Self-calibration and visual SLAM with a multi-camera system on a micro aerial vehicle

Lionel Heng, Gim Hee Lee, Marc Pollefeys

Institutions: DSO National Laboratories, National University of Singapore, ETH Zurich

Published on: 01 Oct 2015 - Autonomous Robots (Springer US)

Topics: Stereo camera, Stereo cameras, Simultaneous localization and mapping and Triangulation (computer vision)

Self-Calibration and Visual SLAM with a Multi-Camera System on a Micro Aerial Vehicle

Lionel Heng, Gim Hee Lee, and Marc Pollefeys
Computer Vision and Geometry Lab, ETH Zürich, Switzerland

***Abstract*—The use of a multi-camera system enables a robot to obtain a surround view, and thus, maximize its perceptual awareness of its environment. An accurate calibration is a necessary prerequisite if vision-based simultaneous localization and mapping (vSLAM) is expected to provide reliable pose estimates for a micro aerial vehicle (MAV) with a multi-camera system. On our MAV, we set up each camera pair in a stereo configuration. We propose a novel vSLAM-based self-calibration method for a multi-sensor system that includes multiple calibrated stereo cameras and an inertial measurement unit (IMU). Our self-calibration estimates the transform with metric scale between each camera and the IMU. Once the MAV is calibrated, the MAV is able to estimate its global pose via a multi-camera vSLAM implementation based on the generalized camera model. We propose a novel minimal and linear 3-point algorithm that uses inertial information to recover the relative motion of the MAV with metric scale. Our constant-time vSLAM implementation with loop closures runs on-board the MAV in real-time. To the best of our knowledge, no published work has demonstrated real-time on-board vSLAM with loop closures. We show experimental results in both indoor and outdoor environments. The code for both the self-calibration and vSLAM is available as a set of ROS packages at https://github.com/hengli/vmav-ros-pkg.**

I. Introduction

Vision-based MAVs are more versatile than laser-based MAVs. Whereas a laser only provides geometry data, a camera can provide both geometry data via stereo and structure-from-motion techniques, and appearance data. A camera is a passive sensor while a laser is an active sensor and is thus susceptible to interference. Furthermore, a camera is lighter and has a smaller footprint. However, utmost care has to be taken when choosing the camera configuration for a vision-based MAV expected to operate robustly in challenging environments. A single-camera configuration introduces limited perceptual awareness, and in turn, flight constraints because if the camera observes too few features for some time, localization can fail and lead to a crash. In the case of a single downward-looking camera [32], the MAV cannot fly too close to the ground which often has little texture, and at the same time, it cannot perform obstacle avoidance due to the absence of a forward-looking camera. In the case of a forward-looking camera, constraints are imposed on the path planning. For example, in [24], any planned path ensures that there are a sufficient number of features for localization. In addition, in [11], a path is planned such that the MAV does not move outside the camera's field of view, and inadvertently crash into an unseen obstacle.

In this paper, we use a multi-camera system on a MAV; together with the use of fish-eye lenses, this camera configuration provides a surround view of the vicinity. Our multi-camera approach is similar in spirit to Furgale et al. [9] who uses a car equipped with multiple cameras configured to provide a surround view. The use of such a multi-camera system eliminates the constraints on path planning associated with a single camera as discussed earlier, and allows for immediate path planning in all directions. Furthermore, localization is more robust as there is always at least one camera that observes a sufficient number of features. Another benefit of using multiple cameras is that we directly infer metric scale.

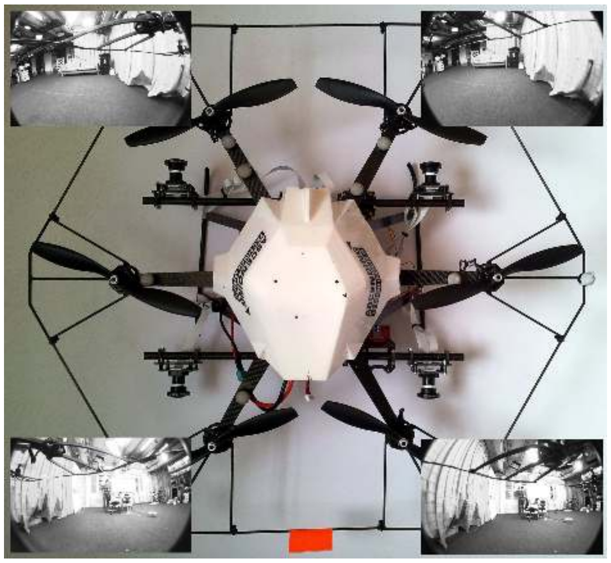


Fig. 1. We show an image from each of the four fish-eye cameras on our MAV platform. Our camera configuration provides a surround view. Each pair of cameras is arranged in a stereo configuration so that 3D scene information is available at all times.

Our MAV is self-contained in the sense that all algorithms necessary for autonomous flight are run on-board. With this self-contained MAV, we circumvent latency and reliability issues related to off-board computing, especially in areas where Wi-Fi reception is poor. On the other hand, more cameras correlate to a requirement for more computational resources, but with recent advances in computing hardware, especially in multi-core processors, it is now computationally feasible to run image processing algorithms on-board a MAV that uses multiple cameras.

For robust operation in the field, we arrange each camera pair in a stereo configuration such that 3D scene information is always available. Thus, we avoid the use of failure-prone map initialization methods required for monocular cameras and multi-camera systems with non-overlapping fields of view.

The use of multiple cameras on a MAV raises two issues we have to resolve before we can realize autonomous flight: finding the inter-sensor transforms, and estimating the MAV's pose with multiple synchronized sensors. We come up with novel methods for self-calibration and pose estimation based on the generalized camera model [23]. Our self-calibration method is designed for a multi-sensor system with multiple calibrated stereo cameras and an IMU; we do not assume overlapping fields of view between any two stereo cameras. This method is based on vSLAM, only assumes that each stereo camera is pre-calibrated, and does not require operator input, fiducial markers or external positioning systems. We are able to estimate the extrinsic calibration parameters with metric scale which is inferred from stereo. We do not know of other existing vSLAM-based self-calibration methods for multiple calibrated stereo cameras. We propose a 3-point minimal and linear solution for motion estimation that uses inertial information to recover the relative motion with metric scale. We incrementally build a graph of keyframes and constraints, and use the double-window optimization method [3] to optimize this graph. This graph optimization enables real-time on-board SLAM with loop closures on a vision-based MAV, which to the best of our knowledge, has not been shown before in published works.

