Leveraging Image-Based Localization for Infrastructure-Based Calibration of a Multi-Camera Rig

Lionel Heng*
Computer Vision and Geometry Lab
ETH Zürich
Universitätstrasse 6, 8092 Zürich
hengli@inf.ethz.ch

Paul Furgale
Autonomous Systems Lab
ETH Zürich
Tannenstrasse 3, 8092 Zürich
paul.furgale@mavt.ethz.ch

Marc Pollefeys
Computer Vision and Geometry Lab
ETH Zürich
Universitätstrasse 6, 8092 Zürich
marc.pollefeys@inf.ethz.ch

Abstract

Most existing calibration methods for multi-camera rigs are computationally expensive, use installations of known fiducial markers, and require expert supervision. We propose an alternative approach called *infrastructure-based calibration* that is efficient, requires no modification of the infrastructure (or calibration area), and is completely unsupervised. In infrastructure-based calibration, we use a map of a chosen calibration area and leverage image-based localization to calibrate an arbitrary multi-camera rig in near real-time. Due to the use of a map, before we can apply infrastructure-based calibration, we have to run a survey phase once to generate a map of the calibration area. In this survey phase, we use a survey vehicle equipped with a multi-camera rig and a calibrated odometry system, and SLAM-based self-calibration to build the map which is based on natural features. The use of the calibrated odometry system ensures that the metric scale of the map is accurate. Our infrastructure-based calibration method does not assume an overlapping field of view between any two cameras, and does not require an initial guess of any extrinsic parameter. Through extensive field tests on various ground vehicles in a variety of environments, we demonstrate the accuracy and repeatability of the infrastructure-based calibration method for calibration of a multi-camera rig. The code for our infrastructure-based calibration method is publicly available as part of the CamOdoCal library at `https://github.com/hengli/camodocal`.

1 Introduction

There is an ongoing and rapid growth in the use of cameras on robotic and human-operated vehicles. From the robotic perspective, a camera is a rich source of appearance information. Furthermore, recent advances

*`http://www.inf.ethz.ch/personal/hengli`

(a) The Volkswagen Golf platforms. The cameras are integrated into the car body.

(b) The Toyota Prius platform. The cameras are mounted on the roof rack.

Figure 1: Each platform is equipped with a set of four fish-eye cameras that provides a surround view of the environment. These platforms facilitate the demonstration of two unsupervised methods, self-calibration and infrastructure-based calibration, to estimate the extrinsic transformations of these multi-camera systems without operator input. No specialized fiducial markers such as chessboards are used.

in both structure-from-motion techniques and computing hardware enable the extraction of geometric information from images in real time. This abundance of sensory information together with the use of multiple cameras to increase the robot's field of view greatly enhances robot perception. From the automotive perspective, multiple cameras are useful for driver assistance applications which help improve road safety. For example, major automotive manufacturers are making surround view systems standard on an increasing number of car models; each of these surround view systems typically comprises four outward-looking cameras. Our three test platforms follow this trend; each platform is equipped with a four-fisheye-camera system that provides a surround view. Two of our test platforms are Volkswagen Golf cars modified for autonomous driving under the auspices of the V-Charge project (Furgale et al., 2013b) while the third platform is a Toyota Prius car retrofitted for sensor data collection. These platforms are shown in Figure 1.

For vehicles equipped with multi-camera systems, so-called *calibration parameters*---transformations between cameras, camera lens parameters, etc.---must be known with a high degree of precision to ensure safe and robust operation in the presence of pedestrians and other vehicles. Although systems can be calibrated in the factory, several parameters will change during extended operation due to normal wear and tear. To everyone, particularly in research and development groups and the automotive after-market industry, calibration and recalibration of multi-camera systems is a continual burden that requires expertise, specialized equipment, and many man-hours. Consequently, it is necessary that we seek accurate, fast, robust, and unsupervised calibration algorithms.

One possibility for calibration of multi-sensor systems is to build "calibration areas" that include special instrumentation for different sensors such as known fiducial markings for cameras or known structural geometry for lasers. The use of specially designed fiducials can resolve appearance and geometric ambiguities and reduce the computational complexity of system calibration. For example, this was the strategy adopted by Geiger et al. (2012) as they were collecting an extensive multi-sensor dataset over a number of months (Geiger et al., 2013). They installed a number of chessboards covering the full field of view of the cameras and laser scanner and used them to recalibrate the vehicle before every run of data collection. As attractive as this method is, it still requires modification of the infrastructure, which could make deployment on the large scale complicated and expensive, and represent yet another barrier to the deployment of autonomous systems equipped with multiple sensors.

In this paper, we introduce a method called *infrastructure-based calibration* that shares positive aspects with the above-mentioned method but relaxes the requirement to modify the infrastructure. This relaxation is made possible by using natural features instead of known fiducial markings. Our infrastructure-based calibration method requires a map of a chosen calibration area, and hence, a prior one-time run of a survey

phase which builds a high-fidelity map of the calibration area. Mapping is made possible via SLAM-based self-calibration which makes use of data collected from a survey vehicle equipped with a multi-camera rig and a calibrated odometry system. SLAM-based self-calibration allows us to build an accurate map of the calibration area with a multi-camera rig for which we do not know the extrinsic parameters beforehand. Once the map is available, we can apply infrastructure-based calibration in which we use the map and leverage image-based localization to calibrate an unlimited number of arbitrary multi-camera rigs. Sections 4 and Section 5 describe in detail the SLAM-based self-calibration and infrastructure-based calibration respectively. We note that each infrastructure-based calibration incurs a small fraction of the high computational cost that comes with SLAM-based self-calibration. This high cost is due to the presence of two computationally expensive steps in SLAM-based self-calibration methods: exhaustive feature matching between different cameras and bundle adjustment. In contrast, these two steps are not required in image-based localization which is used by our infrastructure-based calibration method. Both the SLAM-based self-calibration and infrastructure-based calibration methods are unsupervised; they neither require expert supervision nor a special calibration setup. Furthermore, no initial guesses for the extrinsic parameters or overlapping fields of view between any two cameras are required. Infrastructure-based calibration is therefore a both financially and computationally efficient solution to the continuous unsupervised calibration of large numbers of multi-camera rigs.

The accuracy of our infrastructure-based calibration method is dependent on the accuracy of image-based localization, and in turn, the accuracy of the map of the calibration area. If we are to estimate extrinsic parameters with correct metric scale, the metric scale of the map of the calibration area generated by our SLAM-based self-calibration method must be accurate. Metric accuracy is ensured via a calibrated odometry system on the survey vehicle. We note that the SLAM-based self-calibration method used in the survey phase requires a calibrated odometry system to infer metric scale, but we do not see this requirement as restrictive since odometry data is commonly available on robots and vehicles. Furthermore, the map must be globally consistent which is ensured via loop closures. To illustrate the importance of global consistency, we give an example case in which a location is visited twice during the survey phase. No loop closures are performed, and hence, this location is represented as two distinct places in the map of the calibration area. We calibrate a two-camera rig, and it is possible that during the calibration, we use data associated with the first visit for the first camera, and data associated with the second visit for the second camera. Since the geometric discrepancy between the first and second visits is not rectified, the calibration inaccurately estimates the extrinsics such that the two cameras appear further from each other than expected.

In this paper, we expand on our previous work on infrastructure-based calibration (Heng et al., 2014) by including details of the survey phase. To make infrastructure-based calibration as accurate as possible, we focus on maximizing the accuracy of the map generated by the survey phase. Specifically, we present a significantly improved version of our SLAM-based self-calibration method (Heng et al., 2013) that is able to generate a metrically accurate and globally consistent map. Such a map enables our infrastructure-based calibration method to estimate extrinsic parameters with high accuracy and repeatability for arbitrary multi-camera rigs.

We briefly discuss the improvements to our SLAM-based self-calibration method used in the survey phase. We add robust 6D pose graph optimization so that wrong loop closures can be identified and removed. Furthermore, with globally consistent vehicle pose estimates from pose graph optimization, the joint optimization step is more likely to converge to the correct solution. We also improve the accuracy of feature matching between images from different cameras. Other improvements focus on the joint optimization step. Here, landmarks seen in different cameras tend to account for a small percentage of all landmarks in the map, and to maximize the accuracy of the poses of the cameras with respect to one another, a higher weighting is assigned to landmarks seen in different cameras. We also make use of data from the intrinsic calibration to enforce the constraint that lines are straight in rectified pinhole images; otherwise, overfitting of intrinsic parameters in the joint optimization step may lead to warped lines. Furthermore, we make use of a calibrated odometry system to infer metric scale. In addition, 6D vehicle poses are optimized instead of 3D vehicle poses to account for non-planar ground surfaces in real-world environments. Field experiments are performed to quantify the accuracy and repeatability of the camera extrinsics that are estimated by

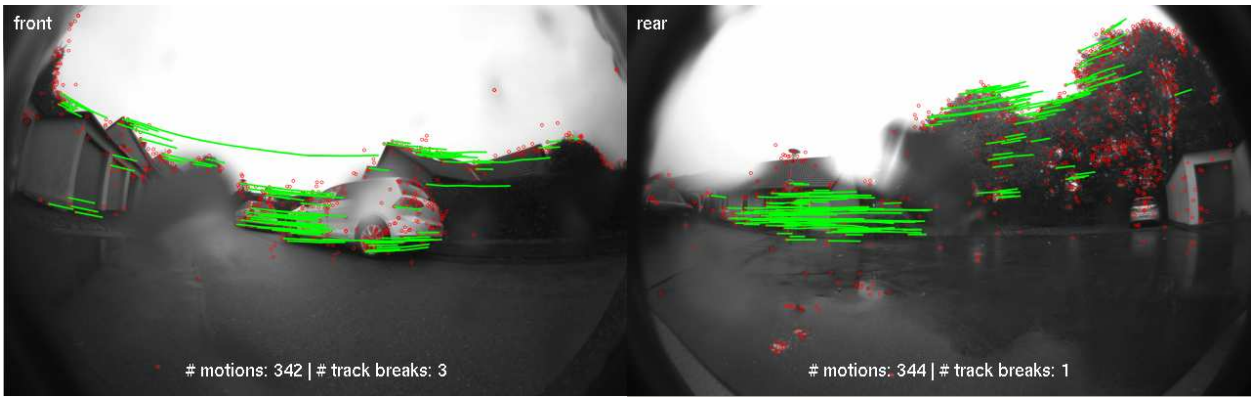


Figure 2: In rainy conditions, water droplets on the camera lens cause significant parts of the image to be obscured, and render such areas unusable for feature tracking. However, our calibration methods are still able to estimate the camera extrinsics with high accuracy.

the improved SLAM-based self-calibration method. Furthermore, we also evaluate the accuracy of the map generated by the improved SLAM-based self-calibration method.

Our work is novel in the following aspects:

1. Our SLAM-based self-calibration and infrastructure-based calibration methods are able to estimate in an unsupervised fashion and without special calibration objects the inter-camera transforms with metric scale for a multi-camera rig. Here, we do not assume overlapping fields of views. To the best of our knowledge, there is no other existing target-free calibration method for multi-camera rigs that is able to estimate the inter-camera transforms with metric scale.

2. We present extensive experiments to show the accuracy of both calibration methods for multi-camera rigs. To the best of our knowledge, no other work on calibration for multi-camera rigs shows experimental validation on a similar scale.

3. To the best of our knowledge, there is no other existing method for multi-camera rigs that makes use of a prior map based on natural features to calibrate a multi-camera rig.

Our calibration methods are designed to be robust; the SLAM-based self-calibration method used in the survey phase explicitly handles breaks in visual odometry and wrong loop closures, and the infrastructure-based calibration method only estimates camera poses given a minimum number of 2D-3D correspondences. For example, in rainy conditions, water droplets can land on the camera lenses, causing significant parts of the images to be obscured and rendered unusable for feature tracking as shown in Figure 2.

2 Related Work

There has been much research on the hand-eye calibration problem (Daniilidis, 1999; Brookshire and Teller, 2012). In general, published generic solutions to the hand-eye calibration problem do not work for vehicles with planar motions, as the height of the camera with respect to the odometry is unobservable. Hence, camera-odometry calibration requires specialized solutions (Antonelli et al., 2010; Censi et al., 2013). Antonelli et al. (2010) finds the extrinsics, camera intrinsics, and odometry parameters simultaneously for a differential drive robot; however, a set of known landmarks is required. Similarly, Censi et al. (2013) finds the extrinsics and odometry parameters for a differential drive robot without the need for known landmarks, but only obtain the three degrees of freedom of the camera-odometry transform. In the case of multi-camera

rigs with minimal overlapping fields of view, simply computing the camera-odometry transform for each camera based on camera and odometry motions is not sufficient to achieve accurate calibration parameters for a multi-camera rig. A feature point may be observed by multiple cameras at different times, and if such information is not made use of, the resulting calibration parameters may not be sufficiently accurate for reprojection of one point in a camera into another camera with a sub-pixel reprojection error. Hence, we look at state-of-the-art methods for calibrating multi-sensor rigs.

2.1 Self-Calibration

We look at unsupervised methods that do not require specific calibration patterns. In the case of lasers, Maddern et al. (2012) proposes a method to estimate the calibration parameters between at least one 3D laser and multiple 2D lasers. They construct a point cloud using a 3D laser, and use this point cloud to obtain the relative transforms between the 3D laser and each 2D laser. Here, they require a 3D laser, and an initial guess of the extrinsic parameters. In the case of both lasers and cameras, Levinson and Thrun (2012) exploit depth discontinuities in laser data and edges in images to estimate the 6-DoF transform between a Velodyne sensor and a camera.

In the case of cameras, Carrera et al. (2011) uses natural features in the environment to estimate the extrinsic parameters up to scale. They propose an extrinsic calibration of a multi-camera rig by using a modified version of MonoSLAM to build a globally consistent sparse map of landmarks for each camera, finding feature correspondences between each pair of maps via thresholded matching between SURF descriptors, and using the 3D similarity transform together with RANSAC to find inlier feature correspondences. At the end, a joint optimization is run to optimize the camera poses, 3D scene points, and robot poses. Our approach is similar in spirit; however, we estimate the extrinsic parameters with metric scale whereas Carrera et al. (2011) does not. Furthermore, we optimize the intrinsics; we find that if the intrinsics are not optimized as part of the bundle adjustment, the calibration accuracy is suboptimal. Our self-calibration method scales well to large environments unlike the approach of Carrera et al. (2011). Here, they use MonoSLAM which is known not to scale to large environments. As a result, our self-calibration method is able to generate a map of a large calibration area that can be used for efficiently calibrating multi-camera rigs. Furthermore in Carrera et al. (2011), the 3D similarity transform step can fail in outdoor environments where the majority of natural features are located far away from the cameras, and their estimated 3D locations can have substantial noise as a result, leading to few inliers.

We note that since SLAM-based calibration methods do not assume a prior map, they have to perform an exhaustive search of feature correspondences between images from different cameras. By relying on a pre-existing map for infrastructure-based calibration, we remove the need to find inter-camera feature correspondences. Furthermore, we do not have to do costly bundle adjustment. Hence, infrastructure-based calibration is far simpler, more robust, and requires a much shorter time compared to SLAM-based calibration methods. We can think of our infrastructure-based calibration method as requiring a one-time fixed cost in terms of map generation but entailing a much lower variable cost per calibration. Hence, our infrastructure-based calibration method is advantageous over SLAM-based calibration methods if many calibrations are needed within a short time span.

2.2 Infrastructure-Based Calibration

All known infrastructure-based calibration methods make use of calibration setups that include known fiducial markings. The use of specially designed fiducials can resolve appearance and geometric ambiguities and reduce the computational complexity of system calibration.

