Hindawi
Journal of Robotics
Volume 2020, Article ID 7362952, 14 pages
https://doi.org/10.1155/2020/7362952

Research Article

CKF-Based Visual Inertial Odometry for Long-Term Trajectory Operations

Trung Nguyen,[1] **George K. I. Mann**,[1] **Andrew Vardy**,[2] **and Raymond G. Gosine**[1]

[1]*Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, NL A1B 3X9, Canada*
[2]*Department of Computer Science and Department of Electrical and Computer Engineering, Memorial University of Newfoundland, St. John's, NL A1B 3X9, Canada*

Correspondence should be addressed to Trung Nguyen; tn0432@mun.ca

Received 11 January 2020; Accepted 16 May 2020; Published 3 June 2020

Academic Editor: L. Fortuna

Copyright © 2020 Trung Nguyen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The estimation error accumulation in the conventional visual inertial odometry (VIO) generally forbids accurate long-term operations. Some advanced techniques such as global pose graph optimization and loop closure demand relatively high computation and processing time to execute the optimization procedure for the entire trajectory and may not be feasible to be implemented in a low-cost robotic platform. In an attempt to allow the VIO to operate for a longer duration without either using or generating a map, this paper develops iterated cubature Kalman filter for VIO application that performs multiple corrections on a single measurement to optimize the current filter state and covariance during the measurement update. The optimization process is terminated using the maximum likelihood estimate based criteria. For comparison, this paper also develops a second solution to integrate VIO estimation with ranging measurements. The wireless communications between the vehicle and multiple beacons produce the ranging measurements and help to bound the accumulative errors. Experiments utilize publicly available dataset for validation, and a rigorous comparison between the two solutions is presented to determine the application scenario of each solution.

1. Introduction

Visual inertial odometry (VIO) employs the sensor fusion between inertial measurement unit (IMU) measurements and camera's image information to enhance the accurate estimation of vehicle trajectory [1, 2]. The VIO system architecture is summarized in Figure 1 where the front-end and back-end computations are designed to exploit the benefits of both sensors, produce reliable ego-motion estimation, and achieve robust performance. The vision front-end computation attempts to track 3D feature points through different camera images. Geometric constraints between multiple camera poses are established to fuse with the IMU data [3–5]. The effectiveness of the process and the estimation accuracy depend on the sensor fusion strategy. Recent literature has introduced multiple sensor fusion solutions with varying requirements of hardware computing resources [2]. However, VIO systems suffer from the inevitable accumulation of error. This limitation makes the system gradually diverge and even fail to track the vehicle trajectory over long-term operation. VIO only produces reliable estimation of the vehicle trajectory in short-term operation and short-distance travel. This issue obviously demands the development of VIO techniques to allow the system to operate for longer duration.

In SLAM and odometry applications, loop closure [6–8] and global pose graph optimization [7] have proven to be effective in addressing the accumulative error issue. These techniques require the entire trajectory and map to relocalize the vehicle estimates and improve the estimation accuracy. Such solutions demand extremely high computational load, memory utilization, and processing time for execution [9, 10]. These costs are challenging in real-time computation and may not be affordable for some microrobotic systems [1, 10, 11], having limited hardware computing capability. For example, the work presented in [10] reported the failure

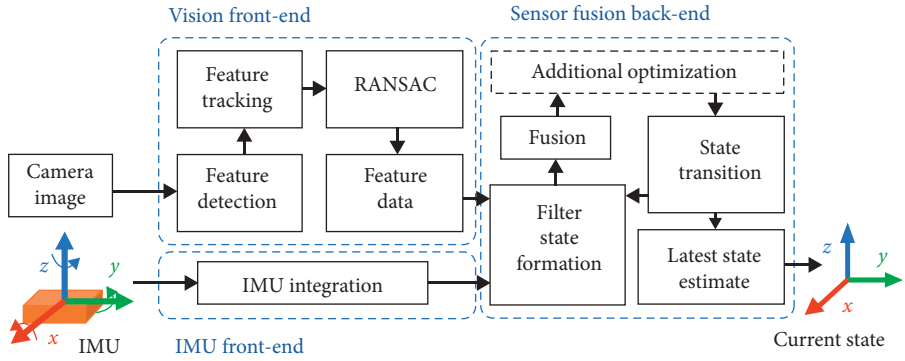

Figure 1: Overview of visual inertial odometry architecture.

to execute these advanced techniques of VINS-Mono [7] on ODROID, an embedded PC with a hybrid processing unit [12]. Also, these solutions are effective only when the vehicle makes a closed-loop trajectory to reobserve some previous landmark features. Not all vehicle trajectories will include loops to activate the loop closure. Alternatively, in [9, 13, 14], a local optimization within the sliding window has been performed. Their implementations did not require the installation of any specialized hardware computing platform and some advanced optimization software library. The effectiveness of this technique depends on the tuning parameters and the criteria for terminating the iteration.

The work described in [4] had developed a tightly coupled cubature Kalman filter- (CKF-) based algorithm for VIO application. This was a filtering approach that also encountered the issue of error accumulation over long-term operation. However, the CKF-based algorithm utilized trifocal tensor geometry (TTG) for the computationally efficient visual measurement update. The TTG-based model had shown enormous potential to implement the iteration for the local optimization. Hence, in this paper, we attempt to implement this idea and improve the filter estimation accuracy over long-term operation.

This paper describes the development of a novel iterated CKF-based VIO that performs multiple corrections on a single measurement. The process locally optimizes the estimate of the current filter state and covariance during the visual measurement update. The system employs maximum likelihood estimate (MLE) based criteria to terminate the optimization process. For comparison, this paper also investigates the combination of VIO and ranging measurements as a second solution. The wireless communications between the vehicle and multiple beacons produce ranging measurements and help to bound the accumulative errors. The integration follows the sequential-sensor-update approach, which enables the operational independence of each measurement update. To summarize, this paper makes the following contributions. Firstly, we utilize the benefit of TTG model to enhance the optimization during the filter update step. The implementation does not increase the system complexity significantly or require the installation of any advanced optimization library. This strategy is suitable for self-localization projects without using any additional sensors. Secondly, we investigate the combination of VIO and ranging measurements to bound the estimation error over long-term operation. This solution can be applied for large-scale navigation projects with multiple known-location beacons. Thirdly, the work described in this paper contributes to the group of VIO filtering approaches. Two solutions are proposed for long-term operation without using any map and the entire trajectory. We also conduct some comparisons to determine the benefits of each solution and application scenario.

The remainder of the paper is organized as follows. Some related works are presented in the next section. Section 3 introduces system coordinates, the original VIO design based on CKF, and the issue of error accumulation. Section 4 describes the first solution for employing iterated CKF, while Section 5 describes the second solution for using ranging sensors. Experimental validation for each solution is also reported accordingly. Finally, Section 6 presents some discussion and conclusion.

2. Related Works

Generally, engineered devices are subject to unavoidable imperfections due to technical limitations of manufacturing processes and material properties. The defects result in many uncertainties (i.e., noise and disturbance) affecting the device's operation [15, 16]. Device developers need to invest in more time and effort to understand the sources and impacts of these uncertainties before proposing any algorithm and designing optimal software architecture. The system, including camera and IMU, also encounters a class of similar problems that raise more challenges in calibration and designing computational procedures [2, 6, 7]. It is worthwhile to study uncertainties in the VIO system to improve its performance in long-term operations.

In the VIO literature, many researchers have addressed the accumulative error issue. Nonlinear optimization approach can minimize the estimation error over long-term operation. For example, Qin et al. employed a global 4-DOF pose graph optimization for the entire trajectory [7]. This strategy consumes a considerable amount of memory and CPU usage, especially when the size of the pose graph grows unbounded as the travel distance increases. A down-sample process was useful in this case to include only all primary key-frames within loop closure constraints. This process can

limit the pose graph database to a certain size but negatively affect the quality of the estimation. The optimization can be enhanced within a sliding window to reduce the computational cost as in [6]. Loop closure approach is another common solution to address estimation drift [7, 17, 18]. If the loop is detected, the vehicle estimate is relocalized based on multiple constraints of camera observations. In general, compared to other existing VIO strategies [10], the VIO systems with these two approaches require considerable computational resources, processing time, and the installation of advanced optimization software libraries such as Google's Ceres Solver [19] and DBoW2 loop detector [20]. These approaches are not preferable for microrobotic systems with limited memory and CPU speed. For example, the work presented in [10] needs to reduce the maximum number of features from 400 to 200, the key-frame of sliding window from 5 to 3, and the number of IMU linked frames from 3 to 2 in order to implement OKVIS [6] on ODROID. Similarly, the implementation of VINS-Mono without loop closure [7] on ODROID requires the reduction of the number of tracked features from 150 to 100. VINS-Mono with loop closure cannot be executed on ODROID [7].

Constructing the optimization process within a sliding window has reduced the computational complexity to apply for resource-constrained devices. Heo et al. upgraded the Multistate Constraint Kalman Filter (MSCKF) structure using the local optimization for all measurements before refining the global states [3, 9, 21]. The deployment attempts to utilize full information within the sliding window to improve the filter performance. The optimal relative pose constraints are inferred to include the relative motion constraints and any prior information during the filter update. Even so, the filter performance in noisy environments is questionable because if only shows a slightly smaller estimation error than the MSCKF in a practical experiment. Alternatively, some researches [13, 14] implemented an iterative procedure to optimize the filter mean and covariance during the filter update. This kind of deployment attempts to minimize the localization errors at the cost of increased computational requirements. However, this increase is feasible for resource-constrained devices [13]. The complexity of the visual measurement model also affects the computational efficiency of the optimization execution. In our previous project [4], we have explored the potential of TTG, which can predict the visual measurements using three camera frames. Moreover, the TTG-based visual measurement model operates as a straightforward function and consumes less computational cost than the traditional model. It will be more efficient if we perform local optimization with the TTG-based model. In this paper, we will implement this idea, determine the benefits of using TTG in local optimization, and compare it with other solutions in an attempt to minimize the estimation error accumulation.

