiDAR sensors, the target point cloud needs to have enough geometric structure to reliably fix the 6 degree of freedoms (DoF) of the pose. Since we use an empty room for the calibration, we added some structural elements in the form of pyramids to the wall. By doing so, we can ensure that we have enough information to fix the degrees of freedom along the wall surfaces.

Given the 3D coordinates of AprilTags and their corresponding image coordinates, one can directly use them in a bundle adjustment to obtain accurate poses and intrinsics of the cameras. We directly use the positions of the AprilTags that can be extracted from the TLS point cloud. As this point cloud is highly dense, the code of the AprilTags is clearly visible in the intensity channel of the scan. Thus, we create an image with orthographic projection (also called orthophoto) of each wall in the room using the intensity channel. Then, we use the standard AprilTag library [24] to extract the 2D subpixel-accurate image coordinates of the AprilTag corners in the orthophotos. As every pixel in the orthophoto has its corresponding 3D coordinate in the TLS cloud, we can easily extract the 3D coordinates of the AprilTag corners by bilinear interpolation at high precision.

Thus, our reference map consists of a point cloud $M = \{({}^m\boldsymbol{p}_i, {}^m\boldsymbol{c}_i)\}$ with $i = 0, \dots, I$ points $\boldsymbol{p}_i \in \mathbb{R}^3$ in Euclidean coordinates with their associated intensity $\boldsymbol{c}_i \in \mathbb{R}$ and a set of $J$ AprilTag corner coordinates, both located in the coordinate frame of the reference map $m$. The coordinate frame of $m$ can be chosen freely, and might be the origin defined by the terrestrial laser scanner's internal frame.

### B. Define Error Functions

To calibrate the sensor setup, we need to define the error function that we want to optimize. Generally spoken, we want to minimize for all sensors, at each timestamp, the errors between their observations and the reference map. In this work, we will focus on LiDAR, and camera sensors, but the procedure can be used for different sensors as long as one can relate the sensor measurements to the reference map. The relation is usually simply transforming sensor observations or their corresponding part from the reference map in a same frame and computing the deviation between those.

In the following, we will denote the transformation of the frames by the right subscript and left superscript as commonly used in physics, e.g., the transformation of a point from the frame $i$ to frame $j$ would be ${}^j\boldsymbol{p} = {}^j\boldsymbol{R}_i\, {}^i\boldsymbol{p} + {}^j\boldsymbol{t}_i$, where ${}^j\boldsymbol{R}_i \in \mathbb{R}^{3\times3}$ is the rotation matrix and ${}^j\boldsymbol{t}_i \in \mathbb{R}^3$ the translation vector. 3D point coordinates will be denoted by $\boldsymbol{p} \in \mathbb{R}^3$, while image coordinates have the variable $\boldsymbol{x} \in \mathbb{R}^2$.

*1) Camera:* For calibrating the cameras, we relate the sensor observations to the reference map by using AprilTags. For the camera $c$ we extract for a specific timestamp $t$ the corners $\{{}^i_t\boldsymbol{x}_j\}$ of all AprilTags that are visible in the current image $i$. As an error metric, we use the reprojection error $e_{\text{camera}}(i,t,j)$ of the AprilTag coordinates, such that for the $j^{\text{th}}$ observation, we yield

$$e_{\text{camera}}(i,t,j) = {}^i_t\hat{\boldsymbol{x}}_j - {}^i_t\boldsymbol{x}_j \tag{1}$$

$${}^i_t\hat{\boldsymbol{x}}_j = \boldsymbol{F}_c\, {}^d_t\hat{\boldsymbol{x}}_j + {}^i\boldsymbol{t}_d \tag{2}$$

$${}^d_t\hat{\boldsymbol{x}}_j = \text{distort}({}^u_t\hat{\boldsymbol{x}}_j) \tag{3}$$

$${}^u_t\hat{\boldsymbol{x}}_j = \text{project}({}^c_t\hat{\boldsymbol{p}}_j) \tag{4}$$

$${}^c_t\hat{\boldsymbol{p}}_j = {}^b\boldsymbol{R}_c^\top\, {}^b_t\hat{\boldsymbol{p}}_j - {}^b\boldsymbol{R}_c^\top\, {}^b\boldsymbol{t}_c \tag{5}$$

$${}^b_t\hat{\boldsymbol{p}}_j = {}^m_t\boldsymbol{R}_b^\top\, {}^m\hat{\boldsymbol{p}}_j - {}^m_t\boldsymbol{R}_b^\top\, {}^m_t\boldsymbol{t}_b. \tag{6}$$

