Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight

Ke Sun, Kartik Mohta, Bernd Pfrommer, Michael Watterson, Sikang Liu, Yash Mulgaonkar, Camillo J. Taylor, and Vijay Kumar

***Abstract*—In recent years, vision-aided inertial odometry for state estimation has matured significantly. However, we still encounter challenges in terms of improving the computational efficiency and robustness of the underlying algorithms for applications in autonomous flight with microaerial vehicles, in which it is difficult to use high-quality sensors and powerful processors because of constraints on size and weight. In this letter, we present a filter-based stereo visual inertial odometry that uses the multistate constraint Kalman filter. Previous work on the stereo visual inertial odometry has resulted in solutions that are computationally expensive. We demonstrate that our stereo multistate constraint Kalman filter (S-MSCKF) is comparable to state-of-the-art monocular solutions in terms of computational cost, while providing significantly greater robustness. We evaluate our S-MSCKF algorithm and compare it with state-of-the-art methods including OKVIS, ROVIO, and VINS-MONO on both the EuRoC dataset and our own experimental datasets demonstrating fast autonomous flight with a maximum speed of 17.5 m/s in indoor and outdoor environments. Our implementation of the S-MSCKF is available at https://github.com/KumarRobotics/msckf_vio.**

***Index Terms*—Localization, aerial systems: perception and autonomy, SLAM.**

I. Introduction

ACCURATE and robust state estimation is of crucial importance for robot autonomy and in particular for micro aerial vehicles (MAVs), where correct pose estimation is essential for stabilizing the robot in the air.

The solution of combining visual information from cameras and measurements from an Inertial Measurement Unit (IMU), usually referred to as Visual Inertial Odometry (VIO), is popular because it can perform well in GPS-denied environments and, compared to lidar based approaches, requires only a small and lightweight sensor package, making it the preferred technique for MAV platforms.

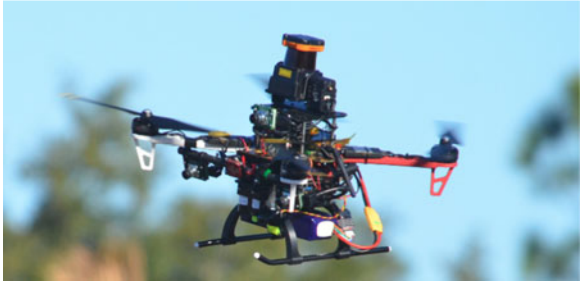


Fig. 1. The 3-kg FALCON robot with an onboard Intel NUC5i7RYH computer, synchronized stereo cameras and IMU, a laser scanner, and a downward facing lidar. Note that only the stereo cameras and the IMU are used for state estimation.

In scenarios such as search and rescue or first response, MAVs have to operate in a wide range of environments that pose challenges to VIO algorithms such as drastically varying lighting conditions, uneven illumination, low texture scenes, and abrupt changes in attitude due to wind gusts or aggressive maneuvering. Thus the VIO not only has to be accurate, it must also be robust.

In order to achieve full autonomy, all software components, from low-level sensor drivers to high-level planning algorithms, have to run onboard in real-time on a computer with similar computational power to a laptop because of the limited payload of a MAV such as the one shown in Fig. 1. The requirement to share onboard resources with other components puts additional pressure on the VIO algorithm to be as efficient as possible and importantly, to not produce excessive intermittent spikes in CPU consumption.

In this letter, we propose a filter-based stereo VIO solution to address such challenges, mostly because they are generally more computationally efficient than competing optimization-based methods. Among the filtering approaches, we choose as a starting point the state-of-art MSCKF [1]–[3] algorithm for its accuracy and consistency. A stereo configuration is preferred over the recently more popular monocular solutions because of its robustness to different environments and motion. Contradicting the widely held belief that stereo vision-based estimation incurs much higher compute cost than monocular approaches, we demonstrate that the proposed stereo VIO is able to achieve similar or even higher efficiency than state-of-art monocular solutions. Our main contributions are:

Manuscript received September 10, 2017; accepted December 27, 2017. Date of publication January 15, 2018; date of current version February 1, 2018. This letter was recommended for publication by Associate Editor H. Andreasson and Editor C. Stachniss upon evaluation of the reviewers' comments. This work was supported in part by the Defense Advanced Research Projects Agency under Grants HR001151626 and HR0011516850, by the U.S. Army Research Laboratory under Grant W911NF-08-2-0004, and by a NASA Space Technology Research Fellowship awarded to M. Watterson. *(Corresponding author: Ke Sun.)*

The authors are with the GRASP Lab, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: sunke@seas.upenn.edu; kmohta@seas.upenn.edu; pfrommer@seas.upenn.edu; wami@seas.upenn.edu; sikang@seas.upenn.edu; yashm@seas.upenn.edu; cjtaylor@seas.upenn.edu; kumar@seas.upenn.edu).

This letter has supplemental downloadable multimedia material available at http://ieeexplore.ieee.org, provided by the authors. The Supplemental Material contains the high-resolution video associated with the paper. This material is 46.1 MB in size.

Digital Object Identifier 10.1109/LRA.2018.2793349

- To the best of our knowledge, the presented work is the first open-source filter-based stereo VIO that can run onboard on a laptop-class computer without GPU acceleration.
- We provide detailed experimental comparisons between the proposed S-MSCKF and state-of-art open-source VIO solutions including OKVIS [4], ROVIO [5], and VINS-MONO [6] in accuracy, efficiency and robustness. The comparison is performed on both the EuRoC [7] dataset and the fast flight dataset using FALCON shown in Fig. 1.
- The fast flight dataset is publicly available at https://github.com/KumarRobotics/msckf_vio/wiki.

The rest of the letter is organized as follows: Section II discusses the related work. Section III presents the mathematical details of the proposed algorithm. In Section IV, we compare our work with different state-of-the-art works in VIO on different datasets and demonstrate the performance of the proposed S-MSCKF with a fully autonomous flight through unstructured and unknown environments. Finally, Section V concludes the letter.

II. Related Work

There is extensive literature on visual odometry, ranging from pure vision-based methods [8]–[12], loosely-coupled VIO solutions [13], [14], to the recently more popular tightly-coupled VIO solutions [1], [4], [5], [15]–[20] which will be the major focus of this letter.