Besides inertial and visual measurements, some researches have included other measurements to improve the system performance in dynamic environments [22] and in some cases where camera images are not useful for navigation [23]. Peretroukhin et al. applied Convolutional Neural Network (CNN) to extract the sun direction directly from the existing image stream [22]. The extraction was used to correct the global orientation. Although the estimation of a sun direction vector improved vehicle trajectory tracking, that solution was not always available, particularly during cloudy weather and nighttime. It only affects the orientation and also requires considerable resources to train and execute the deep learning model of sun detection. Many studies attempt to address the computational issue of deep learning for real-time applications using parallel architecture [24–26]. These solutions accelerate the deep learning execution for constrained hardware with competitive energy efficiency. Also, a Light Detection and Ranging (LIDAR) sensor can help to reduce visual drift in undesired lighting conditions [27, 28]. This solution can improve the positioning accuracy over long-term operation satisfactorily. However, the deployment of LIDAR can raise the issues of power consumption and payload for microrobotic systems. The wireless communication between the vehicle (tag) and known-location beacon (anchor) has been popularly applied to supplement the primary navigation system [27, 29–31]. Such systems commonly detect the Time of Arrival (TOA) of signals encoded in the radio or acoustic waves to conduct a ranging measurement. For example, some researches [30, 31] have deployed ultra-wideband (UWB) radio modules for ranging measurements. This technique suffers from systematic errors such as uncertain wave speed or clock synchronization errors. Consequently, it cannot directly measure the true geometric range, which is why it is called pseudorange measurement [27]. Wang et al. integrated ranging measurements through solving optimization problem [31]. The process attempts to align UWB localization with VIO to produce an optimal estimate for the vehicle position. Similarly, the work presented in [32] formulated the optimization in the scheme of moving horizon estimation using CasADi software optimization library [33]. Despite the heavy computational cost, the experiment result suggested the use of ranging measurements to bound the accumulative errors over a long-term operation. This also demands a better strategy to integrate VIO with ranging measurement for resource-constraint systems. Alternatively, in this paper, we will apply a sequential-sensor-update approach in the scheme of Kalman filter for the multisensor integration. It is interesting to compare the effectiveness of two proposed approaches (i.e., local optimization against additional-ranging integration) in improving the VIO estimate over a long-term operation.

3. Preliminaries

This section attempts to summarize a CKF-based design for VIO sensor fusion algorithm and the benefits of TTG-based measurement model, which was developed in the previous work [4]. This design reveals the issue of the accumulative errors in long-term operation. Two solutions proposed in this paper will be presented in the next section.

3.1. System Coordinates and Notation. Matrices and vectors are denoted in boldface. We use a superscript to indicate that the vector is expressed with respect to a specific reference frame. For example, ${}^{A}\mathbf{v}$ is the vector $\mathbf{v}$ expressed in frame

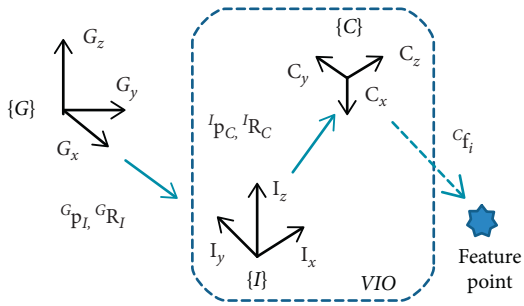

Figure 2: Coordinate system of VIO.

$\{A\}$. The VIO coordinate system is assigned as in Figure 2 with a global frame $\{G\}$, a camera frame $\{C\}$, and an IMU frame $\{I\}$. The VIO system tracks the transformation of $\{I\}$ with respect to $\{G\}$ with the rotation matrix ${}^G\mathbf{R}_I$ and the translation matrix ${}^G\mathbf{p}_I$. Similarly, the transformation of $\{C\}$ with respect to $\{I\}$ is represented by the rotation matrix ${}^I\mathbf{R}_C$ and the translation matrix ${}^I\mathbf{p}_C$. The camera observes the landmark feature point ${}^C\mathbf{f}_i$ in order to perform the filter correction step. Note that rotation matrices have the essential properties of special orthogonal group $\mathrm{SO}(3)$: $\mathbf{R} \in \mathrm{SO3} \doteq \{\mathbf{R} \in \mathbb{R}^{3\times3}: \mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}, \quad \det(\mathbf{R}) = 1\}$. $\mathrm{SO}(3)$ denotes the tangent space to the group $\mathrm{SO}(3)$'s manifold (at the identity) and coincides with the space 3×3 of skew symmetric matrices. $\mathbf{I}$ denotes the identity matrix while $\mathbf{0}$ denotes the zero matrix. We use subscripts to indicate the sizes of these matrices such as $\mathbf{I}_{3\times3}$. We also mention Special Euclidean Group $\mathrm{SE}(3)$ for describing the group of 3D rigid motion, $\mathrm{SE}(3) = \{(\mathbf{R}, \mathbf{p})\colon R \in \mathrm{SO}(3), \mathbf{p} \in \mathbb{R}^3\}$. That group's operations are listed: $\mathbf{T}_1\mathbf{T}_2 = (\mathbf{R}_1\mathbf{R}_2, \mathbf{R}_1\mathbf{p}_2 + \mathbf{p}_1)$ and $\mathbf{T}_1^{-1} = (\mathbf{R}_1^T, -\mathbf{R}_1^T\mathbf{p}_1)$. We use quaternion approach to operate the rotation. $\mathrm{R}(\overline{q})$ is a function producing the rotational matrix from the quaternion $\overline{q}$.

3.2. CKF-Based Visual Inertial Odometry

3.2.1. Filter State Formation and Prediction. The VIO system utilizes IMU data for filter state propagation. At the time t, IMU sensor provides accelerometer and gyroscope measurements (${}^I\mathbf{a}_m$ and ${}^I\omega_m$), which are expressed in three directions (x, y, and z) with metric unit in $\{I\}$ frame and modeled as presented in (1) [3, 34].

$$\begin{aligned} {}^I\mathbf{a}_m(t) &= \mathrm{R}\left({}^I\overline{\mathbf{q}}_G(t)\right)\left({}^G\mathbf{a}(t) - {}^G\mathbf{g}\right) + \mathbf{b}_a(t) + \mathbf{n}_a(t), \\ {}^I\boldsymbol{\omega}_m(t) &= {}^I\boldsymbol{\omega}_m(t) + \mathbf{b}_g(t) + \mathbf{n}_g(t), \end{aligned} \tag{1}$$

where ${}^G\mathbf{g}$ is the gravitational acceleration in the frame $\{G\}$; ${}^I\omega$ is the platform angular velocity in the frame $\{I\}$; ${}^G\mathbf{a}$ is the platform linear acceleration in the frame $\{G\}$; and $\mathbf{b}_a$ and $\mathbf{b}_b$ are bias of accelerometer and gyroscope. The residual noise terms $\mathbf{n}_a$ and $\mathbf{n}_g$ are modeled as zero-mean white Gaussian noise processes. ${}^G\mathbf{v}_I$ denotes the linear velocity of the platform. The filter state includes the IMU state and the last two poses $({}^G\mathbf{p}_{I_1}, {}^G\overline{\mathbf{q}}_{I_1}, {}^G\mathbf{p}_{I_2}, {}^G\overline{\mathbf{q}}_{I_2})$ as $\mathbf{x}_k = [{}^G\mathbf{p}_I^T]$ $[{}^G\overline{\mathbf{q}}_I^T {}^G\mathbf{v}_I^T \mathbf{b}_a^T \mathbf{b}_g^T {}^G\mathbf{p}_{I_1}^T, {}^G\overline{\mathbf{q}}_{I_1}^T, {}^G\mathbf{p}_{I_2}^T, {}^G\overline{\mathbf{q}}_{I_2}^T]^T$. The true filter state is described as a combination of the nominal state $\hat{x}_k$ and the error state $\tilde{x}_k$. In the filter prediction step, 4th order Runge-Kutta numerical integration of kinetic equations (2) is utilized to compute the predicted state [3].

$$\begin{aligned} {}^G\dot{\hat{\mathbf{p}}}_I &= {}^G\hat{\mathbf{v}}_I; \\ {}^G\dot{\hat{\mathbf{v}}}_I &= \mathrm{R}\left({}^G\hat{\overline{\mathbf{q}}}_I\right)\left({}^I\mathbf{a}_m - \hat{\mathbf{b}}_a\right) - {}^Gg; \\ {}^G\dot{\hat{\overline{\mathbf{q}}}}_I &= \frac{1}{2}{}^G\hat{\overline{\mathbf{p}}}_I \otimes \left({}^I\boldsymbol{\omega}_m - \hat{\mathbf{b}}_g\right); \\ \dot{\hat{\mathbf{b}}}_a &= \mathbf{0}_{3\times1}; \\ \dot{\hat{\mathbf{b}}}_g &= \mathbf{0}_{3\times1}; \\ {}^G\dot{\hat{\mathbf{p}}}_{I_1} &= \mathbf{0}_{3\times1}; \\ {}^G\dot{\hat{\overline{\mathbf{q}}}}_{I_1} &= \mathbf{0}_{4\times1}; \\ {}^G\dot{\hat{\mathbf{p}}}_{I_2} &= \mathbf{0}_{3\times1}; \\ {}^G\dot{\hat{\overline{\mathbf{q}}}}_{I_2} &= \mathbf{0}_{4\times1}; \end{aligned} \tag{2}$$

on the other hand, the error state is constructed as $\tilde{x}_k = [{}^G\tilde{\mathbf{p}}_I^T {}^G\delta\boldsymbol{\theta}_I^T {}^G\tilde{\mathbf{v}}_I^T \tilde{\mathbf{b}}_a^T \tilde{\mathbf{b}}_g^T {}^G\tilde{\mathbf{p}}_{I_1}^T {}^G\tilde{\overline{\mathbf{q}}}_{I_1}^T {}^G\tilde{\mathbf{p}}_{I_2}^T {}^G\tilde{\overline{\mathbf{q}}}_{I_2}^T]^T$. We prefer to present the filter rotation error through the error quaternion [3, 35]. Obviously, this presentation will handle the quaternion in its minimal representation and improve numerical stability [34]. In the case of quaternion, we can execute the transformation ${}^G\delta\overline{q}_{I_2}^T = \begin{bmatrix} 1 & (1/2)\delta\theta_I \end{bmatrix}^T$. The true filter state can be computed using