II. Related Work

There is an extensive body of work on calibration for multi-sensor systems. For tractability, we look at current state-of-the-art methods that involve cameras. Kelly and Sukhatme [14] perform visual-inertial SLAM with a camera-IMU system, and at the same time, estimate the calibration parameters including the camera-IMU transform. However, the calibration parameters are only accurate as long as the camera-IMU system experiences constant excitation. Furgale et al. [8] proposes an offline method to find an accurate estimate of the transform and the temporal offset between a camera and an IMU using a calibration pattern. Here, this method makes use of temporal basis functions for parameterization of continuous-time variables, and continuous-time batch estimation. If we use this approach to calibrate each camera-IMU pair, the inter-camera transforms may not be accurate as we do not consider camera-camera constraints in the form of 3D scene points mutually observed by multiple cameras. Brookshire and Teller [4] take a more generic approach similar to hand-eye calibration by calibrating a multi-sensor system based on relative motion measurements for each sensor. In contrast, in our self-calibration, we exploit 3D scene points mutually observed by multiple cameras to obtain a more accurate estimate of inter-camera transforms. Carrera et al. [6], Heng et al. [12] develop self-calibration methods for multi-camera systems that are based on vSLAM. However, [6] only estimates the inter-camera transforms up to scale while [12] infers metric scale from odometry data. In contrast, by utilizing calibrated stereo, our self-calibration method estimates the inter-camera transforms with metric scale without requiring odometry data.

We explore existing work on SLAM with a multi-camera system [13, 15, 5, 31]. Kaess and Dellaert [13] solve an optimization problem comprising pose-point constraints and odometry constraints in order to obtain the pose of the multi-camera system, and do not perform loop closure detection. Kim et al. [15] models a multi-camera system as a spherical camera. The drawback with this spherical model is that the relative motion of the system can only be estimated up to scale. Furthermore, they do not show experiments with their multi-camera system on unmanned aerial vehicles. Carrera et al. [5] uses a forward-looking camera and a backward-looking camera on a ground robot, and implements pose-graph SLAM based on odometry data and loop closure detection. The absence of bundle adjustment in any form limits the metric accuracy of the map. Tribou [31] implements a multi-camera version of PTAM that makes use of a three-camera system with non-overlapping fields of view on a MAV. As initial scene point depths are not known, an initialized map does not have accurate metric scale, and bundle adjustment that runs in a separate thread is relied on to gradually recover metric scale. Due to the cubic complexity of bundle adjustment, the approach of Tribou [31] does not scale to large environments. Furthermore, pose estimates with incorrect metric scale can cause control instability issues. In addition, loop closures are not performed. Li et al. [21] shows that in the case of only intra-camera feature correspondences, and the relative rotation being an identity matrix, the generalized epipolar constraint reduces to the epipolar constraint which only allows us to recover the relative pose up to scale. Due to the absence of inter-camera feature correspondences from non-overlapping fields of view, metric scale cannot be recovered if the MAV moves with minimal rotation. In contrast, wide overlapping fields of views for each stereo camera on our MAV ensure that we always have inter-camera feature correspondences, and thus, are able to obtain pose estimates with metric scale all the time. Furthermore, our use of double-window optimization [3] allows our SLAM implementation to run in constant time, and thus, scale to large environments.

We then look at vision-based MAVs that run real-time pose estimation algorithms on-board. Schauwecker and Zell [27] deploy one downward-looking stereo camera and one forward-looking stereo camera on a MAV, independently estimate the MAV's pose from each camera, and fuse both pose estimates. In contrast, we use all cameras to obtain a single pose estimate. Weiss et al. [32] utilizes both an IMU and downward-looking camera together with a modified version of PTAM to estimate the MAV's pose, and infer scale from accelerometer measurements. Schmid et al. [28] uses a FPGA board to compute depth maps from a forward-looking stereo camera and relies on stereo visual odometry for pose estimation. Shen et al. [29] uses a forward-looking stereo camera with fish-eye lenses. Here, they use gyroscopic measurements to filter out incorrect feature correspondences, and uses a local map to estimate the MAV's pose. We note that the pose estimation in all discussed works is susceptible to drift as loop closure detection is not carried out. To the best of our knowledge, there is no published work on real-time on-board SLAM with loop closures for vision-based MAVs.

A number of works exploit the vertical direction information provided by the IMU to simplify motion estimation algorithms; the vertical direction is not susceptible to drift unlike gyroscopic measurements. However, the IMU present on our MAV is of the MEMS type, and the maximum error of the vertical direction is 3 degrees as measured with a Vicon motion capture system. As a result, we did not get accurate relative pose estimates from using the known vertical in our motion estimation algorithm.

III. MAV Platform

We use a AscTec Firefly equipped with an Intel Core i7 single computer board and the ROS[1] framework for message transmission. The IMU on the AscTec Firefly is time-synchronized to the single computer board. A VRmagic D3 four-camera system is mounted on the AscTec Firefly. In this multi-camera system, each camera is connected to an ARM Cortex A8 board via a proprietary cable. A Lensagon 1.5 mm f/2.0 fish-eye lens with a 185° field of view is fitted to each camera. The IMU on the AscTec Firefly sends a periodic trigger signal at 15 Hz to the ARM board which then grabs pixel-synchronous 754×480 monochrome images from all four cameras. Each time a trigger is sent, the IMU publishes inertial data. In this way, we facilitate inertial-visual fusion by making inertial information available for each image. Furthermore, all sensor measurements are timestamped with respect to the system clock on the single board computer. On the ARM board, we use the RTI Connext DDS middleware[2] to transmit image data over an Ethernet connection to the single computer board. We note that this high-bandwidth image transmission is not possible with ROS as ROS is only able to publish a maximum of 15 images per second on the ARM board given the computational constraints of the ARM processor; still, we leverage ROS for all other message transmissions due to the wide variety of libraries and ease of use that ROS offers for message data manipulation.

To enable autonomous flight, we use the state estimation framework from [32] for robust pose estimation by fusing both inertial data from the IMU and pose data from vSLAM.

For the purpose of brevity, we define the following symbols to be used in this paper. We define a n-camera system to contain cameras $C_1, \ldots, C_n$. In this multi-camera system, there are $\frac{n}{2}$ stereo cameras $S_1, \ldots, S_{\frac{n}{2}}$ and each stereo camera S_i comprises the camera pair $\{C_{2i-1}, C_{2i}\}$. For each camera C_i, we denote its intrinsics as K_{C_i}, and its extrinsics with respect to the IMU's reference frame V as $[R_{C_i}, t_{C_i}]$.

