Monocular Rotational Odometry with Incremental Rotation Averaging and Loop Closure

Chee-Kheng Chng, Alvaro Parra, Tat-Jun Chin, Yasir Latif
School of Computer Science, University of Adelaide

***Abstract*—Estimating absolute camera orientations is essential for attitude estimation tasks. An established approach is to first carry out visual odometry (VO) or visual SLAM (V-SLAM), and retrieve the camera orientations (3 DOF) from the camera poses (6 DOF) estimated by VO or V-SLAM. One drawback of this approach, besides the redundancy in estimating full 6 DOF camera poses, is the dependency on estimating a map (3D scene points) jointly with the 6 DOF poses due to the basic constraint on structure-and-motion. To simplify the task of absolute orientation estimation, we formulate the monocular rotational odometry problem and devise a fast algorithm to accurately estimate camera orientations with 2D-2D feature matches alone. Underpinning our system is a new incremental rotation averaging method for fast and constant time iterative updating. Furthermore, our system maintains a view-graph that 1) allows solving loop closure to remove camera orientation drift, and 2) can be used to warm start a V-SLAM system. We conduct extensive quantitative experiments on real-world datasets to demonstrate the accuracy of our incremental camera orientation solver. Finally, we showcase the benefit of our algorithm to V-SLAM: 1) solving the known rotation problem to estimate the trajectory of the camera and the surrounding map, and 2) enabling V-SLAM systems to track pure rotational motions.**

I. Introduction

Visual odometry [1], [2], [3] (VO) and visual SLAM [4], [5] (V-SLAM) estimate the poses of a moving camera from the captured sequence of images, which is relevant to many real-world technologies such as autonomous driving, UAV, and virtual-reality applications. In particular, *monocular* VO/V-SLAM is of high interest due to its low cost, low power consumption, and its easy-to-setup nature.

Given a set of images within a captured sequence, $\{I_j\}$, monocular VO and V-SLAM systems find the pose

$$\mathbf{T}_j = \begin{bmatrix} \mathbf{R}_j & \mathbf{t}_j \\ 0 & 1 \end{bmatrix} \tag{1}$$

where $\mathbf{R}_j \in SO(3)$ is the camera orientation represented by rotation matrix, and $\mathbf{t}_j \in \mathbb{R}^3$ is its 3D position in a common coordinate system. Apart from estimating the camera pose, V-SLAM systems also maintains (and output) a *global map* with the coordinates of the observed scene points $\mathcal{X} = \{X_i \mid X \in \mathbb{R}^3\}$ which can be used to 1) relocalise the camera when it lost track, and 2) perform loop closure.

There exist some applications when the camera orientation is the sole interest, *attitude estimation* [6], [7], and the *known rotation problem* [8], [9] are among the well studied ones. Attitude estimation predicts the orientation of a moving vehicle based on high-end IMUs. Motivated by the lower cost aspects (in terms of both power consumption and price), Khosravian *et al.* [7] introduced an algorithm to incorporate GPU velocity readings to refine VO's orientation estimates. Secondly, in the V-SLAM domain, given the orientation priors, the known rotation problem [8] can be solved in a quasi-convex framework to obtain the camera position and the observed 3D scene points. We argue that VO and V-SLAM are not the best choices for these applications demanding camera orientation computation alone. Instead, we propose a novel monocular rotational odometry system to estimate absolute camera orientations from relative camera orientations under *epipolar geometry* [10] for which 2D-2D feature correspondences is sufficient. Underpinning our system is a proposed incremental rotation averaging method, which is fast and accurate.

Apart from the system-level advantages, such as a smaller memory footprint, our proposed system exhibits two inherent strengths over VO/V-SLAM systems. Firstly, our method has no requirement in creating and maintaining a local map which is essential for VO and V-SLAM systems. More specifically, VO systems solve PnP [11] which takes 2D-3D feature correspondences to jointly estimate the camera orientation and translation; while V-SLAM systems perform local BA[12] to jointly optimise the camera poses and scene points. The requirement of the 3D scene points mainly stems the translation of the camera, which is unnecessary for our objective.

Secondly, maintaining a 3D map is fundamentally unfeasible during periods of pure rotation motion [10]. It is geometrically impossible to triangulate the depths of 3D scene points at the limit of cameras approaching no parallax. During pure rotation motion, only a panoroma map (with no depth information) can be created for the observed features [13], [14]. However, panoramas can not be used together with the 3D map in the joint estimation frameworks such as PnP and BA. As a partial remedy, most VO/V-SLAM systems rely on some form of model selection [15], [14] to invoke different routines depending on the type of motion (standard rigid motion and pure rotation motion). Without the need of maintaining a map, our proposed method leverages a *motion robust* relative pose estimation algorithm [16], which allows us to estimate the relative orientation in a unified framework.

Our proposed method relates to VO systems in the absence of global mapping. Since the incapacity of globally constraining the camera pose, most of VO systems advocate on reducing frame-to-frame drift. For this reason, recent active research in the field focus on improving front-end components such as feature descriptors [17], feature correspondences selection [3], and accurate single-view depth estimation [18]. However, all of the mentioned algorithms involve deep networks which usually come at the expense of processing time, computation power, and training data. On the contrary, our proposed method uses ORB descriptors [19] (the fastest but less robust) and focuses

on refining the estimates of camera orientation with *rotation averaging* [20], [21] formulated in an incremental fashion (detailed in Sec. II-C).

From our empirical evidence over long sequences in the KITTI dataset [22], incremental rotation averaging alone is insufficient to combat drift accumulation. Hence, we integrated a loop closure component in our system, which invokes global rotation averaging routine to distribute the accumulated drift to the node in the view-graph upon detecting loop-closure. The loop-closure module is built on top of the appearance based loop-closure method in ORB-SLAM2 [4]. Due to the lack of global mapping, we replaced the point clouds based geometrical verification process with a 2D-2D feature matches based algorithm (see Sec.III-E). In the aspect of having a loop-closure module, our system is related to V-SLAM systems. Hence, we compare our method to the state-of-the-art monocular V-SLAM system, ORB-SLAM2[4], over two real-world datasets and demonstrate that our proposed method achieves competitive results.