$$\begin{aligned} {}^G\overline{q}_I &= {}^G\hat{\overline{q}}_I \otimes {}^G\delta\overline{q}_I; \\ {}^G\mathbf{p}_I &= {}^G\hat{p}_I + {}^G\tilde{p}_I; \\ {}^G\mathbf{v}_I &= {}^G\hat{v}_I + {}^G\tilde{v}_I; \\ \mathbf{b}_a &= \hat{b}_a + \tilde{b}_a; \\ \mathbf{b}_g &= \hat{b}_g + \tilde{b}_g; \end{aligned} \tag{3}$$

$$\begin{aligned} {}^G\dot{\tilde{p}}_I &= {}^G\tilde{v}_I; \\ \dot{\tilde{b}}_a &= \mathbf{n}_{wa}; \\ \dot{\tilde{b}}_g &= \mathbf{n}_{wg}; \\ {}^G\delta\dot{\theta}_I &= -S\left({}^I\boldsymbol{\omega}_m - \hat{b}_g\right){}^G\delta\boldsymbol{\theta}_I - \tilde{b}_g - \mathbf{n}_g; \\ {}^G\dot{\tilde{v}}_I &= -R\left({}^G\hat{\overline{q}}_I\right)\mathrm{S}\left({}^I\mathbf{a}_m - \hat{b}_a\right) - R\left({}^G\hat{\overline{q}}_I\right)\tilde{b}_a - \mathrm{R}\left({}^G\hat{\overline{q}}_I\right)\mathbf{n}_a. \end{aligned} \tag{4}$$

The kinetic equations (4) describe the error state operation. These equations are written in the form of $\dot{\tilde{x}}_k = \mathbf{F}_c\tilde{x}_k + \mathbf{G}_c\mathbf{n}_{\mathrm{IMU}}$ with the noise vector $\mathbf{n}_{\mathrm{IMU}} = [\mathbf{n}_g, \mathbf{n}_a, \mathbf{n}_{wa}, \mathbf{n}_{wg}]^T$ associated with variance $\sigma_g^2, \sigma_a^2, \sigma_{wa}^2, \sigma_{wg}^2$, respectively. For discrete time implementation, $\mathbf{F}_c$ is discretized to have $\mathbf{F}_d$ as in [3, 36]. Then, the discrete time covariance $\mathbf{Q}_d$ is derived with the continuous time system noise covariance $\mathbf{Q}_c = \mathrm{diag}(\sigma_g^2, \sigma_a^2, \sigma_{wa}^2, \sigma_{wg}^2)$. The propagation of the state covariance matrix can be performed using the discretized error state propagation $\mathbf{F}_d$ and the error process noise covariance matrices $\mathbf{Q}_d$ as follows:

$$\mathbf{Q}_d = \int_{\Delta t} \mathbf{F}_d(\tau)\mathbf{G}_c\mathbf{Q}_c\mathbf{G}_c^T\mathbf{F}_d^T(\tau)\mathrm{d}\tau;$$
$$\mathbf{P}_{k|k-1} = \mathbf{F}_d\mathbf{P}_{k-1|k-1}\mathbf{F}_d^T + \mathbf{Q}_d. \quad (5)$$

3.2.2. Predicting Visual Measurement Using Trifocal Tensor Geometry. The VIO system utilizes visual measurements for filter update. This step requires performing the 3D feature-point reconstruction before predicting the measurements. This traditional model implements an inverse-depth least-squares Gauss–Newton optimization approach for the reconstruction, which requires considerable time for execution. Alternatively, we apply the point transfer approach using TTG [37], which has low computational complexity and can be used as a nonrecursive function to predict the measurement. This simplification is beneficial to enhance the optimization computation of solution 1. In this section, we summarize the main points of the TTG-based measurement model. Further descriptions about TTG can be found in [4, 37].

Assuming that a feature point is tracked through three consecutive images (i.e., I_1, I_2, and I_3), TTG is applied to predict the visual measurement in the latest image I_3. The computation utilizes three consecutive camera poses and tracked feature point. The image I_1 has a homogeneous coordinate of the feature point $\tilde{\mathbf{m}}_1 = [u_1, v_1, 1]^T$, while u_1 and v_1 are pixel coordinates. Similar denotation is applied with $\tilde{\mathbf{m}}_2$ and $\tilde{\mathbf{m}}_3$. As described in Figure 3, the prediction is achieved through point transferring from the image I_1 to the image I_3 with 2 transfers. In transfer I, we aim to construct the line $\mathbf{l}_2$ in the image I_2, which is perpendicular to the epipolar line $\mathbf{l}_{e2} = \mathbf{R}_{12}^T\mathbf{t}_{12} \times \tilde{\mathbf{m}}_1 = [l_1, l_2, l_3]^T$. The line $\mathbf{l}_2$ is computed as $\mathbf{l}_2 = [l_2, -l_1, -u_2l_2 + v_2l_1]^T$. Then transfer II is applied to predict the corresponding point $\hat{\mathbf{m}}_3 = \mathbf{K}(\sum_i \tilde{\mathbf{m}}_{1,i}\mathbf{T}_i^T)\mathbf{l}_2$, where $\mathbf{K}$ is an intrinsic camera matrix and $\tilde{\mathbf{m}}_{1,i}$ is i^{th} element of $\tilde{\mathbf{m}}_1$. This computation is applied similarly for other tracked feature points. Then, the residual can be calculated using the predicted and actual visual measurements. We can construct the visual measurement model $\mathbf{h}(\mathbf{x}_k, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\})$ as (6). The model also includes the epipolar geometry relations to satisfy the observability constraints of TTG.

$$\mathbf{h}(\mathbf{x}_k, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}) = \begin{bmatrix} \mathbf{K}\left(\sum_i \tilde{\mathbf{m}}_{1,i}\mathbf{T}_i^T\right)\mathbf{l}_2 \\ \tilde{\mathbf{m}}_2^T\mathbf{R}_{12}^T[\mathbf{t}_{12}]_\times\tilde{\mathbf{m}}_1 \\ \tilde{\mathbf{m}}_3^T\mathbf{R}_{23}^T[\mathbf{t}_{23}]_\times\tilde{\mathbf{m}}_2 \\ \tilde{\mathbf{m}}_3^T\mathbf{R}_{13}^T[\mathbf{t}_{13}]_\times\tilde{\mathbf{m}}_1 \end{bmatrix}, \quad (6)$$

where the tensor $\mathbf{T}_i (i = 1, 2, 3)$ is computed from three camera matrices $\mathbf{P}_1 = [\mathbf{I}_{3\times3}|0_{3\times1}]$, $\mathbf{P}_2 = [\mathbf{R}_{12}^T| - \mathbf{R}_{12}^T\mathbf{t}_{12}]$, and $\mathbf{P}_3 = [\mathbf{R}_{13}^T| - \mathbf{R}_{13}^T\mathbf{t}_{13}]$ as $\mathbf{T}_i = \mathbf{a}_i\mathbf{b}_4^T - \mathbf{a}_4\mathbf{b}_i^T$. $\mathbf{a}_i$ and $\mathbf{b}_i$ are the i^{th} columns of the respective camera matrices $(i = 1, \ldots, 4)$. The transformation $(\mathbf{R}_{12} = {}^G\mathbf{R}_{C_1}^T{}^G\mathbf{R}_{C_2}$ and $\mathbf{t}_{12} = {}^G\mathbf{R}_{C_1}^T({}^G\mathbf{p}_{C_2} - {}^G\mathbf{p}_{C_1}))$ can be calculated from three camera poses of the image i and the camera-IMU calibration with ${}^I\mathbf{R}_C$ and ${}^I\mathbf{p}_C$. We apply similar procedure for other transformation matrices.

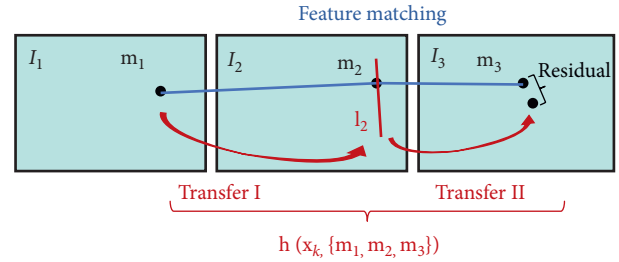

Figure 3: Illustration of predicting the visual measurement using TTG.

3.2.3. Cubature Kalman Filtering for Measurement Update. CKF is applied to handle the high nonlinearity of the TTG-based measurement model. CKF is a Jacobian-free nonlinear filter, which applies a deterministic sampling approach, Spherical Radial Transformation (SRT), and cubature rule to generate a minimal set of sampling points. Propagating these sampling points through the nonlinear functions results in a better approximation of the mean and covariance (Figure 4). CKF shares common properties with UKF but is improved in numerical implementation and system stability. In [38], the authors have addressed the vulnerability of the UKF to system failure associated with the negatively weighted sigma points. The presence of these negative points may cause the system to halt or even fail when taking the square-root operation of the covariance matrix. This vulnerable step is removed in the CKF implementation [4, 38].

CKF is a straightforward implementation of SRT in the recursive estimation. If a system has n state variables, the third order CKF selects $2n$ cubature points in order to compute the standard Gaussian weighted integral:

$$I_{\mathbf{N}}(\mathbf{f}) = \int_{\mathbb{R}^n} \mathbf{f}(\mathbf{x})\mathbf{N}(\mathbf{x}, \hat{\mathbf{x}}, \hat{P})\mathrm{d}\mathbf{x} \cong \sum_{s=1}^{2n} \frac{1}{2n}\mathbf{f}(\hat{\mathbf{x}} + \mathbf{S}\boldsymbol{\xi}_s), \quad (7)$$

where $\mathbf{N}(.,.)$ is the conventional symbol for a Gaussian density and S is the square-root factor of the covariance P satisfying the equality $\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1}\mathbf{S}_{k|k-1}^T$. Cubature points $(s = 1, 2, \ldots, 2n)$ are generated with its particular parameter:

$$\mathbf{X}_{s,k|k-1} = \mathbf{S}_{k|k-1}\boldsymbol{\xi}_s + \hat{x}_{k|k-1};$$
$$\boldsymbol{\xi}_s = \begin{cases} \sqrt{n}\,e_s, & s = 1, 2, \ldots, n, \\ -\sqrt{n}\,e_{s-n}, & s = n+1, n+2, \ldots, 2n, \end{cases} \quad (8)$$

where $e_s \in \mathbb{R}^{n\times1}$ is the s$^{\text{th}}$ elementary column vector.