IV. Self-Calibration

Our self-calibration method utilizes vSLAM with natural features, and estimates the camera-IMU transforms for a multi-sensor system with multiple calibrated stereo cameras and an IMU. Although this method is designed for calibration of a multi-camera system in which each pair of cameras is arranged

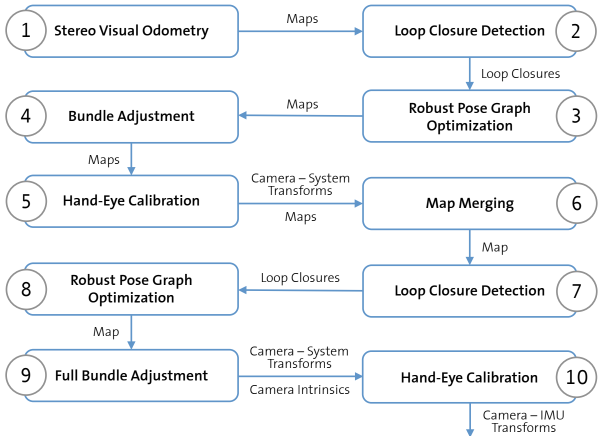


Fig. 2. Our self-calibration pipeline estimates the camera-IMU transforms.

in a stereo configuration, this method can easily be extended to a multi-camera system with at least one stereo camera and any number of monocular cameras. One by-product of the self-calibration is a metrically accurate and globally consistent map which can be used for map-based localization.

Figure 2 shows the pipeline underlying the self-calibration method. From steps 1-4, we obtain a globally consistent map for each stereo camera via stereo vSLAM. Using the poses of the stereo cameras estimated by stereo vSLAM, step 5 gives us an initial estimate of the inter-stereo-camera transforms. Step 6 merges maps from all stereo cameras into a single map. In steps 7-8, we obtain globally consistent pose estimates for the multi-camera system. In addition, we obtain inter-stereo-camera feature correspondences that correspond to loop closures classified as correct. These feature correspondences provide a strong prior for accurate inter-stereo-camera transforms, and in step 9, allow us to recover an accurate estimate of the inter-camera transforms. Step 10 yields the transform between the multi-camera system and the IMU, and thus, the camera-IMU transforms.

A. Stereo Calibration

The stereo calibration outputs the camera intrinsics and stereo transform for each stereo camera. The unified projection model and plumb bob distortion model [22] are used to model the camera intrinsics. In our chessboard-based stereo calibration, we detect the chessboard in each pair of stereo images, and after a minimum number of chessboards is detected, we use the method in [22] to find the initial values for the camera poses and intrinsic parameters. From the camera poses, we infer the stereo camera poses and the stereo transform between the two cameras, and subsequently, use non-linear refinement to optimize the intrinsic parameters, stereo camera poses, and the stereo transform.

B. Extrinsic Calibration

1) Stereo Visual Odometry (VO): In this step, each stereo camera builds its own map. For each stereo frame, we use the CenSurE feature detector [2] implemented in OpenCV to detect features in each image. For each feature, we compute both the ORB feature descriptor [26] and the backprojected

[1]http://www.ros.org

[2]https://www.rti.com/products/dds/index.html

unitary ray. We match features between the two images, and for each feature match with corresponding backprojected rays (r_1, r_2), if $r_2 E r_1$ is less than a threshold where E is the essential matrix representing the stereo transform, we mark the feature match as valid and triangulate the feature match.

We define the reference frame of the stereo camera $\{C_i, C_{i+1}\}$ to be that of camera C_i. To compute the current pose of the stereo camera, we find 3D-2D correspondences between the first images of the previous and current stereo frames, and use the P3P method [17] together with RANSAC to find the pose that corresponds to the highest number of inliers. We optimize the pose via sliding window bundle adjustment. Here, each error residual is equivalent to the dot product between the observed backprojected ray and the ray passing through the camera center and the 3D scene point. This dot product is faster to compute compared to the image reprojection error.

2) Loop Closure Detection: The loop closure detection step is the first of the next three steps that together make the map built by each stereo camera globally consistent. We use the DBoW2 implementation of the vocabulary tree [10]. For a stereo camera $\{C_i, C_{i+1}\}$, we restrict loop closure detection to images from camera C_i. For the first image in each stereo frame, we find the n most similar images, and we filter out matched images whose timestamps are too close to the that of the query frame. This is to avoid unnecessary linking of adjacent frames. Again, we use the P3P method [17] together with RANSAC to find an inlier set of 2D-3D correspondences for each of the n images. We add a loop closure between the query frame and the frame which has the highest number of 2D-3D correspondences provided that this number exceeds a threshold.

3) Robust Pose Graph Optimization: This step finds correct loop closures. At the same time, we obtain globally consistent estimates of the stereo camera poses which are a better initial guess for bundle adjustment compared to pose estimates from stereo VO which have drift. Hence, better convergence is achieved in bundle adjustment. We build a pose graph in which the nodes are the stereo camera poses, and edges between nodes correspond to measurements of relative 6D transforms obtained from either stereo VO or loop closures. We use the approach in Lee et al. [18] to simultaneously optimize this pose graph, and classify loop closures as either correct or wrong. We then merge pairs of duplicate 3D scene points corresponding to correct loop closures.

4) Bundle Adjustment: This step outputs a globally consistent map for each stereo camera. Here, we run bundle adjustment to optimize the stereo camera poses and 3D scene points. The error residuals in the bundle adjustment are based on the dot product of two rays as described in Section IV-B1.

5) Hand-Eye Calibration: In this step, we find an initial estimate of the transforms between stereo cameras. Without loss of generality, we assume that the reference frame of the camera system is the same as that of camera C_1. Using the optimized stereo camera poses as input, we use the hand-eye calibration method [7] to find the transforms between S_1 and S_i for $i = 2, \ldots, \frac{n}{2}$. As a result, we have an initial estimate for the pose of each camera with respect to the camera system's reference frame.

6) Map Merging: In this step, we merge the maps from all stereo cameras. At the beginning, we set the poses of the camera system to be the same as that of S_1. We recompute the scene points observed by S_i for $i > 1$ by multiplying the inverse of the pose of S_i and the original 3D coordinates of the scene points with respect to S_i and which were computed during stereo triangulation in Section IV-B1.

7) Loop Closure Detection: We restrict loop closure detection to images from cameras $C_1, \ldots, C_{n-1}$ and perform the same loop closure detection procedure as described in Section IV-B2.

8) Robust Pose Graph Optimization: We repeat the robust pose graph optimization procedure as described in Section IV-B3. From loop closures identified as correct, we obtain feature correspondences between different stereo cameras, and these feature correspondences allow the full bundle adjustment to recover an accurate estimate of the transforms between the stereo cameras.

