Tightly-Coupled Visual-Inertial Localization and 3-D Rigid-Body Target Tracking

Kevin Eckenhoff, Yulin Yang, Patrick Geneva, and Guoquan Huang

***Abstract*—In this letter we present a novel method to perform target tracking of a moving rigid body utilizing an inertial measurement unit with cameras. A key contribution is the tightly-coupling of the target motion estimation within a visual-inertial navigation system (VINS), allowing for improved performance of both the processes. In particular, we build upon the standard multi-state constraint Kalman filter -based VINS and generalize it to incorporate three-dimensional (3-D) target tracking. Rather than representing the target object as a moving point particle (which is often the case in the literature), we instead utilize a dynamic, 3-D rigid-body model, wherein orientation, position, and their derivatives are estimated, as well as the structure of points on the object. We then leverage visual bearings to this set of features for target motion estimation, rather than requiring continuous observation of a single representative point over the tracking period. Moreover, we propose three motion models which capture most commonly-seen tracking scenarios in practice such as UAVs, fixed-wing aircraft, and ground vehicles over changing slopes and perform an observability analysis with geometric interpretation, providing insights into parameter initialization, and modes of estimation drift. The proposed estimator is validated with bothMonte-Carlo simulations and real-world experiments where it is shown to offer accurate performance even for challenging trajectories that do not completely fit the selected model.**

***Index Terms*—Visual tracking, localization, SLAM, visual-based navigation.**

I. Introduction and Related Work

THE ability of a robot to detect and track moving objects is a key component in a wide variety of application domains such as military surveillance and autonomous driving [1]. In many of these scenarios, the robot is not equipped with perfect knowledge of its state, such as through a motion-capture system, global positioning system (GPS), or through a prior map of the environment. As such, the system must process the data from noisy, onboard sensors to estimate both its own and the target states. This can be seen as a generalization of the classical SLAM problem, wherein the robot attempts to concurrently determine its location and the structure of the *static* environment, which has seen a great amount of research efforts in the past three decades [2].

For many applications of the target tracking problem, such as in unmanned aerial vehicle (UAV) surveillance, payload and computational resources are greatly constrained. This motivates the development of computationally efficient target tracking algorithms that can leverage the information from low-cost, lightweight sensors, while still performing robust and accurate estimation. For such resource constrained platforms, an inertial measurement unit (IMU) with monocular or stereo cameras has recently prevailed as a popular minimal sensing capability for motion estimation, which has been enabled by and driven a large amount of research in recent years into visual-inertial navigation systems (VINS) [3]–[6]. These algorithms can be mainly divided into two broad categories: filtering and batch-based approaches.

In the filtering domain, one of the earliest, and yet still one of the most popular, solutions to the VINS problem is the multi-state constraint Kalman filter (MSCKF) [3]. This approach processes inertial data through the propagation stage of an extended Kalman filter (EKF). Following this, features in the environment measured by the camera are quickly marginalized via a nullspace projection (which can be seen as linear marginalization [7]). This process generates measurements for use in the EKF update which relate *only* to stochastically cloned historical IMU/camera poses, thereby preventing the need to store the features in the state vector. The popularity of this algorithm has led to several extensions, such as performing online camera-to-IMU spatial and temporal calibration [8], fusing rolling shutter cameras [9], and improving consistency [10]. In addition, an inverse form of the MSCKF has recently been developed for use on mobile devices [11]. Batch-based methods, by contrast, solve a nonlinear maximum a posteriori (MAP) estimation problem over either the entire set, or sliding window, of measurements [5]. These methods offer improved robustness and accuracy over filtering methods at a cost of increased computational complexity. The recent development of IMU preintegration has allowed for even more efficient batch-based solutions [4], [12]–[14].

While the function of SLAM is to map the environment while tracking ego-motion, extensions to estimating other moving objects has seen recent research efforts. This coupled problem is known as simultaneous localization, mapping, and moving object tracking (SLAMMOT). Wang et al. [1] decoupled the SLAMMOT problem into two different estimators. Chojnacki and Indelman [15] provided a tightly-coupled batch solution to vision-only SLAMMOT based on light bundle adjustment

Manuscript received September 10, 2018; accepted January 9, 2019. Date of publication January 30, 2019; date of current version February 19, 2019. This letter was recommended for publication by Associate Editor J. Nieto and Editor C. Stachniss upon evaluation of the reviewers' comments. This work was supported in part by the University of Delaware College of Engineering, in part by the NSF under Grant IIS-1566129, in part by the DTRA under Grant HDTRA1-16-1-0039, and in part by Google Daydream. *(Corresponding author: Kevin Eckenhoff.)*

K. Eckenhoff, Y. Yang, and G. Huang are with the Department of Mechanical Engineering, University of Delaware, Newark DE 19716 USA (e-mail: keck@udel.edu; yuyang@udel.edu; ghuang@udel.edu).

P. Geneva is with the Department of Computer and Information Sciences, University of Delaware, Newark DE 19716 USA (e-mail: pgeneva@udel.edu).

This letter has supplementary downloadable multimedia material available at http://ieeexplore.ieee.orghttp://ieeexplore.ieee.org, provided by the authors. This includes a video showing both simulation and real world experiments of the tightly-coupled visual-inertial localization and target tracking algorithm. This material is 13.7MB in size.

Digital Object Identifier 10.1109/LRA.2019.2896472

(LBA) [16], while treating the target as a point particle with constant global velocity. Chen et al. [17] developed an active visual-inertial target tracking system with a UAV. This method separately estimates the sensing robot's state using a VINS algorithm, recovering the target motion by fitting a polynomial to a sequence of relative position measurements from the camera to the target. Lim and Sinha [18] utilized a semi-coupled approach where monocular visual odometry (VO) was run independently, and bearing measurements to a target human of known height were used to recover the scale. The camera estimates were then fused with target measurements in a separate EKF to estimate the position and velocity of both the camera and the target's foot.

A recent work related to our proposed method is by Li et al. [19], who used stereo vision to track the ego-motion of an autonomous vehicle. The pose, motion parameters, and 3D feature structure of objects were then estimated through a separate, loosely-coupled batch optimization, given the output estimates of visual SLAM and a dimensional prior for the objects. It should also be noted that very recently Qiu et al. [20] proposed a loosely-coupled visual-inertial estimation approach for object pose tracking, wherein the sensing device's state was estimated using VINS, while the target's historical poses and feature points were recovered in a separate estimator. While this method displayed impressive ability to track motion over a sliding window of images, it did not directly enforce a motion model on the object's entire pose (only position), and it is not clear if it can be used to reliably predict the target's *future* motion, which is often required in active target tracking systems.