These cubature points are evaluated through the nonlinear measurement model (6) so as to obtain the predicted measurements (9). Then the filter state and covariance are updated using the actual measurement $\mathbf{z}_k$ and the predicted measurement $\hat{\mathbf{z}}_{k|k-1}$ as described in (10) and (11).

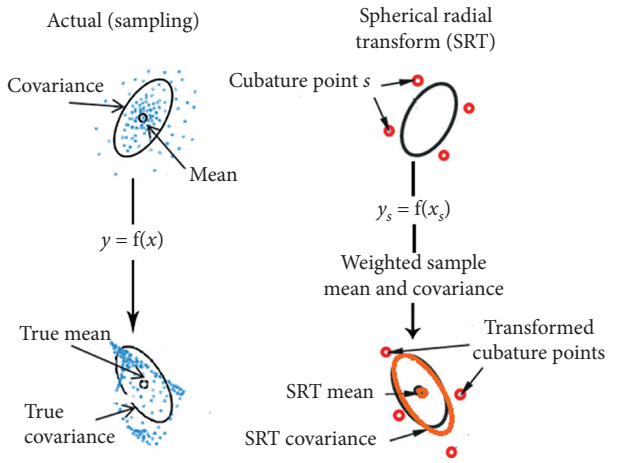

FIGURE 4: Illustration of the spherical-radical transformation for mean and covariance propagation.

$$\mathbf{Z}_{s,k|k-1} = \mathbf{h}\left(\mathbf{X}_{s,k|k-1}, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}\right);$$
$$\hat{\mathbf{z}}_{s,k|k-1} = \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{Z}_{s,k|k-1}; \tag{9}$$

$$\mathbf{P}_{xz} = \sum_{s=1}^{2n} \frac{1}{2n} \left(\mathbf{X}_{s,k|k-1} - \hat{\mathbf{x}}_{s,k|k-1}\right)\left(\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1}\right)^T;$$
$$\mathbf{P}_{zz} = \sum_{s=1}^{2n} \frac{1}{2n} \left(\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1}\right)\left(\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1}\right)^T; \tag{10}$$

$$\mathbf{K}_k = \mathbf{P}_{xz}\left(\mathbf{P}_{zz} + \mathbf{R}_c\right)^{-1};$$
$$\hat{\mathbf{x}}_{k|k} = \mathbf{g}\left(\hat{\mathbf{x}}_{k|k-1}, \mathbf{K}_k\left(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}\right)\right); \tag{11}$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{zz} \mathbf{K}_k^T.$$

3.2.4. CKF-Based VIO's Error Accumulation Issue. After summarizing the main points of the CKF-based VIO design, this paper will analyze the error accumulation issue. For our purpose, we utilize KITTI dataset [39], collected in the city and residential areas. The grayscale image is obtained from the Point Grey Flea2 camera with 1392×512 pixel resolution, $90^\circ \times 35^\circ$ opening angle, and 10 Hz update rate. The IMU measurement is obtained from OXTS RT3003 6-axis L1L2 RTK with 0.02 m/0.1° resolution and 100 Hz update rate. We use synced and rectified data, where the distortion effect is removed from the camera image. The estimation is inevitably subject to the error accumulation (drift) like any other odometry estimate. Figures 5 and 6 present the VIO estimation when the vehicle travels in long distance. We can observe that the Root-Mean-Square Error (RMSE) is accumulated gradually. The estimation drifts from its real trajectory, which can be observed clearly in Figures 5 and 6. In Figure 5 with dataset 2011_09_30_0034, the VIO estimation drifts about 50 m after traveling 900 m. Similarly, the VIO estimation drifts about 45m after traveling 1600m with dataset 2011_09_30_0033 in Figure 6.

The CKF-based system has employed IMU data for the filter state prediction, and camera image for the filter correction. We can say that the drift mainly derives from the poor performance of the camera observation. The visual measurement update step is unable to correct the residuals completely and suppress the drift effectively [4]. The error accumulation can come from various sources:

(i) The camera resolution and calibration are limited to provide reliable measurements in some particular cases such as traveling too fast.

(ii) The monocular camera with front-looking configuration suffers limited depth perception.

(iii) The sensor-fusing algorithm is limited to capturing the uncertainties and eliminating the environment noise.

The first two issues (i)-(ii) are associated with the hardware configuration, while the last issue (iii) derives from the sensor-fusing technique. Additionally, we can observe that the accumulative errors grow unbounded in an assumed unlimited time. The rotation error, which is limited to the range $[-\pi, \pi]$, also contributes to the position error accumulation. In short-term consideration (about 1 second), the drift can be modeled as a linear motion to simplify the problem formation [28]. In long-term consideration, the drift will not grow linearly in the traveled distance. It can be treated as a random process and will increase or decrease depending on the errors in motion vectors [40]. Assuming that the modification of the hardware setup is not an optimal solution, this paper will address the issue through the sensor-fusing algorithm with two proposed solutions in the next sections.

4. Solution 1: Iterated Cubature Kalman Filter

This section presents the first proposed solution to address the error accumulation issue. We upgrade the filter design presented in Section 3 to perform multiple iterations of the visual measurement update step. This advanced computation helps to optimize the estimation and reduce the accumulative errors.

4.1. Iterated Measurement Update. In general, the estimated filter state $\hat{\mathbf{x}}_{k|k}$ is closer to the true filter state than the predicted filter state $\hat{\mathbf{x}}_{k|k-1}$. During the iteration, the estimated state $\hat{\mathbf{x}}_{k|k}^{(j)}$ at j^{th} iterate produces a better approximation to the filter true state than the estimated state $\hat{\mathbf{x}}_{k|k}^{(j-1)}$ at $(j-1)^{\text{th}}$ iterate. The iterated filter state correction is performed as (12):

$$\hat{\mathbf{x}}_{k|k}^{(j)} = \hat{\mathbf{x}}_{k|k}^{(j-1)} + \mathbf{K}_k^{(j)}\left(\mathbf{z}_k - \hat{\mathbf{z}}_k\right);$$
$$\mathbf{K}_k^{(j)} = \mathbf{P}_{xz}^{(j)}\left(\mathbf{P}_{zz}^{(j)} + \mathbf{R}_c\right)^{-1}; \tag{12}$$

$$\mathbf{P}_{k|k}^{(j)} = \mathbf{P}_{k|k}^{(j-1)} - \mathbf{K}_k^{(j)} \mathbf{P}_{zz}^{(j)} \left(\mathbf{K}_k^{(j)}\right)^T. \tag{13}$$

The iteration computes the positive-definite covariance matrices $\mathbf{P}_{k|k}^{(j)}$, $\mathbf{P}_{k|k}^{(j-1)}$, and $\mathbf{P}_{zz}^{(j)}$. For any $j = 1, 2, \ldots, \infty$,

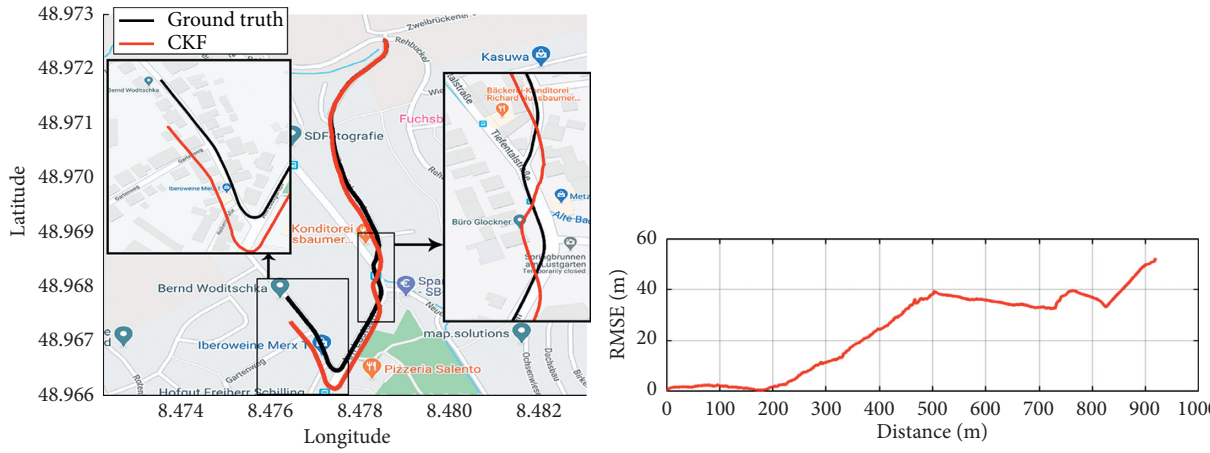

Figure 5: Example of VIO drift in KITTI dataset 2011_09_30_0034.

