Look Both Ways: Bidirectional Sensing for Automatic Multi-Camera Registration

Subodh Mishra[+], Sushruth Nagesh[+], Sagar Manglani[**], Shubham Shrivastava, Graham Mills[**], Punarjay Chakravarty[**] and Gaurav Pandey
Ford Green-Field Labs, Palo Alto, California, USA

***Abstract*— This work describes the automatic registration of a large network (≈ 40) of fixed, ceiling-mounted environment cameras spread over a large area ($\approx 800\ m^2$) using a mobile calibration robot equipped with a single upward-facing fisheye camera and a single backlit ArUco marker for easy detection. The fisheye camera is used to do visual odometry (VO), and the ArUco marker facilitates easy detection of the calibration robot in the environment cameras. In addition, the fisheye camera is also able to detect the environment cameras. This two-way, bidirectional detection constrains the pose of the environment cameras to solve an optimization problem. Such an approach can be used to register a large-scale multi-camera system used for surveillance, automated parking, or robotic applications. This VO based multi-camera automatic registration method has been extensively validated using real-world experiments, and also compared against a similar approach which uses a LiDAR - an expensive, heavier and power hungry sensor.**

***Index Terms*— Multi-camera registration, infrastructure-enabled autonomy, visual odometry, non-linear least-squares optimization.**

I. INTRODUCTION

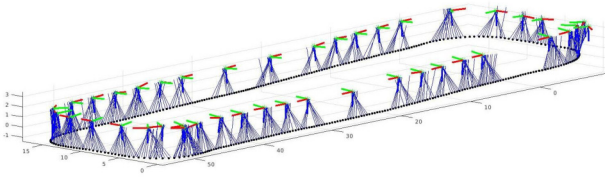


Fig. 1: Plot of estimated poses of multiple cameras (shown as axes) and calibration robot trajectory (dark dots). The blue straight lines are the calibration robot's fiducial marker detections from the environment cameras. The units are in meter.

A distributed sensing system [1] in a parking lot can autonomously park vehicles with drive-by-wire technology and guide their motion in the environment, eliminating the dependence on onboard sensors for perception and localization, especially in controlled environments like parking lots, warehouses, and factory floors. However, for such a network of fixed environment cameras to be useful to an autonomous robot for the aforementioned applications, they need to be registered w.r.t. each other such that the 6 DoF pose of each environment camera is known w.r.t. other environment cameras. A tedious way to determine the camera poses would involve detecting a calibration target (ArUco/Checkerboard) [2] in the common Field of View (FoV) between every pair of environment cameras. This approach requires overlapping field of view (FoV) that cannot always be guaranteed, and tiresome manual labor when a large number of cameras need to registered. In the proposed work, we present an approach to register a large network (≈ 40) of static ceiling-mounted RGB cameras (Figures 2 & 3) with minimal or no overlap between them, spread over a large area ($\approx 800\ m^2$) using a calibration robot (Figure 4) equipped with a single upward looking fisheye camera - for visual odometry (VO) [3] & detection of environment cameras (Figure 3), and a single back-lit ArUco [2] marker - for easy detection of calibration robot in environment cameras (Figure 5). We use VO and ArUco marker detection to bridge the pose between environment cameras. We correct drift by loop closure on visiting previously visited locations (unlike [4]) and determine metric scale from known dimension of ArUco marker (unlike [5]) in the absence of depth sensing (unlike [6], [7]). We look/sense in both directions: downwards from each of the environment camera down onto the robot (as it passes through its FoV, Figure 5) and, upwards from the robot to the environment camera (Figure 3) to determine the pose of each environment camera relative to a common coordinate frame. Apart from driving the calibration robot, all other steps (Section III) are completely automated, requiring no manual feature annotation, detection or matching, for example.

Fig. 2: **Edge nodes:** We register a distributed sensing system of $\sim$ 40 edge devices mounted throughout the ceiling of our test environment. Each edge node comprises of an RGB camera (we call it as environment camera in the paper) and compute. These are powered over ethernet and networked and time-synced to a central server.

[+] S. Mishra (subodh514@tamu.edu) and S. Nagesh (snagesh1@ford.com) have equally contributed to the work.
[**]This work was performed by P. Chakravarty and G. Mills when they were employed at Ford.