In this work, we propose a *tightly-coupled* estimator for visual-inertial localization and target tracking (VILTT) by building upon the computationally efficient MSCKF-based VINS framework and generalizing to incorporate 6DOF rigid-body target tracking of a 3D moving object. In particular, the main contributions of this work are the following:

- We represent the target object as a rigid structure built from features and incorporate this representation into the MSCKF using different motion models. This representation allows for *robust* estimation of the target state even if the same feature is not continuously seen over a trajectory due to changing viewing angles or occlusions.
- We offer an extensive observability analysis of the system with the three proposed target motion models, and show that, besides the four unobservable directions inherited from VINS [10], there will be additional unobservable directions related to the target state, whose geometric interpretations are also provided.
- The proposed tightly-coupled VILTT estimator, with each of the three target motion models, is validated in Monte-Carlo simulations and real-world experiments.

II. Estimation Background

Within the standard MSCKF framework [3], the IMU state of an aided inertial navigation system (INS) is given by:

$$\mathbf{x}_I = \begin{bmatrix} {}^I_G\bar{q}^\top & \mathbf{b}_\omega^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_a^\top & {}^G\mathbf{p}_I^\top \end{bmatrix}^\top \tag{1}$$

where ${}^I_G\bar{q}$ is the unit quaternion of JPL form parameterizing the rotation ${}^I_G\mathbf{R}$ from the global frame $\{G\}$ to the current local frame $\{I\}$ [21], $\mathbf{b}_\omega$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases, and ${}^G\mathbf{v}_I$ and ${}^G\mathbf{p}_I$ are the velocity and position of the IMU expressed in the global frame, respectively. The error state corresponding to (1) is:

$$\delta\mathbf{x}_I = \begin{bmatrix} {}^I\delta\boldsymbol{\theta}_G^\top & \delta\mathbf{b}_\omega^\top & {}^G\delta\mathbf{v}_I^\top & \delta\mathbf{b}_a^\top & {}^G\delta\mathbf{p}_I^\top \end{bmatrix}^\top \tag{2}$$

The relationship between the vector quantities with true value $\mathbf{v}$, mean value $\hat{\mathbf{v}}$, and error state $\delta\mathbf{v}$ takes the form $\mathbf{v} = \hat{\mathbf{v}} + \delta\mathbf{v}$. For quaternions in JPL convention, with true value $\bar{q}$, mean value $\hat{\bar{q}}$, and error state $\delta\boldsymbol{\theta}$, we have $\bar{q} \approx [(\delta\boldsymbol{\theta}/2)^\top \; 1]^\top \otimes \hat{\bar{q}}$, with $\otimes$ as the quaternion multiplication.

A. Inertial State Propagation

An IMU attached to the moving platform provides local linear acceleration and angular velocity measurements. In particular, the measurements $\mathbf{a}_m$ and $\boldsymbol{\omega}_m$, are related to the true values, $\mathbf{a}$ and $\boldsymbol{\omega}$, by:

$$\mathbf{a}_m = \mathbf{a} + {}^I_G\mathbf{R}\,{}^G\mathbf{g} + \mathbf{b}_a + \mathbf{n}_a, \;\; \boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{b}_\omega + \mathbf{n}_\omega \tag{3}$$

where ${}^G\mathbf{g} \simeq [0\,0\,9.81]^\top$ is the global gravity, and $\mathbf{n}_a$ and $\mathbf{n}_\omega$ are the continuous-time Gaussian noises which corrupt the measurements (note that in the rest of this letter we let $\mathbf{n}$ denote zero-mean Gaussian noises). The underlying IMU dynamics are given by [22]:

$$\begin{aligned} {}^I_G\dot{\bar{q}} &= \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\,{}^I_G\bar{q}, \quad {}^G\dot{\mathbf{v}} = {}^I_G\mathbf{R}^\top\mathbf{a}, \quad {}^G\dot{\mathbf{p}} = {}^G\mathbf{v} \\ \dot{\mathbf{b}}_w &= \mathbf{n}_{b\omega}, \quad \dot{\mathbf{b}}_a = \mathbf{n}_{ba} \end{aligned} \tag{4}$$

where $\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{bmatrix}$ and $\lfloor\cdot\rfloor$ denotes the skew symmetric matrix. Using these dynamic models, the EKF propagation can be performed according to [3].

B. MSCKF Feature Update

The first step of the MSCKF is to perform stochastic cloning, such that our state vector also contains the estimates for the poses of the robot during a sliding window of m historical cloning times. At imaging timestep k, the augmented state $\mathbf{x}_k$ becomes:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_I & \mathbf{x}_{cl} \end{bmatrix}^\top \tag{5}$$

$$\mathbf{x}_{cl} = \begin{bmatrix} {}^{I_{k-1}}_G\bar{q}^\top & {}^G\mathbf{p}_{I_{k-1}}^\top & | \cdots | & {}^{I_{k-m}}_G\bar{q}^\top & {}^G\mathbf{p}_{I_{k-m}}^\top \end{bmatrix}^\top \tag{6}$$

where ${}^{I_i}_G\bar{q}$ and ${}^G\mathbf{p}_{I_i}$ refer to the position and orientation of the IMU at imaging timestep i. As the sensor suite moves throughout the environment, the image measurements corresponding to the same tracked feature are collected over the sliding window. Each measurement, $\mathbf{z}_i$, is expressed as a function of the corresponding cloned pose and the global feature position:

$$\begin{aligned} \mathbf{z}_i &= \boldsymbol{\Pi}({}^{C_i}\mathbf{p}_{fs}) + \mathbf{n}_{fi}, \;\; \boldsymbol{\Pi}\left([x\,y\,z]^\top\right) = \begin{bmatrix} \frac{x}{z} & \frac{y}{z} \end{bmatrix}^\top \\ {}^{C_i}\mathbf{p}_{fs} &= {}^C_I\mathbf{R}\,{}^{I_i}_G\mathbf{R}\left({}^G\mathbf{p}_{fs} - {}^G\mathbf{p}_{I_i}\right) + {}^C\mathbf{p}_I \end{aligned} \tag{7}$$

where ${}^C_I\mathbf{R}$ and ${}^C\mathbf{p}_I$ represent the camera-to-IMU calibration parameters for the measuring camera, and ${}^G\mathbf{p}_{fs}$ refers to the position of the static feature in the global frame. In this work we represent features using an inverse depth representation [3] defined by $\mathbf{m}_f = [\alpha\;\beta\;\rho]^\top$ in an arbitrary "anchoring" frame:

$${}^G\mathbf{p}_{fs} = {}^G_{C_a}\mathbf{R}\left(\frac{1}{\rho}\begin{bmatrix}\alpha\\ \beta\\ 1\end{bmatrix}\right) + {}^G\mathbf{p}_{C_a} \tag{8}$$

where C_a denotes the anchoring camera frame (in practice we pick the anchor to be the frame that the feature was first seen from). Utilizing the fact that scaling of the argument vector does not change the projection function, we use the transformed, but equivalent, measurement model:

$$\mathbf{z}_i = \mathbf{\Pi}(\rho^{C_i}\mathbf{p}_{fs}) + \mathbf{n}_{fi} \tag{9}$$

