Visual-Inertial Teach & Repeat for Aerial Robot Navigation

Matias Nitsche[1] Facundo Pessacg[1] Javier Civera[2]

Abstract— **This paper presents a Teach & Repeat (T&R) algorithm from stereo and inertial data, targeting Unmanned Aerial Vehicles with limited on-board computational resources. We propose a tightly-coupled, relative formulation of the visual-inertial constraints that fits the T&R application. In order to achieve real-time operation on limited hardware, we constraint it to motion-only visual-inertial Bundle Adjustment and solve for the minimal set of states. For the repeat phase, we show how to generate a trajectory and smoothly follow it with a constantly changing reference frame. The proposed method is validated with the sequences of the EuRoC dataset as well as within a simulated environment, running on a standard laptop PC and on a low-cost Odroid X-U4 computer.**

I. Introduction

Mimicking a certain trajectory which has been previously followed by a mobile sensor platform is a desirable robotic capability with clear applications; such as structure inspection, environment monitoring or sample-return missions in planetary exploration. This problem is commonly referred to as *teach and repeat* (T&R) navigation.

In these scenarios, localization usually needs to be solved internally by the robot in the absence of reliable external position information such as GPS. Simultaneous Localization and Mapping (SLAM) is a widely adopted solution for GPS-denied navigation, allowing to estimate the robot pose for use in the control loop. However, SLAM typically uses an *absolute* formulation, where all robot poses and landmark positions are referred to a single privileged reference frame (generally the initial robot pose). This choice imposes global map consistency, which cannot be generally guaranteed due to pose estimation drift. Thus, most solutions expect frequent loop-detection followed by global pose-graph relaxation, which becomes very costly for large-scale scenarios.

If the expectation of optimal path-planning is abandoned, global map consistency is actually not required [1]. In other words, only local consistency is really necessary. In fact, it is possible to employ a *relative* formulation of the problem, where poses are expressed as a transformation relative to other nearby pose while landmarks positions are expressed in the initial observing coordinate frame. As a result, constant-time loop-closing becomes possible, obtaining a more efficient approach [2].

The computational cost of the navigation solution is of particular importance for payload-limited hardware platforms, such as small aerial robots. For this reason, relative localization approaches become an attractive solution. On the other hand, the visual-inertial combination is a very convenient sensor choice for agile and small robots, due to its low power demand, cost, size and weight. Inertial data can capture brisk motions, while visual data refers to external features and can remove drift.

Motivated by these two aspects, in this work we present a simple and efficient visual-inertial teach and repeat (VI-T&R) method from stereo and inertial data. Our main contribution is a relative tightly-coupled keyframe-based formulation of the problem, the first one in the literature to our knowledge. We also propose several adaptations for resource-constrained hardware, being our approach suitable for small aerial vehicles. We demonstrate the performance and potential of the proposed approach with experimental results in the EuRoC dataset and in simulation environments.

II. Related Work

The literature on T&R navigation has mostly focused on terrestrial robots equipped with either laser [3] or visual sensors [4]. Works that address aerial navigation are more scarce.

In [5] a proof-of-concept for the T&R navigation of an aerial robot is presented. The proposed method uses a downward-looking camera for pose tracking based on a planar floor assumption. In following work [6], the T&R method was adapted for the case of fixed-wing aerial platforms. However, only offline processing is performed and GPU hardware is required. Moreover, closed-loop control and trajectory planning are left as future work by the authors. More recently, a VI-T&R approach for aerial robots has been proposed in [7], involving both tightly-coupled and loosely-coupled estimators running together. Real-time performance is demonstrated in experiments using a powerful Intel i7 processor. The authors of this work state that for achieving navigation, loop-closing and global bundle-adjustment are performed between teach and repeat phases.