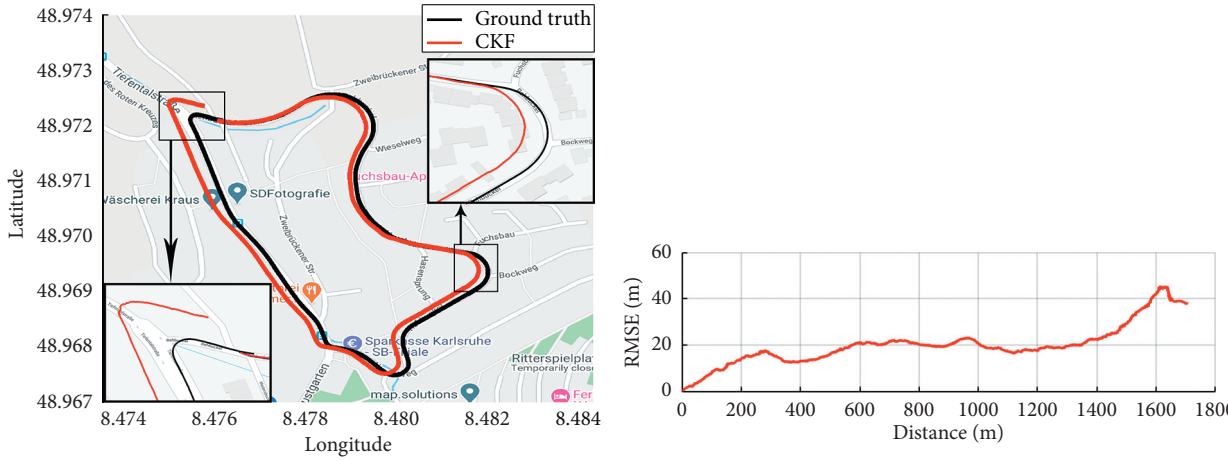

Figure 6: Example of VIO drift in KITTI dataset 2011_09_30_0033.

assuming $\lim_{j\longrightarrow\infty}\mathbf{K}_k^{(j)} \neq 0$, (13) reveals $\mathbf{P}_{k|k}^{(j)} < \mathbf{P}_{k|k}^{(j-1)}$. However, when each element of the matrix $\mathbf{P}_{k|k}^{(j)}$ is bounded, we also have $\lim_{j\longrightarrow\infty}\mathbf{P}_{k|k}^{(j)} = \lim_{j\longrightarrow\infty}\mathbf{P}_{k|k}^{(j-1)}$ [41]. Additionally, it is inferred from (13) that $\lim_{j\longrightarrow\infty}\mathbf{K}_k^{(j)} = 0$ violates the initial assumption ($\lim_{j\longrightarrow\infty}\mathbf{K}_k^{(j)} \neq 0$). Actually, the assumption cannot be hold during the iteration. Therefore, $\lim_{j\longrightarrow\infty}\mathbf{K}_k^{(j)} = 0$. When $\mathbf{K}_k^{(j)} \longrightarrow 0$ with $j > N$, $\mathbf{x}_{k|k}^{(j)} \longrightarrow \mathbf{x}_{k|k}^{(j-1)}$ and $\mathbf{P}_{k|k}^{(j)} \longrightarrow \mathbf{P}_{k|k}^{(j-1)}$. Consequently, the iteration achieves the global convergence.

$$\Delta\hat{\mathbf{x}}_{k|k}^{(j)} = \hat{\mathbf{x}}_{k|k}^{(j)} - \hat{\mathbf{x}}_{k|k}^{(j-1)}. \tag{14}$$

During the iteration, the state variation $\Delta\hat{x}_{k|k}^{(j)}$ is evaluated as (14). Obviously, a predetermined threshold ρ can be set for $\Delta\hat{x}_k^{(j)} < \rho$ to terminate the iteration. However, tuning the threshold ρ is challenging in dynamic noisy environments while the optimal value is not guaranteed to be in the likelihood surface. This limitation gives rise to the development of an alternative approach to terminate the iteration. Having $\mathbf{x}_{k|k} \sim N(\hat{\mathbf{x}}_k^{(j)}, \mathbf{P}_{k|k}^{(j)})$, $\mathbf{z}_k \sim N(\hat{\mathbf{z}}_k, \mathbf{R}_k)$, and $\Delta\mathbf{z}_k^{(j)} = \mathbf{z}_k - \mathbf{z}_k^{(j)}$, a cost function is evaluated at j^{th} iterate:

$$J\left(\mathbf{x}_k^{(j)}\right) = \left(\Delta\hat{\mathbf{x}}_{k|k}^{(j)}\right)^T\left(\mathbf{P}_{k|k}^{(j)}\right)^{-1}\Delta\hat{\mathbf{x}}_{k|k}^{(j)} + \left(\Delta\mathbf{z}_k^{(j)}\right)^T\mathbf{R}_k^{-1}\Delta\mathbf{z}_k^{(j)}. \tag{15}$$

It is complicated and impractical to determine the minimum value of the cost function J, which reveals the MLE of $\mathbf{x}_k$ and $\mathbf{z}_k$ [42]. Instead, we can observe that the inequality condition (16) is always satisfied during the optimization process. In other words, $J(\mathbf{x}_k^{(j)})$ is closer to the maximum likelihood surface than $J(\mathbf{x}_k^{(j-1)})$. Hence, $\mathbf{x}_k^{(j)}$ is a more accurate approximation to MLE than $\mathbf{x}_k^{(j-1)}$.

$$J\left(\mathbf{x}_k^{(j)}\right) < J\left(\mathbf{x}_k^{(j-1)}\right). \tag{16}$$

The inequality is utilized to terminate the iteration. To apply for the cubature measurement update, the inequality condition is rewritten with $\mathbf{P}_{k|k}^{(j)} = \mathbf{S}_{k|k}^{(j-1)}\mathbf{S}_{k|k}^{(j-1)}$ as follows [41]:

$$\begin{aligned}&\left(\Delta\hat{\mathbf{x}}_{k|k}^{(j)}\right)^T\left(\mathbf{S}_{k|k}^{(j-1)}\mathbf{S}_{k|k}^{(j-1)}\right)^{-1}\Delta\hat{\mathbf{x}}_{k|k}^{(j)} + \left(\Delta\mathbf{z}_k^{(j)}\right)^T\mathbf{R}_k^{-1}\Delta\mathbf{z}_k^{(j)}\\ &<\left(\Delta\mathbf{z}_k^{(j-1)}\right)^T\mathbf{R}_k^{-1}\Delta\mathbf{z}_k^{(j-1)}.\end{aligned} \tag{17}$$

4.2. The System Architecture of Iterated CKF. To summarize, the computation of iterated CKF is presented in Algorithm 1. Figure 7 presents the system architecture design implementing iterated CKF. The IMU data is employed for the filter state and

(1) Initialize the filter state and covariance matrix;
(2) **for** $k \in (1, \ldots, \infty)$ **do**
(3) Use IMU data to predict the nominal filter state $\hat{\mathbf{x}}_k$ with 4^{th} order Runge-Kutta numerical integration;
(4) Calculate $\mathbf{F}_d$ and $\mathbf{Q}_d$
(5) Compute the propagated state covariance $\mathbf{P}_{k|k-1}$;
(6) **if** *New Image* $\mathbf{I}_k$ **then**
(7) $I_k \longrightarrow I_3$. Perform feature detection for I_3 and match these features with features $\{m_1\}$ and $\{m_2\}$ to have feature tracking $\{m_1\} \leftrightarrow \{m_2\} \leftrightarrow \{m_3\}$;
(8) Factorize: $\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1}\mathbf{S}_{k|k-1}^T$;
(9) **while** ((*Condition* (17) *is satisfied*) **and** $(j < N)$) **do**
(10) Calculate the variation rate $\Delta\mathbf{x}_{k|k}^{(j)}, \Delta\mathbf{z}_{k|k}^{(j)}$ using (14);
(11) Generate Cubature points using (8);
(12) Compute innovation covariance matrix and the predicted measurement: (9) and (10);
(13) Compute the new filter state and covariance matrix: (12) and (13);
(14) **end**
(15) State transition and rearrange the covariance matrix with respective camara poses;
(16) **end**
(17) **end**

Algorithm 1: Iterated CKF algorithm.

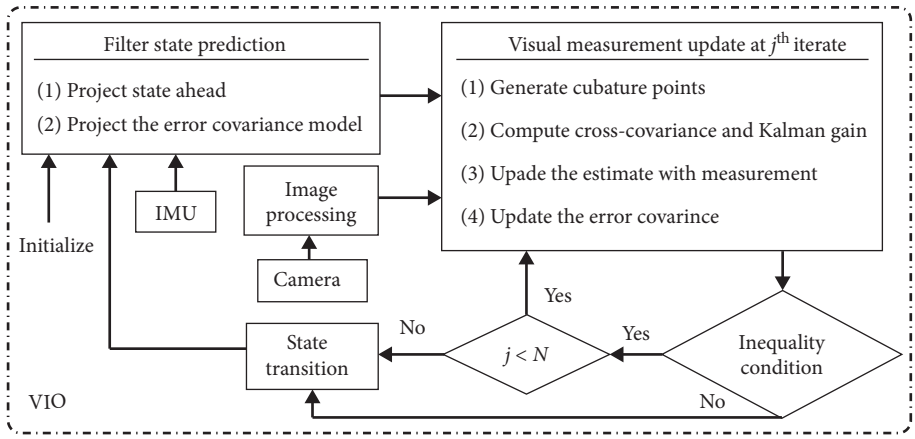

Figure 7: The system architecture implementing iterated CKF with j iterations.

covariance propagation. The camera observation is used for visual measurement update. The block of image processing executes the visual front-end computation. The visual measurement update conducts j^{th} iterate under the inequality condition (17) and maximum iteration $j < N$. If only one iteration is performed, the iterated CKF becomes the CKF. The state transition block guarantees the appropriate filter state and image transition after finishing the measurement update.

4.3. Experimental Validation of Iterated CKF. We also use KITTI dataset for experimental validation because it allows verifying the system performance in long-term operation with long-distance traveling. It also helps to observe the impact of the accumulative errors as well as the effect of implementing each solution. The estimates of these filters are presented in Google™ Maps, where we also provide multiple zoom-in images for comparison. The evaluation criteria are RMSE and the rotation error. The ground-truth data is obtained from GPS/IMU inertial navigation system data.

Figures 8 and 9 present the experiment with KITTI dataset 2011_09_30_0034. The vehicle travels about 900 m. In this case, the iterated CKF has superior performance than the other filters. The CKF and UKF produce accurate estimates within the first distance of 200 m. The employment of the optimization process has effectively decreased the position estimation errors and minimized the accumulative errors. In Figure 8, we only observe the drift in the performance of iterated CKF after 800 m. The rotation estimates of these filters have similar accuracy in Figure 9. The iterated CKF continuously updates the bias of accelerometer and gyroscope at each time instant as in Figure 10. The IMU estimation with only the integration of IMU data is not able to produce reliable estimations and track the vehicle trajectory. The integration with the camera in VIO system has improved the quality of the estimation.