This formulation prevents instability for features at a very far distance. Using the current estimates for the clones in the window, triangulation is performed to recover an estimate for the feature $\hat{\mathbf{m}}_f$. We then linearize the system to obtain the robot state $\mathbf{H}_x$ and 3D feature $\mathbf{H}_f$ Jacobians, as well as the linearized residual system:

$$\delta\mathbf{z} = \mathbf{H}_x\delta\mathbf{x} + \mathbf{H}_f\delta\mathbf{m}_f + \mathbf{n}_f \tag{10}$$

where $\delta\mathbf{z}$ is formed by stacking the individual measurement residuals, $\delta\mathbf{z}_i = \mathbf{z}_i - \mathbf{\Pi}(\hat{\rho}^{C_i}\hat{\mathbf{p}}_{fs})$. The key idea of the MSCKF is to find the matrix $\mathbf{Q}_2$ whose columns span the left nullspace of $\mathbf{H}_f$. Multiplying the above linear system on the left by $\mathbf{Q}_2^\top$, we obtain a new measurement function that depends *only* on the robot state:

$$\delta\mathbf{z}' = \mathbf{H}'_x\delta\mathbf{x} + \mathbf{n}'_f \tag{11}$$

Therefore, we can directly use this measurement to update our state using the well-known EKF update [3] *without* the need to store the measured features in the state. This leads to substantial computational savings as the problem size remains bounded over the entire trajectory.

III. Target State Estimation

A. Target State Representation

Leveraging the lightweight MSCKF framework, we now rigorously incorporate the tracking of an external 3D moving object into the same estimation thread. The first problem that needs to be addressed is *how* to represent the rigid-body target. That is, we need to define what parameters must be estimated to fully define the target and its motion.

One of the simplest representations is that of a point particle, which involves estimating a single position and its derivatives [15]. In reality, however, we often wish to track the motion of an arbitrarily large rigid *object*. Naive use of the point particle model requires picking a single point on the target to serve as a representation of the entire object, losing higher-level geometric understanding. For example, when target tracking using vision sensors, computer vision algorithms will possibly yield many features (corners) on the body of the target. A point particle model requires that we can only use one of these measurements, i.e., the one corresponding to the representative feature. If this feature becomes occluded, for example if the rigid body undergoes a large rotation, we can no longer measure the target, despite the fact that other features on the object can be still be observed. Therefore, we are motivated to instead represent a target as a set of structured points along with a pose.

In particular, we assume that the target evolves as a rigid body. That is, the relative positions of all points that reside on it, as expressed in a local body frame, remain fixed. We pick a representative point to serve as the origin of a local body frame (see Fig. 1). The pose of the target is then defined by the position of this representative point, ${}^G\mathbf{p}_T$, along with an orientation, ${}^T_G\bar{q}$.

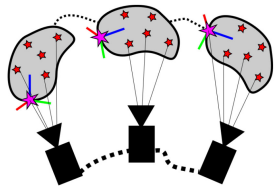

Fig. 1. Illustration of the rigid body target tracking problem. As the sensor suite moves through the environment, bearing measurements to features on the evolving target's body, shown as stars, are collected and tracked through the sequence of images. To represent the pose of the target, a coordinate system is chosen and attached to a representative feature (pink star).

In practice this representative point will often correspond to the first seen target feature that can be reliably extracted over time.

B. Target Measurement Update

Similar to the standard static features [see (7)], image measurements of points on the target's rigid body are collected. By abuse of notation, those measurements corresponding to the representative feature are written as:

$$\mathbf{z}_i = \mathbf{\Pi}({}^{C_i}\mathbf{p}_T) + \mathbf{n}_{fi} \tag{12}$$

$${}^{C_i}\mathbf{p}_T = {}^C_I\mathbf{R}\,{}^{I_i}_G\mathbf{R}\left({}^G\mathbf{p}_T - {}^G\mathbf{p}_{I_i}\right) + {}^C\mathbf{p}_I \tag{13}$$

It is important to note that because the representative point's position is being estimated in this framework, this measurement can be used directly in the EKF update.

Since the number of tracked features that do *not* correspond to the representative point could be very large, we leverage the MSCKF's nullspace projection (i.e., linear marginalization [7]) to limit the state size. In particular, we concurrently maintain a sliding window of *target* poses $\mathbf{T}_i = [{}^{T_i}_G\bar{q}^\top\ {}^G\mathbf{p}_{T_i}^\top]^\top$, where for convenience we let $\mathbf{T}_i$ denote the target pose clone associated with IMU clone i corresponding to the same imaging time. These non-representative feature measurements are given by:

$$\mathbf{z}_i = \mathbf{\Pi}({}^{C_i}\mathbf{p}_{ft}) + \mathbf{n}_{fi} \tag{14}$$

$${}^{C_i}\mathbf{p}_{ft} = {}^C_I\mathbf{R}\,{}^{I_i}_G\mathbf{R}\left({}^G\mathbf{p}_{T_i} + {}^{T_i}_G\mathbf{R}^\top{}^T\mathbf{p}_{ft} - {}^G\mathbf{p}_{I_i}\right) + {}^C\mathbf{p}_I \tag{15}$$

In this case, the unknown feature state is the position of the feature expressed in the frame of the target, ${}^T\mathbf{p}_{ft}$. We then perform the same nullspace projection as the standard MSCKF with all tracks of this dynamic feature allowing for an efficient update that does not depend on the feature state. Alternatively, we can choose to add a small subset of the additional features into the state vector depending on the available resources, which we have found yields a substantial improvement in the accuracy of the target orientation estimate due to reobservations. In fact, we have found experimentally that while these MSCKF-like target features provide short-term orientation information, relying on them solely will yield large orientation drift over a long period of time, thus motivating us to keep a small, sparse set of features in the state.