In other related works, the benefits of the relative formulation of the problem are identified and embraced at various degrees. In [8], while following the usual approach of referencing poses w.r.t the initial coordinate frame, the benefit of expressing landmark positions relative to the observation frame is highlighted. Moreover, authors argue in favor of not fixing the initial pose of state estimation window (typically done to remove the gauge freedom of the solution) for the purpose of avoiding an unbounded growth of the uncertainty of the states, which introduces linearization errors. Similarly, [9] employ the concept of *anchor nodes* for the same purpose, referencing all information to the initial

[1] Instituto de Ciencias de la Computación, ICC-UBA-CONICET, Argentina, {mnitsche,fpessacg}@dc.uba.ar

[2] I3A, Universidad de Zaragoza, Spain, jcivera@unizar.es

978-1-7281-3605-9/19/$31.00 ©2019 IEEE

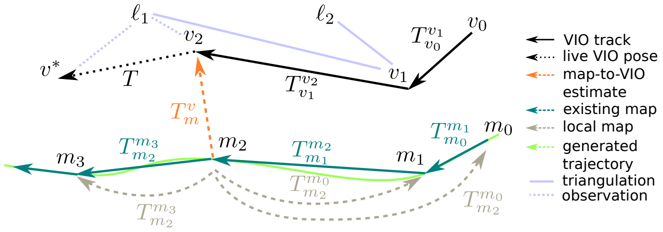


Fig. 1: Map structure: VIO estimates the transformations $T^{v_i+1}_{v_i}$ between keyframes. During the *repeat* phase, a relative transform T^v_m between the active VIO keyframe and the prior map is estimated. A local Euclidean map can be built combining transforms, and a smooth trajectory to be repeated can be generated.

pose of the estimation window. Finally, in [10] visual-inertial estimation over a purely-relative map is presented.

Only very few works in the literature address stereo visual-inertial state estimation on resource-constrained platforms. In [11], a monocular visual-inertial estimation method is presented, demonstrating real-time operation on low-cost hardware by performing motion-only bundle-adjustment (map structure is fixed). In [9] an IEKF solution targeting a lower power ARM computer is proposed. However, the computation time is not reported.

III. Tightly Coupled Visual-Inertial Teach & Repeat

The proposed VI-T&R method is composed of two stages. During the *teach* phase, the robot is localized using stereo visual-inertial odometry (VIO), while also recording the poses of relevant keyframes and observed 3D landmarks into a map. Subsequently, during the *repeat* phase, the robot also localizes itself using VIO but refers its pose with respect to the prior map, which allows to follow the previous trajectory.

The estimated map adopts a relative formulation, similarly to [4], [12]. It is in essence a graph of robot poses (or *keyframes*), where edges represent relative transformations in SE(3). 3D visual landmarks are represented relative to each local frame. The map structure is illustrated in Fig. 1.

A. Visual-Inertial Odometry

The goal of the VIO in the *teach* phase is to estimate the current camera pose $\mathbf{T}$, relative to the last keyframe, and the n most recent inertial states. The full state is defined as

$$\mathbf{x} = \{\mathbf{T}, \mathbf{v}_1, \ldots, \mathbf{v}_i, \ldots, \mathbf{v}_{n-1}, \mathbf{b}^1_g, \ldots, \mathbf{b}^i_g, \ldots, \mathbf{b}^n_g, \mathbf{g}^*\} \tag{1}$$

where $\mathbf{T} = \mathbf{T}^n_{n-1}$ represents the current camera pose w.r.t. last keyframe, $\mathbf{g}^*$ is the normalized gravity vector (we assume known gravity magnitude) expressed in the IMU coordinate frame of the oldest state in the estimation window, $\mathbf{v}_i$ is the velocity expressed in camera frame i and $\mathbf{b}^i_g$ the gyroscope bias for state i.

We choose to include the gravity vector as part of the state given that, under the relative formulation of the problem, we wish to avoid representing poses with respect to a fixed coordinate system, such as a gravity-aligned reference frame. This also allows the correction of the gravity direction estimate, in contrast to most works where it is initially obtained and then used to align the working frame to it. We also use a minimal 2DoF parametrization which removes the unobservable dimensions of global yaw and position.

On the other hand, since the observability of both the gravity vector and the accelerometer biases cannot be guaranteed unless the estimation window includes rich rotations [13], we choose to obtain an initial accelerometer bias via manual calibration and we keep it fixed. While the accelerometer bias might drift over time, its influence is smaller than that of the gyroscope bias and gravity direction. We also exclude the last velocity state $\mathbf{v}_n$, since it is only observable from inertial measurements. Thus, we estimate velocities up to $\mathbf{v}_{n-1}$ and obtain $\mathbf{v}_n$ from inertial integration.