We summarise our contributions as

1) An efficient incremental rotation averaging method[1].
2) A visual rotational odometry system which is robust against any camera motion (Sec. III-B), capable of loop-closing (Sec. III-E), and that achieves state-of-the-art accuracy on real-world datasets (Sec. IV-C).

II. Related Works

A. Monocular Visual Odometry and Visual SLAM systems

Our approach is closer to VO than V-SLAM as we do not require computing and maintaining a global map. The interested reader can find a comprehensive survey for VO in [23]. In essence, conventional VO pipelines first identify feature matches, obtain 2D-3D correspondences after triangulating the scene map, and estimate relative motion between consecutive frames. Finally, they estimate the absolute poses of the camera by chaining the relative motions. An emerging trend in VO systems is the incorporation of deep networks to learn and predict 1) feature correspondences with an optical flow network [3], 2) depths of the observed features with a single image [24], and 3) camera poses [25]. Since conventional VO algorithms do not maintain a global map and do not perform loop closure, they focus on reducing the drifts between each pose estimate with better front-end inputs (i.e., feature correspondences, accurate depth predictions). On the other hand, our proposition uses the fastest but less robust ORB-descriptors for the 2D-2D feature matching process, and we use our proposed incremental and robust rotation averaging algorithm to produce accurate orientation estimates.

In addition to estimating the camera pose, monocular V-SLAM systems also estimate the observed 3D scene points (known as *mapping*). In a broad sense, V-SLAM methods can be classified into three groups: indirect-based, direct-based, and the hybrid group. ORB-SLAM2 [4] is the state-of-the-art system in the indirect-based category. ORB-SLAM2 operates as follows. It first extracts ORB image features and identifies feature matches to previous images. To jointly estimate camera poses and the coordinates of scence points, ORB-SLAM2 leverages in a combination of well-studied geometry algorithms (including relative motion estimation and PnP) with large scale non-linear optimisation frameworks (e.g., pose graph optimisation and bundle adjustment). Engel *et al.* [1] propose a direct based system that utilises all the pixel information instead of selecting feature correspondences and minimises the *photometric loss* instead of the reprojection error as in conventional indirect methods. The system, namely DSO, was later extended with loop-closure in [26]. Our work is closely related V-SLAM works in the loop closure aspect, and specifically close to ORB-SLAM2 as our system extracts and matches ORB-features, and performs loop-closure with DBoW2 featuress [27].

[1]C++ implementation will be available.

B. The View-Graph

Conventional VO and V-SLAM systems maintains a so-called *view-graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with edges $(j, k) \in \mathcal{E}$ connecting images I_j, I_k for which a relative motion can be estimated. Typically, the view-graph relates the image pairs capturing overlapped scenes such that sufficient epipolar constraints exist to obtain the essential matrix $\mathbf{E}_{j,k}$. In our system, the nodes of the view-graph relate to camera orientations, and the edges to the relative orientation between the connected views.

C. Rotation Averaging

In the absence of noise and outliers, the absolute orientation $\mathbf{R}_k$ for an image I_k can be directly estimated by chaining the orientation $\mathbf{R}_j$ of another image I_j to the relative orientation $\mathbf{R}_{j,k}$ between I_j and I_k, for which

$$\mathbf{R}_k = \mathbf{R}_j \mathbf{R}_{j,k} \tag{2}$$

holds. Thus, under those ideal conditions, we could constraint $\mathcal{G}$ to be a "loop" (only connecting temporary adjacent images) and obtain all absolute orientations (up to some arbitrary orientations) by simply chaining them by applying Eq. (2). However, in the VO/V-SLAM application settings where the presence of outliers and noise is inevitable, estimating orientations from $\mathcal{G}$ being a loop will inevitably produce inaccurate results as this setting requires that all relative orientations are precisely computed. Instead, *rotation averaging* solves for the absolute orientations by minimising the discrepancies between relative and absolute orientations across all edges in $\mathcal{G}$

$$\min_{\{\mathbf{R}_j\}} \sum_{(j,k)\in\mathcal{E}} \rho(d(\mathbf{R}_{j,k}, \mathbf{R}_k \mathbf{R}_j^{-1})), \tag{3}$$

where ρ is a loss function (e.g. $\rho(x) = x^2$) and $d : SO(3) \times SO(3) \mapsto \mathbb{R}_+$ is a distance between rotation matrices (e.g., the geodesic distance in SO(3)).

Rotation averaging (3) was first introduced by Govindu [28], and has since been further explored by numerous works [21], [20], [29]. Chatterjee *et al.* [20] proposes an efficient and robust algorithm (L1-IRLS) to solve (3). L1-IRLS is an iterative reweighted least-squares algorithm which finds a local optimum in the Lie group SO(3) with a robust initialisation scheme. Specifically, the initial estimates are obtained by solving an approximated algorithm based on an easy to solve yet robust ℓ_1-based optimisation problem. As default, L1-IRLS takes ρ as the Huber-like

robust loss function and solve of the weights (of the relative rotations) and the absolute rotations alternatively until convergence. Our proposed incremental rotation averaging method (see Sec.III-D) is an extension to L1-IRLS.

Rotation averaging is relevant in the V-SLAM field. Carlone *et al.* [30] propose to first estimate the orientation components to initialise the subsequent pose-graph optimisation. Parra *et al.* [9] propose an L-infinity V-SLAM framework that detaches the camera orientation estimates from the translation component and the observed scene points. Although both of the mentioned works use rotation averaging to estimate the absolute orientation of each camera pose, none of them formulates and solve the problem incrementally.

D. Relative Orientation Estimation

For every incoming frame, we first identify the 2D-2D feature matches with the previously processed frames within a fixed window (more details in Sec. III-B), and estimate their relative orientations. If there are sufficient feature matches, we add a new node to the view-graph and the edges related to the estimated relative orientations.