In the case of lasers, Gao and Spletzer (2010) use pairs of retro-reflective markers with known baselines for extrinsic calibration of a multi-laser system. Features are detected from these pairs of markers, and after a vehicle is driven along multiple loops, inter-loop feature correspondences are found and used to estimate the

extrinsic parameters via second-order cone programming.

In the case of cameras, a majority of existing work (Kumar et al., 2008; Lebraly et al., 2011; Li et al., 2013) requires a pattern board. The additional use of a mirror in Kumar et al. (2008) creates a limitation in which the mirror has to be in the camera's field of view, while at the same time, the entire pattern is visible in the camera. Lebraly et al. (2011) uses two pattern boards, and requires the rig to manoeuvre such that each camera sees both pattern boards at different times. Li et al. (2013) requires neighboring cameras to see some part of the pattern at the same time. We note that the use of a pattern board comes with a viewing constraint that makes calibration of multi-camera rigs non-straightforward. To overcome the viewing constraint associated with the use of a pattern board, Geiger et al. (2012) uses a room setup in which many chessboards are mounted in a configuration that covers the full field of view of the cameras and 3D laser. This room setup was used to recalibrate the multi-sensor setup before every run of data collection for an extensive dataset over a number of months (Geiger et al., 2013). This method requires modification of the infrastructure. In contrast, our infrastructure-based calibration method does not require modification of the infrastructure as it makes use of natural features instead of chessboards.

Existing methods for localization of multi-camera rigs (Tariq and Dellaert, 2004; Kneip et al., 2013; Lee et al., 2013b) assume that the extrinsic parameters are known with a high degree of accuracy. and hence, are not applicable in infrastructure-based calibration. Furthermore, these multi-camera localization methods are sensitive to errors in the extrinsic parameters; a slightly incorrect estimate of the subset of extrinsic parameters corresponding to one camera may lead to the labeling of all inlier 2D-3D feature correspondences observed by that camera as outliers. As a result, that camera is rendered redundant in the multi-camera localization. Our infrastructure-based calibration method estimates all extrinsic parameters accurately such that reliable multi-camera localization is possible.

3 Vehicle Platforms

Each of our vehicle platforms uses four CCD cameras with global shutter and GigE interfaces. On the Volkswagen Golf cars, the cameras built by Allied Vision Technologies provide 1280×800 monochrome images at a user-configured rate of 12.5 Hz, while on the Toyota Prius car, the cameras built by Matrix Vision provide 1280×960 monochrome images at a user-configured rate of 10 Hz. Each camera is fitted with a fisheye lens that has a 185° field of view. For each multi-camera setup, one camera is designated as a master camera which triggers all the other cameras. In this way, the multi-camera setup captures images from all cameras simultaneously.

The calibrated odometry system on each vehicle platform is a black box provided by the car manufacturer; apart from the fact that the odometry system reads in wheel tick data from each wheel and steering wheel angle measurements, and outputs odometry poses, we do not know the internal workings of the odometry system. On each platform, odometry pose data is published at 50 Hz.

RTI Connext DDS[1] and ROS[2] are used for message transmission of image and odometry data on the Volkswagen Golf car and Toyota Prius car respectively. A computer located in the boot logs all data to disk, and the data logs are subsequently processed by our calibration methods.

4 Survey Phase - SLAM-Based Self-Calibration

This section describes our SLAM-based self-calibration method that is used in the survey phase. This method simultaneously produces a map of the calibration area and estimates the inter-camera transforms for the

[1] `http://www.rti.com/products/dds/index.html`
[2] `http://www.ros.org`

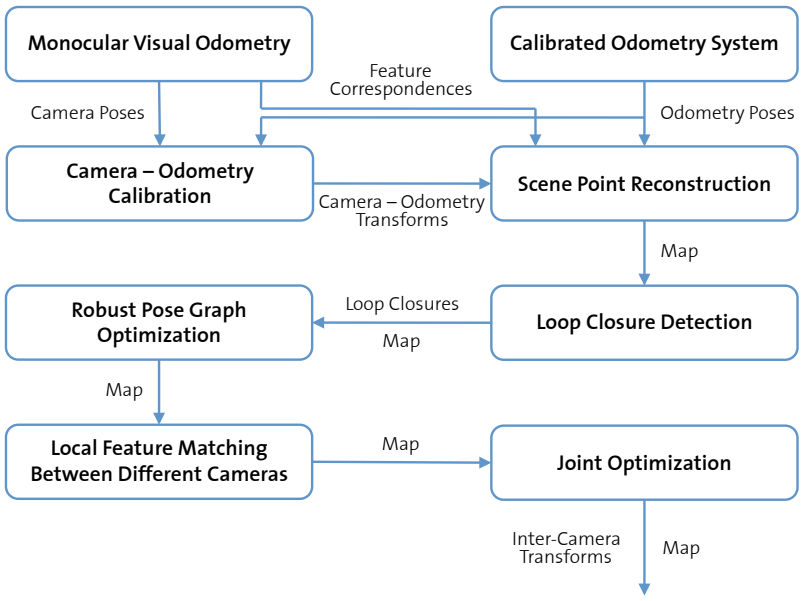


Figure 3: Images and odometry data are input to the self-calibration pipeline which estimates the camera extrinsics, and at the same time, generates a map of the calibration area.

multi-camera rig mounted on the survey vehicle. A calibrated odometry system on the survey vehicle is used to infer metric scale for the map. The generated map is subsequently used for the infrastructure-based calibration method described in Section 5.

Figure 3 shows the pipeline for our self-calibration method. Here, we give a brief outline of the method followed by detailed descriptions in the subsections below.

The monocular visual odometry step detailed in Section 4.1 estimates sets of camera poses and inlier feature tracks for each camera. The camera poses are used by the camera-odometry calibration described in Section 4.2 together with odometry poses to obtain an initial estimate of the camera-odometry transforms. This initial estimate is used together with the inlier feature tracks and odometry data by the scene point reconstruction described in Section 4.3.

By then, we have a map of the calibration area and estimates of the camera-odometry transforms. However, the map is not globally consistent as there are no loop closures. The inter-camera transforms inferred from the camera-odometry transforms are not accurate as we have not taken camera-to-camera constraints into account when computing the camera-odometry transforms in Section 4.2.

To make the map globally consistent, we detect loop closures as described in Section 4.4. Subsequently, we use robust pose graph optimization described in Section 4.5 to identify wrong loop closures and remove them, and at the same time, optimize the pose graph. Furthermore, using the globally consistent vehicle pose estimates from robust pose graph optimization instead of the odometry poses as an initial guess for the joint optimization leads to better convergence behavior. To obtain accurate inter-camera transforms such that a feature point in one camera can be reprojected to another camera with sub-pixel accuracy, we obtain temporal camera-to-camera constraints in the form of feature correspondences between different cameras as described in Section 4.6. Subsequently, the joint optimization described in Section 4.7 uses the temporal camera-to-camera constraints to optimize the inter-camera transforms in addition to intrinsic camera parameters, vehicle poses, and 3D scene points. We now have a metrically accurate and globally consistent map, and accurate inter-camera transforms.

Table 1: A table row corresponds to a step in the self-calibration pipeline. A bullet in cell (r, c) indicates that variable c is modified by step r.

	Camera Intrinsics	Camera-Odometry Transforms	Vehicle Poses	Scene Points
Monocular Visual Odometry				
Camera-Odometry Calibration		•		
Scene Point Reconstruction		•		•
Loop Closure Detection				•
Robust Pose Graph Optimization			•	
Local Feature Matching Between Different Cameras				•
Joint Optimization	•	•	•	•

Table 1 shows which variables are modified by each step in the pipeline. In summary, the pipeline produces optimized estimates of the intrinsic camera parameters and inter-camera transforms for a rig with an arbitrary number of cameras, and a metrically accurate and globally consistent map of the calibration area which can reliably be used by infrastructure-based calibration.

4.1 Monocular Visual Odometry (VO)

The monocular VO step takes images from the multi-camera system and time-synchronized odometry as input, and outputs sets of camera poses and inlier feature tracks for each camera.

In this step, we run monocular VO for each camera in order to obtain a set of camera motions together with inlier feature tracks. The set of camera motions is required for the subsequent step of computing an initial estimate of the camera-odometry transforms in Section 4.2. Similarly, the inlier feature tracks are required for scene point reconstruction in Section 4.3. It is possible for the VO to break occasionally in poorly-textured areas, producing sets of VO estimates with different scales. Nevertheless, in the camera-odometry calibration stage described in Section 4.2, we use all sets of VO estimates to find an initial estimate of the camera-odometry transform for each camera.

In our monocular VO, we use OpenCV's GPU implementation of SURF[3] to extract feature points and their descriptors. We extract a synchronized set of keyframes from all cameras when the corresponding odometry pose is at least 0.2 m away from the odometry pose at which the last set of keyframes was taken. At each iteration, we use the P3P method (Kneip et al., 2011) together with RANSAC to compute the camera pose and identify inlier feature tracks, and subsequently, apply sliding window bundle adjustment. We show the inlier feature tracks and the estimated camera poses for the four fish-eye cameras in Figures 4 and 5 respectively.

4.2 Camera - Odometry Calibration

The camera-odometry calibration step uses the camera poses computed by the monocular VO step and time-synchronized odometry. The output is an initial estimate of the camera-odometry transforms.

[3] `http://docs.opencv.org/modules/nonfree/doc/feature_detection.html`

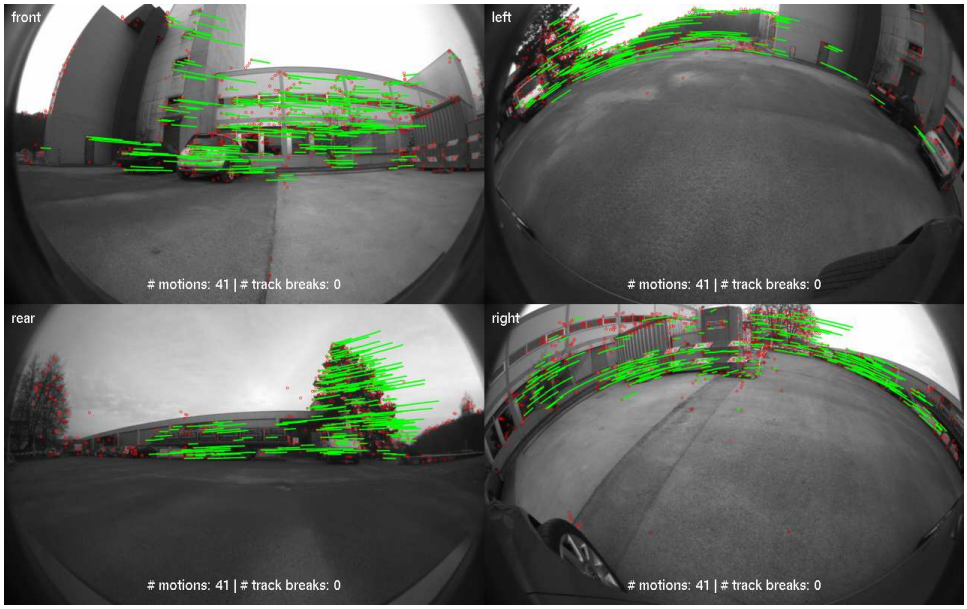


Figure 4: Inlier feature tracks.

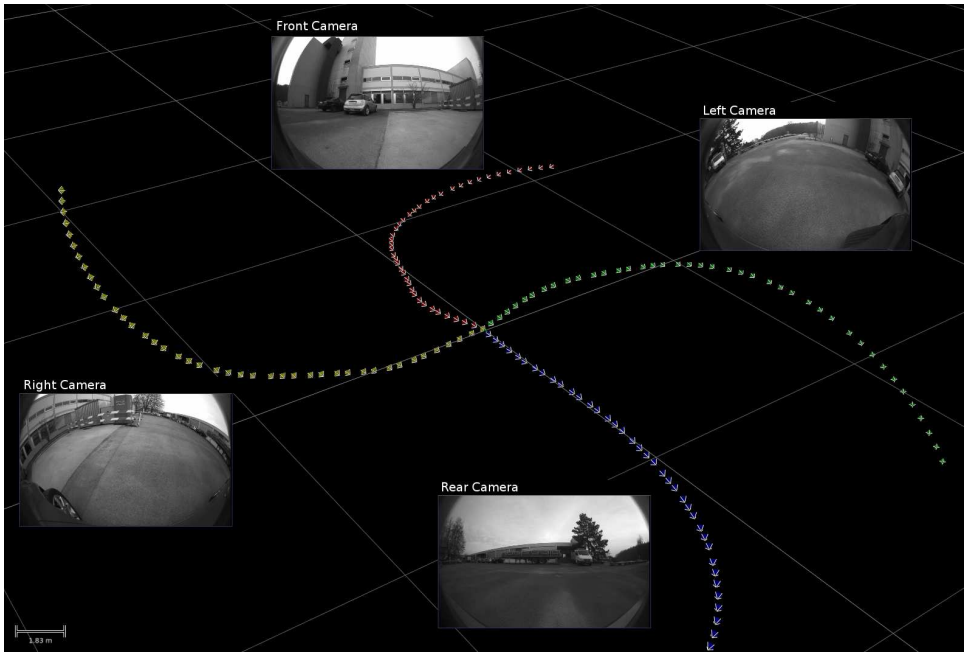


Figure 5: Camera poses estimated by monocular VO for the four fish-eye cameras when the car is making a gradual right turn. Note that the scale of the estimated poses for each camera is different. Our camera-odometry calibration step jointly solves for the different scales and the camera-odometry transforms.

To obtain an initial estimate of each camera-odometry transform, we solve a planar version of the hand-eye calibration problem which relates camera motions to odometry motions via the camera-odometry transform. Camera motions are obtained from the previous monocular VO step. We cannot use existing solutions for the 6-DoF hand-eye calibration problem, as the car moves on a $x-y$ plane most of the time, and thus, the z-component of the camera-odometry translation is unobservable.

It is common for the VO estimation to break occasionally, resulting in several segments of camera poses with different scales from a single camera. The method of Guo et al. (2012) is only able to use one segment of camera poses at most to estimate the camera-odometry transform. Hence, we make modifications to the method to estimate a single camera-odometry transform from an arbitrary number of segments of camera poses. In the first step which is identical to that in Guo et al. (2012), our approach solves for the pitch and roll components of the camera-odometry rotation by minimizing a least-squares cost function. The second step differs from Guo et al. (2012) in the aspect that we estimate multiple scales instead of a single scale; we simultaneously solve for the yaw component of the camera-odometry rotation, the camera-odometry translation, and the scales for all sparse maps by minimizing another least-squares cost function. As the vehicle motion is planar, the z-component of the camera-odometry translation is unobservable, and therefore, set to zero.

For each camera, in the first step, we estimate $\mathbf{q}_{yx}$, and in the second step, we estimate $\mathbf{q}_z$, ${}^{O}\mathbf{t}_C = [t_x, t_y, 0]^T$, and s_j for $j = 1, ..., m$ where ${}^{O}\mathbf{q}_C = \mathbf{q}_z\mathbf{q}_{yx}$ and ${}^{O}\mathbf{t}_C$ are the unit quaternion and translation respectively that transform the camera frame to the odometry frame, $\mathbf{q}_z$ corresponds to the yaw quaternion that corresponds to the yaw component of the camera-odometry rotation, $\mathbf{q}_{yx}$ corresponds to the pitch-roll quaternion which is the product of the pitch and roll quaternions corresponding to the pitch and roll components of the camera-odometry rotation respectively, and s_j is the scale for each of the m sparse maps.