II. SYSTEM DESCRIPTION

A. The Distributed Camera System

The distributed camera system (Figures 2 & 3) comprises several static environment cameras which have minimal to no overlapping FoV. They generate images of size 1080 x 1920 pixels at 30 Hz. We use the factory provided intrinsic calibration parameters for these cameras.

B. The Calibration Robot

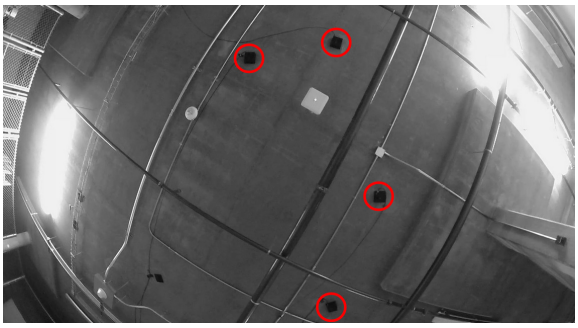


Fig. 3: **Looking up ↑ Edge-Node detection from Robot:** Automated detection of ceiling-mounted environment cameras when viewed from the upward-facing fisheye camera on the calibration robot.

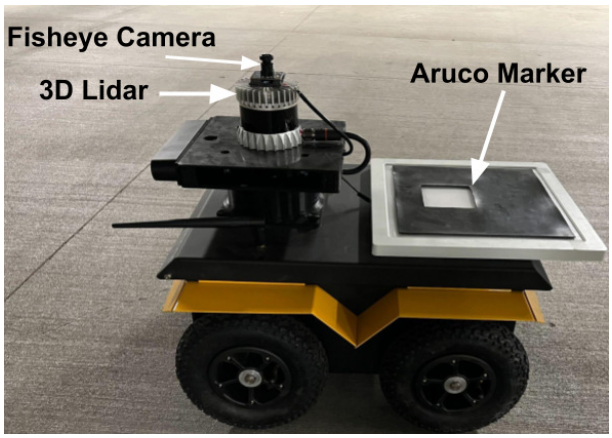


Fig. 4: **Calibration Robot:** Clearpath Jackal with an upward-facing wide-angle fisheye camera F and an ArUco marker M. The ArUco marker is backlit to make detection easier. The robot also carries a LiDAR for comparison between LiDAR and camera-only calibration.

We use a Clearpath Jackal as our calibration robot (Figure 4). It has a rigidly attached upward-facing wide-angle (160°) 2MP fisheye camera which generates images of size 1080 x 1920 pixels at 30 Hz. The fisheye camera faces upwards and tracks static features on the ceiling to perform VO and it also detects environent cameras on the ceilng (using OpenCV [8] blob detection, Figure 3), which we use to further constrain the position of the detected cameras in the estimation procedure. A backlit ArUco marker [2] of known dimensions is attached to the calibration robot so that the robot can be easily detected by the environment cameras (Figure 5). The robot has a 64 Channel Ouster 3D-LiDAR which generates 3D scans at 10 Hz used for doing LiDAR Odometry (LO) [9] based calibration of environment cameras - a method we will compare/bench-mark our approach against.

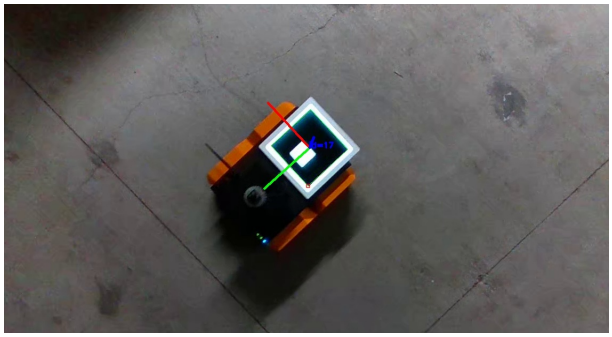


Fig. 5: **Looking down ↓ Robot Detection from Environment Camera:** The Calibration Robot (Figure 4) viewed from an environment camera with axes drawn on the detected backlit ArUco marker.

III. METHOD

Described below is our 3 step approach to solve this problem.