As in the standard MSCKF, an initial estimate of the marginal parameter (non-representative feature) must be obtained in order to perform the nullspace projection [see (10)], as well as variable initialization. To this end, we use the following geometric constraint about the unknown non-representative feature

position, ${}^{T}\mathbf{p}_{ft}$, (see Fig. 1):

$$ {}^{T}\mathbf{p}_{ft} = {}^{T_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_{T_i}\right) + d_i {}^{T_i}_{G}\mathbf{R}\, {}^{G}_{C_i}\mathbf{R}\, {}^{C_i}\mathbf{r}_i \Rightarrow \tag{16} $$

$$ \left\lfloor {}^{T_i}_{G}\mathbf{R}\, {}^{G}_{C_i}\mathbf{R}\, {}^{C_i}\mathbf{r}_i \right\rfloor {}^{T}\mathbf{p}_{ft} = \left\lfloor {}^{T_i}_{G}\mathbf{R}\, {}^{G}_{C_i}\mathbf{R}\, {}^{C_i}\mathbf{r}_i \right\rfloor {}^{T_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_{T_i}\right) \tag{17} $$

where d_i is the unknown depth of the feature in the i-th image, and ${}^{C_i}\mathbf{r}_i$ is the corresponding measured bearing, and ${}^{G}\mathbf{p}_{C_i}$ is the position of the measuring camera. Stacking all measurements (17) for the feature taken over the interval, we build a *linear system* that can be solved efficiently to obtain an estimate of the local feature position. This estimate is then refined by a local BA over the target feature utilizing the collected bearing measurements.

C. Target Stochastic Motion Models

To incorporate target tracking (a dynamic system) into the EKF framework, a motion model for propagation is needed. Unlike the tracking robot, we have *no* access to proprioceptive sensors for prediction of the target's state. As such, we assume a stochastic motion model and jointly estimate its parameters alongside the target's pose. In the following, we advocate three possible motion models that can capture a large class of realistic target tracking scenarios.

1) Model 1. Constant Global Linear Velocity: We first assume constant global linear velocity, ${}^{G}\mathbf{v}_T$, and constant angular velocity, ${}^{T}\boldsymbol{\omega}$, which yields the total target state and its dynamics as:

$$ \mathbf{x}_T^{(1)} = \begin{bmatrix} {}^{T}_{G}\bar{q}^\top & {}^{T}\boldsymbol{\omega}^\top & {}^{G}\mathbf{p}_T^\top & {}^{G}\mathbf{v}_T^\top \end{bmatrix}^\top $$

$$ {}^{T}_{G}\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\Omega}\left({}^{T}\boldsymbol{\omega}\right){}^{T}_{G}\bar{q},\ {}^{G}\dot{\mathbf{p}}_T = {}^{G}\mathbf{v}_T,\ {}^{G}\dot{\mathbf{v}}_T = \mathbf{n}_{tv},\ {}^{T}\dot{\boldsymbol{\omega}} = \mathbf{n}_{t\omega} $$

In particular, we treat both the linear and angular velocities as continuous-time random walks driven by noises $\mathbf{n}_{tv}$ and $\mathbf{n}_{t\omega}$. The strength of the noise values can be used to capture the predictability of the target based on its assumed motion model. This model is ideal for scenarios in which the evolution of the target's orientation and position are decoupled. For example, UAVs can move with full position and yaw control, and thus the orientation is not strictly coupled with position. The error state of this model evolves according to:

$$ \begin{bmatrix} {}^{T}\dot{\delta\boldsymbol{\theta}}_G \\ {}^{T}\dot{\delta\boldsymbol{\omega}} \\ {}^{G}\dot{\delta\mathbf{p}}_T \\ {}^{G}\dot{\delta\mathbf{v}}_T \end{bmatrix} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}} \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} {}^{T}\delta\boldsymbol{\theta}_G \\ {}^{T}\delta\boldsymbol{\omega} \\ {}^{G}\delta\mathbf{p}_T \\ {}^{G}\delta\mathbf{v}_T \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{tv} \\ \mathbf{n}_{t\omega} \end{bmatrix} $$

2) Model 2. Constant Local Linear Velocity: This model assumes that the target exhibits constant velocity as seen from its *local* frame. We therefore replace the global linear velocity in the previous model with local velocity ${}^{T}\mathbf{v}_T$. Such a model can be used for example, for tracking ground vehicles or fixed-wing aircrafts. The target state and its dynamics are given by:

$$ \mathbf{x}_T^{(2)} = \begin{bmatrix} {}^{T}_{G}\bar{q}^\top & {}^{T}\boldsymbol{\omega}^\top & {}^{G}\mathbf{p}_T^\top & {}^{T}\mathbf{v}_T^\top \end{bmatrix}^\top $$

$$ {}^{T}_{G}\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\Omega}\left({}^{T}\boldsymbol{\omega}\right){}^{T}_{G}\bar{q},\ {}^{G}\dot{\mathbf{p}}_T = {}^{G}_{T}\mathbf{R}\, {}^{T}\mathbf{v}_T,\ {}^{T}\dot{\mathbf{v}}_T = \mathbf{n}_{tv}, {}^{T}\dot{\boldsymbol{\omega}} = \mathbf{n}_{tw} $$

with the corresponding error state dynamics:

$$ \begin{bmatrix} {}^{T}\dot{\delta\boldsymbol{\theta}}_G \\ {}^{T}\dot{\delta\boldsymbol{\omega}} \\ {}^{G}\dot{\delta\mathbf{p}}_T \\ {}^{T}\dot{\delta\mathbf{v}}_T \end{bmatrix} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}} \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -{}^{G}_{T}\hat{\mathbf{R}}\lfloor {}^{T}\hat{\mathbf{v}}_T \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & {}^{G}_{T}\hat{\mathbf{R}} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} {}^{T}\delta\boldsymbol{\theta}_G \\ {}^{T}\delta\boldsymbol{\omega} \\ {}^{G}\delta\mathbf{p}_T \\ {}^{T}\delta\mathbf{v}_T \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{tv} \\ \mathbf{n}_{tw} \end{bmatrix} $$

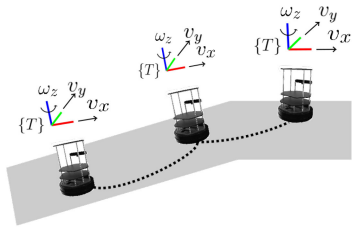

Fig. 2. Illustration of the planar motion model. The target maintains a constant yaw rate and local planar velocity. The noise injected into the model can be used to handle changes in the ground plane.