The *five-points algorithm* [31] is the *"go-to"* method in estimating the relative motion between a pair of frames. However, the method breaks down when the baseline between the input frames vanishes, resulting in undefined essential matrix [10]. One real-world scenario of the mentioned degeneracy is the pure rotation motion (or rotation-only motion) where the camera is rotating around a principle axis and not moving in the 3D euclidean space. Hence, VO/V-SLAM systems usually incorporate model selection techniques (stemming from [15], [14]) to select either essential matrix or homography matrix (which is defined in pure rotation motion) is appropriate to model the current motion. Instead of relying on model selection, we adopted the relative orientation estimation algorithm in [16] which is robust against any motion. Underpinning Kneip *et al.*'s method is the *normal epipolar constraint* (first introduced in [32]), a novel epipolar geometry constraint capable of decoupling the relative orientation and the translation direction between a pair of frames. As such, the constraint is well defined under both standard motion and rotation-only motion.

III. Methods

Alg. 1 describes our system which advocates to keep updated a view-graph as the camera captures new frames. Contrasted to V-SLAM, our view-graph relates absolute (nodes) and relative (edges) orientations alone (i.e., without the translation components as in V-SLAM systems). Fig. 1 depicts the final view-graph for the Carpark dataset (see Sec. IV-A) where each node is drawn with their 3D position we obtained from the ground-truths.

The core optimisation component in our method is the incremental rotation averaging routine (Alg.1, Line 13) which solves a new rotation averaging formulation (Sec. III-D) to anchor the solution within a window of orientations $\mathcal{R}_{\text{window}}$ to the previously estimated orientations out of the window.

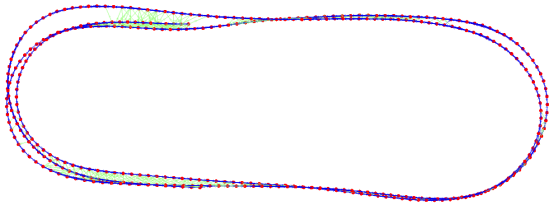


Fig. 1: View-graph of our system for the Carpark dataset. Nodes (in red) are at the camera locations. Edges represent covisibility. Loop-closure edges are highlighted in green.

Algorithm 1 Visual Rotational Odometry

1: **global variables:** View-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
2: **for** each new frame $k = 1, 2, \ldots$ **do**
3: $f \leftarrow k - |\mathcal{F}_{\text{window}}|$.
4: $\mathcal{Y}_k \leftarrow$ feature_extraction(I_k).
5: **for** j = f,...,k-1 **do**
6: $\mathcal{C}_{j,k} \leftarrow$ feature_matching$(\mathcal{Y}_j, \mathcal{Y}_k)$.
7: $(\mathbf{R}_{j,k}, \mathcal{C}'_{j,k}) \leftarrow$ relative_rotation$(\mathcal{C}_{j,k})$.
8: **if** $|\mathcal{C}'_{j,k}| > \theta_{\text{matches}}$ **then**
9: $\mathcal{G} \leftarrow$ upgrade_view-graph$(\mathbf{R}_{j,k})$.
10: **end if**
11: **end for**
12: Obtain $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ for $\mathcal{R}_{\text{window}}$.
13: $\{\mathbf{R}_j\}_{j \in \mathcal{E}'} \leftarrow$ inc_rotation_averaging$(\mathcal{G}')$.
14: **if** loop closure detected **then**
15: $\{\mathbf{R}_j\}_{j \in \mathcal{E}} \leftarrow$ glob_rotation_averaging$(\mathcal{G})$.
16: **end if**
17: **end for**

A. Feature Extraction and Matching

We use ORB descriptors [19] as image features due to its superior speed performance. We adopted identical extraction technique as ORB-SLAM2 [4]. Upon extracting a set of ORB features $\mathcal{Y}_k$ in the current frame I_k (Alg. 1, Line 4), our system performs a local search (Alg. 1, Line 6) to find matches $\mathcal{C}_{j,k} = \{(\mathbf{x}_i, \mathbf{x}'_i) \mid \mathbf{x}_i \in I_j \text{ and } \mathbf{x}'_i \in I_k\}$ in the previous frames $\{I_j\}_{j=f}^{k-1}$, where index f is the first frame within a fixed window $\mathcal{F}_{\text{window}}$. The local search algorithm assumes a high frame rate[2]. Hence, the detected features remain close (in pixel coordinates) to their corresponding features in the neighbouring frames.

B. Motion Robust Relative Orientation Estimation

The conventional estimation of the relative orientation, $\mathbf{R}_{j,k}$, (assuming known camera calibration) from decomposing the essential matrix

$$E_{j,k} = [\mathbf{t}_{j,k}]_\times \mathbf{R}_{j,k}, \tag{4}$$

(where $[\mathbf{t}_{j,k}]_\times$ is the skew-symmetric matrix representation of the cross product operator) is undefined for pure rotation motion ($\mathbf{t}_{j,k} = \mathbf{0}$). Instead, we incorporate in our system (Alg. 1, Line 7) the Kneip's method [16] which estimates the relative orientation independently of the translation. Thus, Kneip's method can correctly estimate $\mathbf{R}_{j,k}$ even if $\mathbf{t}_{j,k} = \mathbf{0}$.

[2]which is a standard-setting in video sequences.

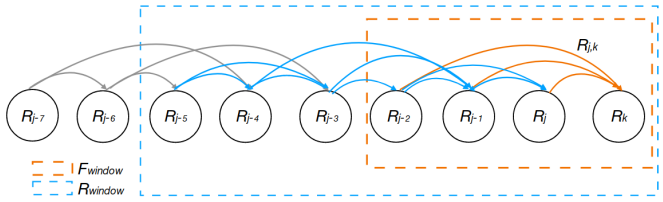


Fig. 2: The local view-graph $\mathcal{G}'$. The nodes out of the $\mathcal{R}_{\text{window}}$ remain constant within our rotation averaging formulation.