We first start with the well-known hand-eye problem using the quaternion representation:

$$ {}^{O_{i+1}}\mathbf{q}_{O_i}\, {}^{O}\mathbf{q}_C = {}^{O}\mathbf{q}_C\, {}^{C_{i+1}}\mathbf{q}_{C_i}, \tag{1} $$

$$ (\mathbf{R}({}^{O_{i+1}}\mathbf{q}_{O_i}) - \mathbf{I})\, {}^{O}\mathbf{t}_C = s_j \mathbf{R}({}^{O}\mathbf{q}_C)\, {}^{C_{i+1}}\mathbf{t}_{C_i} - {}^{O_{i+1}}\mathbf{t}_{O_i}, \tag{2} $$

where ${}^{O_{i+1}}\mathbf{q}_{O_i}$ and ${}^{O_{i+1}}\mathbf{t}_{O_i}$ are the unit quaternion and translation respectively that transform odometry frame i to odometry frame $i+1$, ${}^{C_{i+1}}\mathbf{q}_{C_i}$ and ${}^{C_{i+1}}\mathbf{t}_{C_i}$ are the unit quaternion and translation respectively that transform camera frame i to camera frame $i+1$, $\mathbf{R}(\mathbf{q})$ is the rotation matrix that corresponds to $\mathbf{q}$, and $\mathbf{I}$ is a 3×3 identity matrix. Given ${}^{O_{i+1}}\mathbf{q}_{O_i}$, ${}^{O_{i+1}}\mathbf{t}_{O_i}$, ${}^{C_{i+1}}\mathbf{q}_{C_i}$, and ${}^{C_{i+1}}\mathbf{t}_{C_i}$, we use equations 1 and 2 to estimate ${}^{O}\mathbf{q}_C$ and ${}^{O}\mathbf{t}_C$.

4.2.1 Finding the pitch-roll quaternion

The method to find the pitch-roll quaternion is identical to that in Guo et al. (2012). To estimate $\mathbf{q}_{yx}$, we substitute ${}^{O}\mathbf{q}_C = \mathbf{q}_z\mathbf{q}_{yx}$ into equation 1 and use the fact that rotations around the z-axis commute to get

$$ {}^{O_{i+1}}\mathbf{q}_{O_i}\mathbf{q}_{yx} - \mathbf{q}_{yx}\, {}^{C_{i+1}}\mathbf{q}_{C_i} = 0, \tag{3} $$

which can be written as a matrix vector equation:

$$ \underbrace{(\mathcal{L}({}^{O_{i+1}}\mathbf{q}_{O_i}) - \mathcal{R}({}^{C_{i+1}}\mathbf{q}_{C_i}))}_{\mathbf{S}}\, \mathbf{q}_{yx} = 0, \tag{4} $$

where the matrix which we call $\mathbf{S}$ is a 4×4 matrix and

$$ \mathbf{q} = [w_q\ x_q\ y_q\ z_q]^T, $$

$$ \mathcal{L}(\mathbf{q}) = \begin{bmatrix} w_q & -z_q & y_q & x_q \\ z_q & w_q & -x_q & y_q \\ -y_q & x_q & w_q & z_q \\ -x_q & -y_q & -z_q & w_q \end{bmatrix}, \quad \mathcal{R}(\mathbf{q}) = \begin{bmatrix} w_q & z_q & -y_q & x_q \\ -z_q & w_q & x_q & y_q \\ y_q & -x_q & w_q & z_q \\ -x_q & -y_q & -z_q & w_q \end{bmatrix}. $$

We have two constraints on the unknown $\mathbf{q}_{yx}$:

$$x_{q_{yx}} y_{q_{yx}} = -z_{q_{yx}} w_{q_{yx}}, \quad \text{and} \quad \mathbf{q}_{yx}^T \mathbf{q}_{yx} = 1. \tag{5}$$

Given $n \geq 2$ motions, we build the $4n \times 4$ matrix

$$\mathbf{T} = \begin{bmatrix} \mathbf{S}_1^T & ... & \mathbf{S}_n^T \end{bmatrix}^T, \tag{6}$$

which has rank 2 in the absence of noise. We find the singular value decomposition $\mathbf{T} = \mathbf{U}\mathbf{S}\mathbf{V}^T$; the last two right-singular vectors $\mathbf{v}_3$ and $\mathbf{v}_4$ which are the last two columns of $\mathbf{V}$ span the null space of $\mathbf{T}$:

$$\mathbf{q}_{yx} = \lambda_1 \mathbf{v}_3 + \lambda_2 \mathbf{v}_4. \tag{7}$$

We use the two constraints from equation 5 to solve for λ_1 and λ_2, and therefore, obtain $\mathbf{q}_{yx}$.

4.2.2 Finding the yaw quaternion, scales, and translation

Given that the z-component of the camera-odometry translation is unobservable due to planar motion, we obtain a planar version of equation 2 by setting all z-components to zero:

$$\begin{bmatrix} \cos\phi - 1 & -\sin\phi \\ \sin\phi & \cos\phi - 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} - s_j \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} + {}^{O_{i+1}}\mathbf{t}'_{O_i} = 0, \tag{8}$$

where ϕ is the relative yaw between odometry frames i and $i+1$, $[p_1\ p_2]^T$ are the first two elements of the vector $\mathbf{R}(\mathbf{q}_{yx})\,{}^{C_{i+1}}\mathbf{t}_{C_i}$ and ${}^{O_{i+1}}\mathbf{t}'_{O_i}$ denotes the first two elements of the vector ${}^{O_{i+1}}\mathbf{t}_{O_i}$. $\mathbf{q}_z$ is a function of α, the yaw component of the camera-odometry rotation, such that

$$\mathbf{q}_z = \begin{bmatrix} \cos\frac{\alpha}{2} & 0 & 0 & \sin\frac{\alpha}{2} \end{bmatrix}^T. \tag{9}$$

Given ϕ, $[p_1\ p_2]^T$, and ${}^{O_{i+1}}\mathbf{t}'_{O_i}$, our goal is to estimate t_x, t_y, s_j and α.

We reformulate equation 8 as a matrix vector equation:

$$\begin{bmatrix} \mathbf{J} & \mathbf{K} \end{bmatrix} \begin{bmatrix} t_x \\ t_y \\ -s_j \cos\alpha \\ -s_j \sin\alpha \end{bmatrix} = -\,{}^{O_{i+1}}\mathbf{t}'_{O_i}, \tag{10}$$

where $\mathbf{J} = \begin{bmatrix} \cos\phi - 1 & -\sin\phi \\ \sin\phi & \cos\phi - 1 \end{bmatrix}$ and $\mathbf{K} = \begin{bmatrix} p_1 & -p_2 \\ p_2 & p_1 \end{bmatrix}$.

As the VO for a single camera may break occasionally, we may have more than 1 VO segment. Each VO segment contributes one instance of equation 10. For the m VO segments with $n_1 \geq 2$, ..., $n_m \geq 2$ motions respectively where $n = \sum_{i=1}^{m} n_i$, we stack the m instances of equation 10 into the $2n \times (2 + 2m)$ matrix

$$\mathbf{G} = \begin{bmatrix} \mathbf{J}_1^1 & \mathbf{K}_1^1 & ... & 0 & 0 & ... & 0 & 0 \\ ... & ... & ... & 0 & 0 & ... & 0 & 0 \\ \mathbf{J}_{n_1}^1 & \mathbf{K}_{n_1}^1 & ... & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & ... & \mathbf{J}_1^j & \mathbf{K}_1^j & ... & 0 & 0 \\ 0 & 0 & ... & ... & ... & ... & 0 & 0 \\ 0 & 0 & ... & \mathbf{J}_{n_j}^j & \mathbf{K}_{n_j}^j & ... & 0 & 0 \\ 0 & 0 & ... & 0 & 0 & ... & \mathbf{J}_1^m & \mathbf{K}_1^m \\ 0 & 0 & ... & 0 & 0 & ... & ... & ... \\ 0 & 0 & ... & 0 & 0 & ... & \mathbf{J}_{n_m}^m & \mathbf{K}_{n_m}^m \end{bmatrix}, \tag{11}$$

where the matrices $\mathbf{J}_i^j$ and $\mathbf{K}_i^j$ correspond to the ith motion in VO segment j, and

$$\mathbf{G}\begin{bmatrix} t_x \\ t_y \\ -s_0\cos\alpha_0 \\ -s_0\sin\alpha_0 \\ ... \\ -s_m\cos\alpha_m \\ -s_m\sin\alpha_m \end{bmatrix} = -\begin{bmatrix} {}^{O_1}\mathbf{t}'_{O_0} \\ ... \\ {}^{O_{n+1}}\mathbf{t}'_{O_n} \end{bmatrix}, \tag{12}$$

for which we use the least squares method to find the solution to ${}^{O}\mathbf{t}_C = [t_x\ t_y]^T$, s_j and α_j. We have estimated the scale s_j for each VO segment, and the translation ${}^{O}\mathbf{t}_C$; however, we have m hypotheses of α. We choose the best hypothesis that minimizes the cost function

$$\mathbf{C} = \sum_{i=0}^{n_j}((\mathbf{R}({}^{O_{i+1}}\mathbf{q}_{O_i}) - \mathbf{I})\,{}^{O}\mathbf{t}_C - s_j\mathbf{R}(\alpha)\mathbf{R}(q_{yx})\,{}^{C_{i+1}}\mathbf{t}_{C_i} + {}^{O_{i+1}}\mathbf{t}_{O_i}). \tag{13}$$

We then refine the estimate of $\mathbf{q}_{yx}$, α, ${}^{O}\mathbf{t}_C$, and s_j for $j = 1, ..., m$ by using non-linear optimization to minimize $\mathbf{C}$.

4.3 Scene Point Reconstruction

The scene point reconstruction step takes as input inlier feature tracks found during the monocular VO step, time-synchronized odometry, and the initial estimate of the camera-odometry transforms from the camera-odometry calibration step, and outputs 3D scene points. This step is necessary as 3D scene information is required to compute the 6D transform measurement for each loop closure edge in the pose graph.

We initialize the vehicle poses to be the same as the odometry poses. For each camera, we iterate through every frame, and at each frame, we find all features which are visible in the previous and current frames, and do not correspond to a previously initialized 3D scene point. We triangulate each feature correspondence; if the reprojection error of the resulting 3D scene point in either frame does not exceed a threshold of 2 pixels, we associate the 3D scene point with the feature correspondence. Subsequently, each feature track is associated to the same 3D scene point which the first two features already correspond to. We illustrate the scene point reconstruction step in Figure 6. We use the Ceres solver (Agarwal et al., 2013) to run a bundle adjustment which optimizes the camera-odometry transforms and 3D scene points by minimizing the image reprojection error across all frames for all cameras while keeping the vehicle poses fixed.

4.4 Loop Closure Detection

The loop closure detection step takes as input the camera-odometry transforms, images from the multi-camera system, and 3D scene points computed by the scene point reconstruction step, and outputs loop closure measurements, each of which is a relative 6D transform between two vehicle poses. Here, a loop closure can be between either the same camera, or different cameras.

The 6D transform measurements are used in the robust pose graph optimization step. The loop closure detection step also generates an inlier set of 2D-3D correspondences for each loop closure; if a loop closure is verified by the robust pose graph optimization step to be correct, the corresponding set of 2D-3D correspondences is used to merge duplicate 3D scene points that initially were created from the vehicle visiting the same area multiple times.

We use the DBoW2 implementation of the vocabulary tree to find similar images, and the P3P method (Kneip et al., 2011) together with RANSAC for geometric verification. We add a loop closure between two

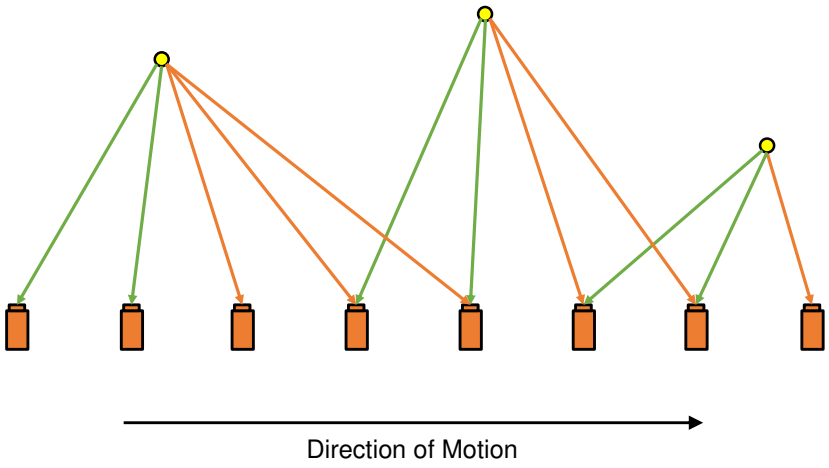


Figure 6: Yellow circles represent 3D scene points. An arrow between a camera and a 3D scene point indicates that the scene point is observed by the camera. Green arrows indicate that the corresponding feature observations are used to triangulate the associated scene point.

frames if the number of inlier 2D-3D feature correspondences exceeds a threshold. These two frames may belong to either the same camera or different cameras. Furthermore, we use information from the 2D-3D feature correspondences to merge duplicate 3D scene points after the loop closure is verified by the robust pose graph optimization step to be correct.

4.5 Robust Pose Graph Optimization

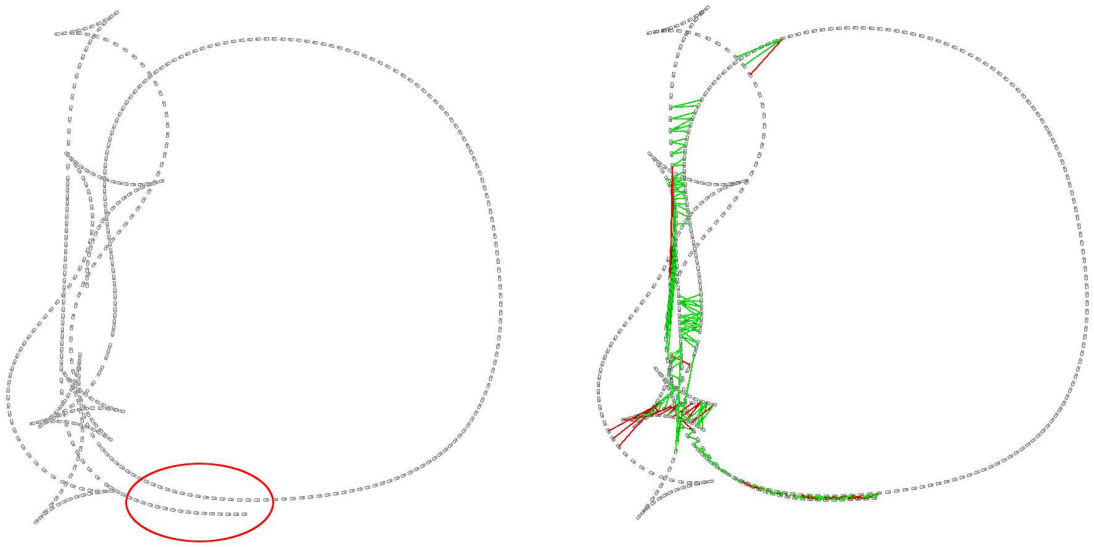
The robust pose graph optimization takes as input time-synchronized odometry, vehicle poses, and loop closure measurements computed by the loop closure detection step, and outputs the corrected vehicle poses.

We build a pose graph in which the nodes are the vehicle poses, and edges between nodes correspond to measurements of relative 6D transforms obtained from either odometry data or loop closures. We use the approach in Lee et al. (2013a) to optimize this pose graph, and at the same time, switch off false loop-closure edges. An example is shown in Figure 7. Figure 7a shows the initial pose graph, and Figure 7b shows the optimized pose graph with all loop closures edges identified as either correct or wrong. Here, we observe that the loop is closed correctly near the bottom of Figure 7b.

After the pose graph optimization, we rigidly move the scene points such that they are consistent with the new vehicle poses. At this point, the camera-odometry transforms are not sufficiently accurate for reprojection of feature points from one camera into another camera with sub-pixel reprojection errors. We solve this issue by finding local feature point correspondences between different cameras, and give details on this step in the next section.