3) Model 3. Local Planar Velocity: In many applications, the target is known to navigate in a locally planar environment (e.g., when tracking a ground vehicle). Rather than assuming pure 2D scenarios, we allow for *changing* ground planes, for example, when a vehicle goes up a ramp before coming to a new elevation (see Fig. 2). A method to handle this plane change is to also estimate the current ground plane and add pseudo-measurements that the target should operate in this plane [23]. We forgo directly estimating the plane by proposing the following stochastic local planar model:

$$ \mathbf{x}_T^{(3)} = \begin{bmatrix} {}^{T}_{G}\bar{q}^\top & \omega_z & {}^{G}\mathbf{p}_T^\top & v_x & v_y \end{bmatrix}^\top $$

$$ {}^{T}_{G}\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\Omega}\left(\begin{bmatrix} n_{\omega x} \\ n_{\omega y} \\ \omega_z \end{bmatrix}\right){}^{T}_{G}\bar{q},\ {}^{G}\dot{\mathbf{p}}_T = {}^{G}_{T}\mathbf{R}\begin{bmatrix} v_x \\ v_y \\ n_{vz} \end{bmatrix} $$

$$ \dot{v}_x = n_{vx},\ \dot{v}_y = n_{vy},\ \dot{\omega}_z = n_{\omega z} $$

In particular, along with the target pose we maintain the scalars v_x, v_y, and ω_z which describe the linear velocity in the local frame and the angular velocity about the z-axis (yaw). This motion model therefore constrains the evolution of the target state to act with constant local velocity in a plane whose normal lies along the axis of rotation. The noises in the other directions allow us to model such effects as uneven roads and the changing of the current ground plane (such as going up a hill). The error

state evolves according to:

$$\begin{bmatrix} {}^T\dot{\delta\boldsymbol{\theta}}_G \\ \dot{\delta\omega}_z \\ {}^G\dot{\delta\mathbf{p}}_T \\ \dot{\delta v}_x \\ \dot{\delta v}_y \end{bmatrix} = \begin{bmatrix} -\lfloor {}^T\hat{\boldsymbol{\omega}} \rfloor & \mathbf{e}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 2} \\ \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \mathbf{0}_{3\times 2} \\ -{}^G_T\hat{\mathbf{R}}\lfloor {}^T\hat{\mathbf{v}}_T \rfloor & \mathbf{0}_{3\times 2} & \mathbf{0}_3 & {}^G_T\hat{\mathbf{R}}[\mathbf{e}_1 \quad \mathbf{e}_2] \\ \mathbf{0}_{2\times 3} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 3} & \mathbf{0}_{2\times 2} \end{bmatrix}$$

$$\begin{bmatrix} {}^T\delta\boldsymbol{\theta}_G \\ \delta\omega_z \\ {}^G\delta\mathbf{p}_T \\ \delta v_x \\ \delta v_y \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{L} \\ \mathbf{0}_{1\times 3} & \mathbf{e}_3^\top \\ {}^G_T\hat{\mathbf{R}}\mathbf{K} & \mathbf{0}_3 \\ \mathbf{J} & \mathbf{0}_{2\times 3} \end{bmatrix} \begin{bmatrix} n_{vx} \\ n_{vy} \\ n_{vz} \\ n_{\omega x} \\ n_{\omega y} \\ n_{\omega z} \end{bmatrix} \tag{18}$$

where

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \mathbf{K} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{J} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

and $\mathbf{e}_i$ is the 3×1 unit vector in the i-th axis direction.

IV. Observability Analysis

In this section, we perform an in-depth observability analysis for the linearized VILTT system with the three target motion models. The key reasons for observability analysis include: (i) it provides a deep insight about the system's geometrical properties [10], [24], [25] and determines the minimum measurement modalities or state parameters needed to initialize the estimator, (ii) it can be used to identify degenerate motions [23], [26] which cause additional unobservable directions and should be avoided in real applications whenever possible, and (iii) the observability constrained (OC)-based methodology as in OC-EKF [24] and OC-VINS [10], that enforce the correct observability properties, can be adopted to improve consistency.

For simplicity, we consider the case where the state vector contains the IMU state $\mathbf{x}_I$, target state $\mathbf{x}_T$ (from Section III-C), one static (environmental) feature ${}^G\mathbf{p}_{fs}$ and one non-representative target feature ${}^T\mathbf{p}_{ft}$:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I^\top & {}^G\mathbf{p}_{fs}^\top & \mathbf{x}_T^\top & {}^T\mathbf{p}_{ft}^\top \end{bmatrix}^\top \tag{19}$$

In analogy to [10], we construct the observability matrix for the linearized VILTT system whose right nullspace spans the unobservable directions. Intuitively, these unobservable directions correspond to state variables that cannot be recovered from the measurement constraints. In the following, due to the space constraint, we present the main results of our observability analysis, while the detailed analysis can be found in our companion technical report [27].

A. Model 1

Given model 1 (constant ${}^G\mathbf{v}_T$ and constant ${}^T\boldsymbol{\omega}$), if all measurements to the static feature, target feature, and representative point are available, the VILTT system will have at least 7 unobservable directions corresponding to the global yaw, global IMU position ${}^G\mathbf{p}_I$, and the target orientation ${}^T_G\mathbf{R}$. Clearly, the first 4 unobservable directions are inherited from VINS [10]. Interestingly, if the measurements of the target's representative point are *unavailable* (e.g., due to occlusion), the system will have one more unobservable direction corresponding to the representative point position along the rotation axis of ${}^T\boldsymbol{\omega}$. In both cases, the unobservable directions related to the target state are the following:

$$\mathbf{N}^{(1)}_{{}^T_G\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor {}^T\hat{\boldsymbol{\omega}} \rfloor\right)^\top & \mathbf{0}_3 & \mathbf{0}_3 & \left(\lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor\right)^\top \end{bmatrix}^\top \tag{20}$$

$$\mathbf{N}^{(1)}_{{}^G\mathbf{p}_T} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \left({}^G_{T_0}\hat{\mathbf{R}}{}^T\hat{\boldsymbol{\omega}}\right)^\top & \mathbf{0}_{1\times 3} & \left(-{}^T\hat{\boldsymbol{\omega}}\right)^\top \end{bmatrix}^\top \tag{21}$$

B. Model 2

Given model 2 (constant ${}^T\mathbf{v}_T$ and constant ${}^T\boldsymbol{\omega}$), if all measurements available, the system will have at least 7 unobservable directions as in model 1. Similarly, if no representative-point measurements are available, the system will have 3 extra unobservable directions that correspond to the full 3D position of the representative point. In both cases, the unobservable directions related to the target state can be respectively found as follows:

$$\mathbf{N}^{(2)}_{{}^T_G\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor {}^T\hat{\boldsymbol{\omega}} \rfloor\right)^\top & \mathbf{0}_3 & \left(\lfloor {}^T\hat{\mathbf{v}} \rfloor\right)^\top & \left(\lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor\right)^\top \end{bmatrix}^\top \tag{22}$$

$$\mathbf{N}^{(2)}_{{}^G\mathbf{p}_T} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \left({}^G_{T_0}\hat{\mathbf{R}}\right)^\top & \left(\lfloor {}^T\hat{\boldsymbol{\omega}} \rfloor\right)^\top & -\mathbf{I}_3 \end{bmatrix}^\top \tag{23}$$