For an image point $\mathbf{x}$, we define its *bearing vector*

$$\mathbf{f} := \frac{\hat{\mathbf{x}}}{\| \hat{\mathbf{x}} \|}, \tag{5}$$

where $\hat{\mathbf{x}}$ is $\mathbf{x}$ in normalised homogeneous coordinates (assuming known camera intrinsics).

Kneip's method solves an iterative eigenvalue rank minimisation problem over a set of normal vectors

$$\mathbf{n}_i := \mathbf{f}'_i \times \mathbf{R}_{j,k}\mathbf{f}_i \tag{6}$$

of the *epipolar plane*, where $(\mathbf{f}_i, \mathbf{f}'_i)$ are corresponding bearing vectors such that $(\mathbf{x}_i, \mathbf{x}'_i) \in \mathcal{C}_{j,k}$. The essential geometry constraint (which is independent of the motion) is that all normal vectors must lie on the same plane.

The eigenvalue rank minimisation problem finds the relative orientation that minimises the smallest eigenvalue $\lambda_{\mathbf{M},\text{min}}$

$$\min_{\mathbf{R}_{j,k}} \lambda_{\mathbf{M},\text{min}} \tag{7}$$

of $\mathbf{M} = \mathbf{N}\mathbf{N}^T$, the covariance matrix of the normal vectors $\mathbf{N} = [\mathbf{n}_1, \ldots, \mathbf{n}_n]$,

To solve (7), we initialise $\mathbf{R}_{j,k}$ as $\mathbf{R}_{j-1,k-1}$ assuming the rotation motion between neighbouring pair of frames should be close to each other[3].

Since $\mathbf{M}$ is a real symmetric and positive definite matrix with rank at most 2 (based on the coplanarity constraint on the normal vectors), it is, therefore, equivalent to an iterative rank minimisation. The optimisation framework is solved in a Levenberg-Marquardt scheme as detailed in [16]. Similar to the *five-point method* [31], Kneip's method is embedded in a RANSAC framework to remove outlying feature correspondences ($\mathcal{C}'_{j,k} \subseteq \mathcal{C}_{j,k}$ are the correspondences after RANSAC in Alg. 1, Line 7).

We add a new connection (j, k) to the view-graph if there are sufficient matches θ_{matches} ($= 100$ in our implementation) after RANSAC (Alg. 1, Line 9).

C. Local View-Graph

We estimate absolute camera orientations over a local sub-graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ of the view-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to incrementally solve for their camera orientations at each frame step (Alg. 1, Line 12).

$\mathcal{V}'$ contains the neighbouring nodes (with orientations in $\mathcal{R}_{\text{window}}$) to the current frame I_k and the adjacent vertices in $\mathcal{V}$ to them (i.e., nodes in $\mathcal{V}$ connected to the neighbouring nodes by an edge in $\mathcal{E}$). Fig. 2 depicts $\mathcal{G}'$.

[3]Similar to ORB-SLAM's constant velocity assumption which initialises the latest frame's camera pose with previously optimised poses

D. Incremental Rotation Averaging

Incrementally updating camera orientations in $\mathcal{G}$ by solving "conventional" rotation averaging (3) over a sub-graph of $\mathcal{G}$ ("local" to the current view) will inevitably produce imprecise results even after correcting drift. This is because the solution in the sub-graph will be anchored to the full graph at a single node (the first node in the sub-graph) with has an equivalent effect in accuracy to chaining orientations as discussed in Sec. II-C. In other words, the accuracy of the estimations in the local graph will be strongly dependant of the previous estimate of the first orientation in the sub-graph. To mitigate this chaining effect on accuracy, we propose an alternative formulation which anchors the solution in the sub-graph not to one but many nodes in the view-graph. To this end, for a window $\mathcal{R}_{\text{window}}$ of the most recent absolute orientation in $\mathcal{G}$, we build its local view-graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ as described in Sec. III-C. Thus, we anchor the solution of

$$\min_{\mathcal{R}_{\text{window}}} \sum_{(j,k)\in\mathcal{E}'} \rho(d(\mathbf{R}_{j,k}, \mathbf{R}_k\mathbf{R}_j^T)), \tag{8}$$

to all previously computed absolute orientations adjacent to the orientations in the window, i.e., the rotation matrices in

$$\mathcal{R}_{\text{c}} = \mathcal{V}' \setminus \mathcal{R}_{\text{window}}. \tag{9}$$

To solve (8), we present an extension of L1-IRLS [20]. L1-IRLS optimises over the $SO(3)$ Lie group by iteratively solving a weighted least-squares problem at the tangent Euclidean space which minimises

$$\left(\sqrt{\Phi}\mathbf{A}\Delta\Omega_{\mathcal{V}} + \sqrt{\Phi}\Delta\Omega_{\mathcal{E}}\right)^2, \tag{10}$$

where Φ is a matrix collecting the weights for the edges in $\mathcal{E}$, A is a matrix encoding $\mathcal{G}$, $\Delta\Omega_{\mathcal{V}}$ is the vector variable associated to the absolute orientations, and $\Delta\Omega_{\mathcal{E}}$ the vector associated to the relative orientations (refer to [33] for details). Instead, we minimises

$$\left(\sqrt{\Phi'}\mathbf{A}'\Delta\Omega_{\text{window}} + \sqrt{\Phi'}\Delta\Omega_{\mathcal{E}'}\right)^2, \tag{11}$$

where now the vector variable $\Delta\Omega_{\text{window}}$ comprises only the absolute orientations in $\mathcal{R}_{\text{window}}$ and $\Delta\Omega_{\mathcal{E}'}$ only relative orientations in $\mathcal{E}'$. Φ' and $\mathbf{A}'$ are the versions for Φ and $\mathbf{A}$ without the unrelated columns to $\Delta\Omega_{\text{window}}$.

E. Loop-Closure

Loop-closure is an integral component in modern V-SLAM system. Our system detects (Alg. 1, Line 14) and solves (Alg. 1, Line 15) loop-closure when the camera captures images of a pre-visited scene.