Existing tightly-coupled VIO solutions can, in general, be classified into optimization-based (e.g., [4], [15], [16]) and filter-based approaches (e.g., [1], [5], [20]). Optimization-based methods obtain the optimal estimate by jointly minimizing the residual using the measurements from both IMU data and images. Most of these systems, such as [15], use sparse features obtained from images as measurements. Methods like this are also called indirect methods. Usenko *et al.* [16] and Forster *et al.* [17] propose a direct method which minimizes photometric error directly in order to exploit more information from the images. The literature shows that optimization-based approaches are able to achieve high accuracy. However, such methods require significant computational resources because of the iterative optimization process, although recent efficient solvers (e.g., [21], [22]) can be run in real time online.

In contrast, filter-based approaches, which generally use the Extended Kalman Filter (EKF) [1], or the Uncented Kalman Filter [19], are much more efficient while achieving accuracy comparable to optimization-based approaches. Huang *et al.* [23], Li *et al.* [2], and Hesch *et al.* [24] also propose the First Estimate Jacobian (FEJ) and the Observability Constraint (OC) to improve consistency of VIO in the filter framework, which in turn improves the estimation accuracy. In more recent work [5], [25], the direct method is used in the filter-based VIO framework to further improve accuracy and robustness.

Only a few VIO solutions are designed for stereo or multi-camera system [4], [16], [26], [27] compared to the vast amount of work on monocular systems. This can be attributed in part to costs associated with processing additional images and matching features. In [26], datasets are collected using stereo visual inertial configuration with cameras running at 6.25 Hz, which are then processed offline. The stereo VIO in [26] is more a proof of concept of IMU pre-integration and does not lend itself to practical implementation. Leutenegger *et al.* [4] propose a more complete optimization framework for multi-camera VIO that is able to run in real time. Usenko [16] introduces direct methods into stereo VIO in order to further improve accuracy. All three solutions are optimization-based approaches requiring powerful CPUs to operate in real time. More recently, Paul *et al.* [27] proposed a filter-based stereo VIO based on the square root inverse filter [18], which demonstrates the possibility of operating a stereo VIO online efficiently, even on a mobile device. Since the implementations of [16] and [18] are not open-sourced, they are not used for comparison in this letter.

III. Filter Description

In the description of the filter setup, we follow the convention in [1]. The IMU state is defined as,

$$\mathbf{x}_I = \begin{pmatrix} {}^I_G\mathbf{q}^\top & \mathbf{b}_g^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_a^\top & {}^G\mathbf{p}_I^\top & {}^I_C\mathbf{q}^\top & {}^I\mathbf{p}_C^\top \end{pmatrix}^\top$$

where the quaternion ${}^I_G\mathbf{q}$ represents the rotation from the inertial frame to the body frame. In our configuration, the body frame is set to be the IMU frame. The vectors ${}^G\mathbf{v}_I \in \mathbb{R}^3$ and ${}^G\mathbf{p}_I \in \mathbb{R}^3$ represent the velocity and position of the body frame in the inertial frame. The vectors $\mathbf{b}_g \in \mathbb{R}^3$ and $\mathbf{b}_a \in \mathbb{R}^3$ are the biases of the measured angular velocity and linear acceleration from the IMU. Finally the quaternion, ${}^I_C\mathbf{q}$ and ${}^I\mathbf{p}_C \in \mathbb{R}^3$ represent the relative transformation between the camera frame and the body frame. Without loss of generality, the left camera frame is used assuming the extrinsic parameters relating the left and right cameras are known. Using the true IMU state would cause singularities in the resulting covariance matrices because of the additional unit constraint on the quaternions in the state vector. Instead, the error IMU state, defined as,

$$\tilde{\mathbf{x}}_I = \begin{pmatrix} {}^I_G\tilde{\boldsymbol{\theta}}^\top & \tilde{\mathbf{b}}_g^\top & {}^G\tilde{\mathbf{v}}_I^\top & \tilde{\mathbf{b}}_a^\top & {}^G\tilde{\mathbf{p}}_I^\top & {}^I_C\tilde{\boldsymbol{\theta}}^\top & {}^I\tilde{\mathbf{p}}_C^\top \end{pmatrix}^\top$$

is used with standard additive error used for position, velocity, and biases (e.g., ${}^G\tilde{\mathbf{p}}_I = {}^G\mathbf{p}_I - {}^G\hat{\mathbf{p}}_I$). For the quaternions, the error quaternion $\delta\mathbf{q} = \mathbf{q} \otimes \hat{\mathbf{q}}^{-1}$ is related to the error state as,

$$\delta\mathbf{q} \approx \begin{pmatrix} \frac{1}{2}{}^G_I\tilde{\boldsymbol{\theta}}^\top & 1 \end{pmatrix}^\top$$

where ${}^G_I\tilde{\boldsymbol{\theta}} \in \mathbb{R}^3$ represents a small angle rotation. With such a representation, the dimension of orientation error is reduced to 3 enabling proper presentation of its uncertainty. Ultimately N camera states are considered together in the state vector, so the entire error state vector would be,

$$\tilde{\mathbf{x}} = \begin{pmatrix} \tilde{\mathbf{x}}_I^\top & \tilde{\mathbf{x}}_{C_1}^\top & \dots & \tilde{\mathbf{x}}_{C_N}^\top \end{pmatrix}^\top$$

where each camera error state is defined as,

$$\tilde{\mathbf{x}}_{C_i} = \begin{pmatrix} {}^{C_i}_G\tilde{\boldsymbol{\theta}}^\top & {}^G\tilde{\mathbf{p}}_{C_i}^\top \end{pmatrix}^\top$$

In order to maintain bounded computational complexity, some camera states have to be marginalized once the number of camera states reaches a preset limit. Discussions of how to choose camera states to marginalize can be found in Section III-D.