C. Model 3

Given model 3 (planar motion with constant ω_z, v_x, and v_y), if all measurements are available, unlike the above two models, the target's roll and pitch will become observable and thus the system has at least 5 unobservable directions, among which 4 are inherited from VINS and 1 corresponds to target orientation yaw. If no measurements of the representative point are available, as in the case of model 2, the system will also gain 3 extra unobservable directions corresponding to the full 3D position of the representative point. In both cases, the unobservable directions related to the target state are given by:

$$\mathbf{N}^{(3)}_{{}^T_G\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{e}_3^\top & \mathbf{0}_1 & \mathbf{0}_{1\times 3} & \hat{v}_y & -\hat{v}_x & \left(\lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3\right)^\top \end{bmatrix}^\top \tag{24}$$

$$\mathbf{N}^{(3)}_{{}^G\mathbf{p}_T} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^G_{T_0}\hat{\mathbf{R}}\mathbf{e}_3\right)^\top & \mathbf{0}_1 & \mathbf{0}_1 & -\mathbf{e}_3^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^G_{T_0}\hat{\mathbf{R}}\mathbf{e}_2\right)^\top & -\hat{\omega}_z & \mathbf{0}_1 & -\mathbf{e}_2^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^G_{T_0}\hat{\mathbf{R}}\mathbf{e}_1\right)^\top & \mathbf{0}_1 & \hat{\omega}_z & -\mathbf{e}_1^\top \end{bmatrix}^\top \tag{25}$$

It can be seen from these results that the VILTT systems will have at least 4 unobservable directions inherited from VINS

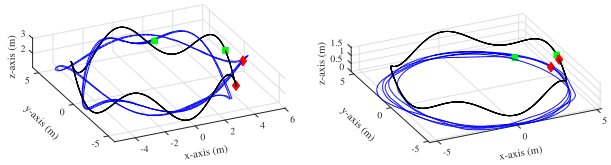

Fig. 3. 3D simulation trajectories of the tracking robot (black) and target (blue): (a) general 3D target motion, and (b) constrained target motion. The total tracking robot's path length is 186 and 165 meters, respectively. The green square and red diamond denote the start and end of the tracking robot and target trajectories, respectively.

which correspond to the initial global yaw and global IMU position (see [10], [26]), while the extra unobservable directions depend on the target motion model selected. Note that it is not trivial to find a best representative point that will be frequently measured, as a representative point that is occluded or cannot be tracked reliably will make the system suffer from the introduction of additional unobservable directions. On the other hand, an unobservable parameter can be initialized arbitrarily (unless some prior information is available). For example, in models 1 and 2, the initial target orientation can be freely chosen due to its unobservability, while in model 3, the orientation initialization procedure needs to be carefully addressed with available measurements.

V. Simulation Results

In this work we used a stereo visual-inertial system, but note that the proposed method can also be deployed in a monocular setting. The RotorS simulator [28], which leverages Gazebo [29], was used to simulate an Asctec Firefly UAV equipped with a stereo visual-inertial sensor as the active tracking robot, while another simulated robot acted as the passive target. Two scenarios were created: (i) both the tracking robot and a target Firefly move with 3D motion, and (ii) a Turtlebot target is constrained to a 2D planar motion while the tracking robot moves through 3D space. The ground-truth IMU readings were corrupted using the realistic sensor characteristics of an ADIS16448 IMU, while image measurements were corrupted by one pixel noise. The rigid-body target was treated as a one meter box with features lying around the surface of the boundary, while static features were simulated around the workspace.

For both static and target feature measurements, occlusions were simulated by checking whether the projection vector intersected the boundary of the target's box, ensuring that blocked features were not used. Fourteen of the target features were maintained in the state vector to prevent orientation drift in the VILTT. We performed 30 Monte-Carlo simulations wherein a single ground truth target and IMU trajectory was collected for each scenario, and each Monte-Carlo run represented a different realization of noise corrupting the corresponding measurements (IMU and bearing). The performance metrics used are the root mean squared errors (RMSE) of: (i) the 6DOF absolute (global) pose (position and orientation) estimates of both the tracking robot and target, and (ii) the relative position estimates between the tracking robot and the target. We note that the relative position error may be of more importance than the absolute error for certain scenarios such as autonomous target following.

We first evaluated the performance of the proposed VILTT with the motion models 1 and 2, where both the tracking robot and target moved along 3D trajectories as depicted in Fig. 3a.

TABLE I
Averaged RMSE Results of the Proposed VILTT in the Case of General 3D Target Motion, Showing Both the Absolute and Relative Accuracy of the Realtime Performance

	Global Velocity		Local Velocity	
Units	m	deg	m	deg
IMU	0.309	2.669	0.302	2.571
Target	0.319	3.559	0.318	5.625
Relative	0.011	-	0.028	-

TABLE II
Averaged RMSE Results of the Proposed VILTT in the Case of Constrained 3D Target Motion, Showing Both the Absolute and Relative Accuracy of the Realtime Performance

	Global Velocity		Local Velocity		Local Planar	
Units	m	deg	m	deg	m	deg
IMU	0.196	1.180	0.196	1.178	0.204	1.039
Target	0.207	1.616	0.207	1.610	0.214	7.567
Relative	0.010	-	0.010	-	0.015	-

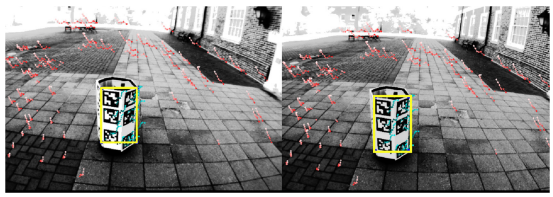

Fig. 4. Example image taken during the second experiment. A Turtlebot equipped with fiducial tags acted as the target. These tags were used to distinguish features both on (light-blue) and off (red) the target by forming a bounding box (yellow).

The Monte-Carlo results are shown in Table I. As evident, both systems were able to achieve high accuracy and recover the target's 6DOF motion (both position and orientation). In this experiment the orientation error for model 1 was much smaller than that for model 2, which is most likely due to the fact that the UAV had mostly decoupled orientation and position control. Interestingly both models had very similar performance in the position estimates for the target and the pose estimates of the IMU. Note that although the estimated target trajectory did *not* exhibit constant global or local velocity exactly, the proposed models were still able to handle these imperfections due to modeling the target's velocities as random walks.