Finally, in order to keep the computational cost low, we perform motion-only Bundle Adjustment in a single-thread, estimating $\mathbf{x}$ at frame rate. In other words, landmarks are not adjusted after initial triangulation. Moreover, only the current camera pose $\mathbf{T}^n_{n-1}$ is estimated, while remainder $\mathbf{T}^{i+1}_i$ transforms are kept fixed. The remainder of variables (linear velocity, gyroscope bias and gravity) are left unfixed. Summing up, the objective function for VIO is defined as:

$$J(\mathbf{x}) = \sum_{m=1}^{M} \left\| \mathbf{e}^V_{m,n-1,n} \right\|_{(\mathbf{Q}^V_m)^{-1}} + \sum_{i=0}^{n-1} \left\| \mathbf{e}^I_{i,i+1} \right\|_{(\mathbf{Q}^I_{i,i+1})^{-1}} \tag{2}$$

where $\mathbf{e}^V_{m,n-1,n}$ is the visual residual related to the most recent camera frame and the m-th landmark (of the most recently observed M landmarks in previous keyframes), $\mathbf{e}^I_{i,i+1}$ is the inertial residual related to pair i and $i+1$ of inertial states, and $\mathbf{Q^V}^{-1}_m$, $\mathbf{Q^I}^{-1}_{i,i+1}$ corresponding information matrices.

Visual Residuals: The visual portion of the VIO problem involves the minimization of a stereo reprojection residual obtained from a pair of matching landmarks $\boldsymbol{\ell} \in \mathbb{R}^3$ and image-feature pair $\mathbf{y} \in \mathbb{R}^4$,

$$\mathbf{e}^V_{m,i,j} = \mathbf{y}^j_m - \pi(\mathbf{T}^j_i, \boldsymbol{\ell}^i_m) \tag{3}$$

where $\pi(\cdot)$ is the left-right stereo-projection function. $\mathbf{T}^j_i$ represents the camera pose j w.r.t coordinate frame i, $\boldsymbol{\ell}^i_m$ the m-th landmark, expressed in frame i and $\mathbf{y}^j_m$ the observation of $\boldsymbol{\ell}^i_m$ from frame j.

On a relative framework, obtaining $\mathbf{T}^j_i$ for keyframes that are not directly linked involves composing the kinematic chain from i to j at every solver iteration [12]. However, given that we only optimize the last transform $\mathbf{T}^n_{n-1}$, the kinematic chain is fixed. Thus, we can precompute the coordinates of tracked landmarks w.r.t. to the latest coordinate frame ($\boldsymbol{\ell}^{n-1}_m$) every time a new keyframe is added.

The covariance matrix $\mathbf{Q^V}_m$ is defined as:

$$\mathbf{Q^V}_m = \mathbf{Y}_m + \mathbf{J}_\pi \mathbf{R}^i_{i+1} \Phi_m \mathbf{R}^i_{i+1}{}^\top \mathbf{J}_\pi^\top \tag{4}$$

where $\mathbf{Y}_m = \sigma^2_y \mathbf{I}$ relates to noise in feature position $\mathbf{y}$, $\mathbf{J}_\pi$ is the Jacobian of the projection function π and $\mathbf{R}^{i+1}_i$

the camera orientation. The covariance matrix Φ_m, corresponding to landmark $\boldsymbol{\ell}_m$, is initially obtained via first-order propagation of the stereo triangulation and later transformed after keyframe insertion to maintain its representation w.r.t. frame i. Including this term is beneficial under a motion-only estimation and represents better the resulting pose uncertainty [14], [4].