A. Estimate Motion of the upward-looking fisheye camera using Visual Odometry (VO)

We use SVO [10] [11] 's front-end to determine the frame to frame motion of the upward-looking fisheye camera on the robot. We eliminate drift in VO by implementing loop closure (using DBoW2 [12] for place recognition) and performing bundle adjustment [13] (using ceres solver [14]). A comparison of trajectories before and after bundle adjustment is shown in Figure 6 (compare red and green).

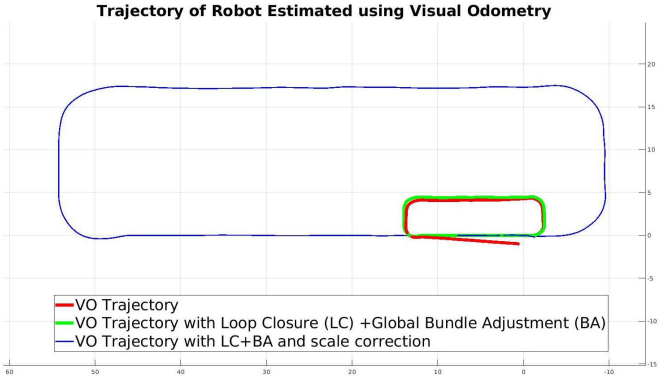


Fig. 6: **Motion estimation using Visual Odometry (VO).** The trajectory in red is the result from SVO [10]'s front-end. We use loop closure [12] and bundle adjustment [14] to reduce the drift in VO (shown in green) and finally we determine the metric scale (in meters) (Section III-B) of the trajectory using the detection of ArUco marker on the calibration robot. The axes are in meter.

B. Determine the scale s of VO & the spatial offset T_M^F between the ArUco marker M and the upward-looking fisheye camera F using ArUco detections

ArUco detection pose measurements $T_M^{C_k} \in SE(3)$ are used to determine the metric scale s of the trajectory estimated by monocular VO and the spatial separation T_M^F between the fisheye camera and the ArUco marker. The motion-based calibration technique [15] is employed for this purpose, using

which we align the motion estimated by VO with the motion of the ArUco marker estimated by its detection in an environment camera C_k. We use pose interpolation to determine the robot pose ${}_{int}T^W_{F_i}$ & ${}_{int}T^W_{F_{i+1}}$ corresponding to timestamps of ArUco measurements $T^{C_k}_{M_i}$ & $T^{C_k}_{M_{i+1}}$ respectively to align ${}_{int}T^{F_i}_{F_{i+1}}$ $(= ({}_{int}T^W_{F_i})^{-1}\,{}_{int}T^W_{F_{i+1}})$ with $T^{M_i}_{M_{i+1}}$ $(= (T^{C_k}_{M_i})^{-1}T^{C_k}_{M_{i+1}})$, and determine the metric scale s of robot trajectory and also the spatial offset T^F_M between the fisheye camera F and the marker M. The residual r_i is given by:

$$r_i = {}_{int}T^{F_i}_{F_{i+1}}\,T^F_M \ominus T^F_M T^{M_i}_{M_{i+1}} \tag{1}$$

${}_{int}T^{F_i}_{F_{i+1}} = \begin{bmatrix} R({}_{int}q^{F_i}_{F_{i+1}}) & s.{}_{int}p^{F_i}_{F_{i+1}} \\ 0 & 1 \end{bmatrix}$. Ceres solver [14] is used to determine s and T^F_M by minimizing a cost function shown in Equation 2.

$$\hat{s}, \hat{T}^F_M = \underset{s\in R, T^F_M \in SE(3)}{\operatorname{argmin}} \sum_{i=0}^{L-1} \rho(\|r_i\|^2) \tag{2}$$

Here L is the number of corresponding motion segments and $\rho()$ is Huber Loss Function. The estimated scale s and spatial offset T^F_M are used to re-scale VO pose estimates (Figure 6) and to estimate the environment camera poses (Section III-C).