4.6 Local Feature Matching Between Different Cameras

The local feature matching step reads as input vehicle poses computed from the robust pose graph optimization step, images from the multi-camera system, and 3D scene points computed from the 2nd scene point reconstruction step. The output is a set of 3D-3D correspondences between different cameras; as each 3D-3D correspondence corresponds to a pair of duplicate scene points, we merge these two scene points into a single scene point.



(a) Pose graph before optimization.

(b) Pose graph after robust optimization.

Figure 7: The pose of the vehicle at each time instance is marked with a black rectangle. In (a), the vehicle drives along the same road at two different times as marked by a red circle. Due to odometry drift, the corresponding trajectories do not coincide. After loop closures in this area, the trajectories are now aligned as shown in (b). In (b), loop-closure edges identified as correct by the pose graph optimization are marked as green while wrong loop closures are marked as red.

Although inter-camera feature correspondences are available from the loop closure detection step, it is typical for the image pair corresponding to an inter-camera feature correspondence to be captured far apart in time, and thus, such a feature correspondence acts as a very weak prior for the accurate estimation of relative camera poses. In contrast, the image pair corresponding to each inter-camera feature correspondence detected in this step is captured close in time, and thus, these feature correspondences act as strong priors.

Here, the cameras are assumed to have minimal overlapping fields of view at most. We iterate over each odometry pose in order of increasing timestamp, and maintain a local frame history for each camera. Note that the larger the local frame history, the longer time the local feature matching step takes. In our implementation, the local frame history for each camera spans a distance of 3 meters as measured from the vehicle poses associated with the frames in the local frame history.

At each odometry pose, for each possible pair of cameras, we do feature matching between the first camera's current frame and those in the second camera's frame history, and find the frame in the history that yields the highest number of inlier 2D-2D correspondences. Instead of using previously extracted features that correspond to existing 3D scene points, we extract and use features from rectified images. Before the feature matching, the image pair is rectified on a common image plane which corresponds to the average rotation between the first camera's pose and the second camera's pose; the camera poses are computed using the current extrinsic estimate and the odometry poses. An example of a rectified image pair between the right and front cameras is shown in Figure 8. This rectification step is to ensure a high number of inlier feature point correspondences. In contrast, we get a very low number of inlier feature point correspondences between images that are distorted and unrectified. These images inherently have large distortion in the case of fisheye cameras, and often, each pair of cameras has significantly different viewpoints. Hence, given a pair of feature descriptors corresponding to the same feature but to two different cameras, it is common for these feature descriptors to be dissimilar, and thus, be not identified as a valid feature correspondence.

For each feature point correspondence (f_1, f_2) obtained from a pair of rectified pinhole images, we find the

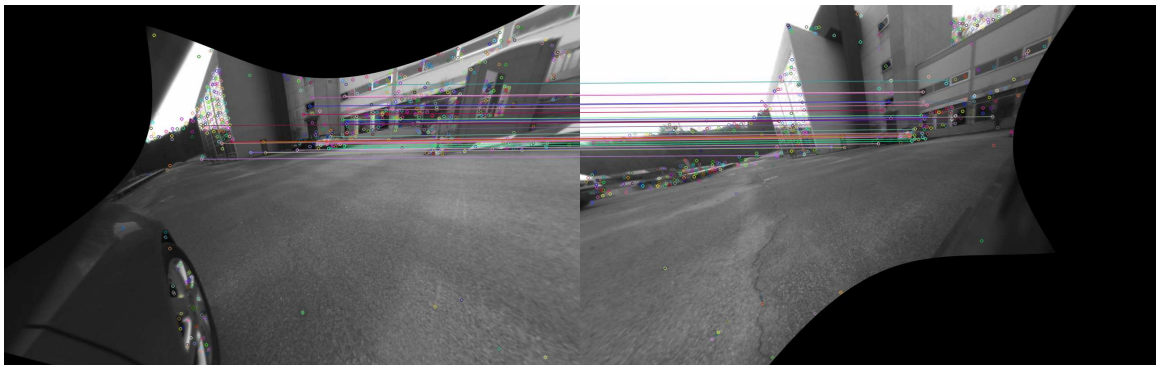


Figure 8: Inlier feature point correspondences between rectified pinhole images in the right and front cameras.

Figure 9: The subset of the same feature point correspondences from Figure 8 that correspond to 3D scene points in the maps is shown with the original images from the right and front cameras.

image coordinates of the two feature points: p_1 and p_2 in the corresponding original images. For feature point f_i where $i = 1, 2$, we look for a feature point g_i in the original image that is associated with a 3D scene point X_i in the map and whose image coordinates are within 1 pixel of p_i. If there is an associated 3D scene point for each of f_1 and f_2, and both scene points are different, we see one scene point as a duplicate of the other, and simply merge the two scene points. We show an example of inlier feature correspondences with associated 3D scene points in Figure 9.

Figure 10 shows a top-level illustration of how our local feature matching between different cameras works. Here, F_t is the current frame F_t captured by the front camera at time t. The local frame history for the right camera contains the last 5 frames captured by the right camera: $R_{t-4}, ..., R_t$. From these 5 frames, we find the frame that has the highest number of feature correspondences with F_t. As the camera pose corresponding to R_{t-2} has the closest viewing direction to the camera pose corresponding to F_t, R_{t-2} has the highest number of feature correspondences with F_t.

This step is critical to finding optimal camera-odometry transforms that allow us to project one scene point observed by one camera into another camera with minimal reprojection error.

4.7 Joint Optimization

We perform joint optimization that optimizes all intrinsics, camera-odometry transforms, vehicle poses, and 3D scene points. We note that the height of each camera with respect to the odometry frame is not observable due to the planar motion of the vehicle. However, this joint optimization recovers the relative heights of the cameras due to the use of feature correspondences between different cameras which are found from Section

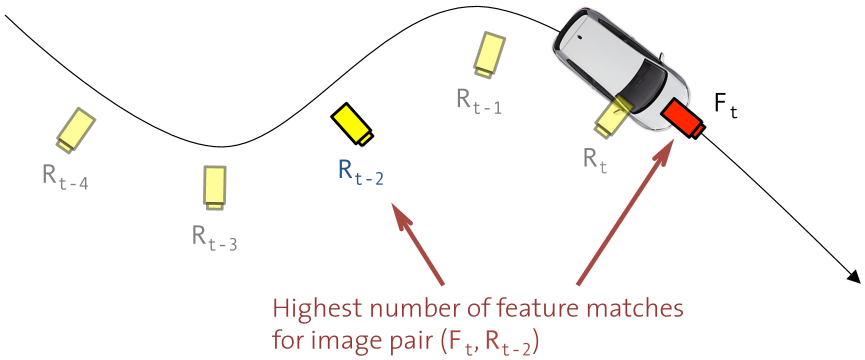


Figure 10: In this example, we do feature matching between the front camera's current frame F_t and those in the right camera's frame history, and find the frame in the history that yields the highest number of feature correspondences. Here, R_{t-2} has the highest number of feature correspondences with F_t since they have the closest viewing directions.

4.6.

The joint optimization is set up as a least squares minimization problem that minimizes a cost function. The cost function is designed to fulfil the following objectives:

1. The map should be accurate.
2. The pose of any camera with respect to all other cameras should be accurate.
3. Lines should be straight in rectified pinhole images.

Here, the cost function comprises three sets of inverse-covariance-weighted residuals:

$$\min_{\mathbf{C}_c,\mathbf{V}_i,\mathbf{Q}_j,\mathbf{T}_c,\mathbf{X}_p} \sum_{c,i,p} w_p \rho\left(\Delta\mathbf{z}_{c,i,p}{}^T \mathbf{W}_1 \Delta\mathbf{z}_{c,i,p}\right) + \sum_{c,j,q} \rho\left(\Delta\mathbf{z}_{c,j,q}{}^T \mathbf{W}_2 \Delta\mathbf{z}_{c,j,q}\right) + \sum_i \mathbf{u}_i{}^T \mathbf{W}_3 \mathbf{u}_i. \tag{14}$$

where

$$\begin{aligned} \Delta\mathbf{z}_{c,i,p} &= \boldsymbol{\pi}_1(\mathbf{C}_c, \mathbf{V}_i, \mathbf{T}_c, \mathbf{X}_p) - \mathbf{p}_{cip} \\ \Delta\mathbf{z}_{c,j,q} &= \boldsymbol{\pi}_2(\mathbf{C}_c, \mathbf{Q}_j, \mathbf{Y}_q) - \mathbf{p}_{cjq} \\ \Delta\mathbf{u}_i &= \mathbf{h}(\mathbf{V}_i, \mathbf{V}_{i+1}) - \mathbf{Z}_{i,i+1} \end{aligned} \tag{15}$$

As long as the residuals are inverse-covariance-weighted and assuming that the residuals have zero-mean Gaussian distributions, the joint optimization outputs a maximum likelihood estimate (Triggs et al., 2000).

$\boldsymbol{\pi}_1$ is a projection function that predicts the image coordinates of the scene point $\mathbf{X}_p$ seen in camera c given the camera's intrinsic parameters $\mathbf{C}_c$, the vehicle pose $\mathbf{V}_i$, and the rigid body transformation from the camera frame to the odometry frame $\mathbf{T}_c$. $\mathbf{p}_{cip}$ is the observed image coordinates of $\mathbf{X}_p$ seen in camera c with the corresponding vehicle pose $\mathbf{V}_i$. Similarly, $\boldsymbol{\pi}_2$ is a projection function that predicts the image coordinates of the chessboard corner point $\mathbf{Y}_q$ seen in camera c given the camera's intrinsic parameters $\mathbf{C}_c$, and the camera pose $\mathbf{Q}_j$. $\mathbf{p}_{cjq}$ is the observed image coordinates of $\mathbf{Y}_q$ seen in camera c whose pose is $\mathbf{Q}_j$. ρ is a robust cost function used for minimizing the influence of outliers. h is a function that computes the relative pose given two consecutive vehicle poses $\mathbf{V}_i$ and $\mathbf{V}_{i+1}$, and $\mathbf{Z}_{i,i+1}$ is the observed relative pose between $\mathbf{V}_i$ and $\mathbf{V}_{i+1}$ as measured from odometry data. We visualize the reference frames and variables in Figure 11.

In equation 14, the first set of residuals corresponds to the sum of image reprojection errors from all feature observations, the second set of residuals corresponds to the sum of image reprojection errors associated with

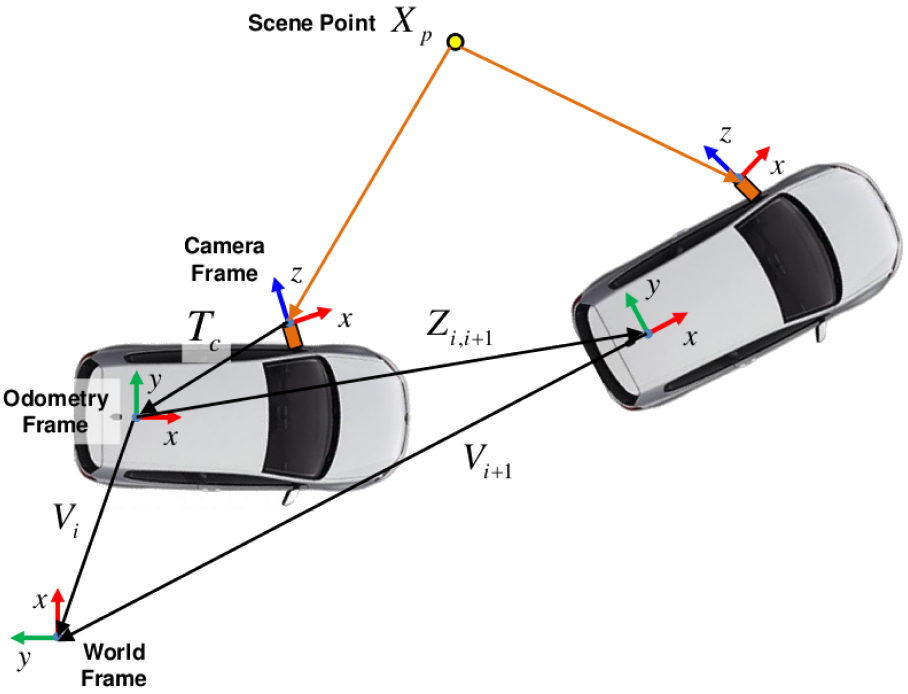
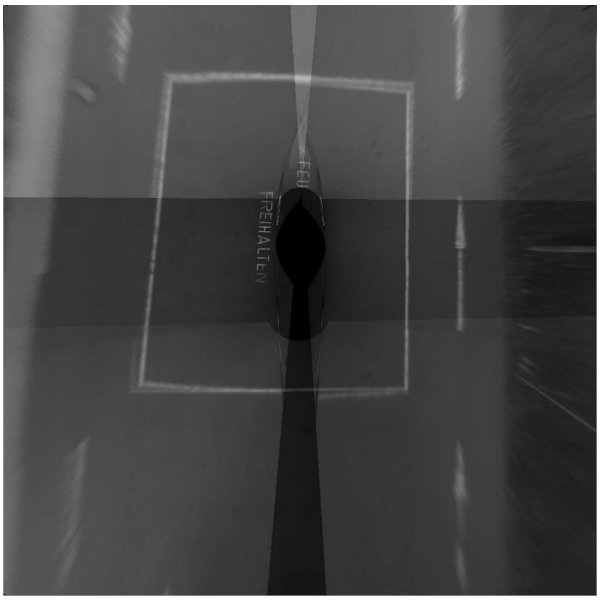


Figure 11: Visualization of the different reference frames and variables described in the joint optimization step.

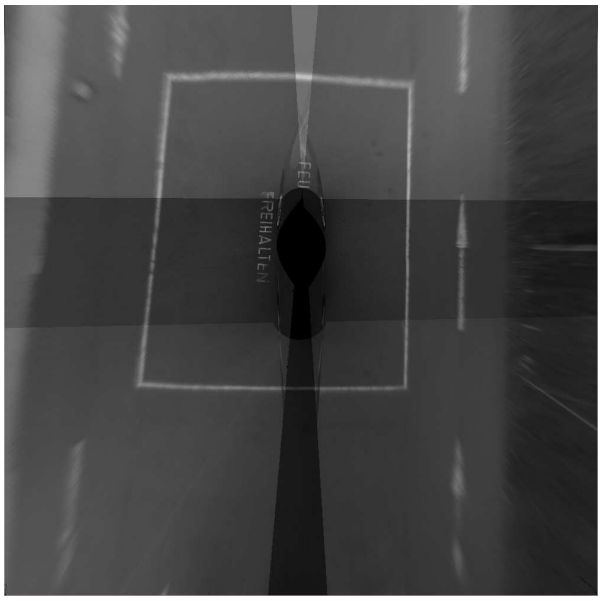
the chessboard corner points, and the third set of residuals corresponds to the sum of relative pose errors as measured by odometry data. Note that no measurement is used more than once.

w_p is a weight assigned to a feature observation associated with scene point $\mathbf{X}_p$. If $\mathbf{X}_p$ is observed by multiple cameras, we choose $w_p = 1$. Otherwise, if $\mathbf{X}_p$ is only observed by one camera, we choose $w_p = \frac{n_m}{n_s}$ where n_m is the number of feature observations associated with scene points observed by multiple cameras and n_s is the number of feature observations associated with scene points observed by a single cameras. This weighting scheme is chosen because scene points observed by multiple fractions usually form a small fraction of all scene points in the map, and the lower weighting of scene points seen by a single camera ensures that the relative camera poses are accurate. $\mathbf{W}_2$ is the inverse of the measurement covariance corresponding to the chessboard corner points; this measurement covariance is computed during the intrinsic camera calibration. Given the fact that feature detectors estimate keypoint coordinates with less accuracy compared to sub-pixel chessboard corner detectors, we obtain $\mathbf{W}_1$ by multiplying $\mathbf{W}_2$ by a fudge factor, which in our case, is 0.1. Similarly, $\mathbf{W}_3$ is the inverse of the measurement covariance corresponding to the relative odometry poses; we compute $\mathbf{W}_3$ using the odometry data and estimated vehicle poses from the robust pose graph optimization step described in Section 4.5.