9) Full Bundle Adjustment: We run full bundle adjustment to optimize the camera intrinsics, camera extrinsics, camera poses, and 3D scene points. In the full bundle adjustment, we minimize a cost function comprising two sets of residuals:

$$\begin{aligned} &\min_{\mathbf{K}_c, \mathbf{P}_i, \mathbf{Q}_j, \mathbf{T}_c, \mathbf{X}_p} \\ &\sum_{c,i,p} w_p \rho\left(||\pi_1(\mathbf{K}_c, \mathbf{P}_i, \mathbf{T}_c, \mathbf{X}_p) - \mathbf{p}_{cip}||^2\right) \\ &+ \sum_{c,j,q} \rho\left(||\pi_2(\mathbf{K}_c, \mathbf{Q}_j, \mathbf{Y}_q) - \mathbf{p}_{cjq}||^2\right). \end{aligned} \tag{1}$$

π_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 $\mathbf{K}_c$, the camera system pose $\mathbf{P}_i$, and the transform from the camera frame to the camera system frame $\mathbf{T}_c$. $\mathbf{p}_{cip}$ is the observed image coordinates of $\mathbf{X}_p$ seen in camera c with the corresponding camera system pose $\mathbf{P}_i$. Similarly, π_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 $\mathbf{K}_c$, 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.

In equation 1, the first set of residuals corresponds to the sum of image reprojection errors of the 3D scene points, and the second set of residuals corresponds to the sum of image reprojection errors of the chessboard corner points. 3D scene points observed by multiple stereo cameras usually make up a small percentage of all scene points in the map. To ensure that these 3D scene points make a significant contribution to accurate estimation of the inter-camera transforms, we assign a higher value to w_p for feature observations that are associated with such 3D scene points. We optimize the camera intrinsic parameters as we wish to make use of the high number of 3D scene points to improve the accuracy of the camera intrinsics.

However, overfitting of intrinsic parameters can occur. To prevent overfitting, and at the same time, ensure metric scale, we include the residuals from the stereo camera calibrations.

10) Hand-Eye Calibration: We find the rotation offset between the reference frames of the IMU and the camera system using the method in [7]. We do not require the translation offset as our three-point algorithm for motion estimation only makes use of the rotation offset. However, the translation offset is required for state estimation which fuses data from both the IMU and vSLAM, and a hand-measurement is sufficiently accurate. After the hand-eye calibration, we have the final estimates of the camera-IMU transforms.

V. Visual SLAM

In this section, we describe the algorithms used in our keyframe-based vSLAM implementation. We propose a novel 3-point algorithm to estimate the relative motion of the MAV with metric scale and with respect to the current keyframe. As this 3-point algorithm makes use of the relative rotation measurement from the IMU via short-term integration of gyroscopic measurements, the accuracy of this relative rotation measurement with respect to the current keyframe drops over time due to gyroscopic drift. Hence, after a certain period of time during which there is no new keyframe, we switch to the pose estimation method [20] which is also based on the generalized camera model and uses 3 correspondences. Our motion estimation technique is far more computationally efficient than the pose estimation technique. Furthermore, our motion estimation algorithm is linear and computes one unique solution, while the pose estimation algorithm is non-linear and returns up to 8 solutions. A computational analysis reveals that the number of arithmetic operations required by the pose estimation algorithm is a very high multiple of that required by the motion estimation algorithm on the order of ten thousands. However, the pose estimation algorithm does not assume any prior information unlike the motion estimation algorithm.

We mark the current frame as a keyframe if the number of correspondences falls below a threshold. Over time, we incrementally build a graph of keyframes and constraints obtained from both visual odometry and loop closures. We choose the double-window optimization method [3] to optimize the graph as the ability of this method to run in constant time makes real-time vSLAM on-board a MAV feasible.

A. Motion Estimation

Here, we discuss in depth our novel 3-point algorithm for motion estimation based on the generalized camera model.

1) Generalized Epipolar Constraint (GEC): Pless [23] introduced the generalized camera model for a multi-camera system which allows for non-central projection. In this model, we replace each image pixel $\mathbf{x}$ in camera C_i with a ray expressed as a Plücker line 6-vector $\mathbf{L}$ that passes through the camera center of C_i and the normalized image point $\hat{\mathbf{x}} = \mathbf{K}_{C_i}^{-1}\mathbf{x}$:

$$\mathbf{L} = \begin{bmatrix} \mathbf{q}^T & \mathbf{q}'^T \end{bmatrix}^T, \tag{2}$$

TABLE I
Comparisons of total number of iterations needed for RANSAC ($w = 0.5$ and $p = 0.99$)

Algorithm	# RANSAC Iterations	# Solutions	Total
Minimal 3-Point	34	1	34
Minimal 6-Point [30]	292	64	18688
Linear 17-Point [23]	603606	1	603606

where $\mathbf{q}$ and $\mathbf{q}'$ are the direction and moment vectors:

$$\mathbf{q} = \mathbf{R}_{C_i}\hat{\mathbf{x}}, \quad \mathbf{q}' = \mathbf{t}_{C_i} \times \mathbf{q}. \tag{3}$$

We write the GEC [23] as

$$\mathbf{L}_2{}^T \underbrace{\begin{bmatrix} \mathbf{E} & \mathbf{R} \\ \mathbf{R} & 0 \end{bmatrix}}_{\mathbf{E}_{GC}} \mathbf{L}_1^T = 0, \tag{4}$$

where $\mathbf{L}_1 \leftrightarrow \mathbf{L}_2$ are two Plücker line vectors representing a ray correspondence between two generalized camera frames V_1 and V_2, and $\mathbf{E}_{GC}$ is the 6×6 generalized essential matrix in which the first item is the conventional essential matrix $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$ where $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation from V_1 to V_2. We note that for the conventional essential matrix, $\mathbf{t}$ is computed only up to scale, but for a generalized camera, the scale of $\mathbf{t}$ can be computed as shown in [21]. In the degenerate case of no inter-camera feature correspondences and $\mathbf{R}$ being an identity matrix, the scale of $\mathbf{t}$ cannot be recovered. However, the wide overlapping fields of view for each stereo camera ensure that inter-camera feature correspondences are always present, and thus, we are always able to compute the scale of $\mathbf{t}$.