Figures 11 and 12 present the experiment with KITTI dataset 2011_09_30_0033. The vehicle travels in a loop about ~1800 m. Similarly, the employment of iteration helps to improve the estimation effectively despite the computational

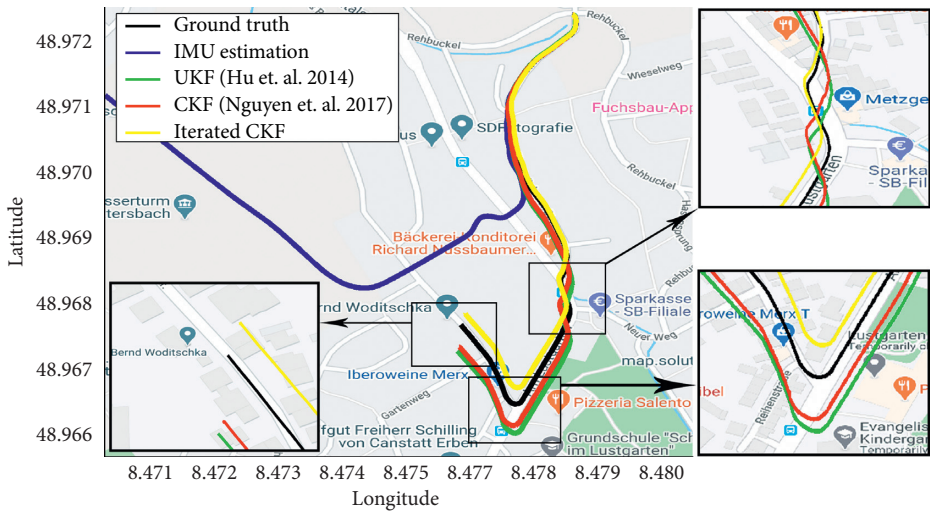

Figure 8: Experimental results presented in Google Maps for 2011_09_30_0034.

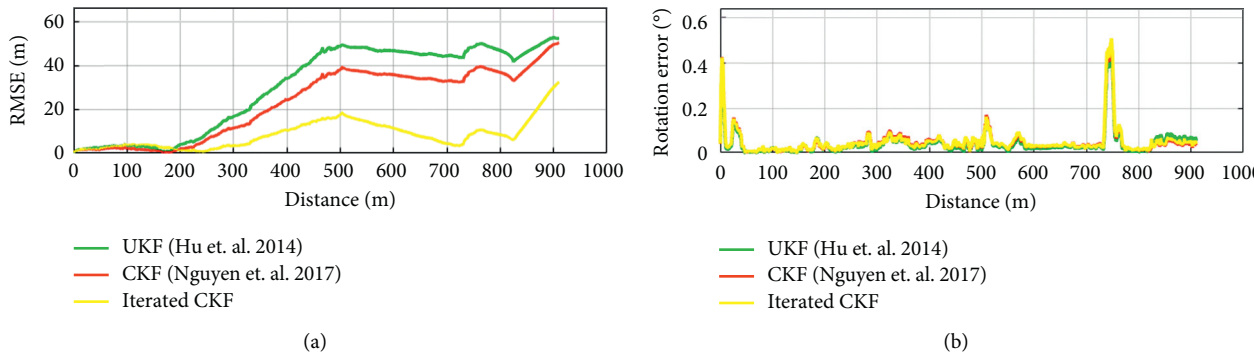

Figure 9: RMSE (a) and rotation error (b) in the experiment with dataset 2011_09_30_0034.

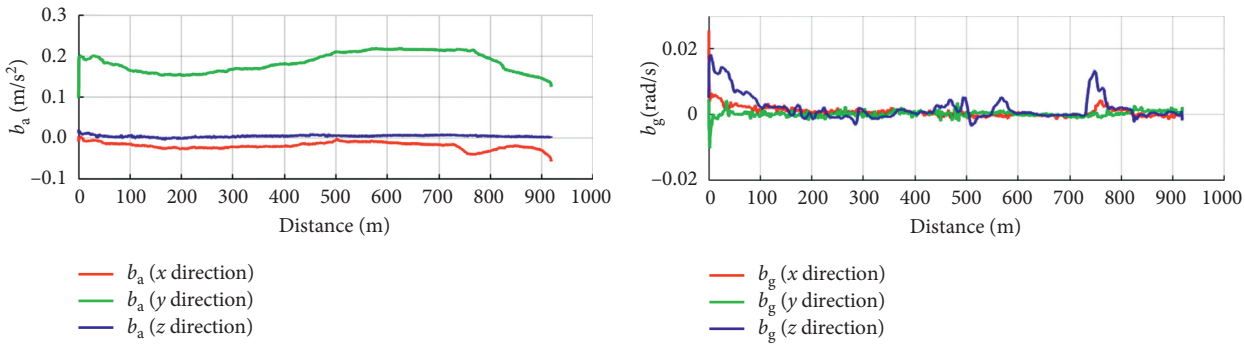

Figure 10: Iterated CKF estimation of accelerometer bias (a) and gyroscope bias (b) with dataset 2011_09_30_0034.

cost. Considering the rotation errors in Figure 12, all three filters have almost similar accuracy. The video demonstration of this solution can be located at youtu.be/-8SWh-cy-Ck.

5. Solution 2: Pseudorange Measurements

This section presents the second proposed solution to address the error accumulation. In solution 2, additional pseudorange measurements are tightly integrated with VIO, which will bound the estimation error and correct the positional drifts. The pseudorange measurement can be established by the wireless transmission between anchor and tag units. The tag unit is mounted on the vehicle and communicates with multiple anchor units, which are installed rigidly at known locations in the environment. The vehicle can be passive [43] or active [30] in the communication process, depending on how many vehicles are employed. Each pseudorange measurement can be modeled as in

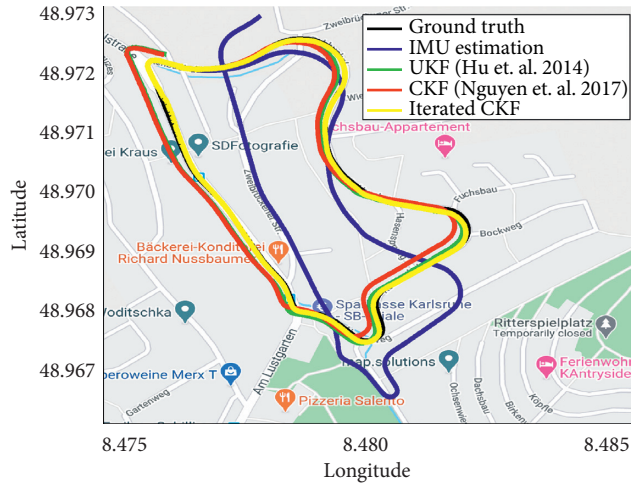

Figure 11: Experimental results presented in Google Maps for dataset 2011_09_30_0033.

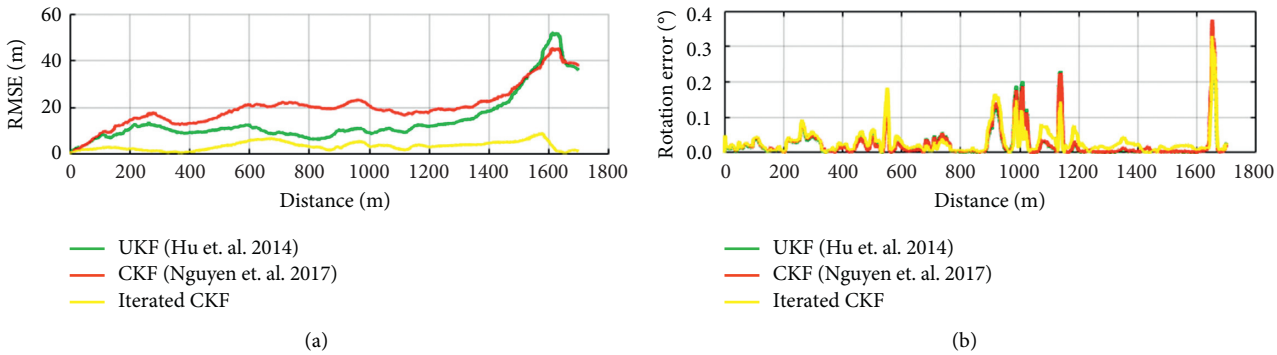

Figure 12: RMSE (a) and rotation error (b) in the experiment with dataset 2011_09_30_0033.

$$z_{r,d} = \left\| {}^{G}\mathbf{p}_{a,d} - {}^{G}\mathbf{p}_{I} \right\|_2 + \zeta_d \beta, \tag{18}$$

where ${}^{G}\mathbf{p}_{a,d}$ is the position of the d^{th} anchor, ${}^{G}\mathbf{p}_{I}$ is the current position of the vehicle, β is a bias of range error model, ζ_d is the coefficient describing the influence of β on the pseudorange measurement, and $\|\cdot\|_2$ is the Euclidean distance. Considering the real-time implementation, multiple hardware modules can be employed such as Decawave [44] or Time Domain's P410 UWB module [45].

5.1. Sequential-Sensor-Update Approach. We set up multiple anchors along the vehicle trajectory as in Figure 13. In reality, multiple pseudorange measurements can enable the vehicle to self-localize by using either Time of Arrival (TOA) or Time Difference of Arrival (TDOA) measurements. In general, that self-localization system requires at least four available ranging measurements for accurate estimation [31, 43]. However, if the vehicle is traveling long distance, it is difficult to always receive enough ranging measurements to perform self-localization. We consider here the case where the ranging measurements are only used to supplement the VIO.

We can synthesize all measurements (i.e., visual and ranging measurements) into a single composite group sensor with only one measurement model and similarly apply CKF for the estimation. However, this group sensor approach assumes that all sensors have similar update rates and that measurements are always available. This assumption does not satisfy our system configuration when the

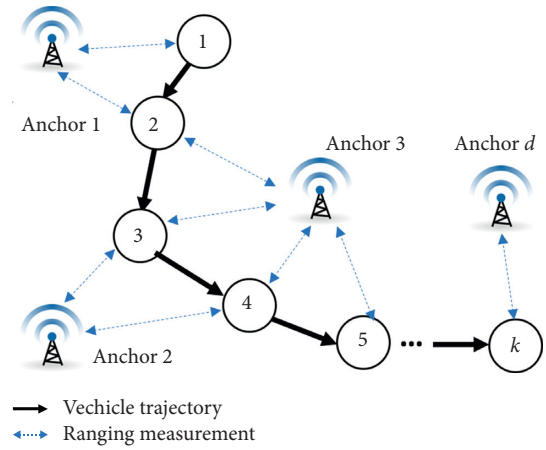

Figure 13: Illustration of wireless communication process with minimum pseudorange measurements.