The camera intrinsics are optimized in the joint optimization. The reason for this optimization is that the intrinsic camera calibration estimates sub-optimal values of the intrinsic parameters due to a biased distribution of chessboard poses and overfitting. The number of 3D scene points observed by the cameras in the extrinsic self-calibration is many times higher than the number of chessboard corner points observed in the intrinsic camera calibration, and by making use of the numerous scene points, we can improve the accuracy of the camera intrinsics in the joint optimization. Figure 12a shows the composite top-view image created from images from four cameras with the intrinsics originally estimated by the chessboard-based intrinsic camera calibration while Figure 12b shows the composite top-view image created with the optimized intrinsics. We clearly see a difference between the two; the composite top-view image created with the optimized intrinsics



(a) A composite top-view image created with original intrinsic parameters.



(b) A composite top-view image created with optimized intrinsic parameters.

Figure 12: Both composite images cover a 20 m × 20 m area. The four camera images blend seamlessly with one another in the composite image on the right, but not in the composite image on the left. This obvious visual difference proves that optimization of the intrinsic parameters in the joint optimization step significantly improves the accuracy of these parameters.

show seamless blending between the four camera images. This visual difference shows that the optimized intrinsics are more accurate than those estimated by the intrinsic camera calibration.

In experiments, we observed that overfitting of camera intrinsic parameters can occur, causing lines to look curved in rectified pinhole images. To avoid this situation, we include the residuals from the intrinsic camera calibration, and thus, enforce the constraint that lines should look straight in rectified pinhole images.

5 Infrastructure-Based Calibration

Given a map of the calibration area which is generated by the SLAM-based self-calibration method in the survey phase, the infrastructure-based calibration method leverages image-based localization to obtain camera poses, and in turn, estimates both the rig poses and camera extrinsics which are further optimized via non-linear refinement. We use a vocabulary tree for image-based localization; it is possible to use other state-of-the-art methods (Sattler et al., 2011; Lim et al., 2012) for image-based localization. In contrast to the SLAM-based self-calibration method, the infrastructure-based calibration method does not require a calibrated odometry system as the map has metric scale, and hence, enables us to directly infer the extrinsic parameters with metric scale via image-based localization.

One advantage of the infrastructure-based calibration method is that it runs in near real-time. In contrast, the self-calibration method in the survey phase and other SLAM-based calibration methods typically takes a few hours. Hence, the infrastructure-based calibration method allows us to perform many calibrations within a short time.

We assume that the cameras used in the infrastructure-based calibration have been intrinsically calibrated beforehand. The pipeline behind the infrastructure-based calibration method is shown in Figure 13. We

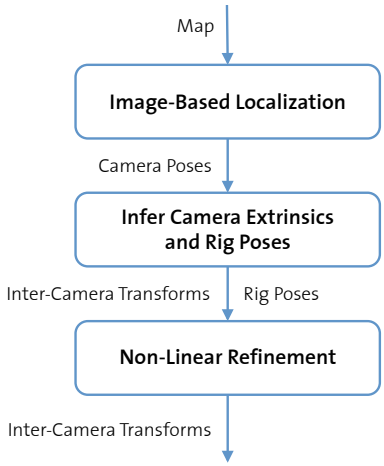


Figure 13: A map and images are input to the infrastructure-based calibration pipeline which generates the camera extrinsics.

describe the calibration process in detail below.

5.1 Image-Based Localization

The image-based localization step takes images from the multi-camera rig and the map as input, and outputs camera poses with respect to the map's reference frame. Image-based localization allows us to infer the camera poses for a given set of frames captured simultaneously from all cameras. First, for each image, we use the vocabulary tree to find the k most similar images. For each candidate, we obtain 2D-2D feature correspondences from matching features between the query image and the candidate, and rejecting probable false matches using the distance ratio test. As the feature points in the candidate image already have corresponding 3D scene points from the map, it is trivial to obtain 2D-3D feature correspondences. We use the P3P method (Kneip et al., 2011) together with RANSAC to find the camera pose together with the inlier set of 2D-3D feature correspondences. For the query image, we choose the camera pose associated with the candidate which has the highest number of inlier 2D-3D feature correspondences. The camera pose is defined to be unknown if the highest number of inlier correspondences does not exceed a threshold, which in our case, is 25. We denote the number of cameras on the multi-camera rig as n. For a particular time step t, we store the set of camera poses $\mathcal{S}_t$ if the following two conditions are met:

1. if at least 2 camera poses are found such that $2 \leq |\mathcal{S}_t| \leq n$, and
2. the minimum distance between the current camera pose and the previous camera pose over all cameras exceeds a threshold, which in our case, is 0.3 m.

The former condition is necessary for the set of camera poses to be useful for calibration, and the latter condition minimizes bias by avoiding situations where the majority of sets of camera poses is concentrated in a few locations. These situations occur when the car is at a standstill or moving slowly. For brevity, we denote the collection that contains all sets of camera poses meeting the two conditions as $\mathcal{W}$.

5.2 Inferring Camera Extrinsics and Rig Poses

In this step, we use $\mathcal{W}$, the collection of sets of camera poses inferred from image-based localization, to obtain an initial estimate for the camera-rig transforms and the rig poses. Since the cameras are rigidly fixed to the

rig, for $\mathcal{S}_t \in \mathcal{W}$, we can express $\mathcal{S}_t$, the set of camera poses, as a function of the rig pose and the camera-rig transforms. Here, the rig pose is the pose of the rig with respect to the map's reference frame, and each camera-rig transform is a rigid body transform from the camera's reference frame to the rig's reference frame. Without loss of generality, we choose the reference frame of the rig to coincide with that of a designated camera that makes up part of the multi-camera rig to be calibrated.

From $\mathcal{W}$, we generate hypotheses of the camera-rig transforms and rig poses. We choose the best hypothesis that minimizes the sum of reprojection errors over all inlier 2D-3D feature correspondences which were used to infer the camera poses in $\mathcal{W}$. This hypothesis gives us the initial estimate of the camera-rig transforms and rig poses. Algorithm 1 provides details on how we infer this initial estimate from $\mathcal{W}$. In the next step, we optimize the initial estimate via non-linear refinement.

Algorithm 1: Algorithm for obtaining an initial estimate of camera-rig transforms and rig poses from $\mathcal{W}$.

input : $\mathcal{W}$

output : Initial estimate of camera-rig transforms and rig poses

$\mathcal{A} = \emptyset$

for $\mathcal{S}_t \in \mathcal{W}$ *such that* $|\mathcal{S}_t| = n$ **do**

- compute a hypothesis of camera-rig transforms $\{\mathbf{H}_1, ..., \mathbf{H}_n\}$ from $\mathcal{S}_t$ where $\mathbf{H}_c$ is the rigid body transformation from camera c to the rig's reference frame
- create an empty hypothesis of rig poses
- **for** $\mathcal{T}_u \in W$ **do**
  - compute $|\mathcal{T}_u|$ estimates of the rig pose from $\mathcal{T}_u$ and $\{\mathbf{H}_1, ..., \mathbf{H}_n\}$
  - use the $|\mathcal{T}_u|$ estimates to obtain an average estimate of the rig pose $\mathbf{P}_u$ at time step u by using:
    1. quaternion averaging (Markley et al., 2007) to obtain the average rotation, and
    2. simple averaging to obtain the average translation
  - add $\mathbf{P}_u$ to the hypothesis of rig poses
- add the hypothesis of camera-rig transforms and rig poses to $\mathcal{A}$

select from $\mathcal{A}$ the best hypothesis of camera-rig transforms and rig poses that minimizes the sum of reprojection errors over all inlier 2D-3D feature correspondences used to infer the camera poses in $\mathcal{W}$

5.3 Non-Linear Refinement

In this step, we minimize the sum of all reprojection errors by optimizing the camera-rig transforms and the rig poses while keeping the coordinates of all 3D scene points fixed.

Formally, we solve the optimization problem

$$\min_{\mathbf{P}_t, \mathbf{T}_c} \sum_{c,t,p} \rho\left(||\boldsymbol{\pi}(\mathbf{C}_c, \mathbf{P}_t, \mathbf{H}_c, \mathbf{X}_p) - \mathbf{p}_{ctp}||^2\right). \tag{16}$$

$\boldsymbol{\pi}$ is a projection function that predicts the image coordinates of the scene point $\mathbf{X}_p$ seen in camera c given the camera's intrinsic parameters $\mathbf{C}_c$, the rig pose $\mathbf{P}_t$, and the rigid body transformation from the camera to the rig's reference frame $\mathbf{H}_c$. $\mathbf{p}_{ctp}$ is the observed image coordinates of $\mathbf{X}_p$ seen in camera c with the corresponding rig pose $\mathbf{P}_t$. ρ is a robust cost function used for minimizing the influence of outliers. We use the Cauchy cost function in this case. We visualize the reference frames and variables in Figure 14.

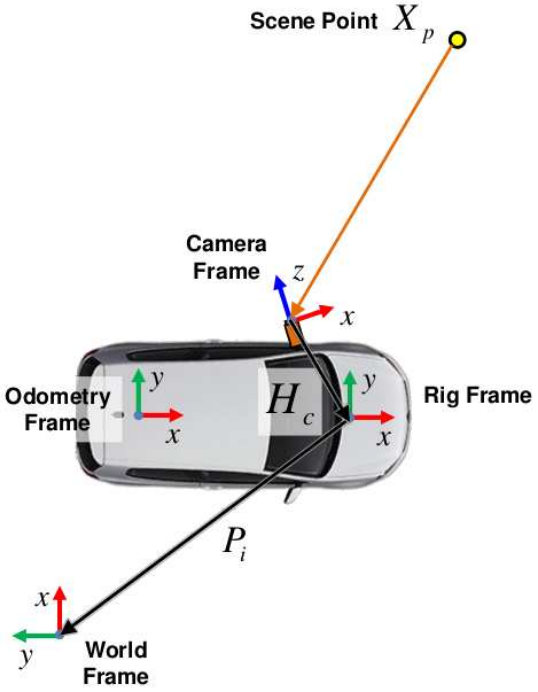


Figure 14: Visualization of the different reference frames and variables described in the non-linear refinement step.

6 Experiments and Results

This section presents a series of experiments to validate our self-calibration method (Section 6.1) and our infrastructure-based calibration method (Section 6.2).

6.1 Self-Calibration

This section presents a variety of experiments to validate the accuracy and repeatability of the calibration parameters estimated by our self-calibration method. Furthermore, we perform additional experiments to validate the metric accuracy and global consistency of the map produced by the self-calibration method.

The first experiment measures the accuracy of the self-calibration method. In this experiment, we run an AR-marker-based extrinsic calibration to provide ground truth data. We compare the results from our self-calibration method against those from the AR-marker-based extrinsic calibration. The second experiment measures the repeatability of the results from the self-calibration method over multiple loops in the same area, and aims to show that the self-calibration method provides stable results. The third experiment evaluates the accuracy of the vehicle poses estimated by our self-calibration method based on the use of GPS/INS measurements as ground truth, and thus, the metric accuracy and global consistency of the map. The fourth experiment evaluates the accuracy of the map based on a visual comparison with aerial imagery.

On an 2.80 GHz Intel Core i7 PC, our self-calibration pipeline takes approximately 3 hours to compute the calibration parameters using 500 frames for each camera; the majority of the time is spent on the local feature matching and joint optimization steps.

Table 2: Kermit: Estimated angles in degrees between the front camera (reference camera) and the other three cameras.

	Our self-calibration	**AR-marker-based calibration**	**Difference**
Left Camera	87.6419°	88.3278°	0.6859°
Rear Camera	177.0829°	177.2843°	0.2014°
Right Camera	89.2839°	89.6838°	0.3999°

Table 3: Grobi: Estimated angles in degrees between the front camera (reference camera) and the other three cameras.

	Our self-calibration	**AR-marker-based calibration**	**Difference**
Left Camera	93.1446°	92.5857°	0.5589°
Rear Camera	168.0953°	168.5607°	0.4654°
Right Camera	91.0565°	90.8517°	0.2048°

6.1.1 Experiment 1 - Accuracy of Calibration Parameter Estimates

We compare our extrinsic calibration results against those generated by an AR-marker-based extrinsic calibration. In this AR-marker-based calibration, three different types of AR markers are placed around the vehicle: camera markers that are visible in at least one camera, floor markers that are placed on the ground around the car, and wheel markers that are placed on the rear wheels to help identify the nominal odometry frame. The cameras on the vehicle together with a calibrated handheld camera capture images of the markers. Subsequently, the markers are detected in all images, and a bundle adjustment is run to optimize the poses of the markers and cameras. At the end of the bundle adjustment, the poses of the cameras are extracted with respect to the nominal odometry frame; the locations of the floor markers define the ground plane, and the wheel markers define the rear axis of the vehicle.

We compare the results for two different Volkswagen Golf cars which we name Kermit and Grobi respectively. For both cars, we tabulate in Tables 2 and 3 the estimated angles between the front camera and the other three cameras for both our estimated extrinsics and the AR-marker-based extrinsics. Similarly, we tabulate in Tables 4 and 5 the unit estimated translations between the front camera and the other three cameras to ignore the effect of scale. For Kermit, we show a visual comparison of the final estimated extrinsics with the extrinsics computed by the AR-marker-based calibration in Figure 15. A green sphere marks the origin of the odometry frame. We observe that for both cars, both sets of extrinsics estimated by our self-calibration method and the AR-marker-based calibration almost coincide; the scale of the extrinsics estimated by our self-calibration method is 2.7% and 0.8% smaller on average for Kermit and Grobi respectively.

Table 4: Kermit: Estimated unit translations in meters between the front camera (reference camera) and the other three cameras.

	Our self-calibration	**AR-marker-based calibration**	**Difference**
Left Camera	$\begin{bmatrix} -0.5464 \\ -0.2159 \\ -0.8092 \end{bmatrix}$ m	$\begin{bmatrix} -0.5502 \\ -0.2420 \\ -0.7992 \end{bmatrix}$ m	$\begin{bmatrix} 0.0038 \\ 0.0261 \\ 0.0100 \end{bmatrix}$ m
Rear Camera	$\begin{bmatrix} -0.0074 \\ -0.1070 \\ -0.9942 \end{bmatrix}$ m	$\begin{bmatrix} -0.0160 \\ -0.1343 \\ -0.9908 \end{bmatrix}$ m	$\begin{bmatrix} 0.0086 \\ 0.0273 \\ 0.0034 \end{bmatrix}$ m
Right Camera	$\begin{bmatrix} 0.5306 \\ -0.2261 \\ -0.8169 \end{bmatrix}$ m	$\begin{bmatrix} 0.5274 \\ -0.2454 \\ -0.8134 \end{bmatrix}$ m	$\begin{bmatrix} 0.0032 \\ 0.0193 \\ 0.0035 \end{bmatrix}$ m

Table 5: Grobi: Estimated unit translations in meters between the front camera (reference camera) and the other three cameras.