2) Minimal 3-Point Algorithm: We estimate R from short-term integration of gyroscopic measurements between the two frames V_1 and V_2. Substituting $\mathbf{L}_1 = [\mathbf{q}_1^T \; \mathbf{q}_1'^T]^T$ and $\mathbf{L}_2 = [\mathbf{q}_2^T \; \mathbf{q}_2'^T]^T$ into and rearranging equation 4 in the form $\mathbf{At} = \mathbf{b}$, we get

$$\underbrace{\mathbf{q}_1^T \mathbf{R}^T [\mathbf{q}_2]_\times}_{\mathbf{A}} \mathbf{t} = \underbrace{-\mathbf{q}_1^T \mathbf{R}^T \mathbf{q}_2' - \mathbf{q}_1'^T \mathbf{R}^T \mathbf{q}_2}_{\mathbf{b}}. \tag{5}$$

As $\mathbf{t}$ has three unknown variables, we require 3 Plücker line correspondences to solve for $\mathbf{t}$. Given 3 Plücker line correspondences, we construct the 3×3 matrix

$$\mathbf{C} = \begin{bmatrix} \mathbf{A}_1^T & \mathbf{A}_2^T & \mathbf{A}_3^T \end{bmatrix}^T, \tag{6}$$

and the 3-vector

$$\mathbf{D} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \end{bmatrix}^T, \tag{7}$$

and solve the system of linear equations $\mathbf{Ct} = \mathbf{D}$ to obtain $\mathbf{t}$. We note that this 3-point algorithm is linear and does not require scene point triangulation.

3) Robust Estimation: We make our 3-point algorithm robust to outliers by implementing it within RANSAC. We determine the best solution by choosing the solution that has the highest number of inliers.

The number of iterations k needed in RANSAC for a n-point algorithm is given by $k = \frac{\ln(1-p)}{\ln(1-w^n)}$, where n is the number of correspondences, w is the probability that any selected correspondence is an inlier, and p is the probability that all the selected correspondences are inliers. The total number of iterations m needed to run RANSAC while evaluating s solutions is given by $m = k \times s$. Table I shows the number of iterations required by our 3-point algorithm; for comparison, we include the 6-point minimal solution [30] and linear 17-point solution [23] to the GEC problem. We observe that our 3-point algorithm requires much fewer iterations compared to the 6-point and 17-point solutions; in other words, our 3-point algorithm is very computationally efficient when estimating the relative motion. We then optimize the relative motion estimate returned by RANSAC using non-linear refinement and the inlier set of correspondences associated with the relative motion estimate.

B. Pose Estimation

When there is no new keyframe for some time, the relative rotation measurement between the current keyframe and the current frame becomes inaccurate due to gyroscopic drift, and using such rotation measurements will cause our 3-point algorithm for motion estimation to return inaccurate estimates. In this case, we switch to the pose estimation algorithm [20] based on 3D scene points observed in the current keyframe. In this pose estimation algorithm, we require only 3 correspondences to recover the absolute pose associated with the current frame. The pose estimation algorithm returns up to 8 solutions, but in most cases, 2 solutions are returned. As in motion estimation, we use the RANSAC framework for robustness to outliers, and to find the best solution that corresponds to the highest number of inliers.

C. Location Recognition

To ensure that the map in the immediate vicinity of the MAV is globally consistent, we find loop closures. We use the same P3P RANSAC technique discussed in Section IV-B2 to find loop closures.

D. Optimization

For graph optimization, we use the double-window optimization method [3] which we implement using Google's Ceres Solver [1]. In double-window optimization, we simultaneously optimize both an inner window and outer window which correspond to sets of pose-point constraints and sets of pose-pose constraints respectively. For robustness to outliers, we use the Huber and Cauchy robust cost functions respectively for the inner and outer windows. We make one modification to the double-window optimization such that we include residuals, each of which corresponds to the error between the vertical direction associated with the MAV's estimated pose and the vertical direction measurement from the IMU. This modification ensures that the map is aligned with the ground plane.

VI. Experiments and Results

We conduct experiments to evaluate the accuracy of our self-calibration method and vSLAM implementation. We use a Vicon motion capture system in some experiments for the sole purpose of collecting ground truth data. The results of these experiments are described in detail in this section.

TABLE II
COMPARISON OF OUR ESTIMATED INTER-SENSOR TRANSFORMS WITH GROUND-TRUTH ESTIMATES FROM THE VICON-BASED CALIBRATION METHOD. ${}^{i}\mathbf{R}_j$ IS THE ROTATION GIVEN IN ROLL, PITCH, AND YAW ANGLES BETWEEN SENSORS i AND j. ${}^{i}\mathbf{t}_j$ IS THE TRANSLATION BETWEEN SENSORS i AND j.

	Our self-calibration	Difference with Vicon-based calibration
${}^{C_1}\mathbf{R}_{C_2}$	$[-0.06^\circ\ 0.20^\circ\ -1.39^\circ]$	$[0.001^\circ\ 0.035^\circ\ 0.016^\circ]$
${}^{C_1}\mathbf{t}_{C_2}$	$[31.89\ -0.10\ 0.04]$ cm	$[0.15\ 0.01\ 0.11]$ cm
${}^{C_1}\mathbf{R}_{C_3}$	$[177.41^\circ\ 0.85^\circ\ 176.59^\circ]$	$[0.177^\circ\ 0.025^\circ\ 0.029^\circ]$
${}^{C_1}\mathbf{t}_{C_3}$	$[1.38\ 3.31\ -28.21]$ cm	$[0.67\ 0.14\ 0.55]$ cm
${}^{C_1}\mathbf{R}_{C_4}$	$[176.20^\circ\ 0.34^\circ\ 177.63^\circ]$	$[0.183^\circ\ 0.060^\circ\ 0.086^\circ]$
${}^{C_1}\mathbf{t}_{C_4}$	$[32.66\ 1.47\ -27.96]$ cm	$[0.73\ 0.06\ 0.61]$ cm
${}^{IMU}\mathbf{R}_{C_1}$	$[-94.09^\circ\ 2.42^\circ\ -92.07^\circ]$	$[0.231^\circ\ 0.050^\circ\ 0.187^\circ]$

A. Self-Calibration

We use a Vicon motion capture system to evaluate the accuracy of the parameters estimated by our self-calibration method. To obtain ground-truth estimates, we devise a Vicon-based calibration method in which we use the Vicon system to track the pose of both the MAV and a chessboard moving across the field of view of each camera. Hence, the transform between the chessboard and MAV is known at any time step. From intrinsic camera calibration which also computes the camera poses with respect to the chessboard, we obtain an initial guess of both the intrinsic camera parameters and the transform between each camera and the MAV. We optimize these intrinsic parameters and camera-MAV transforms via non-linear refinement. From Vicon pose measurements and IMU measurements, we perform a hand-eye calibration to find a ground-truth estimate of the rotation between the MAV and the IMU, and thus, the camera-IMU transforms.

Prior to running our self-calibration method, we separately calibrate each stereo camera on the MAV. Subsequently, we carry the MAV along a figure-8 path several times while varying the height of the MAV above the ground. During this manoeuvre, the MAV's heading is roughly parallel to the direction of motion. This figure-8 manoeuvre ensures that all calibration parameters are fully observable.