C. Estimate the poses of the environment cameras

1) Looking Down: For a single camera C_k of the distributed system, we gather all the ArUco pose measurements $\{T^{C_k}_{M_i}\}_{i=0:N_k-1}$ which measure the poses of the ArUco marker in the camera C_k frame as the calibration robot drives under it, and use their respective timestamps to determine the corresponding robot/fisheye camera pose $\{{}_{int}T^W_{F_i}\}_{i=0:N_k-1}$ using interpolation. For ArUco measurements $\{T^{C_k}_{M_i}\}_{i=0:N_k-1}$ from camera C_k, and the corresponding interpolated robot poses $\{{}_{int}T^W_{F_i}\}_{i=0:N_k-1}$, the pose of environment camera C_k can be determined from Equation 3 by solving a non-linear least-squares optimization problem using ceres library [14].

$$\hat{T}^W_{C_k} = \underset{T^W_{C_k}\in SE(3)}{\operatorname{argmin}} \sum_{i=0}^{N_k-1} \rho\left(\left\| T^W_{C_k} T^{C_k}_{M_i} \ominus {}_{int}T^W_{F_i} \hat{T}^F_M \right\|^2\right) \tag{3}$$

Here N_k is the number of ArUco detection pose measurements made by camera C_k and $\rho()$ is Huber Loss Function. We perform this optimization (Equation 3) for all environment cameras $\{C_k\}_{k=1:N}$ to estimate their respective 6-DoF pose using respective ArUco detection pose measurements made by looking down on the calibration robot driving below.

2) Looking Up - Position Refinement: In this step our goal is to refine the positions $\{p^W_{C_k}\}_{k=0:N-1} \in R^3$ of the estimated camera poses $\{T^W_{C_k}\}_{k=0:N-1} \in SE(3)$ by minimizing the re-projection error between the projection $\pi(K, T^W_{F_j}, p^W_{C_k})$ of the estimated environment camera position $p^W_{C_k}$ and the corresponding pixel detection $\begin{bmatrix} u \\ v \end{bmatrix}_{kj}$ (obtained using OpenCV's blob detection algorithm) on the upward facing fisheye camera image. We associate $\pi(K, T^W_{F_j}, p^W_{C_k})$ to $\begin{bmatrix} u \\ v \end{bmatrix}_{kj}$ by performing a nearest neighbour search. The residual for minimizing the reprojection error is given in Equation 4.

$$r_{kj} = \begin{bmatrix} u \\ v \end{bmatrix}_{kj} - \pi(K, T^W_{F_j}, p^W_{C_k}) \tag{4}$$

r_{kj} can be defined as the reprojection error of the k^{th} environment camera when viewed from the j^{th} robot key-frame pose. Here K & π are the intrinsic calibration parameters of the fisheye camera and the fisheye projection model respectively. We solve this minimization problem (Equation 5) using ceres library [14].

$$\{\hat{p}^W_{C_k}\}_{k=0:N-1} = \underset{\{p^W_{C_k}\}_{k=0:N-1}\in R^{3\times 1}}{\operatorname{argmin}} \sum_{j=0}^{M-1}\sum_{k=0}^{N-1} w_{kj}\rho(\left\|r_{kj}\right\|^2) \tag{5}$$

Here $w_{kj} = 1$ if environment camera C_k is visible in the j^{th} robot key-frame otherwise $w_{kj} = 0$ and $\rho()$ is Cauchy Loss Function. Figure 1 shows a plot of estimated environment cameras, ArUco detection pose measurements and the calibration robot trajectory after the registration procedure. Equation 5 is similar to solving a Bundle Adjustment problem.

IV. EXPERIMENTS & RESULTS

To evaluate our method, we collect two datasets by driving the calibration robot under the environment cameras of the distributed system. We register 43 and 38 environment cameras in datasets 1 & 2 respectively. In the absence of ground truth, we collect 3D scans from the LiDAR on the calibration robot for performing qualitative comparison of the proposed VO based solution against a state of the art LiDAR Odometry LO [9] based approach. One of the immediate advantages of the VO based method over the LO based approach is that one can perform qualitative verification of environment camera registration by projecting environment cameras' positions on the fisheye camera image.