	Our self-calibration	**AR-marker-based calibration**	**Difference**
Left Camera	$\begin{bmatrix} -0.5059 \\ -0.3085 \\ -0.8055 \end{bmatrix}$ m	$\begin{bmatrix} -0.5273 \\ -0.2906 \\ -0.7985 \end{bmatrix}$ m	$\begin{bmatrix} 0.0214 \\ 0.0179 \\ 0.0070 \end{bmatrix}$ m
Rear Camera	$\begin{bmatrix} 0.0372 \\ -0.1835 \\ -0.9823 \end{bmatrix}$ m	$\begin{bmatrix} 0.0145 \\ -0.1788 \\ -0.9838 \end{bmatrix}$ m	$\begin{bmatrix} 0.0227 \\ 0.0047 \\ 0.0015 \end{bmatrix}$ m
Right Camera	$\begin{bmatrix} 0.5732 \\ -0.2614 \\ -0.7766 \end{bmatrix}$ m	$\begin{bmatrix} 0.5513 \\ -0.2804 \\ -0.7858 \end{bmatrix}$ m	$\begin{bmatrix} 0.0219 \\ 0.0190 \\ 0.0092 \end{bmatrix}$ m

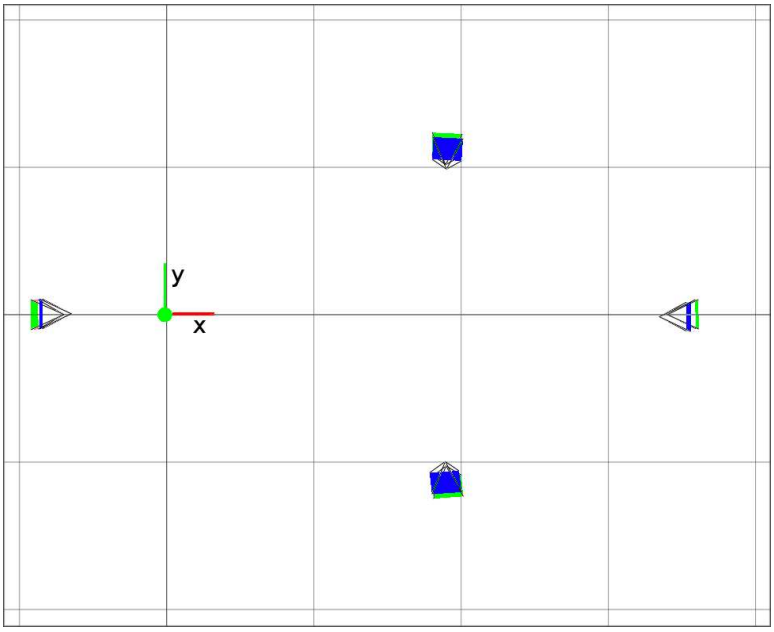
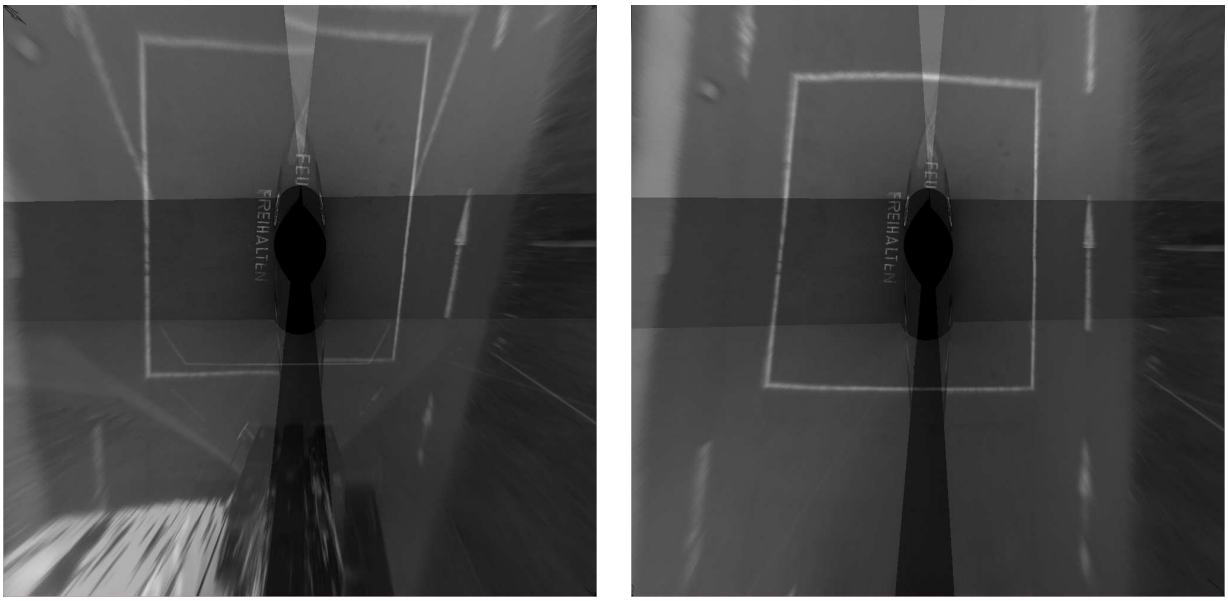


Figure 15: Comparison of extrinsics generated by our pipeline and the AR-marker-based calibration method for Kermit. The green sphere marks the origin of the odometry frame while the green and red axes represent the x-axis and y-axis of the odometry frame respectively. The cameras corresponding to the extrinsics estimated by our self-calibration method are colored blue while those corresponding to the AR-marker-based extrinsics are colored green. The grid resolution is 1 m.

We are unable to compare our approach with Carrera et al. (2011) as their 3D similarity transform step fails in our experimental settings. The reason is that in outdoor environments, a vast majority of scene points are usually located far away from the cameras, and thus, the estimates of their 3D coordinates have significant noise.

Calibration algorithms are notoriously difficult to evaluate because of the difficulty of obtaining ground truth values for the parameters being estimated. Hence, we resort to evaluating our method by demonstrating the enhanced performance of algorithms that use the calibration parameters. We argue that the extrinsics estimated by our self-calibration method are more accurate than those estimated by the AR-marker-based extrinsic calibration. We propose two qualitative tests to support our argument: creating a composite top-view image that is formed by projecting the image from each camera onto the ground plane using the given calibration parameters, and performing motion stereo based on the plane sweep algorithm which takes three consecutive images and corresponding odometry poses and computes a depth map based on these three images. In the first test, the calibration parameters are assumed to be perfect if images from all cameras blend seamlessly. Similarly, in the second test, the calibration parameters are assumed to be accurate if the computed depth map from motion stereo is dense and accurate.

Composite Top-View Image: We create a composite top-view image by using the calibration parameters to project the image from each of the four cameras onto the ground, and merging the resulting 4 images via alpha blending. The image covers a 20 m $\times$ 20 m area with the vehicle's odometry frame located at the center of the image. Figure 16 shows the composite top-view images created with both sets of calibration parameters. Seamless blending is observed in the image that corresponds to our self-calibration method, while no seamless blending is observed at all in the image that corresponds to the AR-marker-based extrinsic calibration. This visual difference shows that our self-calibration method produces very accurate calibration parameters.



(a) A composite top-view image created with calibration parameters estimated by the AR-marker-based calibration.

(b) A composite top-view image created with calibration parameters estimated by the self-calibration.

Figure 16: Both composite top-view images are of a 20 m $\times$ 20 m area centered on the car. The four camera images blend seamlessly with one another in the composite image on the right, but not in the composite image on the left. This visual difference shows that our self-calibration method produces very accurate calibration parameters.

Motion Stereo: We perform an experiment to show that the estimated calibration parameters are more accurate than that of the AR-marker-based parameters; we use a multi-view plane-sweep-based stereo algo-

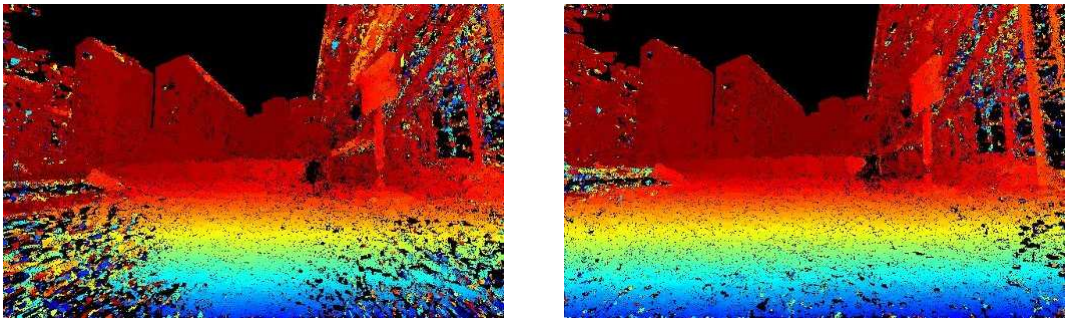


Figure 17: 3-view plane sweep stereo with the front camera. Forward camera motion is challenging for motion stereo. The left depth map is computed using the AR-marker-based parameters while the right depth map is computed using our estimated parameters.

Table 6: Mean and standard deviation of the rotation and translation vectors for each camera.

	Rotation Vector			**Translation Vector**		
Left Camera	$\begin{bmatrix} -0.0155 \\ -1.4958 \\ -0.0116 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0007 \\ 0.0047 \\ 0.0007 \end{bmatrix}$	$\begin{bmatrix} -0.6026 \\ -0.1225 \\ -1.3954 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0043 \\ 0.0012 \\ 0.0033 \end{bmatrix}$
Rear Camera	$\begin{bmatrix} -0.0063 \\ 3.0932 \\ 0.0435 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0002 \\ 0.0034 \\ 0.0018 \end{bmatrix}$	$\begin{bmatrix} 0.1065 \\ -0.0024 \\ -1.6114 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0039 \\ 0.0013 \\ 0.0034 \end{bmatrix}$
Right Camera	$\begin{bmatrix} 0.0103 \\ 1.5602 \\ -0.0143 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0006 \\ 0.0056 \\ 0.0008 \end{bmatrix}$	$\begin{bmatrix} 0.7073 \\ -0.0060 \\ -1.3041 \end{bmatrix}$	$\pm$	$\begin{bmatrix} 0.0043 \\ 0.0012 \\ 0.0027 \end{bmatrix}$

rithm (Yang and Pollefeys, 2003) with semi-global block matching to compute dense depth maps from rectified pinhole images for each camera using only the odometry poses and calibration parameters without any camera pose optimization. The plane sweep is extremely sensitive to the accuracy of the calibration parameters; we compare the depth maps computed by the plane sweep using the calibration parameters estimated by our self-calibration method with the depth maps computed with the calibration parameters generated by the AR-marker-based calibration. We show in Figure 17 that the use of our estimated parameters results in a more accurate dense map especially on the ground compared to the use of the AR-marker-based parameters. This result demonstrates the higher accuracy of our estimated parameters, and can be attributed to the fact that our pipeline explicitly makes use of odometry data to estimate the camera-odometry transforms, and makes use of a high number of features over a wide range of depths to ensure optimal accuracy.

6.1.2 Experiment 2 - Repeatability of Calibration Parameter Estimates

This experiment looks at the repeatability of the camera extrinsics estimated by our self-calibration method. We drive 25 loops with the same camera configuration in a slowly changing environment in which cars and pedestrians move around. For each loop, we use the same set of camera intrinsics, and estimate the camera-odometry transforms. In turn, for each camera, we compute the rotation and translation vectors that correspond to its pose with respect to the front camera. We tabulate the means and standard deviations of the rotation and translation vectors for each camera in Table 6. In addition, we illustrate the distribution of the rotation and translation vectors over the 25 loops in the form of a box plot in Figure 18. For ease of comparison, we subtract the vectors from their means in this box plot.

We note that the standard deviations of the rotation and translation vectors are very small, and therefore, conclude that the extrinsics estimated by our self-calibration method are highly repeatable.

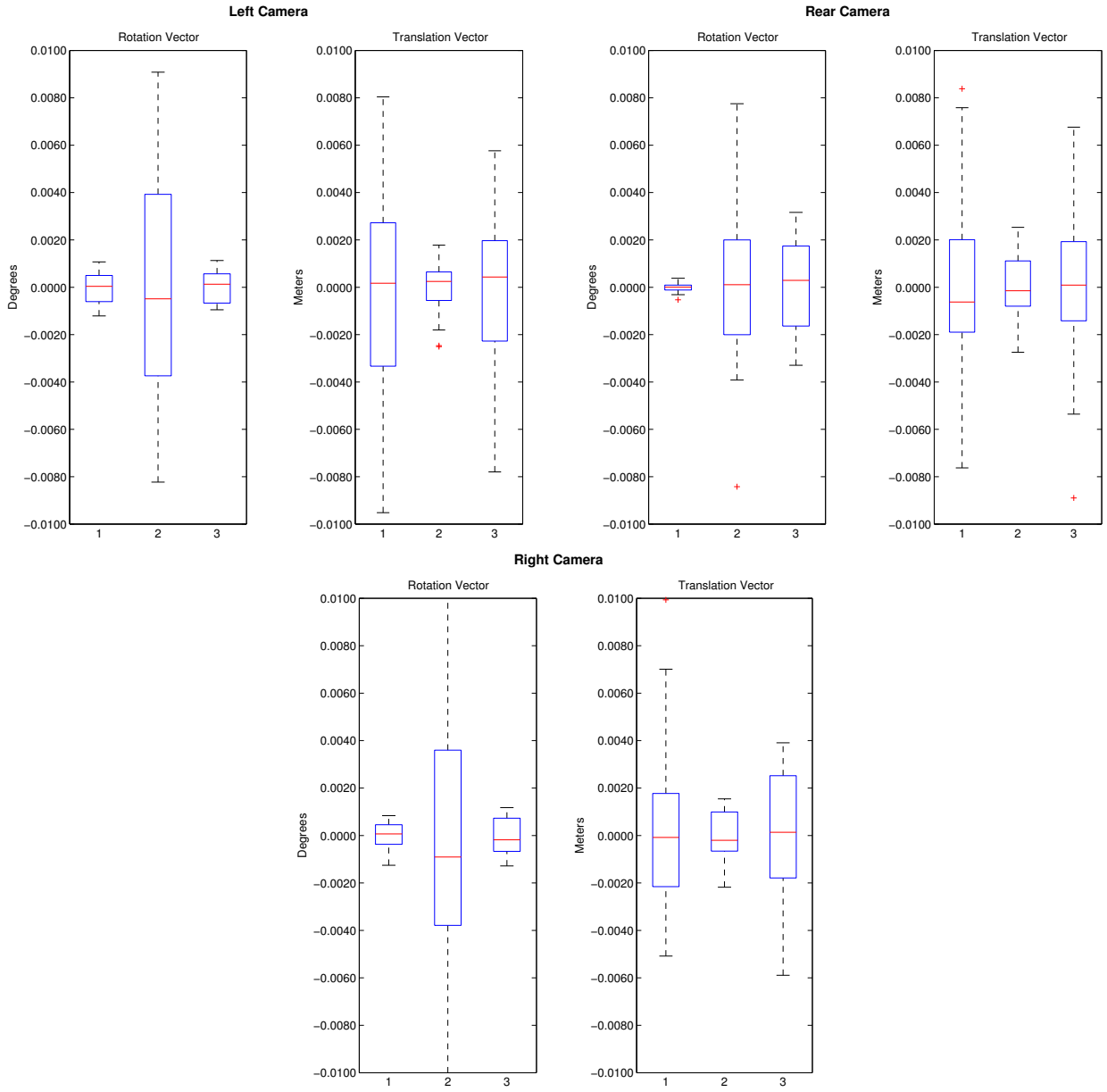


Figure 18: A box plot of the rotation and translation vectors about the mean for the left, rear, and right cameras with respect to the front camera.