Namely, we transform the AprilTag coordinate ${}^m\hat{\boldsymbol{p}}_j \in \mathbb{R}^3$ in Eq. (6) first from the frame of the reference map $m$ over the base-link $b$ (a local coordinate system on the robot) into its camera frame $c$ using Eq. (5). From there, we project the point in Eq. (4) depending on the type of camera into a unified intrinsic free camera frame $u$ in which we apply the non-linear camera distortions, i.e., as seen in Eq. (3). After the distortion, we obtain in Eq. (2) the AprilTag coordinate ${}^u_t\hat{\boldsymbol{x}}_j \in \mathbb{R}^2$ in the image frame $i$ by applying the focal length $\boldsymbol{F}_c \in \mathbb{R}^{2\times2}$ and the principal point ${}^i\boldsymbol{t}_d \in \mathbb{R}^2$. The focal length matrix $\boldsymbol{F}_c$ is a diagonal matrix with $[f_x, f_y]$ on the main diagonal. For the distortion, we use tangential and radial distortions similar to OpenCV [3]:

$${}^d_t\hat{\boldsymbol{x}}_j = \text{distort}({}^u_t\hat{\boldsymbol{x}}_j) \tag{7}$$

$${}^d_t\hat{\boldsymbol{x}}_j = {}^u_t\hat{\boldsymbol{x}}_j(1 + \sum_{n=1}^{N} k_{n,c} r^{2n})^\tau + 2\, {}^u_t\hat{\boldsymbol{x}}_j\, {}^u_t\hat{\boldsymbol{x}}_j^\top \boldsymbol{p}_c + r^2\boldsymbol{p}_c, \tag{8}$$

with the radial coefficients $\{k_{n,c}\}$, and the tangential coefficients $\boldsymbol{p}_c = [p_2, p_1]^\top$. The parameter $\tau$ can be changed for different radial distortion modeling, i.e., the classical Brown's distortion model has $\tau = 1$, while the division model has $\tau = -1$. For the projection, we either use the classical pinhole or equidistant model [30] as follows:

$$\text{project}\left(\begin{bmatrix}x\\y\\z\end{bmatrix}\right) = \begin{cases} \begin{bmatrix}\frac{x}{z} & \frac{y}{z}\end{bmatrix}^\top & \text{if } c \text{ is pinhole} \\ \begin{bmatrix}\frac{x}{r_{xy}}\text{atan2}\left(r_{xy}, z\right) \\ \frac{y}{r_{xy}}\text{atan2}\left(r_{xy}, z\right)\end{bmatrix} & \text{if } c \text{ is fisheye,} \end{cases} \tag{9}$$

with the Euclidean distance in the image $r_{xy} = \sqrt{x^2 + y^2}$.

Additionally, we add a prior on the AprilTag coordinates $\hat{\boldsymbol{p}}_j$, defined by:

$$e_{\text{prior}}(j) = \hat{\boldsymbol{p}}_j - \boldsymbol{p}_j^{(0)}, \tag{10}$$

where $\boldsymbol{p}_j^{(0)}$ denotes the initially extracted AprilTag coordinates from the TLS map. By this, we can incorporate the uncertainty in the AprilTag extraction without giving too much freedom for pushing errors from the camera model into the AprilTag coordinates.

*2) LiDAR:* For estimating the extrinsics and intrinsics of the LiDAR sensors, we try to align the point clouds as good as possible with the reference map. Therefore, this part is similar to classical point cloud registration methods [2]. We use the classical point-to-plane error function, as often used in ICP [4]. For this, we compute for the dense and accurate reference map the normals for each point based on the local neighborhood of each point, respectively. The key difference from most ICP-based methods is, that we do not try to independently align each point cloud with the reference, but jointly optimize all sensors and scans together. Thus, we optimize not only one pose per scan, but the whole kinematic chain. This results in

$$e_{\text{LiDAR}}(l,t,j) = {}^{m}\boldsymbol{n}_k^{\top}({}^{m}_{t}\boldsymbol{R}_b\,{}^{b}\hat{\boldsymbol{p}}_j + {}^{m}_{t}\boldsymbol{t}_b - {}^{m}\boldsymbol{p}_k) \tag{11}$$

$${}^{b}\hat{\boldsymbol{p}}_j = {}^{b}\boldsymbol{R}_l\,{}^{l}\boldsymbol{p}_j \left( s_l + \frac{o_l}{\|{}^{l}\boldsymbol{p}_j\|} \right) + {}^{b}\boldsymbol{t}_l, \tag{12}$$

where ${}^{m}\boldsymbol{p}_k$ and ${}^{m}\boldsymbol{n}_k$ are the corresponding points and normals of the j[th] source point ${}^{l}\boldsymbol{p}_j$.

We estimate as intrinsics a scale factor $s_l$ and offset $o_l$ for each LiDAR to address systematics in the range measurements. The correspondences are obtained by searching for each LiDAR point ${}^{l}\boldsymbol{p}_j$ the closest point in the reference map $M$. Due to structural elements, like the pyramids in our reference, we are able to use this procedure not only for 3D multi-beam LiDAR sensors, but also for the commonly used 2D profile LiDAR.