Inertial Residuals: We use the preintegrated residuals of [15], adapted to estimate relative motions. Inertial preintegration adopts a body-centered frame, removing the dependency from the variables being optimized. Thus, integration does not have to be repeated for each iteration. This results in a straightforward formulation in the context of relative localization as it actually reduces to a comparison between the relative rotation and translation obtained using both vision and inertial information. The inertial residual $\mathbf{e}^I_{i,i+1} = \left(\mathbf{e}_\mathbf{R}, \mathbf{e}_\mathbf{v}, \mathbf{e}_\mathbf{p}, \mathbf{e}_{\mathbf{b_g}}\right)$ is defined as:

$$\begin{aligned}
\mathbf{e}_\mathbf{R} &= \log\left(\varDelta\mathbf{R}(\mathbf{b}^g_i)^\top \mathbf{R}^j_i\right)^\vee \\
\mathbf{e}_\mathbf{v} &= \mathbf{R}^C_I(\mathbf{v}_j - \mathbf{v}_i) - \mathbf{g}_i \Delta t - \varDelta\mathbf{v}({\mathbf{b_g}}^i, \mathbf{b_a}) \\
\mathbf{e}_\mathbf{p} &= \mathbf{t}^j_i - \mathbf{R}^C_I \mathbf{v}_i \Delta t - \frac{1}{2}\mathbf{g}_i \Delta t^2 - \varDelta\mathbf{p}({\mathbf{b_g}}^i, \mathbf{b_a}) \\
\mathbf{e}_{\mathbf{b_g}} &= {\mathbf{b_g}}^j - {\mathbf{b_g}}^i
\end{aligned}$$

where $\varDelta\mathbf{R}, \varDelta\mathbf{v}, \varDelta\mathbf{p}$ are the preintegrated IMU measurements (using known $\mathbf{b_a}$ and corrected using jacobians w.r.t. $\mathbf{b_g}$ variation during optimization), $\{\mathbf{R}^j_i, \mathbf{t}^j_i\}$ stands for the relative transformation between key-frame i and j, already expressed in the IMU frame and $\mathbf{R}^C_I$ the orientation of the camera in the IMU frame. Note that the gravity vector is expressed in frame i by applying the fixed transform $\mathbf{R}^i_1$ (precomputed before optimization). ${\mathbf{Q}^\mathbf{I}}_{i,i+1}$ can be computed incrementally as in [15]. We also add to this term the uncertainty of fixed variables such as $\mathbf{t}^j_i$, $\mathbf{R}^j_i$ and $\mathbf{R}^i_1$, which helps to mitigate the possible side-effects of early fixation.

State Initialization: For inertial state estimation we assume $\mathbf{b_a}$ known (and thus is fixed). In practice, a reasonable estimate for $\mathbf{b_a}$ can be obtained, for example, by obtaining average accelerometer readings at rest on each axis in opposing directions. Afterwards, an initial value for $\mathbf{g}$ can be obtained by subtracting $\mathbf{b_a}$ from these average readings. State initialization then continues by running both visual-only (VO) and visual-inertial (VI) estimators in parallel. First, the VI estimator initializes $\mathbf{b_g}$ using $\mathbf{T}$ and $\mathbf{v}$ as obtained from VO estimation. Next, $\mathbf{b_g}$ is kept fixed while linear velocities and gravity vector are refined. Once these have converged, initialization is complete and $\mathbf{T}$ is estimated from the initialization seed.

Pose prediction: While the VIO algorithm runs close to camera rate, for the purpose of closed-loop control of an aerial robot, we also run a pose prediction thread at IMU rate by integrating angular velocity and linear acceleration.

Active matching: For landmark tracking, every time a new VIO keyframe (KF) is inserted, we first transform the previously tracked landmarks, expressed in current KF i, towards the new KF j by applying the uncertain transform (see [4]). The 3D landmark and its covariance matrix are projected to the image and only features which fall inside the corresponding high-confidence ellipse are considered. A match is then established with its nearest neighbour in descriptor-space, up to a maximum distance threshold. As VIO produces high-confidence estimates, this approach reduces false positives and greatly limits the search area.

B. Reference map localization

In order to follow a previously taught trajectory, during the *repeat* phase it is first necessary to establish a relative transform $\mathbf{T}^i_{i^*}$ between the last VIO reference keyframe i and closest keyframe i^* in the reference map (see Fig. 1). In order to decouple VIO from the reference map localization, these steps are solved independently.