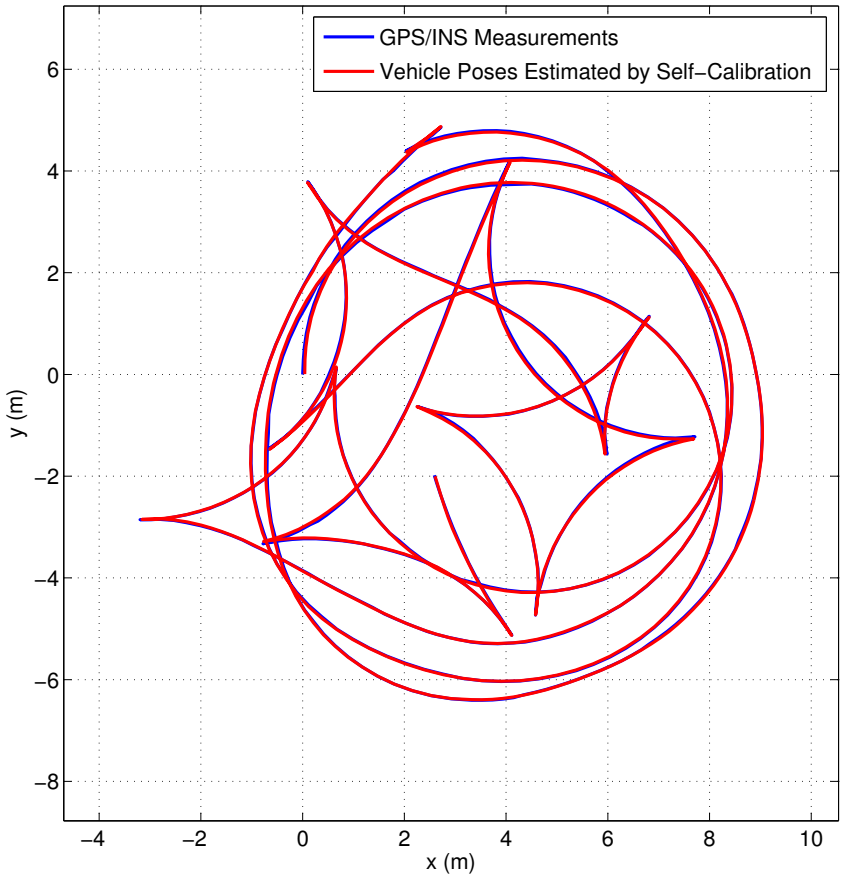


Figure 19: A top-view plot of the vehicle poses (red) estimated by our self-calibration method together with corresponding GPS/INS measurements (blue).

6.1.3 Experiment 3 - Pose Accuracy

In this experiment, we perform a quantitative evaluation of the accuracy of the vehicle poses estimated by our self-calibration method. We use 6-DoF pose measurements from a GPS/INS system as ground truth.

We use our self-calibration method to generate a map from a 180-meter trajectory in an outdoor environment. 652 sets of synchronized images from all cameras were used in the self-calibration, and in turn, 652 vehicle poses were estimated. We show a top-view plot of the vehicle poses together with the corresponding GPS/INS measurements in Figure 19. The vehicle poses and GPS/INS measurements are marked as red and blue respectively.

Based on the GPS/INS measurements, the vehicle poses estimated by our self-calibration method have an average rotation and translation error of 0.140° and 3.51 cm respectively. These low errors show that the estimated poses are accurate, and in turn, show that our self-calibration method produces a metrically accurate and globally consistent map. This global accuracy is extremely important for our infrastructure-based calibration method as it gives the method the ability to estimate accurate extrinsics for a multi-camera rig.

6.1.4 Experiment 4 - Metric Map Accuracy

We use our self-calibration method to generate a map from a 308-meter trajectory. This map has 885734 2D feature points and 197836 3D scene points, and an average reprojection error of 0.37 pixels. In Figure 20, we show this map together with its predecessor which was generated at the beginning of the pipeline with intrinsics estimated from the intrinsic camera calibration, and extrinsics estimated from the camera-odometry calibration step. In the visualized map, each 3D scene point is colored according to the camera it was first observed in. 3D scene points observed by the front, left, rear, and right cameras are colored red, green, blue, and yellow respectively.

We observe in Figure 20 that the optimized intrinsics and extrinsics correspond to a more clearly-defined map. This observation shows that the optimized parameters are more accurate than the initial parameters. Furthermore, in this figure, we also show the map overlaid on aerial imagery which is manually aligned. This overlay shows that the map aligns well with the aerial imagery on a global scale, and therefore, is metrically accurate.

6.2 Infrastructure-Based Calibration

This section presents an evaluation of our infrastructure-based calibration method. We perform experiments in both an indoor parking garage and outdoor urban environment on the ETH campus. Figure 21 shows images of both areas taken by the front camera. We design our experiments to demonstrate that our infrastructure-based calibration exhibits a high level of accuracy in both indoor and outdoor environments in the presence of moving cars and pedestrians.

Before we conduct the experiments, we run the survey phase once to build a map for both areas as shown in Figures 20 and 22a. To determine the accuracy of our estimated camera extrinsics, we compare the estimated camera extrinsics against those estimated by the self-calibration method in the survey phase. We first compute the pose of each camera with respect to the first camera for both sets of extrinsics. Then, we use these relative poses to compute the rotation error and two types of translation errors: the angle between the two translation vectors, and the norm of the difference between the two translation vectors. These three error metrics are used to give a quantitative measure of the accuracy of the camera extrinsics estimated by our infrastructure-based calibration method.

On average, it takes 0.5 seconds to estimate the camera poses from each set of images assuming that each camera has its own dedicated thread which processes all images coming from that camera. Computing the initial estimate of the camera extrinsics together with non-linear refinement typically takes no more than 5 seconds.

6.2.1 Experiment - Indoor Parking Garage

In this experiment, a vehicle is driven along one loop with the same camera configuration that was used to generate the map. This loop trajectory differs from that taken by the vehicle during the data collection for building the map. This experiment aims to show that our estimated extrinsics and those estimated by the self-calibration method are the same.

Figure 22a shows the map generated by the self-calibration method that is used in the infrastructure-based calibration. Figure 22b shows the subset of scene points that are from the map and used for the calibration. Furthermore, Figure 22b shows the estimated camera poses which are used to infer the camera extrinsics.

We tabulate in Table 7 the three error metrics for our estimated extrinsics, using the extrinsics estimated by the self-calibration method as ground truth. The infrastructure-based calibration estimated the extrinsics from 167 sets of camera poses with an average of 3.05 camera poses per set. A total of 37860 2D-3D

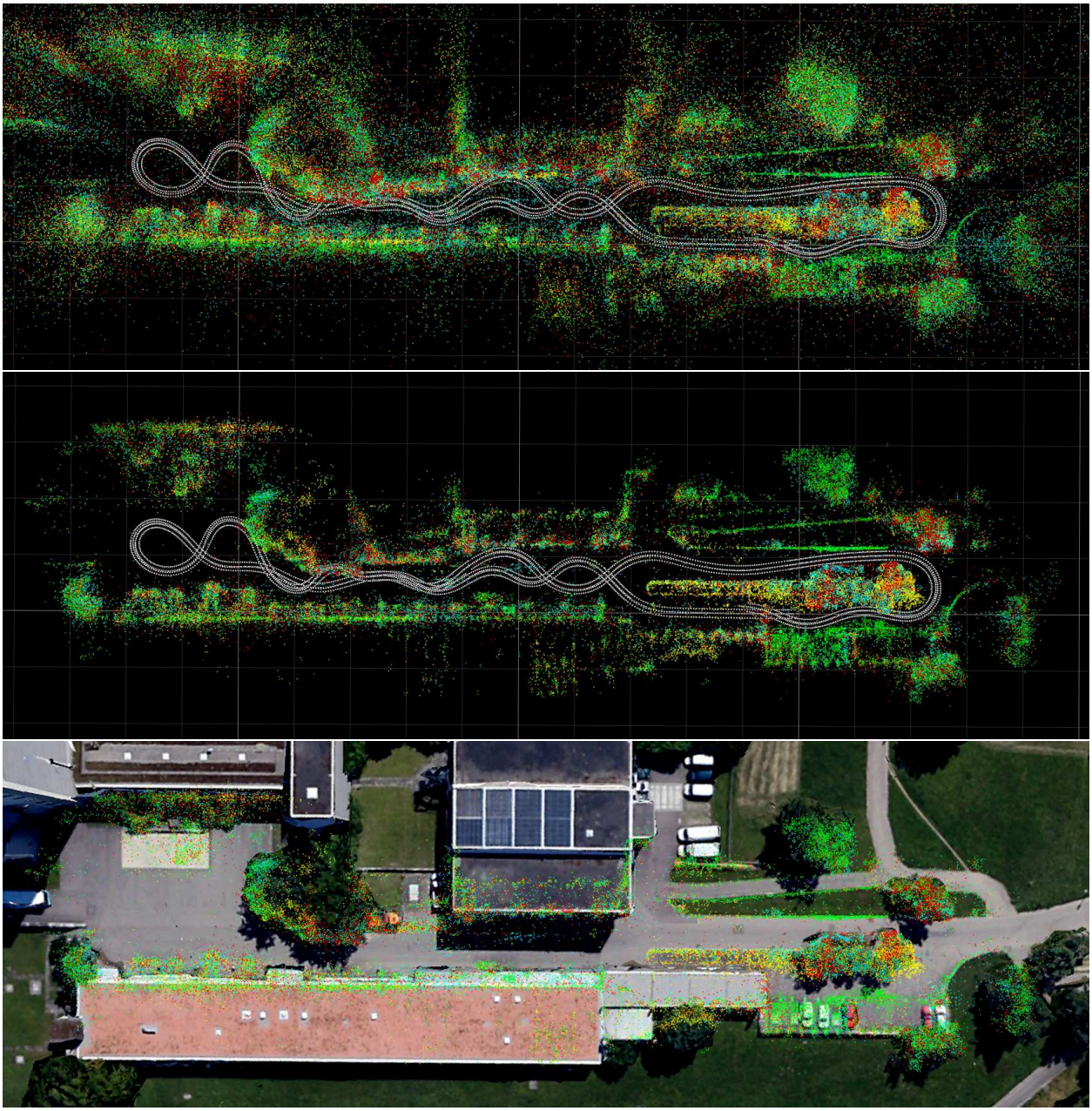
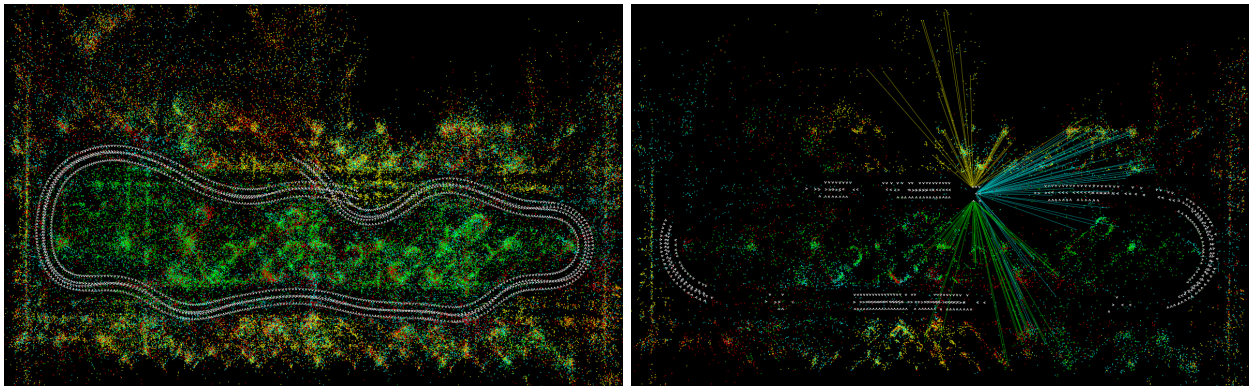


Figure 20: Each 3D scene point is colored based on the camera it was first observed in. The distance between consecutive major grid lines marked by thick white lines is 50 meters. The distance between consecutive minor grid lines marked by thin white lines is 10 meters. (a) The map with intrinsics estimated from the intrinsic camera calibration, and extrinsics estimated from the camera-odometry calibration step. (b) The map with optimized intrinsics and extrinsics. (c) The map is overlaid on aerial imagery.

(a) (b)

Figure 21: (a) Indoor parking garage and (b) outdoor urban environment as seen from the front camera



(a) (b)

Figure 22: An experiment in an indoor parking garage as shown in Figure 21a. (a) The map generated by the self-calibration method. The camera poses marked in white illustrate the twisting path taken by the car. (b) The subset of scene points from the map and which is used for infrastructure-based calibration is shown. The camera poses inferred from PnP are marked as white triangles. Here, the car approaches the end of a loop, and the current set of camera poses are derived from 2D-3D correspondences visualized as lines from the camera poses to the scene points.

Table 7: Indoor experiment: error metrics for the extrinsics estimated by our infrastructure-based calibration method.

	Left Camera	**Rear Camera**	**Right Camera**
Rotation error (deg)	0.0044	0.0032	0.0088
Translation error (deg)	0.0563	0.0468	0.0349
Translation error (m)	0.0016	0.0022	0.0021

Figure 23: The left camera is mounted on a three-way tripod head, which in turn, is mounted on a sliding plate. This sliding plate is attached to the car roof via screws.

correspondences were used. The initial estimates of the extrinsics and rig poses had an associated average reprojection error of 0.99 pixels which reduced to 0.69 pixels after non-linear refinement. It is observed from the results in Table 7 that the extrinsics estimated by our method are virtually the same as those estimated by the self-calibration method.

6.2.2 Experiments - Outdoor Urban Environment

We run a total of four experiments. In each of the first three experiments, the Prius is driven in one loop in the same scene and with a different camera configuration. As in the indoor experiment, this loop trajectory differs from that taken by the Prius during the data collection for building the map. These three experiments aim to show that our calibration can reliably estimate the camera extrinsics for a camera configuration different from that used for building the map. For the fourth experiment, we drive the Prius over 25 loops for 1 hour with the same camera configuration that was used to generate the map. This experiment shows the impact of a changing environment on the calibration accuracy; during this one hour, cars and pedestrians continually move around, and plants and trees sway significantly in moderate wind conditions. From odometry measurements, the average distance of each loop in all experiments is 308 m.

We use a three-way tripod head and a sliding plate in Figure 23 to ensure that camera configuration changes can be measured as precisely as possible.

Experiment 1: The left and right cameras are moved towards the front of the car by 10.0 cm as measured with a ruler. The infrastructure-based calibration used 418 sets of camera poses with an average of 3.26 camera poses per set. Based on the results from our method, the left and right cameras are deemed to have moved 10.08 cm and 10.36 cm respectively. These estimates closely agree with the hand measurements.