### C. Collecting Measurements

For our calibration setup, the measuring process is easy: we only assume that the measurements from each sensor are obtained at discrete points in time. By this, we can optimize the pose of the sensor system $r_t$ based on all the observations taken at the same timestamp $t$ from all sensors. This does not necessarily mean that all the sensors need to be hardware-triggered at the exact same time and with the same frame rate; we only need to keep the scene and sensors static while taking the measurements. We strongly suggest recording in a stop-and-go manner, i.e., (1) move the sensor system, (2) measure with each sensor while standing still, and (3) repeat steps 1 and 2 as much as needed. Thereby, we also avoid the motion distortion in the measurements, e.g., motion blur in the cameras, rolling shutter effects, or motion distortion in the LiDAR scans.

For an optimal calibration result, the observations should cover the full field of view of the sensor. For example for a camera, one should not have only observations in the center but rather distributed over the whole image. In our room, this recommendation is more or less fulfilled automatically, because all walls, including the ceiling have a sufficient coverage of AprilTags.

Since we do not rely on any human interaction like moving checkerboards, but only on a static environment (like a separate calibration room), this method is well suited for full automation, e.g., in an industrial production line.

### D. Estimate Initial Guess

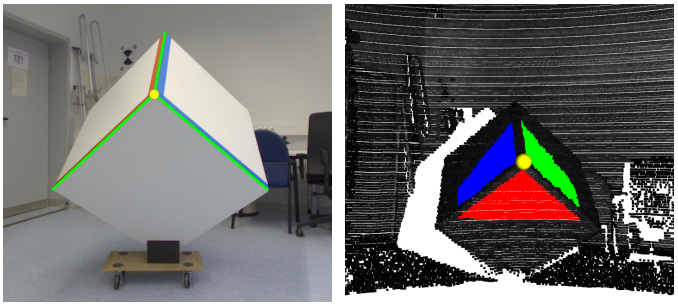
Using the Gauss-Newton model to solve the non-linear optimization problem requires initial values for the parameters. As parameters, we have the 6 DoF pose parameters $\{({}^{m}_{t}\boldsymbol{R}_b, {}^{m}_{t}\boldsymbol{t}_b)\,\forall t\}$, i.e., the transformation parameters from the frame of the base-link $b$ to the frame of the reference map $m$, as well as the extrinsics, i.e., the transformations $\{({}^{b}\boldsymbol{R}_s, {}^{b}\boldsymbol{t}_s)\,\forall s\}$ from the sensor frame $s$ into the base-link frame $b$. We choose the first camera as base-link, but this choice is arbitrary. Additionally, we need for each sensor the intrinsics, e.g., focal length, principal point, and distortion coefficients for each camera, as well as scale and offset for each LiDAR. The offset can model a bias in the range measurements, while the scale can compensate when the LiDAR sensor systematically over or underestimates the ranges proportional to the distance.

As initial guess for the intrinsics of the LiDAR sensors, we assume $s_l \approx 1$ and offset $o_l \approx 0$. The intrinsics of the cameras are estimated by the well-established method by Zhang [33]. Since this requires all points to lie on a plane, we only use the AprilTags from the wall, which has the most visible tags. We use multiple frames with at least 3 visible tags to ensure a reliable estimation. The initial extrinsics $\{({}^{b}\boldsymbol{R}_s, {}^{b}\boldsymbol{t}_s)\,\forall s\}$ can be taken using construction plans of the multi-sensor system, measuring by hand, or computing the relative transformation between the sensors and the base-link from a direct solution. In our experiments, it was sufficient to provide the extrinsics with a couple of centimeters and degrees accuracy, i.e., a simple ruler is sufficient.

We obtain the poses $\{({}^{m}_{t}\boldsymbol{R}_b, {}^{m}_{t}\boldsymbol{t}_b)\,\forall t\}$ by estimating independently for each timestamp the pose of one of the sensors in the reference map. We use the Perspective-n-Point (PnP) [15] algorithm when taking one/ multiple cameras to estimate the poses of the base-link sensor in the reference map. In the upcoming experiments (Sec. V), we take for each timestamp the camera with the most visible AprilTags to estimate the pose. In the case of calibrating only multiple LiDAR sensors, we can also use global registration techniques. We used, for example, the approach by Rusu et al. [21] that uses feature-based correspondences with FPFH features, and searches for the best fit using RANSAC, which provided a sufficient estimation for an initial guess.

### E. Joint Optimization

To obtain the statistically optimal solution for the calibration parameters, we optimize all the sensors in a joint least squares adjustment. Each sensor is rigidly connected to the platform and thus correlated to the other sensors. We use the Gauss-Newton-Model for optimization. We obtain an estimate of the



(a) Camera image (b) LiDAR point cloud

Fig. 2: Measurement of cube corners for evaluation. In the camera image (a) the corner point (yellow) is the intersection of the image edges (green). In the point cloud (b), the corner (yellow) is the intersection of 3 planes estimated using RANSAC on basis of the red, green and blue points.