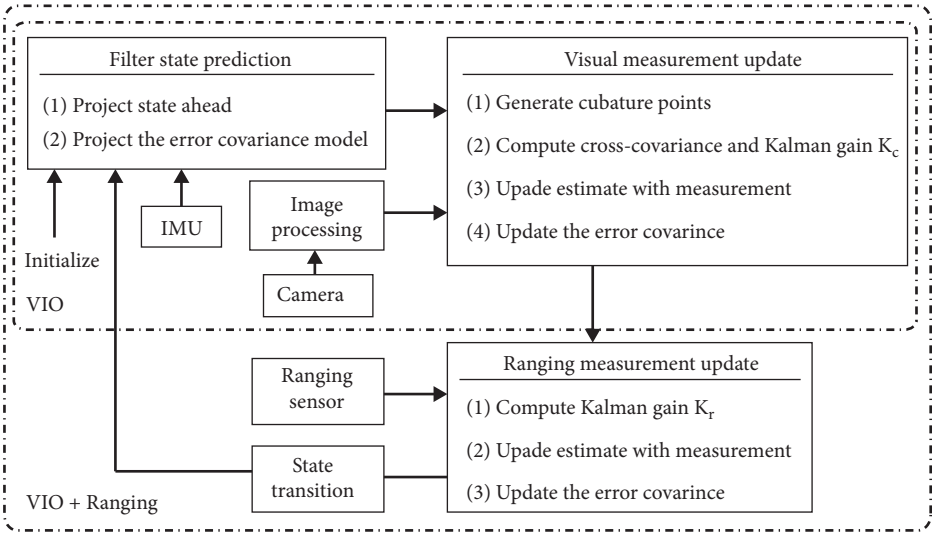

Figure 14: System architecture for integrating pseudorange measurement.

camera and ranging sensors operate independently. The number of available ranging sensors may vary depending on the communication process. As a result, the sequential-sensor method will be applied in this study. The sequential-sensor-update approach considers each sensor's observation as a separate and independent realization. Each sensor will operate following a specific observation model, which can be incorporated into the filter operation in a sequential manner [46]. A set of all observations made by the camera up to time k will be denoted by $\mathbf{Z}_c^k \triangleq \{z_c(1), z_c(2), \ldots, z_c(k)\}$; a set of all observations made by the pseudorange sensor up to time k will be denoted by $\mathbf{Z}_r^k \triangleq \{z_r(1), z_r(2), \ldots, z_r(k)\}$; hence, the set of all observations made by two sensors (camera and pseudorange) up to time k is constructed by $\mathbf{Z}_{c,r}^k \triangleq \{\mathbf{Z}_c^k \cup \mathbf{Z}_r^k\}$.

Figure 14 presents the system architecture when integrated with single pseudorange measurement. It can be extended for multiple pseudorange measurements. Notably, the VIO system is the principle module while pseudorange sensory system is secondary. These two systems are independent, where the pseudorange measurement update cannot intervene in the VIO operation. After performing visual measurement update, if the ranging measurement is not available for some reason, the system will proceed to the state transition. This property helps to sustain trajectory tracking even in the case where there is no communication with any anchor.

The matrix $\mathbf{H}_{r,k} = (\partial \mathbf{h}/\partial \mathbf{x})|_{\hat{\mathbf{x}}_{k|k-1}}$ describes how the filter states are mapped to the pseudorange measurement outputs, while it is computed by applying first-order Taylor series approximations to the pseudorange model. Meanwhile, the filter state includes three poses at times k, $k-1$, and $k-2$, which are guaranteed to satisfy the constraint of trifocal tensor geometry and epipolar geometry. Consequently, the pseudorange measurement model will be reconstructed so that it also corrects three poses at each time step:

$$\mathbf{z}_{r,k} = \begin{bmatrix} z_{r,k} & z_{r,k-1} & z_{r,k-2} \end{bmatrix}^T. \tag{19}$$

In the following equations, "r" denotes the ranging measurement while "c" denotes the visual measurement using camera. The filter state and covariance updated by the visual measurement are presented as $\hat{\mathbf{x}}_{c,k|k}$ and $\mathbf{P}_{c,k|k}$. When a new ranging measurement arrives, the system will use it to perform the filter update step following the sequential-sensor-update approach [46, 47]. This approach also helps to handle the asynchronous data. The pseudorange innovation is computed as (20). The Kalman gain $\mathbf{K}_{r,k}$ is calculated with variance $\mathbf{R}_r = \mathrm{diag}[v_r, v_r, v_r]$ as in (21). Then the filter state $\hat{\mathbf{x}}_{cr,k|k}$ and covariance $\mathbf{P}_{cr,k|k}$ corrected by both measurements can be computed as in (22).

$$\tilde{z}_{r,k} = \mathbf{z}_{r,k} - \mathbf{H}_{r,k}\hat{x}_{c,k|k}; \tag{20}$$

$$\mathbf{K}_{r,k} = \frac{\mathbf{P}_{c,k|k}\mathbf{H}_{r,k}^T}{\mathbf{H}_{r,k}\mathbf{P}_{c,k|k}\mathbf{H}_{r,k}^T + \mathbf{R}_r}; \tag{21}$$

$$\begin{aligned} \hat{\mathbf{x}}_{cr,k|k} &= \hat{\mathbf{x}}_{c,k|k} + \mathbf{K}_{r,k}\tilde{\mathbf{z}}_{r,k}; \\ \mathbf{P}_{cr,k|k} &= \mathbf{P}_{c,k|k} - \mathbf{K}_{r,k}\mathbf{H}_{r,k}\mathbf{P}_{c,k|k}. \end{aligned} \tag{22}$$

The pseudorange sensor is employed to supplement VIO. It is important to identify and reject pseudorange measurement outliers before fusing with VIO. Mahalanobis distance d_k of innovation covariance $\mathbf{S}_{r,k}$ and innovation $\tilde{\mathbf{z}}_{r,k}$ is measured to form the validation measurement gate, which defines the region in the pseudorange measurement space where valid measurements can be found. Any measurement outside that region is considered as an outlier and will not be integrated with VIO. The gate threshold γ_G is used to reject the pseudorange outliers. The innovation covariance $\mathbf{S}_{r,k}$ is computed before evaluating the validation gate as in

$$\begin{aligned} d_k^2 &= \tilde{\mathbf{z}}_{r,k}^T \mathbf{S}_{r,k} \tilde{\mathbf{z}}_{r,k} \leq \gamma_G; \\ \mathbf{S}_{r,k} &= \mathbf{H}_{r,k}\mathbf{P}_{c,k|k}\mathbf{H}_{r,k}^T + \mathbf{R}_{r,k}. \end{aligned} \tag{23}$$

5.2. Experimental Validation of Solution 2. For comparison with solution 1, we also utilize KITTI dataset to validate

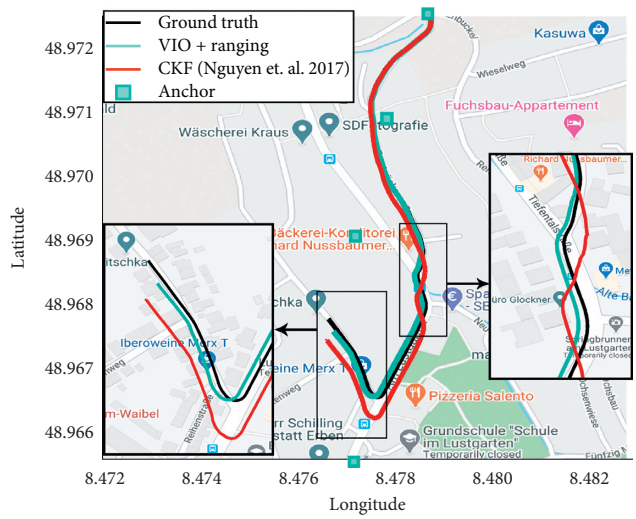

Figure 15: Position estimation result of dataset 2011_09_30_0034 presented on Google Maps.

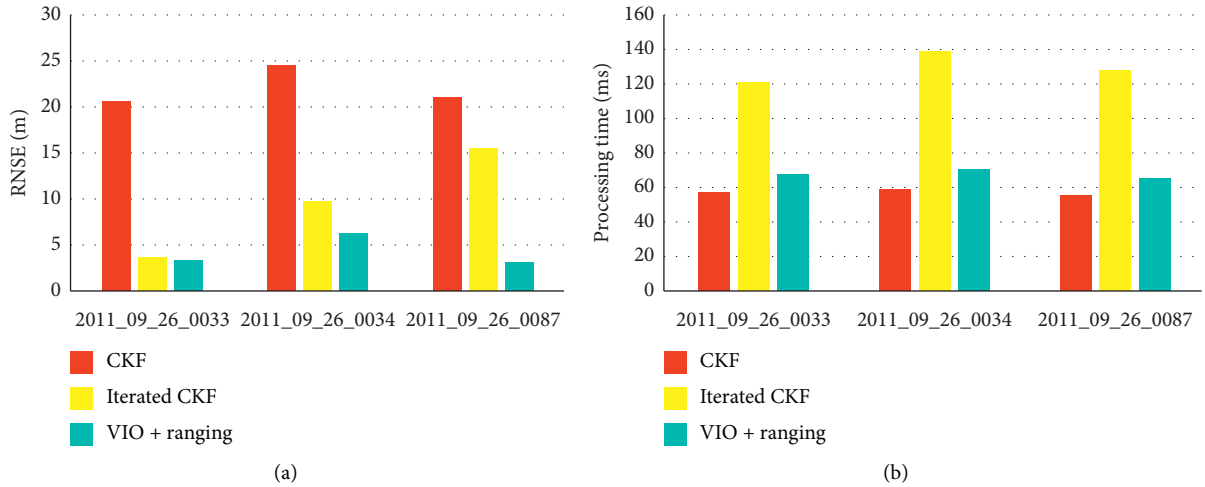

Figure 16: Average RMSE (a) and processing time (b) of position estimation between these solutions.

solution 2. We simulate the ranging system using GPS data. In Figure 15, multiple anchors are presented as cyan square marker. We denote the estimation of solution 2 as VIO-+Ranging. Figures 15 and 16 shows the improvement of estimation accuracy. The use of additional pseudorange sensors helps to bound the estimation error over long-term operation. Additionally, in these zoom-in images of Figure 15, we can observe some specific locations where the vehicle is estimated off the road. The system does not consider the issue of obstacle avoidance, which can be solved using laser-range sensor such as LIDAR.