A. Verification using Re-projection Error

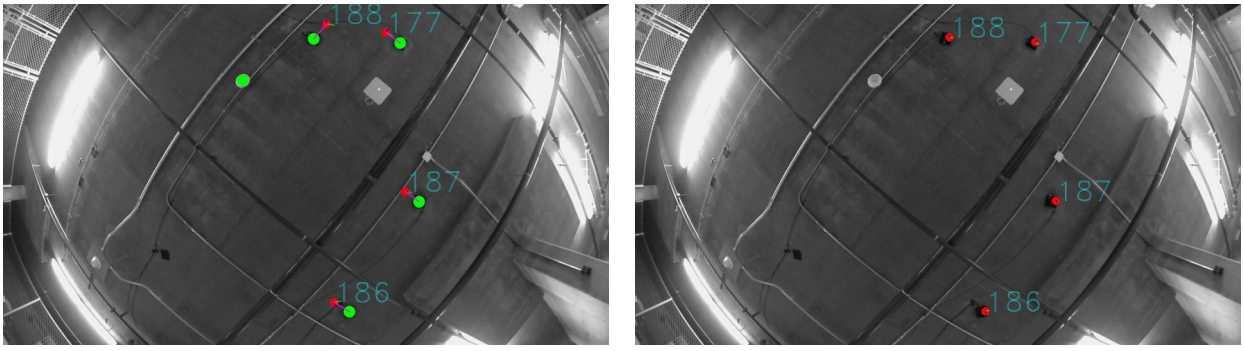
We present the qualitative and quantitative reduction in environment camera re-projection errors due to refinement (Section III-C.2) of estimated environment camera positions in Figures 7a & 7b and Table I, respectively.

Reprojection Error (Pixel)		
	Before Refinement	Post Refinement
Dataset 1	51.642	10.428
Dataset 2	54.245	5.675

TABLE I: Root Mean Squared Re-projection Error calculated by measuring the reprojection error of the estimated distributed camera positions on each fisheye camera keyframe. The reprojection error post refinement is smaller.

B. Comparison with Lidar Odometry (LO) based method

In the absence of ground truth, we compare our method against the state of the art Direct LiDAR Odometry (DLO [9]), and then find environmental camera poses. The LO based approach implements only Section III-C.1 for the



(a) **Re-projection before camera position refinement.** The detected environment cameras (green) do not align with projection of environment cameras (red).

(b) **Re-projection after camera position refinement.** The projection of estimated environment cameras coincide with corresponding environment cameras on the fisheye image.

Fig. 7: Re-projection of distributed camera positions before and after re-projection error based refinement of camera positions.

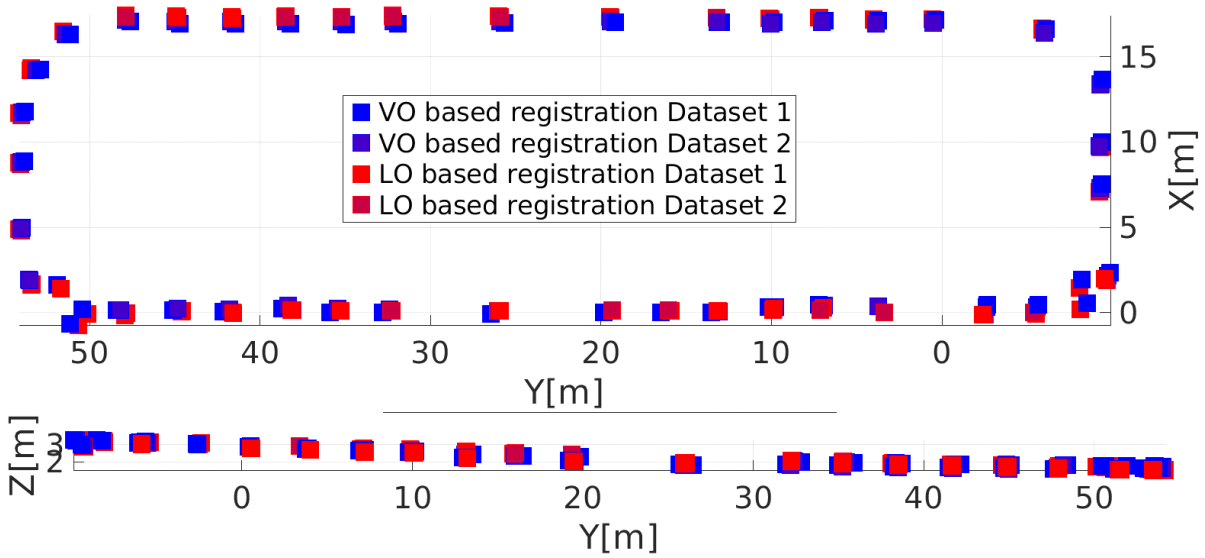


Fig. 8: Comparison of estimated env. camera poses between the VO and LO methods. Top and side view of env. camera positions estimated by VO and LO based approaches for dataset 1 & 2.