Our loop-closure routine consists of two sub-routines: 1) loop candidates detection, and 2) loop candidates validation. We adopted ORB-SLAM2's loop candidates detection routine due to its superior runtime and robustness. In essence, our system maintains a database that stores the DBoW2 features of every processed frame. For every incoming frame, the database is being queried with the latest frame to identify matching candidates.

For the second part of the loop-closure, we invoke our feature extraction to relative orientation computation routines (Alg.1, Line 4 to Line 7) and validate the legitimacy of the candidates based on the number of inlier feature matches $|\mathcal{C}'_{j,k}|$. This process is similar than in ORB-SLAM2, but instead of computing the similarity $Sim(3)$ transform with two sets of 3D point clouds (seen in the loop-closing pair of frames), we compute the relative orientations, given 2D-2D feature correspondences in the loop closing pair of frames. Upon adding the loop-closure edges, we solve a system-wide rotation averaging problem (Alg. 1, Line 15).

IV. Experiments

We first provide empirical results to demonstrate the quality of our orientation estimates. Then, we showcase two potential SLAM applications for our proposed method: the known rotation problem (Sec. IV-D), and pure rotation motion (Sec. IV-E). Our system requires of only three hyperparameters: the size of $\mathcal{F}_{\text{window}}$, the size of $\mathcal{R}_{\text{window}}$, and θ_{matches}. We set them to $|\mathcal{F}_{\text{window}}| = 4$, $|\mathcal{R}_{\text{window}}| = 10$, and $\theta_{\text{matches}} = 100$ for all the experiments.

We implemented our proposed method in C++ and MATLAB R2019b. All the experiments were conducted on a standard PC with six Intel i5-8400 CPU @ 2.80 GHz cores. The relative orientation estimation module (Sec. III-B) was adopted from the OpenGV [34] library.

A. Datasets and Evaluation Metrics

KITTI odometry benchmarking [22] is one of the most popular datasets in the VO/V-SLAM community. We use the 11 sequences (00-10) with provided camera pose ground truths. Besides, we also perform experiments on our own dataset - the Carpark dataset. This sequence was collected with a camera and an IMU sensor attached on a driving vehicle in an open car park. The sequence has 1600 images which involves four strong rotations chunks (lead by turning motion) which is challenging for monocular VO and V-SLAM systems.

We benchmark with standard evaluation metrics. The first metric is the *average rotation error* (as proposed in [22])

$$r_{\text{err}} := \frac{1}{|\mathcal{F}|} \sum_{(i,j)\in\mathcal{F}} \angle((\hat{\mathbf{R}}_i^T \hat{\mathbf{R}}_j)^T (\mathbf{R}_i^T \mathbf{R}_j)), \tag{12}$$

where $\mathcal{F}$ is a set of pair of frame indices $\{(i,j)\}$, where the distance between frames in each pair varies from 100m to 800m[4], $\hat{\mathbf{R}}$ and $\mathbf{R}$ are the ground-truth and the estimated orientations, and $\angle(\cdot)$ returns the angle of the rotation matrix.

Our second metric is the rotation component of the *Relative Pose Error* (RPE) [35] for which we used two variants over n camera poses. For a pre-defined "time-step" Δ, the first variant is the RMSE over the $m := n - \Delta$ angular residuals

$$E_i := \angle((\hat{\mathbf{R}}_i^T \hat{\mathbf{R}}_{i+\triangle})^T (\mathbf{R}_i^T \mathbf{R}_{i+\triangle})). \tag{13}$$

More explicitly,

$$\text{RMSE}(E_{1:n}, \triangle) := \left(\frac{1}{m} \sum_{i=1}^{m} E_i^2 \right)^{1/2}. \tag{14}$$

[4]See the official site http://www.cvlibs.net/datasets/kitti/eval_odometry.php

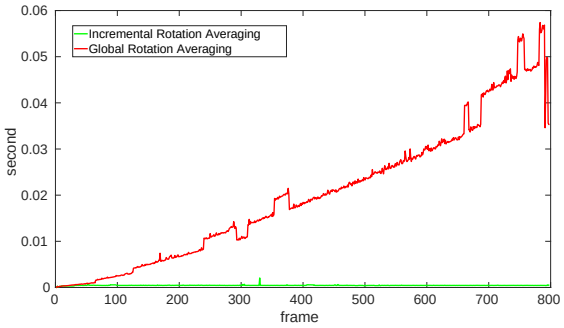


Fig. 3: Runtime comparison for rot. averaging in an incremental and global fashion on the KITTI 03 seq. Our incremental strategy exhibits constant runtime per frame (avg. 0.45ms).

TABLE I: Ablation comparison for the incremental rotation averaging strategy. The significant RPE_1 error gap (lower the better) demonstrates the effectiveness of our incremental solution. Best results are in bold.

		Baseline		Ours w/o loop closure	
Dataset	Sequence	RPE_1 [deg]	RPE_n [deg]	RPE_1 [deg]	RPE_n [deg]
KITTI	00	0.36	8.67	**0.13**	**3.03**
	01	0.27	10.34	**0.25**	**4.77**
	02	0.29	16.03	**0.080**	**3.92**
	03	0.28	5.47	**0.053**	**0.65**
	04	0.04	1.08	**0.034**	**0.50**
	05	0.25	11.36	**0.053**	**2.27**
	06	0.18	4.72	**0.049**	**1.41**
	07	0.28	7.49	**0.065**	**1.04**
	08	0.27	9.21	**0.055**	**3.18**
	09	0.28	9.85	**0.051**	**1.37**
	10	0.38	13.25	**0.061**	**2.30**
Carpark		0.95	3.48	**0.14**	**1.05**

As recommended by [35], $\triangle$ should be set to 1 for VO systems that consider only frame-to-frame accuracy. As such the error reflects the rotation drift between consecutive pair of frames. Meanwhile, for V-SLAM system that emphasis on the overall drifting as well, we use the second RPE variant. The second RPE variant iterates $\triangle$ over a set of numbers ranging the first frame to the last frame n, and

$$\text{RMSE}(E_{1:n}) := \frac{1}{n} \sum_{\triangle=1}^{n} \text{RMSE}(E_{1:n}, \triangle) \tag{15}$$

averages the root-mean-square-error over n.