6. Discussion and Conclusion

This section will discuss the pros and cons of the two proposed solutions, which helps to determine the scenario application of each in terms of system accuracy and hardware implementation. Solution 1 implements iterated CKF, which optimizes the latest filter state and covariance during the measurement update. Solution 2 employs the pseudorange measurements to bound the estimation errors.

Estimation accuracy: The experimental results reveal how each solution can improve estimation accuracy. Figures 15 and 16 evaluates average RMSE of position estimation for each solution. The effectiveness of solution 1 depends on the termination criteria, which decide the number of iterations. Meanwhile, the placement of the pseudorange sensors will influence the number of required ranging corrections to bound the estimation errors, which in turn affects the outcome of solution 2. Two proposed solutions can be classified in the group of VIO filtering approaches. Future work will compare the proposed solutions with the smoothing approaches (e.g., [7, 17]).

Computational cost: 16 measures the average processing time between these solutions. Both approaches can reduce error at the cost of increasing the processing time. The implementation of the iterated CKF increases the processing time about by 125%, while for solution 2

the cost is reduced by 20%. The iterated CKF consumes more processing time than the VIO+Ranging due to the dimension of the measurement model. Solution 1 executes multiple iterations of the visual measurement update to optimize the estimation, while solution 2 only needs about 1–3 pseudorange measurements. The dimension of the feature-based measurement model is much larger than that of the pseudorange measurement model. Solution 1 increases the computational cost significantly, which may limit applicability to micro- and nanorobotic applications. To improve the computational efficiency, we can select a small subset of tracked landmark features, which in turn decreases the dimension of the visual measurement model. Alternatively, the use of specialized computing platforms (e.g., FPGA and GPU) can achieve parallel processing and speed up the VIO execution [48]. Discussions about selecting optimal hardware for implementation are beyond the scope of this paper.

Although both solutions allow the VIO estimation to operate for longer duration, the hardware/software requirement of each solution is another consideration for implementation. Solution 1 with MLE-based optimization does not require the installation of advanced optimization library such as Google's Ceres Solver [19] and CasADi [33]. It is feasible to use the same system configuration for implementation even with hardware constrained platforms. On the other hand, solution 2 requires the setup of multiple anchors along the vehicle trajectory. It cannot be applied to unknown environments.

Overall, both solutions allow the VIO to operate for a longer duration without using a map andexecuting the optimization of the entire trajectory. A VIO developer will consider these mentioned advantages and disadvantages of each solution, the possible hardware configuration, and budget limitation before deciding which solution to select for upgrading his available VIO system. In general, solution 2 is more suitable for large-scale navigation projects, while solution 1 is preferable for stand-alone self-localization projects.

Data Availability

The data used to support the findings of this study are available from the public scene datasets.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), C-CORE J. I. Clark Chair, Memorial University of Newfoundland (MUN), and RDC Ocean Industries Student Research Awards (5404-1774-101).

References

[1] C. Kanellakis and G. Nikolakopoulos, "Survey on computer vision for UAVs: current developments and trends," *Journal of Intelligent and Robotic Systems*, vol. 87, no. 1, pp. 141–168, 2017.

[2] F. Santoso, M. A. Garratt, and S. G. Anavatti, "Visual-inertial navigation systems for aerial robotics: sensor fusion and technology," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 260–275, 2017.

[3] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE, Roma, Italy, April 2007.

[4] T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "Developing computationally efficient nonlinear cubature Kalman filtering for visual inertial odometry," *Journal of Dynamic Systems, Measurement and Control*, vol. 141, no. 8, 2019.

[5] X. I. Wong and M. Majji, "Extended kalman filter for stereo vision-based localization and mapping applications," *Journal of Dynamic Systems, Measurement, and Control*, vol. 140, no. 3, 2018.

[6] S. Leutenegger, S. Lynen, M. Bosse, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[7] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," in *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, https://arxiv.org/pdf/1708.03852.pdf.

[8] F. Zheng, G. Tsai, Z. Zhang, S. Liu, C.-C. Chu, and H. Hu, "Trifo-VIO: robust and efficient stereo visual inertial odometry using points and lines," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3686–3693, Madrid, Spain, October 2018.

[9] S. Heo, J. Cha, and C. G. Park, "EKF-based visual inertial navigation using sliding window nonlinear optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 7, pp. 2470–2479, 2019.

[10] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Brisbane, Australia, May 2018.

[11] S. Zhao, F. Lin, K. Peng, X. Dong, B. M. Chen, and T. H. Lee, "Vision-aided estimation of attitude, velocity, and inertial measurement bias for UAV stabilization," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 81, no. 3-4, pp. 531–549, 2016.

[12] Odroid Platforms, http://www.hardkernel.com/main/products/prdt_info.php.

[13] D. G. Kottas and S. I. Roumeliotis, "An iterative kalman smoother for robust 3D localization on mobile and wearable devices," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Seattle, WA, USA, May 2015.

[14] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[15] M. Bucolo, A. Buscarino, C. Famoso, L. Fortuna, and M. Frasca, "Control of imperfect dynamical systems," *Nonlinear Dynamics*, vol. 98, no. 4, pp. 2989–2999, 2019.

[16] Y. Xiao, X. Ruan, J. Chai, X. Zhang, and X. Zhu, "Online IMU self-calibration for visual-inertial systems," *Sensors*, vol. 19, no. 1624, pp. 1–26, 2019.

[17] R. Mur-Artal and J. D. Tardos, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[18] R. Mur-Artal, J. M. M. Montiel, and J. D. Tards, "Orb-slam: a versatile and accurate monocular slam," *IEEE Transactions on Robotics*, vol. 31, no. 5, 2015.

[19] S. Agarwal and K. Mierle, "Ceres solver," 2018, http://ceres-solver.org.

[20] D. Galvez-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.

[21] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[22] V. Peretroukhin, L. Clement, and J. Kelly, "Reducing drift in visual odometry by inferring sun direction using a Bayesian convolutional neural network," in *Proceedings of the IEEE International Conference on Robotics & Automation*, Singapore, May 2017.

[23] D. Caruso, A. Eudes, M. Sanfourche, and D. Vissi, "Robust indoor/outdoor navigation through magneto-visual-inertial optimization-based estimation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, Canada, September 2017.

[24] P. Arena, A. Basile, M. Bucolo, and L. Fortuna, "An object oriented segmentation on analog CNN chip," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, no. 7, pp. 837–846, 2003.

[25] Y. Ma, Y. Cao, S. Vrudhula, and J.-S. Seo, "Optimizing the convolution operation to accelerate deep neural networks on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 7, pp. 1354–1367, 2018.

[26] A. Ardakani, C. Condo, M. Ahmadi, and W. J. Gross, "An architecture to accelerate convolution in deep neural networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 4, pp. 1349–1362, 2018.

[27] T. A. Johansen, J. M. Hansen, and T. I. Fossen, "Nonlinear observer for tightly integrated inertial navigation aided by pseudo-range measurements," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 139, no. 1, 2017.

[28] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Zhuhai, China, May 2015.

[29] B. Claus, J. H. Kepper, S. Suman, and J. C. Kinsey, "Closed-loop one-way-travel-time navigation using low-grade odometry for autonomous underwater vehicles," *Journal of Field Robotics*, vol. 35, no. 4, pp. 421–434, 2018.

[30] M. W. Mueller, M. Hamer, and R. D. Andrea, "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Seattle, WA, USA, May 2015.

[31] C. Wang, H. Zhang, T.-M. Nguyen, and L. Xie, "Ultra-wideband aided fast localization and mapping system," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, Canada, September 2017.

[32] T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "Developing moving horizon estimation based ranging measurement for supporting vision-aided inertial navigation system," in *Proceedings of the ASME 2017 Dynamic Systems and Control Conference*, Tysons, VA, USA, October 2017.

[33] Casadi Library, https://github.com/casadi/casadi/wiki.

[34] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3d attitude estimation: a tutorial for quaternion algebra," Technical Report, University of Minnesota, Department of Computing Science and Engineering, Minneapolis, MN, USA, 2005.

[35] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011.

[36] K. Sun, K. Mohta, B. Pfrommer et al., "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics And Automation Letters*, vol. 3, no. 2, 2018.

[37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, UK, 2004.

[38] I. Arasaratnam and S. Haykin, "Cubature kalman filters," *IEEE Transactions on Automatic Control*, vol. 54, no. 6, 2009.

[39] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: the KITTI dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[40] H. Liu, R. Jiang, W. Hu, and S. Wang, "Navigation drift analysis for visual odometry navigation," *Computing and Informatics*, vol. 33, 2014.

[41] R. Zhan and J. Wan, "Iterated unscented Kalman filter for passive target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, pp. 1155–1163, 2007.

[42] J. Mu and Y.-L. Cai, "Iterated cubature Kalman filter and its application," in *Proceedings of the IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*, Kunming, China, March 2011.

[43] A. Ledergerber, M. Hamer, and R. D. Andrea, "A robot self-localization system using one-way UltraWideband communication," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3131–3137, Hamburg, Germany, September 2015.

[44] decaWave DWM1000, http://www.decawave.com.

[45] Time Domain's p-410 uwb Module, http://www.timedomain.com/p400-mrm.php.

[46] H. Durrant-Whyte, *Multi Sensor Data Fusion: Course Notes*, Australian Centre for Field Robotics, The University of Sydney, Sydney, Australia, 2001.

[47] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: a review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013.

[48] Z. Zhang, A. Suleiman, L. Carlone, V. Sze, and S. Karaman, "Visual-inertial odometry on chip: an algorithm-and-hardware co-design approach," in *Proceedings of the Robotics: Science and Systems XIII*, Cambridge, MA, USA, July 2017.