A. Process Model

The continuous dynamics of the estimated IMU state is,

$$
\begin{aligned}
I_G\dot{\hat{\mathbf{q}}} &= \frac{1}{2}\Omega(\hat{\boldsymbol{\omega}})^I_G\hat{\mathbf{q}}, \quad \dot{\hat{\mathbf{b}}}_g = \mathbf{0}_{3\times 1},\\
G\dot{\hat{\mathbf{v}}} &= C\left({}^I_G\hat{\mathbf{q}}\right)^\top \hat{\mathbf{a}} + {}^G\mathbf{g},\\
\dot{\hat{b}}_a &= \mathbf{0}_{3\times 1}, \quad {}^G\dot{\hat{\mathbf{p}}}_I = {}^G\hat{\mathbf{v}},\\
I_C\dot{\hat{\mathbf{q}}} &= \mathbf{0}_{3\times 1}, \quad {}^I\dot{\hat{\mathbf{p}}}_C = \mathbf{0}_{3\times 1}
\end{aligned}
\tag{1}
$$

where $\hat{\boldsymbol{\omega}} \in \mathbb{R}^3$ and $\hat{\mathbf{a}} \in \mathbb{R}^3$ are the IMU measurements for angular velocity and acceleration respectively with biases removed, i.e.,

$$\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}_m - \hat{\mathbf{b}}_g, \quad \hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a$$

Meanwhile,

$$\Omega\left(\hat{\boldsymbol{\omega}}\right) = \begin{pmatrix} -[\hat{\boldsymbol{\omega}}_\times] & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{pmatrix}$$

where $[\hat{\boldsymbol{\omega}}_\times]$ is the skew symmetric matrix of $\hat{\boldsymbol{\omega}}$. $C(\cdot)$ in (1) is the function converting quaternion to the corresponding rotation matrix. Based on (1), the linearized continuous dynamics for the error IMU state follows,

$$\dot{\tilde{\mathbf{x}}}_I = \mathbf{F}\tilde{\mathbf{x}}_I + \mathbf{G}\mathbf{n}_I \tag{2}$$

where $\mathbf{n}_I^\top = \left(\mathbf{n}_g^\top \; \mathbf{n}_{wg}^\top \; \mathbf{n}_a^\top \; \mathbf{n}_{wa}^\top\right)^\top$. The vectors $\mathbf{n}_g$ and $\mathbf{n}_a$ represent the Gaussian noise of the gyroscope and accelerometer measurement, while $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are the random walk rate of the gyroscope and accelerometer measurement biases. $\mathbf{F}$ and $\mathbf{G}$ are shown in Appendix A.

To deal with discrete time measurement from the IMU, we apply a 4th order Runge-Kutta numerical integration of (1) to propagate the estimated IMU state. To propagate the uncertainty of the state, the discrete time state transition matrix of (2) and discrete time noise covariance matrix need to be computed first,

$$
\begin{aligned}
\boldsymbol{\Phi}_k &= \boldsymbol{\Phi}(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}} \mathbf{F}(\tau)d\tau\right)\\
\mathbf{Q}_k &= \int_{t_k}^{t_{k+1}} \boldsymbol{\Phi}(t_{k+1}, \tau)\mathbf{G}\mathbf{Q}\mathbf{G}\boldsymbol{\Phi}(t_{k+1}, \tau)^\top d\tau
\end{aligned}
$$

where $\mathbf{Q} = \mathbb{E}\left[\mathbf{n}_I\mathbf{n}_I^\top\right]$ is the continuous time noise covariance matrix of the system. Then the propagated covariance of the IMU state is,

$$\mathbf{P}_{II_{k+1|k}} = \boldsymbol{\Phi}_k\mathbf{P}_{II_{k|k}}\boldsymbol{\Phi}_k^\top + \mathbf{Q}_k$$

By partioning the covariance of the whole state as,

$$\mathbf{P}_{k|k} = \begin{pmatrix} \mathbf{P}_{II_{k|k}} & \mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^\top & \mathbf{P}_{CC_{k|k}} \end{pmatrix}$$

the full uncertainty propagation can be represented as,

$$\mathbf{P}_{k+1|k} = \begin{pmatrix} \mathbf{P}_{II_{k+1|k}} & \boldsymbol{\Phi}_k\mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^\top\boldsymbol{\Phi}_k^\top & \mathbf{P}_{CC_{k|k}} \end{pmatrix}$$

When new images are received, the state should be augmented with the new camera state. The pose of the new camera state can be computed from the latest IMU state as,

$${}^C_G\hat{\mathbf{q}} = {}^C_I\hat{\mathbf{q}} \otimes {}^I_G\hat{\mathbf{q}}, \quad {}^G\hat{\mathbf{p}}_C = {}^G\hat{\mathbf{p}}_C + C\left({}^I_G\hat{\mathbf{q}}\right)^\top {}^I\hat{\mathbf{p}}_C$$

And the augmented covariance matrix is,

$$\mathbf{P}_{k|k} = \begin{pmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{pmatrix}\mathbf{P}_{k|k}\begin{pmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{pmatrix}^\top \tag{3}$$

where $\mathbf{J}$ is shown in Appendix B.

B. Measurement Model

Consider the case of a single feature f_j observed by the stereo cameras with pose $({}^{C_i}_G\mathbf{q}, {}^G\mathbf{p}_{C_i})$. Note that the stereo cameras have different poses, represented as $({}^{C_{i,1}}_G\mathbf{q}, {}^G\mathbf{p}_{C_{i,1}})$ and $({}^{C_{i,2}}_G\mathbf{q}, {}^G\mathbf{p}_{C_{i,2}})$ for left and right cameras respectively, at the same time instance. Although the state vector only contains the pose of the left camera, the pose of the right camera can be easily obtained using the extrinsic parameters from the calibration. The stereo measurement, $\mathbf{z}_i^j$, is represented as,

$$\mathbf{z}_i^j = \begin{pmatrix} u_{i,1}^j \\ v_{i,1}^j \\ u_{i,2}^j \\ v_{i,2}^j \end{pmatrix} = \begin{pmatrix} \frac{1}{{}^{C_{i,1}}Z_j} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \frac{1}{{}^{C_{i,2}}Z_j} \end{pmatrix}\begin{pmatrix} {}^{C_{i,1}}X_j \\ {}^{C_{i,1}}Y_j \\ {}^{C_{i,2}}X_j \\ {}^{C_{i,2}}Y_j \end{pmatrix} \tag{4}$$