Henceforth we refer Eq. (14) as RPE_1 and Eq. (15) as RPE_n. Note that RPE_n is the toughest metric since it involves the furthest pair of frames among the three metrics.

B. Ablation study of the incremental rotation averaging

We first evaluate the effectiveness of our incremental rotation averaging component by comparing against a baseline without this component. Instead, the baseline estimates the absolute orientations by chaining relative orientations (Eq. (2)). We deactivate the the loop-closure component (Lines 14-16 in Alg. 1) in both our method and the baseline.

TABLE II: Comparison of our proposed method against state-of-the-art monocular VO and V-SLAM system. The best results are in bold; second best results are underlined.

Datasets	Seq.	DF-VO[3]			ORB-SLAM2[4]			Ours		
		r_{err} [deg/100m]	RPE_1 [deg]	RPE_n [deg]	r_{err} [deg/100m]	RPE_1 [deg]	RPE_n [deg]	r_{err} [deg/100m]	RPE_1 [deg]	RPEn [deg]
KITTI	00	0.58	**0.12**	4.06	**0.33**	0.38	2.27	0.41	0.13	**1.88**
	01	17.04	9.95	124.33	**0.42**	0.70	6.10	0.86	**0.26**	**5.53**
	02	0.52	**0.072**	4.37	**0.30**	0.17	**2.47**	0.40	0.081	3.06
	03	0.39	**0.049**	0.95	**0.21**	0.062	**0.35**	0.30	0.052	0.66
	04	**0.25**	0.036	0.45	0.31	0.063	**0.33**	0.33	**0.035**	0.54
	05	0.30	**0.048**	1.62	**0.26**	0.14	**1.16**	0.34	0.055	1.27
	06	0.30	**0.037**	**0.89**	**0.24**	0.11	1.07	0.27	0.053	0.94
	07	**0.27**	**0.037**	**0.78**	0.49	0.081	1.12	0.41	0.075	0.99
	08	0.32	**0.050**	2.24	**0.32**	0.090	**1.83**	0.39	0.055	3.38
	09	**0.29**	**0.045**	1.70	0.31	0.11	1.47	0.31	0.050	**1.08**
	10	**0.37**	**0.053**	**1.36**	0.38	0.13	2.75	0.49	0.062	2.57
Carpark		-	-	-	0.32	**0.14**	**0.41**	**0.27**	**0.14**	0.48

As explained earlier, RPE_1 is the relevant comparing metric for this experiment since our method can be regarded as a pure VO system when loop-closure is deactivated. The results in Tab. I show that the incremental rotation averaging strategy has a substantial impact on our system, significantly outperforming the baseline on every sequence. On average, we outperformed RPE_1 in 0.181 degrees over the KITTI dataset, and in 0.81 degrees on the Carpark dataset. We also reported RPE_n values to compare against the results with loop-closure reported in Tab. II. Note that the performance of our method with the loop-closure module activated improves significantly on all the sequence with loops (e.g., KITTI sequences 00, 02, 05, 06, 07, 09, and in the Carpark dataset).

A major advantage of our incremental rotation strategy is scalability. By fixing the absolute orientations to estimate to a constant $|\mathcal{R}_{\text{window}}|$ at every frame step, solving our proposed formulations (Eq. (8)) takes constant runtime. To demonstrate the speed gain, we compare against solving "conventional" rotation averaging (3) over the entire view-graph at every frame step. Fig. 3 plots the runtimes for the comparison. The runtime for solving rotation averaging over the entire view-graph grows as the number of frame processed increases. Meanwhile, our incremental strategy maintains a constantly low processing time (0.45ms on average) at every frame step.

C. Accuracy of Camera Orientation

We compare against the state-of-the-art VO and V-SLAM systems, namely DF-VO [3] and ORB-SLAM2[4], and reported results in Tab. II. As for the results of the DF-VO method, we downloaded DF-VO's estimates (KITTI) from their official github repo[5]. As of ORB-SLAM2, we ran it on all tested sequences five times and report the median results. Similarly, we report the median of five different runs of our proposed method.

Our system achieves competitive performance to the state-of-the-art on both the KITTI and the Carpark datasets over all metrics. The RPE_1 evaluation shows that DF-VO's frame-to-frame drift is the best among the three methods. ORB-SLAM2 tends to have a higher frame-to-frame error due to its frame management scheme that skips consecutive frames with similar appearances. In terms of RPE_n, our method outperforms ORB-SLAM2 in 5 KITTI sequences (00, 01, 06, 07, and 09). As mentioned earlier, comparing RPE_n with DF-VO is unfair due to the lack of loop-closing ability.

[5]https://github.com/Huangying-Zhan/DF-VO

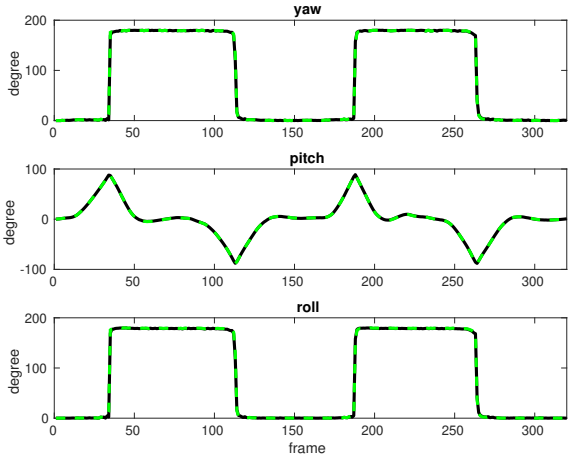


Fig. 4: The yaw, pitch, roll comparisons between ground-truth (black solid line) and our proposed method (green dash line) on the Carpark dataset. The dataset has four strong rotation chuncks (see "pitch" plot) caused by vehicle in turning motion. Our method manage to produce high accuracy orientation estimates from 2D-2D feature correspondences alone.