The figure-8 manoeuvre took 90 seconds and the travelled distance was 54.16 m. Our self-calibration took 16 minutes, and the resulting average reprojection error associated with the generated map was 0.659 pixels. The map had 23957 scene points with an average scene point depth of 4.36 m.

We report in the first column of Table II the estimated rotation in terms of roll, pitch, and yaw angles, and translation of C_i with respect to C_1 for $i = 2, 3, 4$. We also report the estimated rotation and translation of C_1 with respect to the IMU. The second column of Table II shows the differences between the estimated rotations/translations and those estimated by the Vicon-based calibration method. We observe that our estimated inter-camera transforms are accurate with rotation and translation errors not exceeding 0.183° and 0.73 cm respectively.

The average error of the MAV's positions estimated by our self-calibration method using Vicon measurements as ground truth is 1.28 cm. This low error indicates that our self-calibration produces an accurate map, and in turn, accurate calibration parameters.

B. vSLAM

We conduct a combination of both simulation and real-world experiments. The simulation experiments are designed to provide a quantitative analysis of the accuracy of our 3-point algorithm for motion estimation. Two real-world experiments are carried out in different settings to verify the accuracy of the poses estimated by our vSLAM implementation. In both real-world experiments, the MAV flies autonomously by relying on pose estimates from vSLAM which are input to the state estimator, and we manually send velocity commands to the MAV via a remote control. In the first experiment, the MAV moves along multiple loops in an indoor environment, and we use the Vicon motion capture system to record the MAV poses. In the second experiment, the MAV flies in a horizontal loop in an outdoor environment, and we use the loop closure error metric to evaluate the pose accuracy.

In all real-world experiments, the input to our vSLAM implementation consists of 754×480 images from the 4-camera system together with inertial data, and our vSLAM implementation runs at 7-12 Hz with the inner window size and outer window size for the double-window optimization set to 15 and 50 respectively.

1) Simulation Experiments: Here, we run simulations to quantify the accuracy of our 3-point algorithm for motion estimation, and compare the accuracy against that of the 2-point algorithm based on the Ackermann motion model [19] and that of the linear 17-point algorithm [23]. In each simulation, we use the same multi-camera system setup on the MAV. For each trial, we generate a random relative motion (θ, ρ) where θ and ρ are the relative yaw and scale respectively of the Ackermann motion defined in Lee et al. [19]. θ and ρ are assigned random values within the ranges of $[0.05, 0.15]$ rad and $[0.25, 0.75]$ m respectively. The 3D scene points are randomly generated within the range of $[-10, 10]$ m with respect to the world reference frame. Point correspondences are obtained by reprojecting the 3D scene points into the cameras, and we ensure that each 3D scene point is seen by at least one camera over two consecutive frames.

Figures 3(a) and 3(b) show the average translation and rotation errors over 1000 trials, and a range of pixel noise levels between 0 and 1 pixels and with a 0.1 pixel interval. Following Quan and Lan [25], we define the translation error as $2||\mathbf{t} - \tilde{\mathbf{t}}||/(||\mathbf{t}|| + ||\tilde{\mathbf{t}}||)$, where $\mathbf{t}$ and $\tilde{\mathbf{t}}$ are the estimated and ground truth translations. The rotation error is defined as the norm of the Euler angles from $\mathbf{R}\tilde{\mathbf{R}}^T$, where $\mathbf{R}$ and $\tilde{\mathbf{R}}$ are the estimated and ground truth rotation matrices. We can see that the errors from the linear 17-point algorithm are significantly higher, while the 2-point and our 3-point algorithms similarly show low errors.

Figures 3(c) and 3(d) show the translation and rotation errors over 1000 trials, and a range of IMU noise levels between 0 and 0.6 degrees and with an interval of 0.1 degrees. The pixel noise is kept fixed at 0.5 pixels. At each IMU noise level, we collectively corrupt the roll, pitch and yaw angles with noise. We can see that the error from the linear 17-point algorithm is significantly higher than our 3-point algorithm despite the fact that the relative rotation measurement from the IMU is corrupted with noise. Note that the IMU used on our MAV has a maximum error of 0.1 degrees over two consecutive frames.

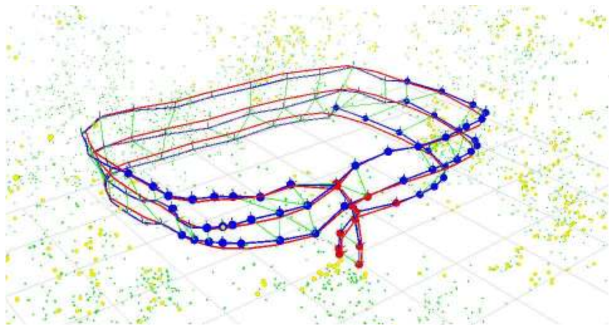


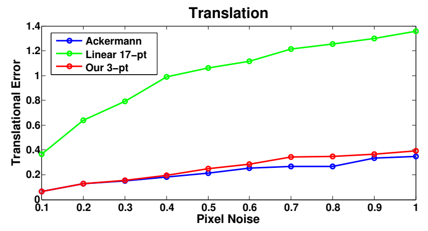
Fig. 4. We show the keyframe graph at the end of a 3-loop flight in an indoor environment. The red and blue spheres represent the keyframe poses belonging to the inner and outer windows respectively. A blue line traces the keyframe positions while a red line traces the corresponding Vicon-measured positions.

We also look at how each algorithm performs when we allow relative motion along the z-axis, and here, the Ackermann motion constraint is violated. Figures 3(e) and 3(f) show the translation and rotation errors as we set the z-component of the relative translation from 0 to 0.6 m with a step of 0.1 m. The pixel noise is kept fixed at 0.5 pixels. We can see that the errors from the 2-point algorithm increase as the Ackermann constraint is increasingly violated. In contrast, the errors from our 3-point algorithm remain relatively constant, and at the same time, they are significantly lower than the errors from the linear 17-point algorithm.

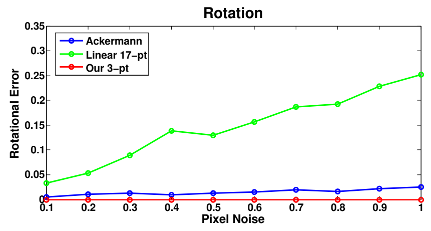
2) Indoor Experiment: In this indoor experiment, the MAV flies 3 loops over a distance of 45.48 m, starting each loop at an increasing height. Figure 4 shows the state of the keyframe graph at the end of the flight. In this figure, the keyframes belonging to the inner and outer windows are marked with red and blue spheres respectively. Green lines between keyframe pairs indicate loop closures. Green points represent 3D scene points in the map while yellow points represent 3D scene points observed in the inner window. A blue line traces the keyframe positions as estimated by our vSLAM implementation. A red line traces the corresponding Vicon-measured keyframe positions.