(a) Calibration room (b) Reference point cloud

Fig. 3: The calibration environment is equipped with AprilTags for the camera calibration and structural elements for the LiDAR. (a) shows a picture of the room, and (b) shows the corresponding point cloud that is used as a reference target. The point cloud is obtained by a Faro Focus3D-X130 terrestrial laser scanner.

accuracy of the parameters using the inverse of the normal equation system. The covariances of the observations should be chosen such that the standardized residuals are approximately standard normal distributed.

In each iteration of the Gauss-Newton algorithm, we update the correspondences for the point clouds to ensure always having the closest points, as also done in ICP. Errors in the initial parameters can lead to wrong associations. Therefore, we use the Geman-McClure robust kernel to reduce the impact of those points.

Once the Gauss-Newton method is converged, we disable the robust kernel and optimize a second time without the robust kernel, removing outliers that are further away than three times the specified sensors standard deviation ($3\sigma$ bound).

## IV. LiDAR-to-Camera Evaluation Method

Estimating all parameters in a joint least squares adjustment allows for obtaining the analytical covariances, especially when choosing realistic covariances in the optimization. But due to imperfect assumptions about the model, covariances, correlations, and linearization, the estimated covariances might be too optimistic. Thus, we evaluate the system separately. Additionally, it allows us to perform an independent comparison to other calibration methods.

Since we are interested in using the camera data in combination with the LiDAR sensors, for example, to project the points into the images (as shown in Fig. 1), we focus on the analysis of the calibration between these sensors. We propose an evaluation method for the independent assessment of the accuracy between LiDAR sensors and cameras. Finding reliable corresponding points in both sensors allows us to compute the reprojection error.

Throughout our experiments, we saw that picking distinct points in the point cloud or range image of the LiDAR sensors was not precise enough. This is due to the limited resolution of the LiDAR, for an accurate evaluation, the resolution of the evaluation method should be higher than the accuracy of the calibration, otherwise aliasing effects can occur. Therefore, we propose to use a cube to find distinct corners in the image and in the point cloud. By extracting three visible planes of the cube from the point cloud, we can compute the point of intersection. We guide this process by manually selecting a point near the cube corner. The corresponding point in the image can be extracted by finding the three edges between the three planes. One can compute the accuracy of the image by projecting the detected corner from the LiDAR into the image and computing the residual at the intersection point of the three image edges. As another metric, we use the distance between the estimated point in the LiDAR and the viewing ray to the corner in the image. The first gives a residual in pixels, while the latter provides a metric error in 3D space. A visualization of the cube measurements are depicted in Fig. 2. Note that the cube is not needed for calibration, only to evaluate the calibration results independently.

## V. Experimental Evaluation

The main focus of this work is a calibration procedure that reliably works for different sensor setups. In the following, we will first look at the main sensor setup and the used calibration environment. Afterward, we evaluate our proposed method and compare it to other approaches. In the end, we will look at the calibration results of different sensor setups to show that our method is of general use.

### A. Experimental Setup

In this work, we aim at calibrating multi-sensor perception systems with the help of a specifically designed calibration environment. The calibration environment is depicted in Fig. 3 and the main sensor system we will look at can be seen in Fig. 5 (a). It is equipped with four Basler Ace cameras facing the front, left, right and to the rear of the vehicle. Additionally, the system has an Ouster OS1-128 LiDAR scanner with 128 beams and a 45° vertical field of view that is mounted horizontally. A second Ouster OS1-32 is mounted vertically and has 32 vertical beams. All sensors are PTP time-synchronized.

As calibration environment, we place 119 AprilTags at the four walls and the ceiling in an otherwise empty room. The 3D coordinates of the tags are extracted as described in Sec. III-A. We mounted structural elements in the shape of pyramids to the walls to fix all DoF of the pose for the LiDAR scan.

### B. Calibration Evaluation

The main goal of this work is to provide a reliable calibration method for multi-sensor perception systems. Therefore,

TABLE I: Calibration evaluation

| Model | RMSE [pix] | RMSE [m] |
|---|---|---|
| Scene-based [13] | 30.71 | 0.064 |
| $Ca^2Lib$ [7] | 11.13 | 0.024 |
| CV2O3D | 4.17 | 0.010 |
| Ours | **2.51** | **0.007** |

we look in this experiment into the accuracy of our method and compare it to other approaches. We will compare our calibration environment-based approach against $Ca^2Lib$ [7], a LiDAR-to-camera calibration approach using a chessboard for calibration, and a natural scene-based approach [13]. Since the approaches calibrate one laser with one camera, we perform this four times to obtain the calibration between all the sensors. Additionally, the approach assumes the cameras to be intrinsically calibrated; therefore, we provide the necessary intrinsics. Multi-sensor calibrations in a specially designed calibration environment are very rare; therefore, we implemented a baseline using standard tools provided by OpenCV [3] and Open3D [35], which we will denote as CV2O3D. For this, we register each sensor to the reference map. The extrinsics between the sensors can be obtained by computing the relative pose between the sensors. We can do this for each timestamp independently, and after removing outliers, estimate the mean transformation. For the camera, we first estimate the intrinsics using Zhangs method [33], followed by estimating the pose in the map using the classic PnP algorithm using the AprilTag coordinates. The poses of the LiDAR in the map can be estimated using a RANSAC-based global registration, followed by a point-to-plane ICP for fine registration between the scans and the reference point cloud map. The difference between our approach is that the standard tools only allow for independent estimation of the sensor states, while our approach is optimizing all the poses, extrinsics, and intrinsics jointly.