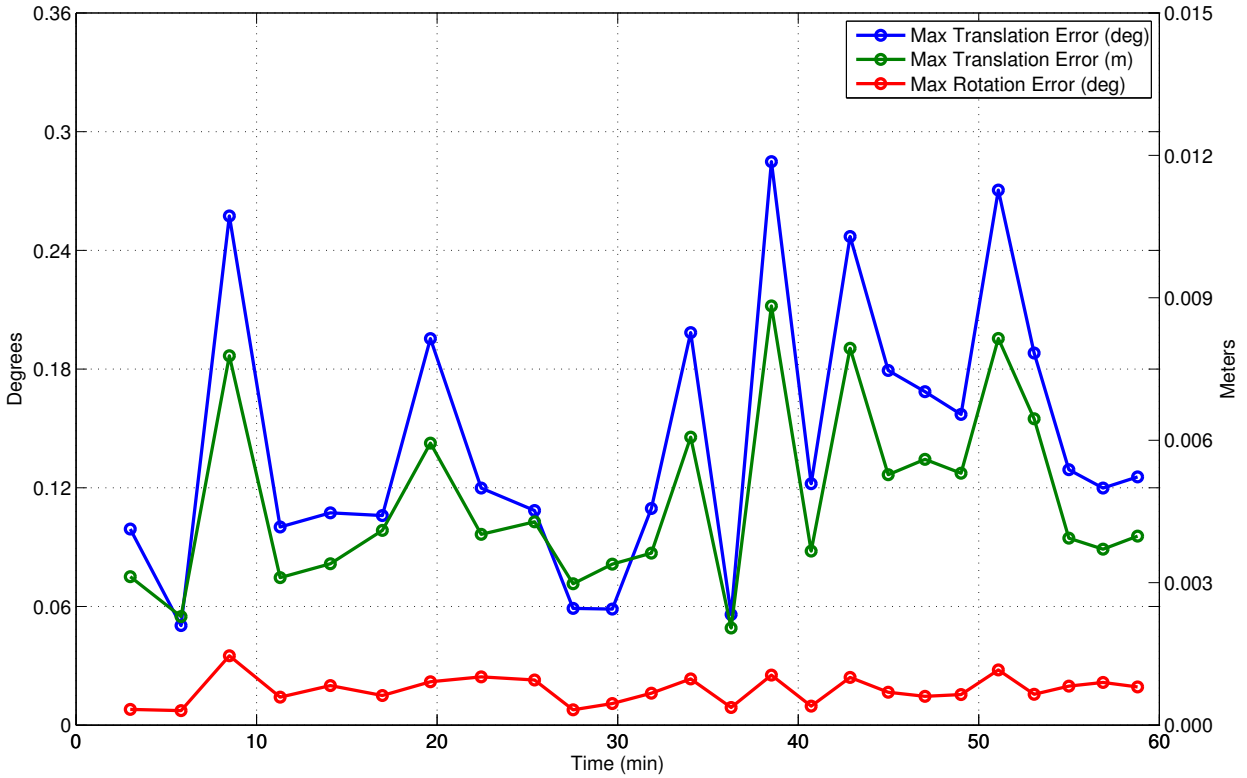


Figure 24: A plot of the maximum errors against the time at which each loop was completed.

Experiment 2 The left camera is rotated around its x-axis towards the ground by 30° as measured with a scale that is available for each axis of movement of the three-way tripod head. The infrastructure-based calibration used 411 sets of camera poses with an average of 3.14 camera poses per set. Our method estimates the left camera to have rotated about its x-axis by 29.9°. This estimate closely agrees with the hand measurement.

Experiment 3 The left camera is rotated around its z-axis towards the front of the car by 15°. The infrastructure-based calibration used 381 sets of camera poses with an average of 3.20 camera poses per set. Our method estimates the left camera to have rotated around its x-axis by 14.3°. This estimate closely agrees with the hand measurements.

Experiment 4 For each of the 25 loops, we estimate the extrinsics, and find the maximum of the three error metrics among all cameras. We then plot in Figure 24 these maximum errors against the time at which each loop was completed. Furthermore, we use a box plot in Figure 25 to show the distribution of the maximum errors over the 25 loops. These plots show that our calibration method is still very accurate regardless of changes in the environment, as the maximum rotation and translation errors do not exceed 0.035°, 0.28°, and 0.88 cm respectively. We note that these errors are higher than those from the indoor parking garage experiment as the average depth of the scene points in the indoor parking garage is smaller. In other words, the nearer the scene points to the camera, the more accurate the calibration parameters.

7 Discussions

Our past use of an AR-marker-based extrinsic calibration method motivated the development of our infrastructure-based calibration method. The AR-marker-based calibration method required us to prepare

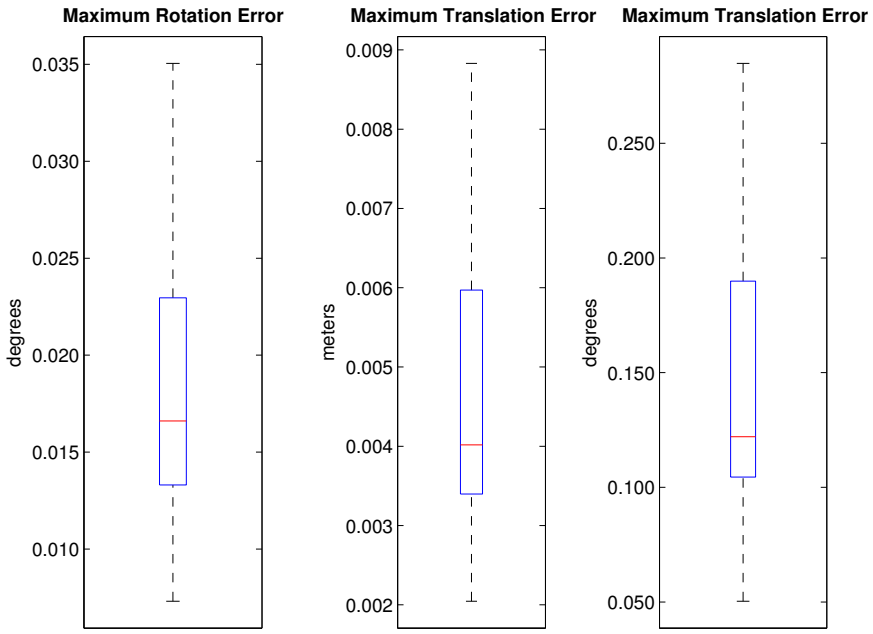


Figure 25: A box plot of the maximum errors for 25 loops.

many AR markers and a calibrated camera for taking images of the scene in conjunction with the multi-camera system. This method was time-consuming, and required a great deal of manual and repetitive work. Hence, we set out to develop a calibration pipeline that was unsupervised, and minimized operator involvement.

Our original implementation of the SLAM-based self-calibration method (Heng et al., 2013) had several flaws that led to an inaccurate map of the calibration area, and in turn, inaccurate estimates of the extrinsic parameters. On some occasions, wrong loop closures caused the map to become distorted, and the extrinsics to become inaccurate. The relative inter-camera poses were not sufficiently accurate for a seamless composite top-view image. Furthermore, optimization of the intrinsics in the bundle adjustment led to overfitting; lines in rectified pinhole images were no longer straight after joint optimization. This line warping issue caused problems for multi-view stereo applications that depended on rectified pinhole images. Lastly, scale drift occurred during joint optimization, causing the scale of both the extrinsics and the map to become inaccurate. Hence, in this paper, we added robust pose graph optimization to solve the false loop closure problem, and added multiple sets of inverse-covariance-weighted residuals in the joint optimization to improve the accuracy of the relative inter-camera poses, prevent overfitting of the intrinsics, and avoid scale drift.

Our infrastructure-based calibration method is able to produce accurate calibration parameters such that the resulting composite top-view image created from a multi-camera system shows seamless blending between images from all cameras. The infrastructure-based calibration consistently produces accurate calibration parameters as long as the map is metrically accurate, and is an accurate representation of the environment.

However, the infrastructure-based calibration method can fail in three cases:

1. If a map of a dynamically changing environment is used for the infrastructure-based calibration long after it was created, the image-based localization step may yield either too few or erroneous camera poses, and produce inaccurate calibration parameters as a result. Here, the image-based localization will not work well as expected because the map does not accurately represent the calibration area.

2. The map is metrically inaccurate and not globally consistent. There are three causes which cause a

degradation in the quality of the map provided by our self-calibration method:

(a) Wrong loop closures are considered in the construction of the map. During self-calibration, robust pose graph optimization may fail in self-similar environments in which a wrong loop closure occurs more often than a correct loop closure. As a result, wrong loop closures may be misclassified as correct, and in turn, the map becomes distorted, and the estimates of the extrinsics are inaccurate. However, we use a visualization tool to visually inspect the map after the completion of the self-calibration, and we observe that if the map visually looks correct given knowledge of the environment layout, the extrinsics are usually accurate.

(b) The odometry system on the survey vehicle is not calibrated. The scale accuracy of the map generated by our self-calibration method is closely tied to the scale accuracy of odometry data. If odometry data reports inaccurate scale information such that the distance travelled by the vehicle is either overestimated or underestimated, the scale of the map will be inaccurate. In turn, the scale of the extrinsics estimated by the infrastructure-based calibration is inaccurate. Nevertheless, if we ensure that the odometry system on the survey vehicle is always well-calibrated, the resulting map and extrinsics will be accurate in terms of scale.

(c) There are too few features in the environment due to lack of texture. Monocular visual odometry may break very often, loop closures may be difficult to detect, and feature correspondences between different cameras may be few. Hence, without sufficient features in the environment, we cannot obtain a metrically accurate and globally consistent map.

8 Conclusions

Through extensive field experiments, our SLAM-based self-calibration method is proven to provide a metrically accurate and globally consistent map of the calibration area during the survey phase. As a result, our infrastructure-based calibration method computes highly accurate calibration parameters for a multi-camera rig with non-overlapping fields of view. Due to the use of natural features, our infrastructure-based calibration method can be used in an arbitrary environment without the need for special calibration setups or environment modifications. Most importantly, the calibration is quick and unsupervised. Hence, no substantial time-related and material-related costs are incurred from using our method to calibrate a multi-camera rig.

Our extensive experiments clearly demonstrate the high accuracy of both our self-calibration and infrastructure-based calibration methods. Both methods are demonstrated to work both indoors and outdoors. We show that our infrastructure-based calibration method can infer calibration parameters for camera configurations which significantly differ from that used to build the map during the survey phase. We also show changes in the environment to have no impact on the calibration accuracy.

We see our infrastructure-based calibration method as having huge potential benefits for the robotics community and automotive industry as the method is unsupervised, robust, easy to use, and produces accurate calibration parameters. The implementations for the self-calibration and infrastructure-based calibration methods are available as part of the CamOdoCal library at `https://github.com/hengli/camodocal`.

9 Future Work

We plan to work on an online version of the calibration. The motivation is that several cameras on our car platforms are mounted on movable parts, for example, a car door, and that vision applications will fail when the cameras are moved. Furthermore, wear and tear causes the camera extrinsics to change slowly over time. An online calibration avoids the need for drivers to periodically bring their vehicles equipped with multi-camera systems to the workshop for recalibration. To deal with unobservable parameters during online calibration, we will explore the use of the method proposed by Maye et al. (2013), which locks unobservable

directions in parameter space, and chooses informative measurements to keep the computational complexity limited.

In addition, we will look at extending the usable lifespan of the map used for the infrastructure-based calibration by exploring time-invariant feature descriptors and illumination-invariant imaging.

Finally, we would like to investigate the possibility of integrating our recent work on continuous-time state estimation into the calibration pipeline. This would allow the estimation of the time offsets between the odometry and image measurement streams (Furgale et al., 2013a), and enable the pipeline to work with rolling shutter cameras (Oth et al., 2013) that are ubiquitous in the automotive industry.

Acknowledgments

The first author is funded by the DSO National Laboratories Postgraduate Scholarship. In addition, this work is supported in part by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant #269916 (V-Charge). We thank Christian Häne for the plane sweep stereo implementation, and Mathias Bürki and Jerome Maye for their hard work in setting up the Prius experimental platform for data collection.

References

Agarwal, S., Mierle, K., and Others (2013). Ceres solver. `https://code.google.com/p/ceres-solver/`.

Antonelli, G., Caccavale, F., Grossi, F., and Marino, A. (2010). Simultaneous calibration of odometry and camera for a differential drive mobile robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5417--5422.

Brookshire, J. and Teller, S. (2012). Extrinsic calibration from per-sensor egomotion. In *Robotics: Science and Systems (RSS)*.

Carrera, G., Angeli, A., and Davison, A. (2011). Slam-based automatic extrinsic calibration of a multi-camera rig. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2652--2659.

Censi, A., Franchi, A., Marchionni, L., and Oriolo, G. (2013). Simultaneous calibration of odometry and sensor parameters for mobile robots. *IEEE Transactions on Robotics*, 29(2):475--492.

Daniilidis, K. (1999). Hand-eye calibration using dual quaternions. *International Journal of Robotics Research (IJRR)*, 18(3):286--298.

Furgale, P., Rehder, J., and Siegwart, R. (2013a). Unified temporal and spatial calibration for multi-sensor systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1280--1286, Tokyo, Japan.

Furgale, P., Schwesinger, U., Rufli, M., Derendarz, W., Grimmett, H., Mühlfellner, P., Wonneberger, S., Rottmann, J. T. S., Li, B., Schmidt, B., Nguyen, T. N., Cardarelli, E., Cattani, S., Brüning, S., Horstmann, S., Stellmacher, M., Mielenz, H., Köser, K., Beermann, M., Häne, C., Heng, L., Lee, G. H., Fraundorfer, F., Iser, R., Triebel, R., Posner, I., Newman, P., Wolf, L., Pollefeys, M., Brosig, S., Effertz, J., Pradalier, C., and Siegwart, R. (2013b). Toward Automated Driving in Cities using Close-to-Market Sensors, an Overview of the V-Charge Project. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 809--816, Gold Coast, Australia.

Gao, C. and Spletzer, J. (2010). On-line calibration of multiple lidars on a mobile vehicle platform. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 279--284.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 32(11):1231--1237.

Geiger, A., Moosmann, F., Car, O., and Schuster, B. (2012). Automatic calibration of range and camera sensors using a single shot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3936--3943.

Guo, C., Mirzaei, F., and Roumeliotis, S. (2012). An analytical least-squares solution to the odometer-camera extrinsic calibration problem. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3962--3968.

Heng, L., Bürki, M., Lee, G. H., Furgale, P., Siegwart, R., and Pollefeys, M. (2014). Infrastructure-based calibration of a multi-camera rig. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Heng, L., Li, B., and Pollefeys, M. (2013). Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1793--1800.

Kneip, L., Furgale, P., and Siegwart, R. (2013). Using multi-camera systems in robotics: Efficient solutions to the npnp problem. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3770--3776.

Kneip, L., Scaramuzza, D., and Siegwart, R. (2011). A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2969--2976.

Kumar, R., Ilie, A., Frahm, J., and Pollefeys, M. (2008). Simple calibration of non-overlapping cameras with a mirror. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1--7.

Lebraly, P., Royer, E., Ait-Aider, O., Deymier, C., and Dhome, M. (2011). Fast calibration of embedded non-overlapping cameras. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 221--227.

Lee, G. H., Fraundorfer, F., and Pollefeys, M. (2013a). Robust pose-graph loop-closures with expectation-maximization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 556--563.

Lee, G. H., Li, B., Pollefeys, M., and Fraundorfer, F. (2013b). Minimal solutions for pose estimation of a multi-camera system. In *International Symposium on Robotics Research (ISRR)*.

Levinson, J. and Thrun, S. (2012). Automatic calibration of cameras and lasers in arbitrary scenes. In *International Symposium on Experimental Robotics (ISER)*.

Li, B., Heng, L., Köser, K., and Pollefeys, M. (2013). A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1301--1307.

Lim, H., Sinha, S., Cohen, M., and Uyttendaele, M. (2012). Real-time image-based 6-dof localization in large-scale environments. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1043--1050.

Maddern, W., Harrison, A., and Newman, P. (2012). Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3096--3102.

Markley, F., Cheng, Y., Crassidis, J., and Oshman, Y. (2007). Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(1):12--28.

Maye, J., Furgale, P., and Siegwart, R. (2013). Self-supervised calibration for robotic systems. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 473--480, Gold Coast, Australia.

Oth, L., Furgale, P., Kneip, L., and Siegwart, R. (2013). Rolling shutter camera calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1360--1367.

Sattler, T., Liebe, B., and Kobbelt, L. (2011). Fast image-based localization using direct 2d-to-3d matching. In *IEEE International Conference on Computer Vision (ICCV)*, pages 667--674.

Tariq, S. and Dellaert, F. (2004). A multi-camera 6-dof pose tracker. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 296--297.

Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (2000). Bundle adjustment a modern synthesis. In Triggs, B., Zisserman, A., and Szeliski, R., editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298--372. Springer Berlin Heidelberg.

Yang, R. and Pollefeys, M. (2003). Multi-resolution real-time stereo on commodity graphics hardware. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 211--217.