RMSD between corresponding environment cameras						
	ϕ°	θ°	ψ°	X[m]	Y[m]	Z[m]
$LO_1 \leftrightarrow VO_1$	0.9582	0.9220	4.0727	0.2819	0.3579	0.0947
$LO_1 \leftrightarrow VO_2$	0.3604	1.3453	5.0665	0.2281	0.1314	0.1104
$LO_2 \leftrightarrow VO_1$	0.9768	1.3204	1.6038	0.2662	0.3376	0.1029
$LO_2 \leftrightarrow VO_2$	0.3777	0.7215	1.6997	0.2099	0.1433	0.1182

TABLE II: RMSD for estimated env. camera poses between VO and LO based approaches for both the datasets. Here LO_i : environment cameras estimated from LO based approach in dataset i & VO_j : Corresponding environment cameras estimated from VO based approach in dataset j.

estimation process as two way detection is not possible when using a LiDAR. As LiDAR measures 3D points in metric units, it is not necessary to determine the metric scale of LO, but the spatial offset between the LiDAR and the ArUco marker is determined using motion based calibration [15] method discussed in Section III-B. Our goal is to discover if the VO based solution presented here gives comparable results. In order to compare the absolute environment camera poses we represent estimated variables in a common frame of reference (using a method similar to [16]). Visualization of the environment camera positions for both the approaches and both datasets is presented in Figure 8. The RMS difference (RMSD) between corresponding environment cameras poses estimated using both the VO and LO based approaches for both the datasets are presented in Table II, where we observe that the results from VO and LO are comparable, with the maximum rotation difference being 5.07° and maximum translation difference being 35.8 cm.

V. Discussion and Application

The proposed VO based registration gives comparable results as LO based registration. Also, a camera is smaller, lighter, cheaper, and consumes less power, making its use possible with cheaper robotic platforms that can be used instead of the one used in this work. We use the registered environment camera system to track objects and autonomously drive vehicles in the environment where this system is installed. A video for the same can be found here[1].

[1] https://youtu.be/8ddsdvJ8q38?si=sp_c7h1P9fUKVobg

REFERENCES

[1] S. Gopalswamy and S. Rathinam, "Infrastructure enabled autonomy: A distributed intelligence architecture for autonomous vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 986–992, 2018.

[2] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognit.*, vol. 47, pp. 2280–2292, 2014.

[3] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[4] H. Huang, N. Li, H. Guo, Y.-L. Chen, and X. Wu, "Calibration of non-overlapping cameras based on a mobile robot," in *2015 5th International Conference on Information Science and Technology (ICIST)*, pp. 328–333, 2015.

[5] T. Pollok and E. Monari, "A visual slam-based approach for calibration of distributed camera networks," pp. 429–437, 08 2016.

[6] E. Ataer-Cansizoglu, Y. Taguchi, S. Ramalingam, and Y. Miki, "Calibration of non-overlapping cameras using an external slam system," in *2014 2nd International Conference on 3D Vision*, vol. 1, pp. 509–516, 2014.

[7] K. Koide and E. Menegatti, "Non-overlapping rgb-d camera network calibration with monocular visual odometry," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9005–9011, 2020.

[8] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[9] K. Chen, B. T. Lopez, A.-a. Agha-mohammadi, and A. Mehta, "Direct lidar odometry: Fast localization with dense point clouds," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2000–2007, 2022.

[10] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 15–22, 2014.

[11] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2016.

[12] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, pp. 1188–1197, October 2012.

[13] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski, "Bundle adjustment in the large," in *Computer Vision – ECCV 2010* (K. Daniilidis, P. Maragos, and N. Paragios, eds.), (Berlin, Heidelberg), pp. 29–42, Springer Berlin Heidelberg, 2010.

[14] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 3 2022.

[15] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor arrays," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4843–4850, 2015.

[16] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7244–7251, 2018.