We evaluated all approaches using the same cube dataset (see Sec. IV) with 34 measured cube corners using the horizontal OS1-128 LiDAR and all cameras. The position and distance to the cube varies such that we have a high FoV coverage. Tab. I shows the RMSE errors. Our approach is outperforming the scene-based calibration [13], checkerboard baseline $Ca^2Lib$ [7] as well as CV2O3D, which uses exactly the same data as our approach. We believe that our configuration in the calibration environment is more stable since we take in each timestamp the observations of all sensors into account and thus obtain better results. Additionally, with the checkerboard, one is only taking the few observations that lie on the board. The scene-based approach has a lot of potential correspondences, but finding the correct ones, especially based on the not so reliable intensity, might be hard without incorporating at least some outliers. In the calibration environment, on the other hand, we can use all the points for the ICP, and due to the AprilTags have fixed correspondences for the cameras.

### C. Model Analysis

In this experiment, we want to show the impact of different parameterizations of the sensor models to validate our design choices. Note, that this should be done for each sensor system to choose the right model for each sensor. The results are depicted in Tab. II. When looking at the configurations [A]-[C], and [H] one can see the impact of estimating intrinsics of the LiDAR. The results without estimating a scale and offset parameter [A] are notably worse than estimating either of those ([B] or [C]). While the best is achieved when estimating both, see [H]. When further analyzing the residuals between the LiDAR points and the reference map after optimization, as depicted in Fig. 4, one can observe systematic errors when optimizing without the intrinsics (blue). The residuals should be normal distributed around zero, but both seem to systematically measure around 2 centimeters too short. Additionally, the distributions of the residuals are not completely symmetric. When optimizing with the intrinsics (green), the residuals look normal distributed around zero, therefore, indicating that no further systematics (like beam-wise intrinsics) are needed.

TABLE II: Ablation of different models

| Camera | | LiDAR | | Metric | |
|---|---|---|---|---|---|
| Model | Degree | Bias | Scale | RMSE [pix] | RMSE [m] |
| [A] D | 3 | ✗ | ✗ | 3.91 | 0.008 |
| [B] D | 3 | ✗ | ✓ | 3.22 | 0.009 |
| [C] D | 3 | ✓ | ✗ | 2.62 | 0.008 |
| [D] B | 2 | ✓ | ✓ | 2.72 | 0.008 |
| [E] B | 3 | ✓ | ✓ | 2.54 | 0.008 |
| [F] D | 1 | ✓ | ✓ | 4.89 | 0.013 |
| [G] D | 2 | ✓ | ✓ | 2.75 | 0.008 |
| [H] D | 3 | ✓ | ✓ | **2.51** | **0.007** |

TABLE III: Synthetic dataset: RMSE of the parameters

| Sensor | Parameter | CV2O3D | Ours |
|---|---|---|---|
| Cameras | Translation [mm] | 4.11 | 0.79 |
| | Rotation [°] | 0.255 | 0.010 |
| | Principal Point [pix] | 3.74 | 0.60 |
| | Focal Length [pix] | 1.32 | 0.31 |
| | Distortion [pix] | 0.94 | 0.11 |
| LiDAR | Translation [mm] | 8.32 | 0.85 |
| | Rotation [°] | 0.524 | 0.010 |
| | Bias [mm] | N/A | 2.82 |
| | Scale | N/A | 0.0011 |

Different image distortion parameterizations are depicted in [D] - [H], where model B depicts Brown's distortion model and D the division model as discussed in Sec. III-B1. The degree column denotes the degree of the polynom used to model the radial distortion. In summary, the first-degree polynomial is substantially worse than the second or third polynomial. Brown's and the division model evaluate quite similarly for same degrees.

### D. Evaluation on Synthetic Data

To validate our method, we evaluate our method on a synthetic dataset. This enables ground-truth reference parameters to which we can compare. For generating realistic synthetic data, we utilize the terrestrial laser scan by rendering images and LiDAR scans from the dense point cloud (see Fig. 3b) given a predefined set of poses, extrinsics and intrinsics of all the sensors. Those values where chosen to be like [H] from Tab. II to have a realistic parameter set and trajectory.