Note that the dimension of $\mathbf{z}_i^j$ can be reduced to $\mathbb{R}^3$ assuming the stereo images are properly rectified. However, by representing $\mathbf{z}_i^j$ in $\mathbb{R}^4$, it is no longer required that the observations of the same feature on the stereo images are on the same image plane, which removes the necessity for stereo rectification. In (4), $\left({}^{C_{i,k}}X_j \; {}^{C_{i,k}}Y_j \; {}^{C_{i,k}}Z_j\right)^\top$, $k \in \{1, 2\}$, are the positions of the feature, f_j, in the left and right camera frame, $C_{i,1}$ and $C_{i,2}$, which are related to the camera pose by,

$$
\begin{aligned}
{}^{C_{i,1}}\mathbf{p}_j &= \begin{pmatrix} {}^{C_{i,1}}X_j \\ {}^{C_{i,1}}Y_j \\ {}^{C_{i,1}}Z_j \end{pmatrix} = C\left({}^{C_{i,1}}_G\mathbf{q}\right)\left({}^G\mathbf{p}_j - {}^G\mathbf{p}_{C_{i,1}}\right)\\
{}^{C_{i,2}}\mathbf{p}_j &= \begin{pmatrix} {}^{C_{i,2}}X_j \\ {}^{C_{i,2}}Y_j \\ {}^{C_{i,2}}Z_j \end{pmatrix} = C\left({}^{C_{i,2}}_G\mathbf{q}\right)\left({}^G\mathbf{p}_j - {}^G\mathbf{p}_{C_{i,2}}\right)\\
&= C\left({}^{C_{i,2}}_{C_{i,1}}\mathbf{q}\right)\left({}^{C_{i,1}}\mathbf{p}_j - {}^{C_{i,1}}\mathbf{p}_{C_{i,2}}\right)
\end{aligned}
$$

The position of the feature in the world frame, ${}^G\mathbf{p}_j$, is computed using the least square method given in [1] based on the current estimated camera poses. Linearizing the measurement model at the current estimate, the residual of the measurement can be approximated as,

$$\mathbf{r}_i^j = \mathbf{z}_i^j - \hat{\mathbf{z}}_i^j = \mathbf{H}_{C_i}^j\tilde{\mathbf{x}}_{C_i} + \mathbf{H}_{f_i}^j\,{}^G\tilde{\mathbf{p}}_j + \mathbf{n}_i^j \tag{5}$$

where $\mathbf{n}_i^j$ is the noise of the measurement. The measurement Jacobian $\mathbf{H}_{C_i}^j$ and $\mathbf{H}_{f_i}^j$ are shown in Appendix C.

By stacking multiple observations of the same feature f_j, we have,

$$\mathbf{r}^j = \mathbf{H}_{\mathbf{x}}^j \tilde{\mathbf{x}} + \mathbf{H}_f^j \, {}^G\tilde{\mathbf{p}}_j + \mathbf{n}^j$$

As pointed out in [1], since ${}^G\mathbf{p}_j$ is computed using the camera poses, the uncertainty of ${}^G\mathbf{p}_j$ is, therefore, correlated with the camera poses in the state. In order to ensure the uncertainty of ${}^G\mathbf{p}_j$ does not affect the residual, the residual in (5) is projected to the null space, $\mathbf{V}$, of $\mathbf{H}_f^j$, i.e.,

$$\mathbf{r}_o^j = \mathbf{V}^\top \mathbf{r}^j = \mathbf{V}^\top \mathbf{H}_{\mathbf{x}}^j \tilde{\mathbf{x}} + \mathbf{V}^\top \mathbf{n}^j = \mathbf{H}_{\mathbf{x},o}^j \tilde{\mathbf{x}} + \mathbf{n}_o^j \tag{6}$$

Based on (6), the update step of the EKF can be carried out in a standard way.

C. Observability Constraint

As has been shown in [2], [23], the EKF-based VIO for 6-DOF motion estimation has four unobservable directions corresponding to global position and rotation along the gravity axis, i.e., yaw angle. A naive implementation of EKF VIO will gain spurious information on yaw. This is due to the fact that the linearizing point of the process and measurement step are different at the same time instant.

There are different methods for maintaining the consistency of the filter, including the First Estimate Jacobian EKF (FEJ-EKF) [23], the Observability Constrained EKF (OC-EKF) [28], and Robocentric Mapping Filter [29]. In our implementation, OC-EKF is applied for two reasons as discussed in [23], i) unlike FEJ-EKF, OC-EKF does not heavily depend on an accurate initial estimation, ii) comparing to Robocentric Mapping Filter, camera poses in the state vector can be represented with respect to the inertial frame instead of the latest IMU frame so that the uncertainty of the existing camera states in the state vector is not affected by the uncertainty of the latest IMU state during the propagation step.

D. Filter Update Mechanism

Two types of delayed measurement updates are described in [1]. The measurement update step is executed when either the algorithm loses a feature or the number of camera poses in the state reaches the limit. In our implementation, we inherit the same update mechanism with modifications for real-time considerations. As suggested in [1], one third of the camera states are marginalized once the buffer is full, which can cause sudden jumps in computational load in real-time implementations. It is desired that one camera state is marginalized at each time step in order to average out the computation. However, removing the observation of a feature at one camera state is not practical in the MSCKF framework since the observation contains no information about the relative transformation between the camera states. Mathematically, this is due to the fact that the null space of $\mathbf{H}_{f_i}^j$ in (5) is a subspace of the null space of $\mathbf{H}_{C_i}^j$ (see Appendix D), which results in a trivial measurement model based on (5).

In our implementation, two camera states are removed every other update step. All feature observations obtained at the two camera states are used for measurement update. Note that because of the reason mentioned above, the two stereo measurements of the two camera states are only useful if they are of the same feature. It is understood that such frequent removal of camera states can cause some valid observations to be ignored. In practice, we found that the estimation performance is barely affected although fewer observations are used. To select the two camera states to be removed, we apply a keyframe selection strategy similar to the two-way marginalization method proposed in [30]. Based on the relative motion between the second latest camera state and its previous one, either the second latest or the oldest camera state is selected for removal. The selection procedure is executed twice to find two camera states to remove. Note that the latest camera state is always kept since it has the measurements for the newly detected features.