To provide a better insight of our orientation estimates, we convert the rotation matrix into the Euler angles (yaw, pitch, and roll) and plot it against the ground-truth of the Carpark dataset in Fig. 4. We observe that the orientation variation throughout the sequence is significant and our estimates closely align with the ground-truths.

D. Application A - The Known Rotation Problem

An application of our system is to provide the camera orientations $\{\mathbf{R}_j\}$ to then estimate the camera positions $\{\mathbf{t}_j\}$ and the scene coordinates $\{\mathbf{X}_i\}$ by solving the known rotation problem [36]

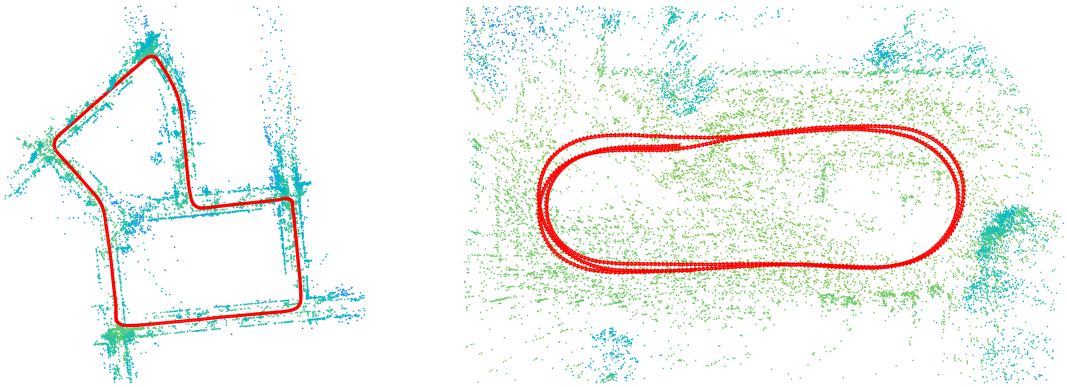
$$\begin{aligned} &\min_{\{\mathbf{t}_j\},\{\mathbf{X}_i\}} \max_{i,j} \|\mathbf{x}_{i,j} - f(\mathbf{X}_i|\mathbf{R}_j,\mathbf{t}_j)\| \\ &\text{s.t.} \quad \mathbf{R}_j^{(3)}\mathbf{X}_i + \mathbf{t}_j^{(3)} > 0 \quad \forall (i,j) \in \mathcal{E}, \end{aligned} \tag{16}$$

where $f(\cdot)$ projects scene points into the image plane.

The objective of the known rotation problem is quasi-convex allowing global optimisation. We used the fast Res-Int solver [37] to solve (16). Finally we refine all the variables $(\{\mathbf{R}_j\},\{\mathbf{t}_j\},\{\mathbf{X}_i\})$ with BA. Fig. 5 displays the results.

E. Application B - Pure Rotation Motion

Modern monocular visual SLAM systems (e.g., ORB-SLAM2, OpenVSLAM [38]) utilise BA as the core engine to jointly estimate the camera poses and scene points. One



(a) KITTI seq. 07 (b) Carpark dataset

Fig. 5: The SLAM outputs of Application A (Sec. IV-D) for (a) the the KITTI dataset seq. 07, and (b) the Carpark dataset.

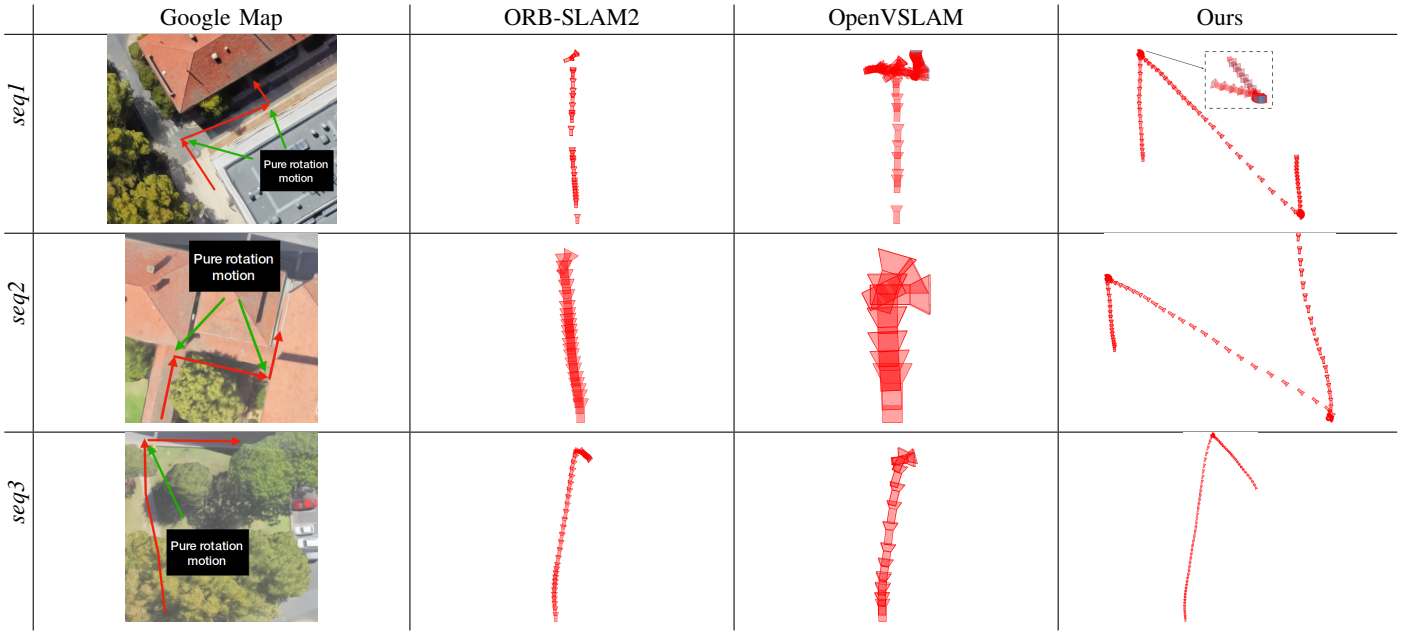


Fig. 6: Camera trajectories (sampled for better visualisation) for ORB-SLAM2, OpenVSLAM and our method over three sequences that contains rotation-only motions. The first column shows the approx. trajectories (captured from GoogleMap).