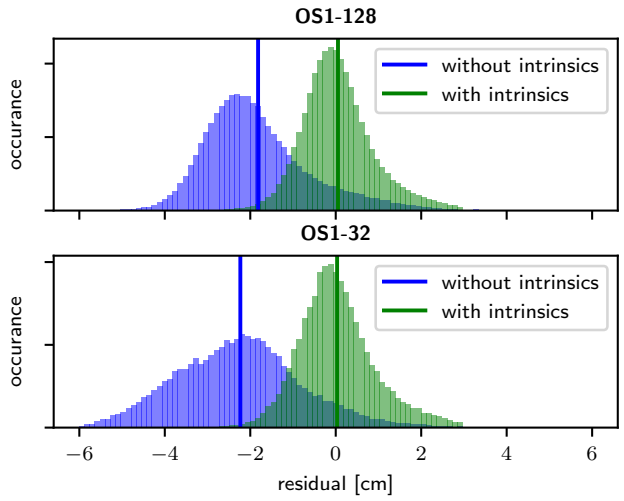



Fig. 4: Histograms of point-to-plane residuals between the LiDAR points and the reference map after adjustment with or without estimating the LiDAR intrinsics (range scale and offset). The vertical lines denote the mean. Without calibrating the range measurements, one can see a constant offset off around 2 cm; both LiDAR sensors underestimate the range.

We add 2 cm of isotropic Gaussian noise to the points of the LiDAR scan. In Tab. III the RMSE's of the individual parameters w.r.t the ground-truth parameters over all cameras and LiDAR sensors are displayed. Our approach is able to outperform CV2O3D, the best performing baseline in the previous experiments. This shows the advantage of a combined adjustment over an individual calibration.

### *E. Calibration of different Perception Systems*

To show the versatility of our system, we will show the calibration results for different perception systems with different sensors and configurations. For this, we will provide quantitative results in the form of the analytical covariances for the relative and absolute poses, as well as qualitative results to provide a more intuitive way to see how well the sensors are calibrated and the observations are aligned to the map. The analytical standard deviations of the relative poses between the sensors (extrinsics), as well as the standard deviation of the absolute poses are shown in Tab. IV. Both show that the translation can be estimated with below millimeter accuracy, while the rotation angles have standard deviations of around 0.06°. Propagating these errors into the image leads to errors with around 2.6 pix standard deviation, which is in line with the measured 2.51 pix RMSE from Sec. V-B; indicating that our system can obtain realistic covariances. Qualitative results can be found in Fig. 5. For each sensor, the observations from one timestamp can be seen in the calibration environment. The point clouds from the LiDAR are well aligned with the walls. The camera observations of the AprilTag coordinates are visualized by the corresponding ray in the reference map frame. One can see that the camera rays intersect the AprilTag coordinates of the reference point cloud $M$. Note that for the estimation, not only the observations from one timestamp, but from around 50 timestamps at different positions are used for a reliable estimation.

TABLE IV: Standard deviation of the relative and absolute poses

| Pose | Platform | x [mm] | y [mm] | z [mm] | rx [°] | ry [°] | rz [°] |
|---|---|---|---|---|---|---|---|
| relative | IPB Car | 1.26 | 1.24 | 1.04 | 0.0575 | 0.0578 | 0.0574 |
| | Youbot | 1.06 | 1.06 | 1.06 | 0.0583 | 0.0594 | 0.0585 |
| | Dingo | 1.07 | 1.06 | 1.08 | 0.0592 | 0.0634 | 0.0601 |
| absolute | IPB Car | 1.01 | 1.0 | 1.02 | 0.0575 | 0.0578 | 0.0574 |
| | Youbot | 1.03 | 1.03 | 1.06 | 0.0585 | 0.0592 | 0.0574 |
| | Dingo | 1.04 | 1.03 | 1.03 | 0.0597 | 0.0621 | 0.0574 |

## VI. Limitations and Future Work

In this chapter, we briefly want to discuss the advantages and disadvantages of our proposed method, as well as possible future research directions that can emerge from here. The main disadvantage we can see is that the setup can be quite costly; we rely on a precise high-resolution point cloud obtained by a terrestrial laser scanner (but needed only once) and have, in the best case, a dedicated room that we can modify to be a good calibration environment. A great advantage however is that once the calibration environment is prepared, the calibration does not require any special knowledge, and the whole process can be completely automated. One interesting future direction is to investigate how to reduce the costly hardware without compromising on the calibration quality.