E. Image Processing Frontend

In our implementation, the FAST [31] feature detector is employed for its efficiency. Existing features are tracked temporally using the KLT optical flow algorithm [32]. It is shown in [27] that descriptor-based methods for temporal feature tracking are better than KLT-based methods in accuracy. In our experiments, we find that descriptor-based methods require much more CPU resource with small gain in accuracy, making such methods less favorable in our application. Note that we also use the KLT optical flow algorithm for stereo feature matching, which further saves computation compared to descriptor-based methods. Empirically, corner features with depths greater than 1m can be reliably matched across the stereo images using KLT tracking with a 20 cm baseline stereo configuration. Two types of outlier rejection procedure are implemented in the image processing frontend. A 2-point RANSAC is applied to remove outliers in temporal tracking. In addition, a circular matching similar to [33] is performed between the previous and current stereo image pairs to further remove outliers generated in the feature tracking and stereo matching steps.

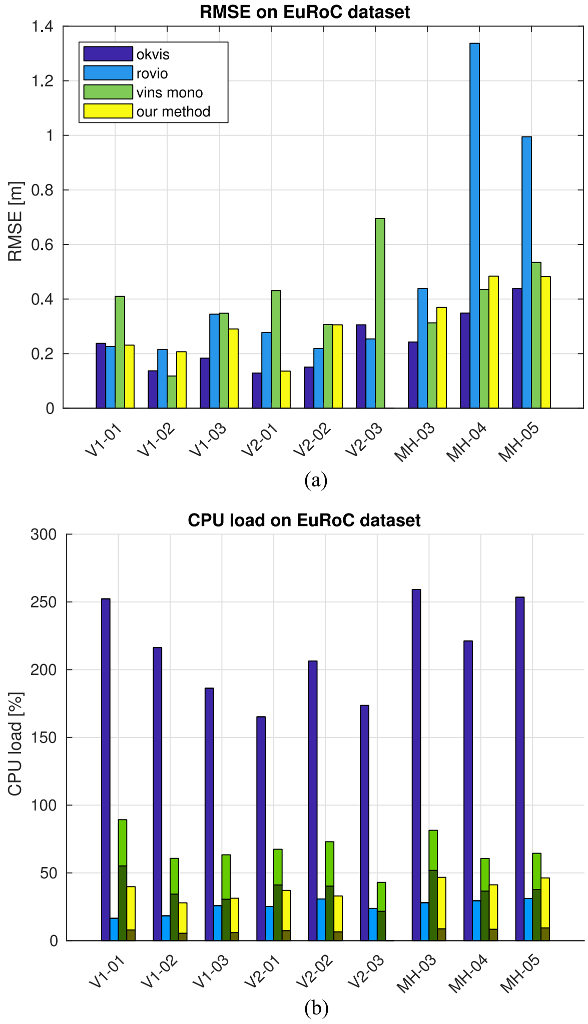
IV. Experiments

In order to evaluate the proposed method, three different kinds of experiments were performed. First, the proposed method was compared with state-of-art visual inertial odometry algorithms including OKVIS [4], ROVIO [5], and VINS-MONO [6] on the EuRoC dataset [7]. Second, we demonstrate the robustness of the proposed algorithm on high speed flights reaching maximum speeds of 17.5 m/s on a runway environment. In both of the experiments, the loop closure functionality of VINS-MONO is disabled in order to just compare the odometry of different approaches. Note that although all of the algorithms used in the comparison are capable of estimating extrinsic parameters between the IMU and camera frames online, the offline calibration parameters are provided in the experiments for optimal performance. Finally, we show a representative application of the proposed S-MSCKF in an experiment that combines estimation, with control and planning for autonomous flight in an unstructured and unknown environment which includes a warehouse, a wooded area and a runway.

A. EuRoC Dataset

The EuRoC datasets were collected with a VI sensor [34] on a MAV, which includes synchronized 20 Hz stereo images and 200 Hz IMU messages. The aggressive rotation and significant lighting change make the dataset challenging for vision-based state estimation. We compare our results on the EuRoC dataset with three representative VIO systems, OKVIS (stereo-optimization), ROVIO (monocular-filter), and VINS-MONO (monocular-optimization). Including the proposed method, the four visual inertial solutions are different combinations of monocular, stereo, filter-based, and optimization-based methods, which may provide insights into the pros and cons of the various approaches. For the monocular approaches, only the images from the left camera are used.

Fig. 2 shows the Root Mean Square Error (RMSE) and the average CPU load of the four methods on the Dataset. The CPU load is measured with NUC6i7KYK equipped with quad-core i76770HQ 2.6Hz. The proposed method does not work properly on `V2_03_difficult`. The reason is that we use the KLT optical flow algorithm [32] for both temporal feature tracking and stereo matching to improve efficiency. The continuous inconsistency in brightness between the stereo images in `V2_03_difficult` causes failures in the stereo feature matching, which then results in divergence of the filter. On the remaining datasets, the accuracy of the four different approaches is similar except ROVIO has larger error in the machine hall datasets which may be caused by the larger scene depth compared to the Vicon room datasets. For the CPU usage, the filter-based methods, both monocular and stereo, are more efficient compared with optimization based methods, which makes the filter-based approaches favorable for on-board real-time application. Between OKVIS and VINS-MONO, OKVIS has more CPU usage mainly because it uses Harris corner detector [35] and BRISK [36] descriptor for both temporal and stereo matching. Also, the backend of OKVIS is run at the fastest possible rate comparing to 10 Hz fixed in VINS-MONO. In the proposed S-MSCKF, around 80% of the computation is caused by the frontend including feature detection, tracking and matching. The filter itself takes about 10% of one core at 20 Hz. Our proposed method provides a good compromise between accuracy and computational efficiency.



(a)

(b)

Fig. 2. (a) Root-mean-square error (RMSE) and (b) average CPU load of OKVIS, ROVIO, VINS-MONO, and the proposed method on the EuRoC dataset. The parameters used for each method are the same as the values given in the corresponding Github repositories. Statistics are averaged over five runs on each dataset. For VINS-MONO and S-MSCKF, the frontend and backend are run as separate ROS nodes. The lighter color represents the CPU usage of the frontend while the darker color represents the backend. Note that the backend of VINS-MONO is run at 10 Hz because of limited CPU power.