Based on Vicon measurements, the average error of the 110 keyframe positions is 6.34 cm. We observe that the errors associated with the keyframes not belonging to either the inner or outer windows, and especially at the boundary of the outer window, are higher as these keyframes are not included in the optimization.

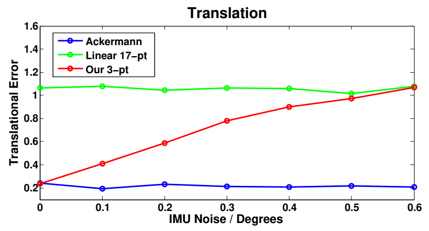
We re-run the vSLAM implementation with real-time playback of the data logged from this experiment, and make one change to the double-window optimization such that the



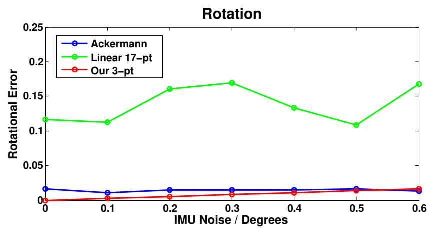
(a)



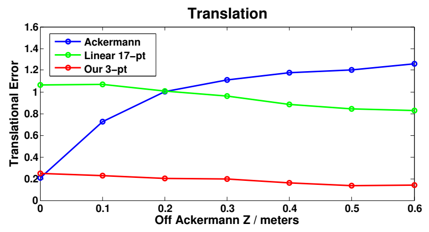
(b)



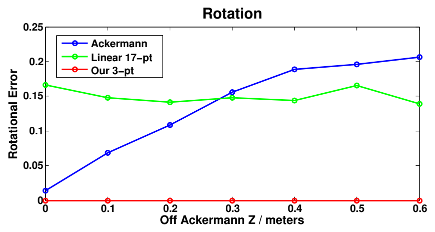
(c)



(d)



(e)



(f)

Fig. 3. Comparisons of translation (no units) and rotation (rad) errors from the Ackermann 2-pt, linear 17-point, and our 3-point algorithms in simulation over (a)-(b) image pixel noise, (c)-(d) IMU noise with pixel noise fixed at 0.5 pixels, and (e)-(f) off Ackermann motion along the z-axis with pixel noise fixed at 0.5 pixels.

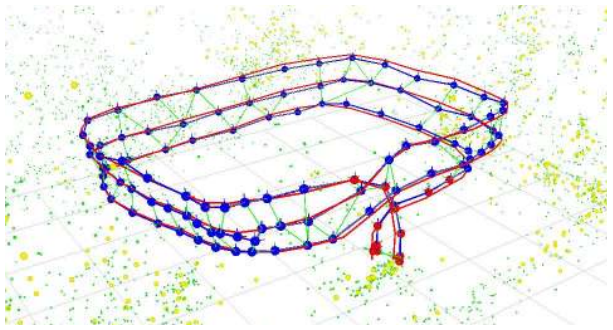


Fig. 5. We increase the size of the outer window to include all keyframes except those in the inner window. As a result, the keyframe position accuracy increases, but the running time of the double-window optimization now scales with the number of keyframes instead of being constant.

outer window includes all keyframes except those in the inner window. Figure 5 shows the state of the keyframe graph at the end of the flight. As expected, the average error of the keyframe positions decreases to 4.57 cm. However, with this outer window setting, the double-window optimization no longer runs in constant time, and thus, is not capable of real-time performance.

The results show that by switching from optimization over all keyframes to optimization with fixed-size inner and outer windows, the accuracy of the keyframe positions slightly decreases. However, the keyframe poses are still reasonably accurate for tasks such as mapping and path planning.

3) Outdoor Experiment: In this outdoor experiment, we disable the loop closure detection as we want to evaluate the pose drift in the absence of loop closures. Here, the MAV flies one large loop on hilly terrain, and both the start and end points are the same. The flight time is 185 seconds, and the travelled distance is 112.98 m. At the end of the flight, the resulting keyframe graph has 820 keyframes. Figure 6 shows the MAV's path estimated by our vSLAM implementation. In this figure, a purple square and red circle mark the start and end points of the flight. The distance between the start and

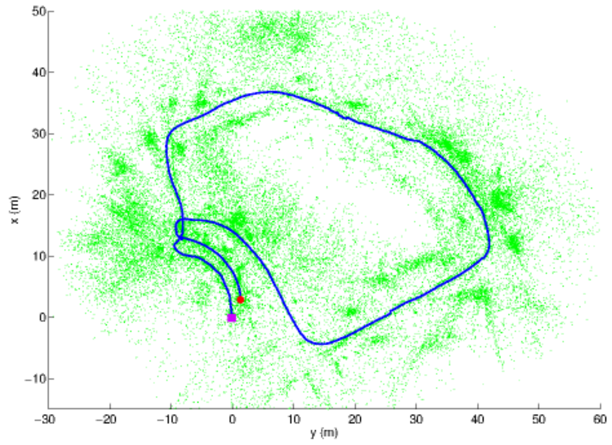


Fig. 6. In an outdoor experiment, we plot the (x, y) positions of the MAV estimated by our vSLAM implementation. A purple square and red circle mark the start and end points which are the same. Green dots represent 3D scene points.

end point is 3.31 m, and thus, the loop closure error is 2.93%.

VII. Conclusions

Our vSLAM-based self-calibration method produces accurate extrinsic calibration parameters with metric scale for a MAV with multiple stereo cameras as long as there is a sufficient number of inter-camera feature correspondences, and a majority of scene points are close to the cameras. Through real-world experiments in indoor and outdoor environments, we demonstrate real-time on-board vSLAM with loop closures on a MAV with multiple cameras. The MAV is able to perform autonomous flight based on the pose estimates from vSLAM. Future work will focus on adding capabilities such as dense mapping and 3D exploration to our MAV platform.

Acknowledgments

The first author is funded by the DSO National Laboratories Postgraduate Scholarship. This work is partially supported by the SNSF V-MAV grant (DACH framework).

References

[1] S. Agarwal, K. Mierle, and Others. Ceres solver, 2013. https://code.google.com/p/ceres-solver/.

[2] M. Agrawal, K. Konolige, and M. Blas. Censure: Center surround extremas for realtime feature detection and matching. In *Computer Vision ECCV 2008*, volume 5305 of *Lecture Notes in Computer Science*, pages 102–115. Springer Berlin Heidelberg, 2008. doi: 10.1007/978-3-540-88693-8_8. URL http://dx.doi.org/10.1007/978-3-540-88693-8_8.