To obtain $\mathbf{T}^i_{i^*}$, we perform visual-only localization against the reference map by minimizing the reprojection error between observations in keyframe i and the corresponding reference map landmarks observed in keyframe i^*, with the following cost function:

$$J(\mathbf{x}) = \sum_{m=1}^{M} \left\|\mathbf{e}^V_{m,i^*,i}\right\|_{(\mathbf{Q}^V_{m,i^*,i})^{-1}} + \left\|\mathbf{T}^i_{i^*} \ominus \check{\mathbf{T}}^i_{i^*}\right\|_{(\mathbf{Q}_{\check{\mathbf{T}}^i_{i^*}})^{-1}} \quad (5)$$

where $\check{\mathbf{T}}^i_{i^*}$ is a prior for the unknown transform.

Initially (and whenever relocalization is triggered), i^* is unknown and is found by Bag-of-Words [16] (BoW) matching between observations in i and those of the complete reference map (during map construction we store the BoW representation for each keyframe). The prior $\check{\mathbf{T}}^i_{i^*}$ is then obtained via PnP estimation under a RANSAC scheme using correspondences established via nearest-neighbors in descriptor-space, considering only candidates with matching BoW feature clusters.

In subsequent steps, after a new VIO keyframe j is inserted, the prior is first updated to represent the pose of j w.r.t. keyframe i^* as $\check{\mathbf{T}}^i_{i^*}\mathbf{T}^j_i$. The keyframe j^* closest to the keyframe j in the *local map* starting from i^* is set as the new reference. The new prior $\check{\mathbf{T}}^j_{j^*}$ is then obtained as $\mathbf{T}^{i^*}_{j^*}\check{\mathbf{T}}^j_{i^*}$. Finally, optimization (5) is performed to obtain the resulting $\mathbf{T}^j_{j^*}$. Landmark matches between j and j^* are established by projecting landmarks observed in j^* to j, using $\check{\mathbf{T}}^j_{j^*}$, and the previously described active-matching procedure.

If a minimum number of correspondences fails to be established, indicating a badly predicted prior or that an unmapped area is being explored, relocalization based on BoW is triggered. Until successfully relocalized, $\check{\mathbf{T}}^j_{j^*}$ is used as the localization result, resulting in purely predicted map-to-VIO pose.

C. Path following

In order to repeat the taught path, a continuous trajectory needs to be obtained to allow for smooth path following. This is particularly challenging when employing a relative formulation, since there is a continuously changing reference frame. To address this, we proceed as follows. We first compute a smooth trajectory from the neighboring poses of

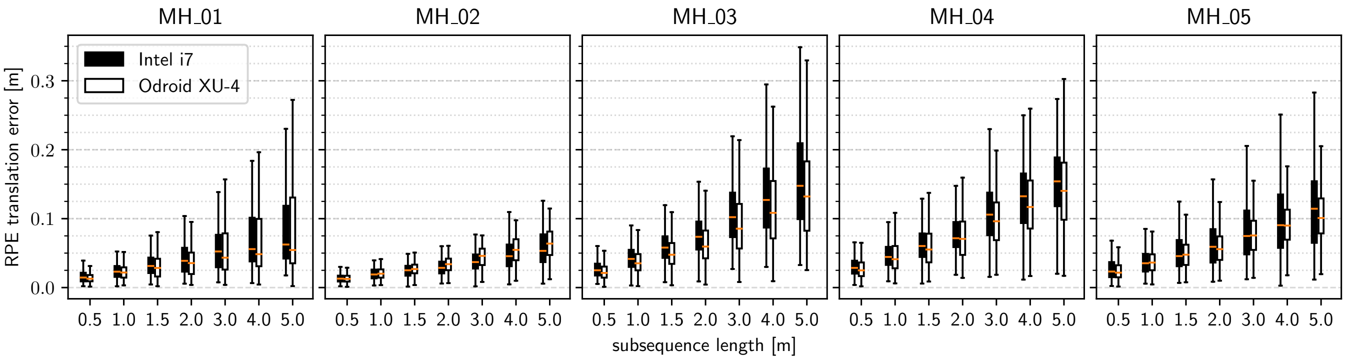


Fig. 2: Translation component of RPE measured over all possible subsequences of different lengths from 0.5 m to 5.0 m, for the map built during the *teach* phase for each EuRoC dataset MH sequence, for Laptop i7 and Odroid XU-4 computers.

keyframes i^*. To do so, we re-use the local map built during localization. Linear and angular velocities associated with each keyframe (as estimated during the *teach* phase) are used as constraints for a polinomial trajectory generator [17]. For the desired arrival time at each keyframe we use the travel time relative to the initial pose of the local map, as observed during the teach phase. As a result, the obtained trajectory closely resembles the one originally taught.