B. Fast Flight Dataset

To further test the robustness of the proposed S-MSCKF, the algorithm is evaluated on four fast flight datasets with top speeds of 5 m/s, 10 m/s, 15 m/s, and 17.5 m/s respectively collected over an airport runway. During each run, the quadrotor is commanded to go to a waypoint 300 m ahead and return to the starting point. Our configuration includes two forward-looking PointGrey CM3-U3-13Y3M-CS cameras[1] running at 40 Hz with resolution 960×800 and one VectorNav VN-100 Rugged IMU[2] running at 200 Hz. The whole sensor suite is synchronized based on the trigger signal from the IMU. To achieve proper image exposure under varying lighting conditions, the camera's internal auto-exposure is disabled and replaced by a fast-adapting external controller that maintains constant average image brightness. The controller uses only the left image for brightness measurement, but then applies identical shutter times and gains to both cameras simultaneously. Fig. 3 shows some example images of the datasets. The dataset is publicly available at https://github.com/KumarRobotics/msckf_vio/wiki.

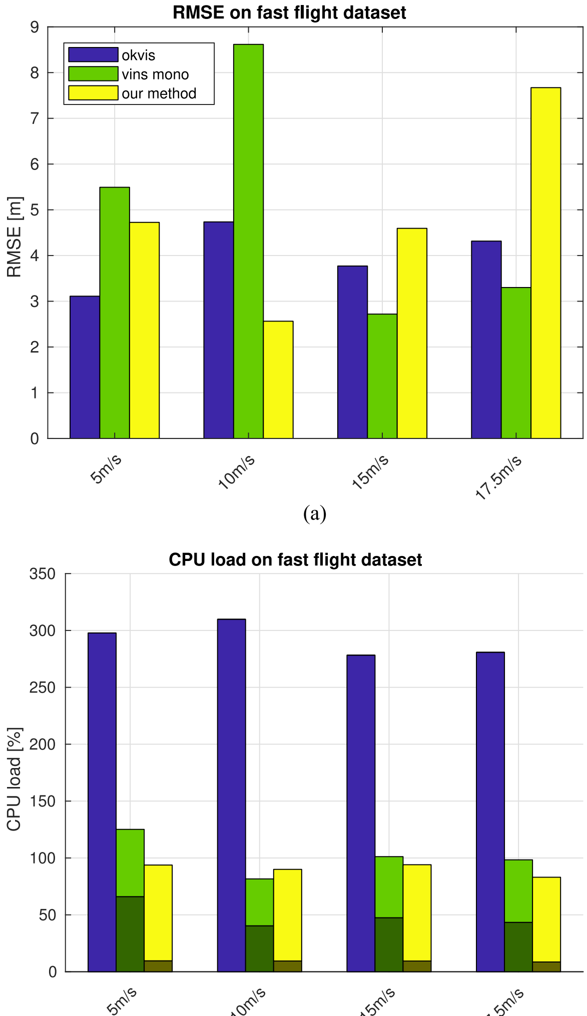
Fig. 4 compares the accuracy and CPU usage of different VIO solutions on the fast flight dataset. The result of ROVIO is omitted in the comparison since it has significant drift in scale which results in much lower accuracy compared to other

[1] https://www.ptgrey.com

[2] http://www.vectornav.com/products/vn-100

(a) (b)

Fig. 3. Example images in the fast flight datasets. (a) Images when the quadrotor is hovering. (b) Images when the quadrotor is accelerating.

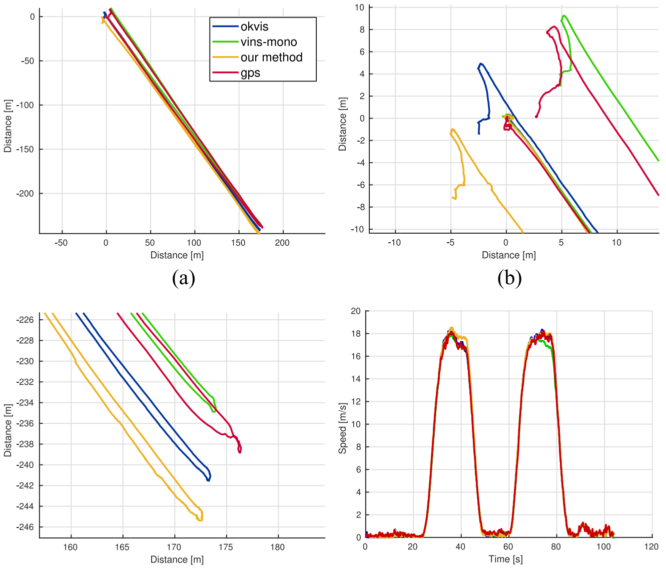


(a)

(b)

Fig. 4. (a) RMSE and (b) CPU load of OKVIS, VINS-MONO, and the proposed method averaged over five runs on each dataset. As in the EuRoC dataset test, the CPU load of VINS-MONO and our method is shown as combinations of front and back end. The backend of VINS-MONO is run at 10 Hz.

methods. The accuracy is evaluated by computing the RMSE of estimated and GPS position only in the x and y directions after proper alignment in both time and yaw. From the experiments, it can be observed that the S-MSCKF achieves the lowest CPU usage while maintaining similar accuracy comparing with other solutions.



(a) (b)

(c) (d)

Fig. 5. (a) Aligned trajectories, (b) the starting point, (c) the goal location, and (d) speed profiles in the dataset with top speed at 17.5 m/s.

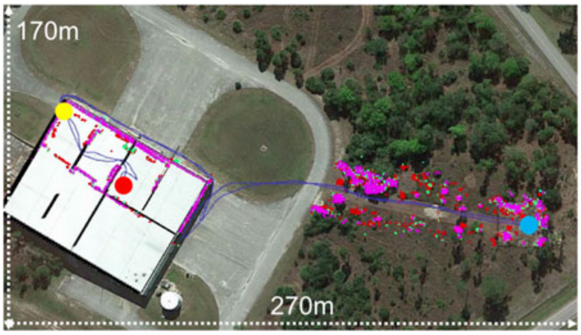


Fig. 6. The global map and round-trip trajectory overlaid on the Google satellite map in an fully autonomous flight experiment. The blue, red, and yellow dots represents the starting point, goal location, and the only entrance of the warehouse respectively. The global laser point cloud is registered using the estimation produced by the S-MSCKF. Over 700 m trajectory, the final drift is around 3 m under 0.5% of the total traveled distance.