[3] H. Strasdat amd A. Davison, J. Montiel, and K. Konolige. Double window optimisation for constant time visual slam. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. doi: 10.1109/ICCV.2011.6126517. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6126517.

[4] J. Brookshire and S. Teller. Extrinsic calibration from per-sensor egomotion. In *Robotics: Science and Systems (RSS)*, 2012. URL http://www.roboticsproceedings.org/rss08/p04.html.

[5] G. Carrera, A. Angeli, and A. Davidson. Lightweight slam and navigation with a multi-camera rig. In *European Conference on Mobile Robots (ECMR)*, 2011. URL http://www.doc.ic.ac.uk/~gcarrera/ecmr2011.pdf.

[6] G. Carrera, A. Angeli, and A. Davison. Slam-based automatic extrinsic calibration of a multi-camera rig. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. doi: 10.1109/ICRA.2011.5980294. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5980294.

[7] K. Daniilidis. Hand-eye calibration using dual quaternions. *International Journal of Robotics Research (IJRR)*, 18(3):286–298, 1999. doi: 10.1177/02783649922066213. URL http://ijr.sagepub.com/content/18/3/286.refs.

[8] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013. doi: 10.1109/IROS.2013.6696514. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696514.

[9] P. Furgale, U. Schwesinger, M. Rufli, W. Derendarz, H. Grimmett, P. Muhlfellner, S. Wonneberger, J. Timpner, S. Rottmann, Bo Li, B. Schmidt, T.N. Nguyen, E. Cardarelli, S. Cattani, S. Bruning, S. Horstmann, M. Stellmacher, H. Mielenz, K. Koser, M. Beermann, C. Hane, L. Heng, G. H. Lee, F. Fraundorfer, R. Iser, R. Triebel, I. Posner, P. Newman, L. Wolf, M. Pollefeys, S. Brosig, J. Effertz, C. Pradalier, and R. Siegwart. Toward automated driving in cities using close-to-market sensors: An overview of the v-charge project. In *IEEE Intelligent Vehicles Symposium (IV)*, 2013. doi: 10.1109/IVS.2013.6629566. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6629566.

[10] D. Galvez-Lopez and J. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics (T-RO)*, 28(5):1188–1197, 2012. doi: 10.1109/TRO.2012.2197158. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6202705.

[11] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Autonomous obstacle avoidance and maneuvering on a vision-guided mav using on-board processing. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. doi: 10.1109/ICRA.2011.5980095. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5980095.

[12] L. Heng, B. Li, and M. Pollefeys. 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)*, 2013. doi: 10.1109/IROS.2013.6696592. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696592.

[13] M. Kaess and F. Dellaert. Visual slam with a multi-camera rig. Technical report, 2006. URL http://www.cc.gatech.edu/~kaess/pub/Kaess06tr.pdf.

[14] J. Kelly and G. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *International Journal of Robotics Research (IJRR)*, 30(1):56–79, 2011. doi: 10.1177/0278364910382802. URL http://ijr.sagepub.com/content/30/1/56.refs.

[15] J. Kim, M. Hwangbo, and T. Kanade. Motion estimation using multiple non-overlapping cameras for small unmanned aerial vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2008. doi: 10.1109/ROBOT.2008.4543678. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4543678.

[16] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007. doi: 10.1109/ISMAR.2007.4538852. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4538852.

[17] L. Kneip, D. Scaramuzza, and R. Siegwart. 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)*, 2011. doi: 10.1109/CVPR.2011.5995464. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5995464.

[18] G.H. Lee, F. Fraundorfer, and M. Pollefeys. Robust pose-graph loop-closures with expectation-maximization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013. doi: 10.1109/IROS.2013.6696406. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696406.

[19] G.H. Lee, F. Fraundorfer, and M. Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. doi: 10.1109/CVPR.2013.354. URL http://ieeexplore.ieee.org/xpls/abs_all.

jsp?arnumber=6619198.

[20] G.H. Lee, F. Fraundorfer, and M. Pollefeys. Minimal solutions for pose estimation of a multi-camera system. In *International Symposium on Robotics Research (ISRR)*, 2013. URL http://www.inf.ethz.ch/personal/glee/papers/MultiCamPose_ISRR2013.pdf.

[21] H. Li, R. Hartley, and J. Kim. A linear approach to motion estimation using generalized camera models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. doi: 10.1109/CVPR.2008.4587545. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4587545.

[22] C. Mei and P. Rives. Single view point omnidirectional camera calibration from planar grids. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2007. doi: 10.1109/ROBOT.2007.364084. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4209702.

[23] R. Pless. Using many cameras as one. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. doi: 10.1109/CVPR.2003.1211520. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1211520.

[24] S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research (IJRR)*, 28(11-12):1448–1465, 2009. doi: 10.1177/0278364909341659. URL http://ijr.sagepub.com/content/28/11-12/1448.refs.

[25] L. Quan and Z. D. Lan. Linear n-point camera pose determination. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 21, pages 774–780, 1999. doi: 10.1109/34.784291. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=784291.

[26] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. doi: 10.1109/ICCV.2011.6126544. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6126544.

[27] K. Schauwecker and A. Zell. On-board dual-stereo-vision for the navigation of an autonomous mav. *Journal of Intelligent and Robotic Systems*, 74(1-2):1–16, 2014. doi: 10.1007/s10846-013-9907-6. URL http://dx.doi.org/10.1007/s10846-013-9907-6.

[28] K. Schmid, T. Tomic, F. Ruess, H. Hirschmuller, and M. Suppa. Stereo vision based indoor/outdoor navigation for flying robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013. doi: 10.1109/IROS.2013.6696922. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696922.

[29] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor. In *Robotics: Science and Systems (RSS)*, 2013. URL http://www.roboticsproceedings.org/rss09/p32.html.

[30] H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström. Solutions to minimal generalized relative pose problems. In *OMNIVIS*, 2005. URL http://www.vis.uky.edu/~stewe/publications/stewenius_05_omnivis_sm26gen.pdf.

[31] M. Tribou. *Relative Pose Estimation Using Non-overlapping Multicamera Clusters*. PhD thesis, University of Waterloo, 2014. URL https://uwspace.uwaterloo.ca/handle/10012/8141.

[32] Stephan Weiss, Markus W. Achtelik, Simon Lynen, Michael C. Achtelik, Laurent Kneip, Margarita Chli, and Roland Siegwart. Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics (JFR)*, 30(5):803–831, 2013. ISSN 1556-4967. doi: 10.1002/rob.21466. URL http://dx.doi.org/10.1002/rob.21466.