drawback of such approach is that BA cannot handle rotation-only motion as the 3D scene points can not be triangulated[6]. We captured three sequences where our mobile phone is attached to a gimbal and rotate around a principal axis to simulate such a motion. Fig. 6 depicts the resulting poses of ORB-SLAM2, OpenVSLAM (latest implementation of ORB-SLAM2-like monocular SLAM pipeline), and our method on these sequences. Note that the commercial based mobile phone camera is not precisely calibrated[7] which causes the drift in the estimated trajectory.

Most of the examples in Fig. 6 (except (seq1, ORB-SLAM2) and (seq3, OpenVSLAM)) show that the tracking of the camera pose stopped during long pure rotation motion part of the sequence. Note that both methods employ a scene point selection heuristic (based on reprojection error) to decide if the newly triangulated point should be added to the existing local map. During pure rotation motion, most of the triangulated points were successfully trimmed away by the heuristic and eventually went into "relocalisation mode" when their local map run out of 3D points. However, in some cases where the heuristic fails to reject these points and continue to track the camera poses, the scale of the camera trajectory after the pure rotation motion became inaccurate as for (seq3, ORB-SLAM2), and (seq1, OpenVSLAM).

Since our method detaches the orientation component of the camera poses from the joint estimation framework, the estimation of the camera orientation remains unaffected during pure rotation motion. We skip the known rotation problem and the BA routines (as explained in Sec. IV-D) in the rotation-only frames and update the camera orientations with the outputs of our proposed method. Meanwhile, we fix the camera positions during those motions.

[6]We are referring to the scenarios where the pure rotation motion is long enough where the camera is turned to a new scene and observed a new scene where none of the features were triangulated previously.

[7]We calibrated with a checkerboard.

V. Conclusion

We presented a novel monocular rotational odometry system that is capable of producing accurate camera orientation estimates. The key aspects of proposed method are 1) its motion robustness, 2) a new constant time incremental rotation averaging solver, and 3) the ability to perform loop closure. Our proposed method achieve state-of-the-art accuracy on real-world datasets. Lastly, two potential applications of our method were demonstrated.

References

[1] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[2] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[3] H. Zhan, C. S. Weerasekera, J. Bian, and I. Reid, "Visual odometry revisited: What should be learnt?" *arXiv preprint arXiv:1909.09803*, 2019.

[4] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[5] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[6] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods," *Journal of guidance, control, and dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[7] A. Khosravian, T.-J. Chin, I. Reid, and R. Mahony, "A discrete-time attitude observer on so (3) for vision and gps fusion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5688–5695.

[8] K. Sim and R. Hartley, "Recovering camera motion using l\infty minimization," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 1230–1237.

[9] A. Parra, T.-J. Chin, A. Eriksson, and I. Reid, "Visual slam: Why bundle adjust?" in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2385–2391.

[10] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[11] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[12] C. Engels, H. Stewénius, and D. Nistér, "Bundle adjustment rules," *Photogrammetric computer vision*, vol. 2, no. 2006, 2006.

[13] C. Pirchheim, D. Schmalstieg, and G. Reitmayr, "Handling pure camera rotation in keyframe-based slam," in *2013 IEEE international symposium on mixed and augmented reality (ISMAR)*. IEEE, 2013, pp. 229–238.

[14] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Höllerer, "Live tracking and mapping from both general and rotation-only camera motion," in *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2012, pp. 13–22.

[15] P. Torr, A. W. Fitzgibbon, and A. Zisserman, "Maintaining multiple motion model hypotheses over many views to recover matching and structure," in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 485–491.

[16] L. Kneip and S. Lynen, "Direct optimization of frame-to-frame rotation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2352–2359.

[17] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 37–45.

[18] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "Demon: Depth and motion network for learning monocular stereo," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5038–5047.

[19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[20] A. Chatterjee and V. Madhav Govindu, "Efficient and robust large-scale rotation averaging," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 521–528.

[21] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *International journal of computer vision*, vol. 103, no. 3, pp. 267–305, 2013.

[22] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[23] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[24] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1983–1992.

[25] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1851–1858.

[26] X. Gao, R. Wang, N. Demmel, and D. Cremers, "Ldso: Direct sparse odometry with loop closure," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2198–2204.

[27] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[28] V. M. Govindu, "Lie-algebraic averaging for globally consistent motion estimation," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1. IEEE, 2004, pp. I–I.

[29] A. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin, "Rotation averaging and strong duality," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 127–135.

[30] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert, "Initialization techniques for 3d slam: a survey on rotation estimation and its use in pose graph optimization," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 4597–4604.

[31] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 756–777, 2004. [Online]. Available: https://doi.org/10.1109/TPAMI.2004.17

[32] L. Kneip, R. Siegwart, and M. Pollefeys, "Finding the exact rotation between two images independently of the translation," in *European conference on computer vision*. Springer, 2012, pp. 696–709.

[33] A. Chatterjee and V. M. Govindu, "Robust relative rotation averaging," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 958–972, 2017.

[34] L. Kneip and P. Furgale, "Opengv: A unified and generalized approach to real-time calibrated geometric vision," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1–8.

[35] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 573–580.

[36] F. Kahl and R. Hartley, "Multiple-view geometry under the l-infinity norm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1603–1617, 2008.

[37] Q. Zhang, T.-J. Chin, and H. Minh Le, "A fast resection-intersection method for the known rotation problem," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3012–3021.

[38] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: a versatile visual slam framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2292–2295.