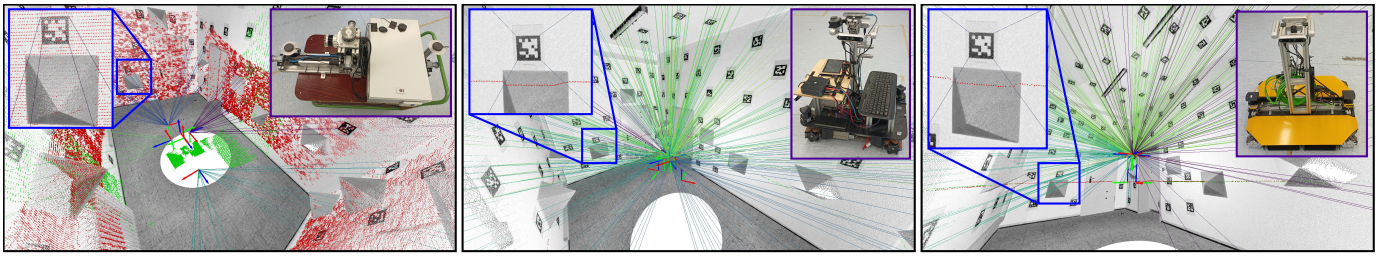
In this work, we are restricted to calibrating perception sensors. Odometry sensors, inertial measurement units (IMU), or global navigation systems (GNSS) are harder to incorporate into the pipeline since our approach relates the measurements to the reference map and not between the poses of the system at different timestamps. Incorporating an odometry sensor, or IMU, probably requires the integration of the measurements between two timestamps and relating the measured movement to the poses of the system, while for GNSS one would need a globally referenced outdoor environment.

## VII. Conclusion

In this paper, we presented an approach for calibrating the intrinsics and extrinsics of perception sensors for robotic systems. The main idea is to exploit a precise reference map as a target for all the sensors. We equipped the environment with structural elements and uniquely identifiable targets to ensure correct correspondences and resolving ambiguities for the calibration. A joint least-squares adjustment of all the sensor observations is used to estimate the statistically optimal solution. This allows us to successfully calibrate different multi-sensor systems. We calibrated different modalities of multi-beam LiDAR, profile scanners, wide-angle cameras, as well as fisheye cameras. For evaluating camera-to-LiDAR calibration, we propose an independent method to compare different calibration approaches. Our experiments suggest that our proposed approach provides accurate extrinsic and intrinsic calibration.

## Acknowledgments

We thank Ignacio Vizzo, Perrine Aguiar, Benedikt Mersch, and Nicky Zimmermann for working countless hours on the sensor setups.



(a) IPB Car (b) Youbot (c) Dingo

Fig. 5: Visualization of the LiDAR scans and image rays to the reference map for different sensor setups. (a) IPB Car mount is a roof-top mount equipped with 4 Basler Ace wide-angle cameras and 2 Ouster OS1 multi-beam LiDAR. (b) The Youbot is a ground vehicle equipped with 2 Hokuyo UTM-30LX profile scanners and one Realsense T265 that has 2 fisheye lenses. (c) Our robot "Dingo" is equipped with 2 SICK TIM781S profile scanners and one FLIR Blackfly S fisheye camera. Both, the Dingo and the Youbot have 4 Realsense D435 facing front, left, right, and rear. The point clouds are well aligned with the reference scan. The camera rays corresponding to the detected pixels as corners of the AprilTag intersect the corners of the reference point cloud.

## REFERENCES


[1] J. Beltrán, C. Guindel, A. De La Escalera, and F. García. Automatic Extrinsic Calibration Method for LiDAR and Camera Sensor Setups. In *Proc. of the IEEE Intl. Conf. on Intelligent Transportation Systems (ITSC)*, 2022.

[2] P. Besl and N. McKay. A Method for Registration of 3D Shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 14(2):239–256, 1992.

[3] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 120:122–125, 2000.

[4] Y. Chen and G. Medioni. Object Modelling by Registration of Multiple Range Images. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 1991.

[5] W. Dong and V. Isler. A novel method for the extrinsic calibration of a 2d laser rangefinder and a camera. *IEEE Sensors Journal*, 18(10):4200–4211, 2018.

[6] C. Fang, S. Ding, Z. Dong, H. Li, S. Zhu, and P. Tan. Single-shot is enough: Panoramic infrastructure based calibration of multiple cameras and 3D LiDARs. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[7] E. Giacomini, L. Brizi, L. Di Giammarino, O. Salem, P. Perugini, and G. Grisetti. Ca2Lib: Simple and Accurate LiDAR-RGB Calibration Using Small Common Markers. *Sensors*, 24(3), 2024.

[8] X. Gong, Y. Lin, and J. Liu. 3D LIDAR-Camera Extrinsic Calibration Using an Arbitrary Trihedron. *Sensors*, 13(2):1902–1918, 2013.

[9] E. Heinz, C. Holst, H. Kuhlmann, and L. Klingbeil. Design and evaluation of a permanently installed plane-based calibration field for mobile laser scanning systems. *Remote Sensing*, 12(3), 2020.

[10] J.K. Huang and J.W. Grizzle. Improvements to Target-Based 3D LiDAR to Camera Calibration. *IEEE Access*, 8:134101–134110, 2020.