Note that compared to the experiments with the EuRoC dataset, the proposed method spends more computational effort on the image processing frontend. One cause is the higher image frequency and resolution, while the other is that the aggressive flight induces shorter feature lifetime which then requires more frequent new feature detection. Fig. 5 shows the aligned trajectories and speed profiles in the dataset with top speed at 17.5 m/s.

C. Autonomous Flight in Unstructured Environments

The proposed S-MSCKF has been thoroughly tested in various field experiments. In this section, we show an example of a fully autonomous flight where the robot has to first navigate through a wooded area, then look for an entrance into a warehouse, find a target, then return to the starting point. This experiment is illustrative since it includes a combination of different kinds of environments as well as common challenges for vision-based estimation including feature poverty, aggressive maneuvers, and significant changes in lighting conditions during indoor-outdoor transitions.

Fig. 6 shows the global laser point cloud and round-trip trajectory overlaid on the Google satellite map. Note that, during the experiment, the laser measurement is used for mapping only. The state estimation is solely based on the stereo cameras and IMU as the sensor configuration given in Section IV-B. Over 700 m round-trip trajectory, the final drift is around 3 m, which is less than 0.5% of the total traveled distance despite the combination of various challenges along the flight. More details of this trial can be found in the supplementary video.[3]

V. Conclusion

In this letter, we present a new filter-based stereo visual inertial state estimation algorithm that uses the Multi-State Constraint Kalman Filter. We demonstrate the accuracy, efficiency, and robustness of our Stereo Multi-State Constraint Kalman Filter (S-MSCKF) using the EuRoC dataset as well as autonomous flight experiments in indoor and outdoor environments, comparing the computational efficiency and performance with state-of-art methods. We show that the S-MSCKF achieves robustness with a modest computational budget for aggressive, three-dimensional maneuvering, fast flight of speeds up to 17.5 m/s, indoor/outdoor transition, and indoor navigation in cluttered environments. Finally, we describe the S-MSCKF implementation which is available at https://github.com/KumarRobotics/msckf_vio.

Since the global position and yaw is not observable in a VIO system as explained in Section III-C, the uncertainty of the corresponding directions will keep growing as the robot travels. It can be observed in the experiments that the filter estimation may jump or even diverge once the prior uncertainty is large. Our future work is addressing the possibility of planning intelligent trajectories for a quadrotor such that the growth in uncertainty of the VIO estimator is slower, which then helps extend the effective range of the robot.

Appendix A

The $\mathbf{F}$ and $\mathbf{G}$ in (2) are,

$$\mathbf{F} = \begin{pmatrix} -\lfloor \hat{\boldsymbol{\omega}}_{\times} \rfloor & -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor \hat{\mathbf{a}}_{\times} \rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{pmatrix}$$

and,

$$\mathbf{G} = \begin{pmatrix} -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{pmatrix}$$

Appendix B

The state augmentation Jacobian, $\mathbf{J}$, given in (3), is of the form,

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_I & \mathbf{0}_{6\times6N} \end{pmatrix}$$

where $\mathbf{J}_I$ is,

$$\mathbf{J}_I = \begin{pmatrix} C\left({}^I_G\hat{\mathbf{q}}\right) & \mathbf{0}_{3\times9} & \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor {}^I\hat{\mathbf{p}}_{C\times} \rfloor & \mathbf{0}_{3\times9} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{I}_3 \end{pmatrix}$$

Note that $\mathbf{J}_I$ given above corrects the typo in [1, eq. (16)].

Appendix C

Following the chain rule, $\mathbf{H}^j_{C_i}$ and $\mathbf{H}^j_{f_i}$, in (5), can be computed as,

$$\begin{aligned} \mathbf{H}^j_{C_i} &= \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,1}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} + \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,2}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} \\ \mathbf{H}^j_{f_i} &= \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,1}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial {}^G\mathbf{p}_j} + \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,2}}\mathbf{p}_j} \cdot \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial {}^G\mathbf{p}_j} \end{aligned} \tag{7}$$

where,

$$\begin{aligned} \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,1}}\mathbf{p}_j} &= \frac{1}{{}^{C_{i,1}}\hat{Z}_j} \begin{pmatrix} 1 & 0 & -\frac{{}^{C_{i,1}}\hat{X}_j}{{}^{C_{i,1}}\hat{Z}_j} \\ 0 & 1 & -\frac{{}^{C_{i,1}}\hat{Y}_j}{{}^{C_{i,1}}\hat{Z}_j} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ \frac{\partial \mathbf{z}^j_i}{\partial {}^{C_{i,2}}\mathbf{p}_j} &= \frac{1}{{}^{C_{i,2}}\hat{Z}_j} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & -\frac{{}^{C_{i,2}}\hat{X}_j}{{}^{C_{i,1}}\hat{Z}_j} \\ 0 & 1 & -\frac{{}^{C_{i,2}}\hat{Y}_j}{{}^{C_{i,1}}\hat{Z}_j} \end{pmatrix} \\ \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} &= \begin{pmatrix} \lfloor {}^{C_{i,1}}\hat{\mathbf{p}}_{j\times} \rfloor & -C\left({}^{C_{i,1}}_{G}\hat{\mathbf{q}}\right) \end{pmatrix} \\ \frac{\partial {}^{C_{i,1}}\mathbf{p}_j}{\partial {}^G\mathbf{p}_j} &= C\left({}^{C_{i,1}}_{G}\hat{\mathbf{q}}\right) \\ \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} &= C\left({}^{C_{i,1}}_{C_{i,2}}\mathbf{q}\right)^\top \begin{pmatrix} \lfloor {}^{C_{i,1}}\hat{\mathbf{p}}_{j\times} \rfloor & -C\left({}^{C_{i,1}}_{G}\hat{\mathbf{q}}\right) \end{pmatrix} \\ \frac{\partial {}^{C_{i,2}}\mathbf{p}_j}{\partial {}^G\mathbf{p}_j} &= C\left({}^{C_{i,1}}_{C_{i,2}}\mathbf{q}\right)^\top C\left({}^{C_{i,1}}_{G}\hat{\mathbf{q}}\right) \end{aligned} \tag{8}$$