In the second simulation, we validated the performance of the proposed VILTT with the three target models where the target exhibited constrained motion as depicted in Fig. 3b. The Monte-Carlo results of RMSE values are shown in Table II. Clearly, all models were able to generate accurate trajectories for the target. We note that for these results the orientation error of model 3 (planar motion) cannot be directly comparable to that of the other two models, as the planar model is attempting to estimate a partially observable target orientation (only the yaw of target orientation is unobservable), while the other models estimate the change in orientation from the arbitrarily initial value. It is interesting to note that even though the target did not move with ideal constant velocity, as it was driven by hand, all models were able to handle this deviation from the assumed motion model. Surprisingly, for this scenario the planar model actually gave the worst result of the three models in terms of target position. This is most likely due to the fact that the imperfections in each model are being captured by the propagation noise whose characterization may greatly impact the accuracy of estimation. It is expected that better characterization of these noise levels would

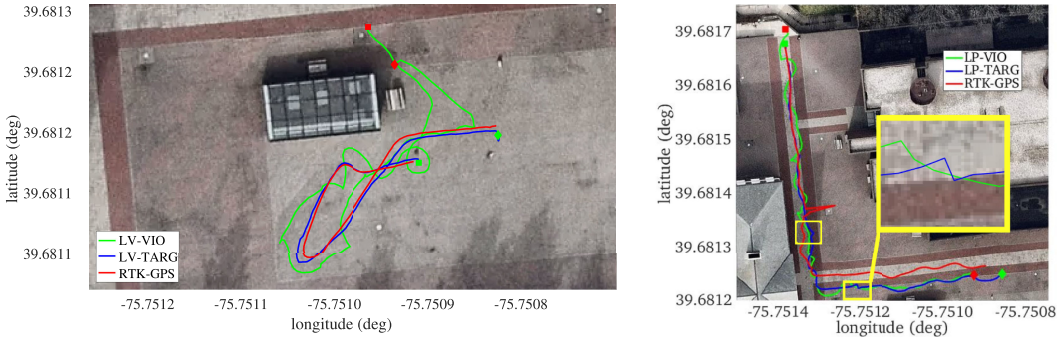

Fig. 5. Top-down views of the trajectories generated by the proposed VILTT estimator in two real-world experiments. Note that only the RTK-GPS measurements (red) of the target were available. (a) The VIO (green) traveled a total distance of 85 meters, with the target (blue) moving 47 meters. The start and end positions are denoted by a square and diamond of opposite colors, respectively. (b) In this scenario, the jump in the RTK groundtruth is due to GPS multipath errors from nearby buildings. The proposed VILTT successfully tracks the target over its 81 meter long trajectory, despite losing sight multiple times (areas within the yellow boundary boxes). This re-observance typically causes a (possibly) large target correction (see blown-up box).

yield better results, which will be investigated in future research. We note that in this work, the noises were chosen to attempt to capture the motions being executed by the simulations, rather than directly simulating target motion noise by drawing from a known distribution.

VI. Experimental Results

In our real-world tests, to prove the concept of the proposed VILTT estimator, we simplified the target segmentation through the use of attached fiducial markers (see Fig. 4) around the target that can be reliably detected so that we can focus on the evaluation of the target estimation accuracy. Of course, more sophisticated target detection (e.g., based on deep learning) can be leveraged for a given application domain, which however is out of the scope of this work that is focusing on the estimator design for the VILTT system. We calculated a bounding box of the target as the min/max image coordinates of the extracted fiducial markers, and updated the bounding box over a small sliding window to robustify it to failures in tag extraction. This information was used to ensure that features were properly classified as either on or off the moving target. We found in this experimental setup that the target had to be within two or three meters to the camera in order to provide reliable tag extraction, otherwise, the system was not able to reliably detect the target. After extraction and computation of the target bounding box, we performed KLT tracking of sparse feature points [30]. Specifically, we initialized new features using FAST [31] feature extraction in a uniform grid pattern. KLT tracking was performed from both left-left and right-right camera images temporally, as well as left-right for stereo matches. We performed 8-point RANSAC on the static features and rejected measurements if they failed any of the three tracks. When features were lost or reached their maximum track length, they were processed using the MSCKF update step (either static or target-based). In this experiment, we maintained the features corresponding to the center and top left corner of each fiducial tag in the state vector. We should point out that while we used fiducial tag corners in our filter, we did *not* use any knowledge of the tag's size in our estimation, rather we relied on them *only* for target detection. Images were collected at a rate of 20 Hz using a VI-Sensor [32], while our system was able to process these images at a faster rate (on average 30 Hz) on an Intel i7-4700MQ CPU @ 2.40GHz. For comparison, the standard MSCKF takes 0.017 seconds per frame, while the proposed method takes an average of 0.031 seconds per frame (approximate $1.8\times$ computational increase).

We evaluated the proposed VILTT estimator on one of datasets that we collected outdoor on the University of Delaware campus, where an RTK-GPS was attached to the target robot (Turtlebot) to allow for groundtruth comparison and the target traveled on a semi-planar brick surface. The tracking robot (IMU/camera platform) was first initialized without the target in view, and after 31 seconds of motion, the target entered the view of the moving platform and was successfully initialized. Multiple loops around the target were made to showcase the ability to still localize the target without observation of the representative point. The tracking robot, target, and RTK-GPS paths are overlaid onto satellite imagery as shown in Fig. 5a. The first 50 seconds of the target and RTK-GPS trajectories were used to compute a "best fit" transformation to align the two frame of references. For clarity, we present only the local velocity model in Figure 5a, but all three models were able to successfully track the target. Clearly, the target position estimates closely follow the path of the RTK-GPS. We also utilized the fiducial tags to compute the error in the estimated relative position between the IMU and target during times when the representative tag is visible. Although we note that "ground-truth" relative positions from tag extraction might not be very accurate, we still found that our estimator yielded an RMSE of 0.044 m.

In addition, we performed a second experiment wherein the target executed a mostly straight trajectory with small sinusoidal perturbations, followed by a sharp turn and continued motion. In this scenario, loss of sight of the target across a few seconds occurred due to actively facing the camera away from the target multiple times along the trajectory, with the maximum time of lost tracking being 7.4 seconds. While this sometimes led to large target updates upon reobserving the Turtlebot after a long period of pure propagation, the proposed VILTT was still able to perform *continuous* estimation of the target, with the resulting trajectory from using the local planar model shown in Fig. 5b. However, we do note that during our experiments we found that viewing the target from a new angle (no previously seen target features are detected) upon reobservation after a period of tracking loss such that new target features are initialized with extremely poor estimates, could lead to divergence of the filter upon reobservation. If this behavior is detected (such as through Mahalanobis distance testing of incoming target measurements), we recommend that the target be reinitialized. For this trajectory,

the RMSE for the relative position from the estimator referenced to the output value from tag extraction was 0.06 m.