[11] K. Huang and C. Stachniss. On Geometric Models and Their Accuracy for Extrinsic Sensor Calibration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[12] K. Irie, M. Sugiyama, and M. Tomono. Target-less camera-lidar extrinsic calibration using a bagged dependence estimator. In *Proc. of the Intl. Conf. on Automation Science and Engineering (CASE)*, 2016.

[13] K. Koide, S. Oishi, M. Yokozuka, and A. Banno. General, Single-shot, Target-less, and Automatic LiDAR-Camera Extrinsic Calibration Toolbox. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[14] J. Kümmerle and T. Kühner. Unified Intrinsic and Extrinsic Camera and LiDAR Calibration under Uncertainties. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[15] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *Intl. Journal of Computer Vision (IJCV)*, 81:155–166, 2009.

[16] X. Liu, C. Yuan, and F. Zhang. Targetless Extrinsic Calibration of Multiple Small FoV LiDARs and Cameras using Adaptive Voxelization. *IEEE Trans. on Instrumentation and Measurement*, 71:1–12, 2022.

[17] E. Olson. Apriltag: A robust and flexible visual fiducial system. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2011.

[18] G. Pandey, J. McBride, S. Savarese, and R. Eustice. Extrinsic Calibration of a 3D Laser Scanner and an Omnidirectional Camera. In *Proc. of the Conf. on Advancements of Artificial Intelligence (AAAI)*. Elsevier, 2010.

[19] G. Pandey, J. McBride, S. Savarese, and R. Eustice. Automatic Targetless Extrinsic Calibration of a 3D Lidar and Camera by Maximizing Mutual Information. In *Proc. of the Conf. on Advancements of Artificial Intelligence (AAAI)*, 2012.

[20] S.A. Rodriguez F., V. Fremont, and P. Bonnifait. Extrinsic calibration between a multi-layer lidar and a camera. In *Proc. of the IEEE Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 2008.

[21] R. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009.

[22] D. Scaramuzza, A. Harati, and R. Siegwart. Extrinsic Self Calibration of a Camera and a 3D Laser Range Finder from Natural Scenes. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[23] J. Schneider, T. Läbe, and W. Förstner. Incremental Real-time Bundle Adjustment for Multi-camera Systems with Points at Infinity. *ISPRS Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XL-1/W2:355–360, 2013.

[24] The APRIL Robotics Laboratory at the University of Michigan. Apriltag 3. https://github.com/AprilRobotics/apriltag. Accessed: April 2024.

[25] D. Tsai, S. Worrall, M. Shan, A. Lohr, and E.M. Nebot. Optimising the selection of samples for robust lidar camera calibration. In *Proc. of the IEEE Intl. Conf. on Intelligent Transportation Systems (ITSC)*, 2021.

[26] Y.X. Tsai, R. Shao, P. Gui, B. Li, and L. Wang. Infrastructure Based Calibration of a Multi-Camera and Multi-LiDAR System Using Apriltags. In *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018.

[27] S. Verma, J. Berrio, S. Worrall, and E. Nebot. Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences. In *Proc. of the IEEE Intl. Conf. on Intelligent Transportation Systems (ITSC)*, 2019.

[28] W. Wang, K. Sakurada, and N. Kawaguchi. Reflectance Intensity Assisted Automatic and Accurate Extrinsic Calibration of 3D LiDAR and Panoramic Camera Using a Printed Chessboard. *Remote Sensing*, 9(8):851, 2017.

[29] J. Weng, P. Cohen, M. Herniou, et al. Camera Calibration with Distortion Models and Accuracy Evaluation. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 14(10):965–980, 1992.

[30] Y. Xiong and K. Turkowski. Creating image-based VR using a self-calibrating fisheye lens. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1997.

[31] G. Yan, F. He, C. Shi, P. Wei, X. Cai, and Y. Li. Joint Camera Intrinsic and LiDAR-Camera Extrinsic Calibration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[32] Q. Zhang and R. Pless. Extrinsic Calibration of a Camera and Laser Range Finder (improves camera calibration). In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[33] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(11):1330–1334, 2000.

[34] L. Zhou, Z. Li, and M. Kaess. Automatic Extrinsic Calibration of a Camera and a 3D LiDAR using Line and Plane Correspondences. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[35] Q. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.

# CERTIFICATE OF REPRODUCIBILITY

The authors of this publication declare that:

1) The software related to this publication is distributed in the hope that it will be useful, support open research, and simplify the reproducability of the results but it comes without any warranty and without even the implied warranty of merchantability or fitness for a particular purpose.

2) *Louis Wiesmann* primarily developed the implementation related to this paper. This was done on Ubuntu 22.04.

3) *Lucas Nunes* verified that the code can be executed on a machine that follows the software specification given in the Git repository available at:

   https://github.com/PRBonn/ipb_calibration

4) *Lucas Nunes* verified that the experimental results presented in this publication can be reproduced using the implementation used at submission, which is labeled with a tag in the Git repository and can be retrieved using the command:

   `git checkout wiesmann2024ral`