[3] https://youtu.be/jxfJFgzmNSw

APPENDIX D

By defining the following short-hand notation from (8)

$$\frac{\partial \mathbf{z}_i^j}{\partial^{C_{i,1}}\mathbf{p}_j} = \begin{pmatrix}\mathbf{J}_1\\ \mathbf{0}\end{pmatrix}, \quad \frac{\partial \mathbf{z}_i^j}{\partial^{C_{i,2}}\mathbf{p}_j} = \begin{pmatrix}\mathbf{0}\\ \mathbf{J}_2\end{pmatrix}$$

$$\frac{\partial^{C_{i,1}}\mathbf{p}_j}{\partial \mathbf{x}_{C_{i,1}}} = \mathbf{H}_1, \quad \frac{\partial^{C_{i,1}}\mathbf{p}_j}{\partial^{G}\mathbf{p}_j} = \mathbf{H}_2, \quad C\left({}^{C_{i,1}}_{C_{i,2}}\mathbf{q}\right) = \mathbf{R},$$

the measurement Jacobian in (7) can be compactly written as

$$\mathbf{H}_{C_i}^j = \begin{pmatrix}\mathbf{J}_1\mathbf{H}_1\\ \mathbf{J}_2\mathbf{R}^\top\mathbf{H}_1\end{pmatrix}, \quad \mathbf{H}_{f_i}^j = \begin{pmatrix}\mathbf{J}_1\mathbf{H}_2\\ \mathbf{J}_2\mathbf{R}^\top\mathbf{H}_2\end{pmatrix}.$$

Assuming $\mathbf{v} = \left(\mathbf{v}_1^\top,\ \mathbf{v}_2^\top\right)^\top \in \mathbb{R}^4$ is the left null space of $\mathbf{H}_{f_i}^j$, then,

$$\mathbf{v}^\top\mathbf{H}_{f_i}^j = \left(\mathbf{v}_1^\top\mathbf{J}_1 + \mathbf{v}_2^\top\mathbf{J}_2\mathbf{R}^\top\right)\mathbf{H}_2 = \mathbf{0}$$

Since $\mathbf{H}_2 = C\left({}^{C_{i,1}}_{G}\hat{\mathbf{q}}\right)$ is a rotation matrix, $\mathrm{rank}\left(\mathbf{H}_2\right) = 3$ which implies that $\mathbf{v}_1^\top\mathbf{J}_1 + \mathbf{v}_2^\top\mathbf{J}_2\mathbf{R}^\top = \mathbf{0}$. With such property, it immediately follows that $\mathbf{v}$ is also the left null space of $\mathbf{H}_{C_i}^j$,

$$\mathbf{v}^\top\mathbf{H}_{C_i}^j = \left(\mathbf{v}_1^\top\mathbf{J}_1 + \mathbf{v}_2^\top\mathbf{J}_2\mathbf{R}^\top\right)\mathbf{H}_1 = \mathbf{0}$$

Therefore, a single stereo measurement cannot be directly used for measurement update.

REFERENCES

[1] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3565–3572.

[2] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *Int. J. Robot. Res.*, vol. 32, no. 6, pp. 690–711, 2013.

[3] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation," in *Proc. Robot.: Sci. Syst. Conf.*, 2013, pp. 241–248.

[4] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

[5] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 298–304.

[6] T. Qin, P. Li, Z. Yang, and S. Shen, "VINS-Mono," 2017. [Online]. Available: https://github.com/HKUST-Aerial-Robotics/VINS-Mono

[7] M. Burri *et al.*, "The EuRoC micro aerial vehicle datasets," *Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1157–1163, 2016.

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

[9] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan: Dense 3D reconstruction in real-time," in *Proc. IEEE Intell. Veh. Symp.*, 2011, pp. 963–968.

[10] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PP, no. 99, Apr. 2017.

[12] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 834–849.

[13] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 957–964.

[14] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 4974–4981.

[15] Z. Yang and S. Shen, "Monocular visual–inertial state estimation with online initialization and camera–IMU extrinsic calibration," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 39–51, Jan. 2017.

[16] V. Usenko, J. Engel, J. Stückler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1885–1892.

[17] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proc. Robot.: Sci. Syst. Conf.*, 2015.

[18] K. Wu, A. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices," in *Proc. Robot.: Sci. Syst. Conf.*, 2015.

[19] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. J. Robot. Res.*, vol. 30, no. 1, pp. 56–79, 2011.

[20] K. Tsotsos, A. Chiuso, and S. Soatto, "Robust inference for visual-inertial sensor fusion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5203–5210.

[21] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[22] S. Agarwal *et al.*, "Ceres solver.," 2012. [Online]. Available: http://ceres-solver.org

[23] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators," *Int. J. Robot. Res.*, vol. 29, no. 5, pp. 502–528, 2010.

[24] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Consistency analysis and improvement of vision-aided inertial navigation," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 158–176, Feb. 2014.

[25] M. L. Xing Zheng, Zack Moratto and A. I. Mourikis, "Photometric patch-based visual-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3264–3271.

[26] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 61–76, Feb. 2012.

[27] M. K. Paul, K. Wu, J. A. Hesch, E. D. Nerurkar, and S. I. Roumeliotis, "A comparative analysis of tightly-coupled monocular, binocular, and stereo VINS," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 165–172.

[28] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Observability-constrained vision-aided inertial navigation," MARS Lab, Dept. Comput. Sci. Eng., Univ. Minnesota, Minneapolis, MN, USA, Tech. Rep. 2012-001, 2012.

[29] J. A. Castellanos, R. Martinez-Cantin, J. D. Tardós, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robot. Auton. Syst.*, vol. 55, no. 1, pp. 21–29, 2007.

[30] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Initialization-free monocular visual-inertial estimation with application to autonomous MAVs," in *Proc. Int. Symp. Exp. Robot.*, 2014, pp. 211–227.

[31] M. Trajković and M. Hedley, "Fast corner detection," *Image Vision Comput.*, vol. 16, no. 2, pp. 75–87, 1998.

[32] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.*, Apr. 1981, pp. 674–679.

[33] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme," in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 486–492.

[34] J. Nikolic *et al.*, "A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 431–437.

[35] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conf.*, Manchester, U.K., 1988, pp. 147–151.

[36] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. IEEE Int. Conf. Comput. Vision*, 2011, pp. 2548–2555.