VII. Conclusions and Future Work

In this letter, we have developed a tightly-coupled state estimation framework that cooperatively estimates a 3D rigid-body moving target and visual-inertial sensor platform, which was shown to achieve accurate realtime performance. In particular, we represented the rigid object with both the 6DOF target pose (orientation and position) and the features attached to it, allowing for a robust tracking of the object even when the representative point feature is not observed. Moreover, to encompass a broad range of realistic target tracking scenarios, three stochastic target motion models have been introduced, along with a thorough observability analysis of the corresponding linearized VILTT systems. Monte-Carlo simulations and real-world experiments have validated the feasibility of the proposed VILTT estimator.

In the future, we plan to investigate proper characterization of the target motion noises, as these, if chosen poorly, can actually lead to degradation of the resulting IMU estimate due to the introduction of inconsistency into the filter, (despite the expected increase in accuracy due to tightly-coupling the localization and tracking [33]). We note, however, in the extreme case where good estimates for these noise parameters are not available, the proposed system can be turned into a decoupled estimator by setting all Jacobians with respect to the IMU and its clones to zero for all target measurements. While this should theoretically offer decreased accuracy compared to properly characterized tightly-coupled estimation, it can protect the IMU estimates from inconsistency. In addition, we plan to extend this system to perform multi-object target tracking along with target detection and automatic model selection.

References

[1] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *Int. J. Robot. Res.*, vol. 26, no. 9, pp. 889–916, Sep. 2007.

[2] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

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

[4] 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.*, Rome, Italy, 2015.

[5] 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.

[6] Z. Huai and G. Huang, "Robocentric visual-inertial odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Madrid, Spain, 2018, pp. 6319–6326.

[7] Y. Yang, J. Maley, and G. Huang, "Null-space-based marginalization: Analysis and algorithm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, 2017, pp. 6749–6755.

[8] M. Li and A. Mourikis, "Online temporal calibration for Camera-IMU systems: Theory and algorithms," *Int. J. Robot. Res.*, vol. 33, no. 7, pp. 947–964, Jun. 2014.

[9] C. Guo, D. Kottas, R. DuToit, A. Ahmed, R. Li, and S. Roumeliotis, "Efficient visual-inertial navigation using a rolling-shutter camera with inaccurate timestamps," in *Proc. Robot.: Sci. Syst.*, Berkeley, CA, USA, 2014.

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

[11] 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.*, Rome, Italy, 2015.

[12] K. Eckenhoff, P. Geneva, and G. Huang, "High-accuracy preintegration for visual-inertial navigation," in *Proc. Int. Workshop the Algorithmic Foundations Robot.*, San Francisco, CA, USA, 2016.

[13] K. Eckenhoff, P. Geneva, and G. Huang, "Continuous preintegration theory for graph-based visual-inertial navigation," 2018, arXiv:1805.02774.

[14] T. Qin, P. Li, and S. Shen, "VINS-MONO: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.

[15] M. Chojnacki and V. Indelman, "Vision-based dynamic target trajectory and ego-motion estimation using incremental light bundle adjustment," *Int. J. Micro Air Veh.*, vol. 10, no. 2, pp. 157–170, 2018.

[16] V. Indelman and F. Dellaert, "Incremental light bundle adjustment: Probabilistic analysis and application to robotic navigation," in *New Develop. in Robot Vision*. Berlin, Germany: Springer-Verlag, 2015, vol. 23, pp. 111–136.

[17] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *Proc. Int. Conf. Intell. Robots Syst.*, Deajeon, Korea, 2016, pp. 446–453.

[18] H. Lim and S. N. Sinha, "Monocular localization of a moving person onboard a quadrotor MAV," in *Proc. Int. Conf. Robot. Autom.*, Seattle, WA, USA, 2015, pp. 2182–2189.

[19] P. Li, T. Qin, and S. Shen, "Stereo vision-based semantic 3-D object and ego-motion tracking for autonomous driving," in *Proc. 15th Eur. Conf. Comput. Vision*, Munich, Germany, 2018, pp. 664–679.

[20] K. Qiu, T. Qin, H. Xie, and S. Shen, "Estimating metric poses of dynamic objects using monocular visual-inertial fusion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Madrid, Spain, 2018, pp. 62–68.

[21] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," Dept. Comp. Sci. Eng., Univ. Minnesota, Minneapolis, MN, USA, Tech. Rep. 2005-002, Mar. 2005.

[22] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*. Reston, VA, USA: American Institute of Aeronautics and Astronautics, 1997.

[23] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, "Vins on wheels," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 5155–5162.

[24] G. Huang, "Improving the consistency of nonlinear estimators: Analysis, algorithms, and applications," Ph.D. dissertation, Dept. Comput. Sci. and Eng., Univ. of Minnesota, Minneapolis, MN, USA, 2012.

[25] A. Martinelli, "Visual-inertial structure from motion: Observability and resolvability," in *Proc. Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, 2013, pp. 4235–4242.

[26] Y. Yang and G. Huang, "Aided inertial navigation with geometric features: Observability analysis," in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, Australia, 2018, pp. 2334–2340.

[27] Y. Yang, K. Eckenhoff, P. Geneva, and G. Huang, "Observability analysis for tightly-coupled visual-inertial rigidbody target tracking," Univ. Delaware, Newark, DE, USA, Tech. Rep. RPNG-TARGET, Jul. 2018. [Online]. Available: http://udel.edu/ yuyang/downloads/tr_target.pdf

[28] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "RotorS—A Modular Gazebo MAV Simulator Framework," in *Robot Operating System (ROS): The Complete Reference (Volume 1)*. Cham, Switzerland: Springer International Publishing, 2016, pp. 595–625.

[29] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. Int. Conf. Intell. Robots Syst.* Sendai, Japan, 2004, pp. 2149–2154.

[30] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. Artificial Intell.*, Vancouver, BC, Canada, 1981, pp. 674–679.

[31] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, Jan. 2010.

[32] 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.

[33] F. M. Mirzaei, A. I. Mourikis, and S. I. Roumeliotis, "On the performance of multi-robot target tracking," in *Proc. IEEE Int. Conf. Robot. Autom.*, Rome, Italy, 2007, pp. 3482–3489.