To follow the generated trajectory, we first obtain a setpoint by sampling the polinomial with a time *cursor* t, which advances in real-time. The difficulty here lies in that, whenever the reference keyframe i^* is redefined, t cannot be used to sample the new trajectory, since $t = 0$ now represents a different pose. To handle this situation, the trajectory is built from a local map including both past and future poses such that an overlap between the previous and current trajectories exists. As the relative travel time of each segment is fixed, we obtain the new cursor t' as $t - t_0$, where t_0 is the arrival time of the initial node of the new trajectory w.r.t. the old trajectory's initial node.

Finally, based on the map-localization result and the desired setpoint, we compute a pose error which is used as input to a closed-loop cascaded PID controller, allowing to compensate for external perturbations such as wind.

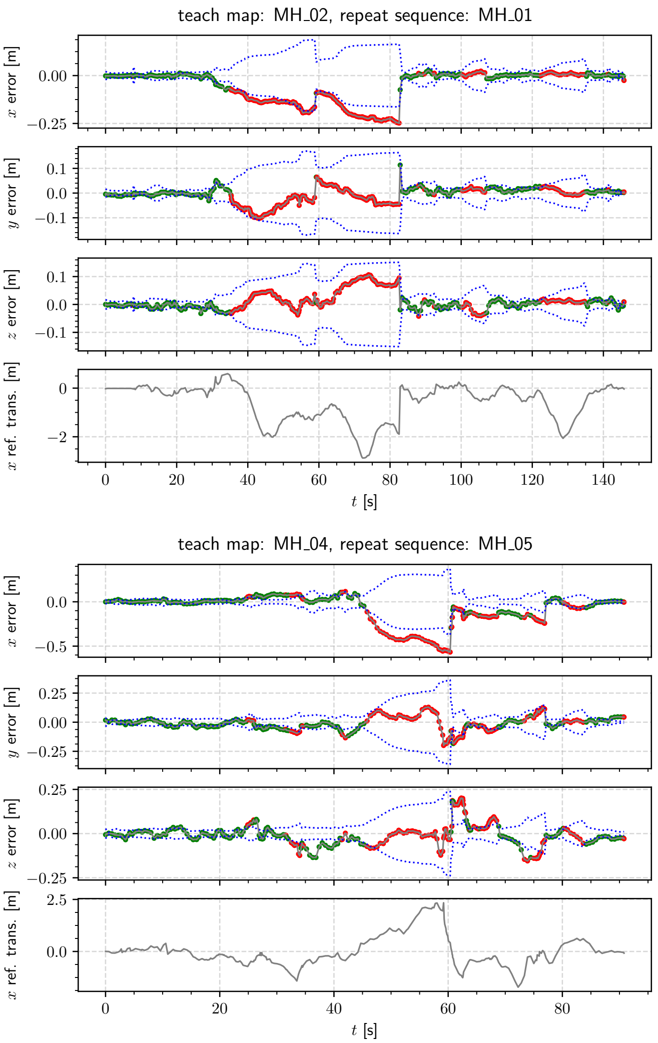


Fig. 3: Translation errors for map localization on EuRoC dataset and 3σ uncertainty region (dotted blue line). Green indicates successful localization, and red VIO predictions. Bottom plot corresponds to groundtruth longitudinal distance between VO and map reference keyframes.

IV. Experimental Results

The proposed method was implemented in ROS, using the Ceres [18] solver for state estimation. As our embedded hardware platform we use a low-power ARM Odroid-XU4 single-board computer, while in our laptop we have an Intel Core i7-3632QM processor running at 2.2 GHz. To reduce the image processing overhead on the Odroid, we use SIMD-optimized ORB and BRIEF algorithms [19]. For experiments, we used the EuRoC MAV dataset [20] to assess localization accuracy, while rotorS [21] and Gazebo simulator were used to analyze navigation performance during repeat phase. We also show the computational cost of the method when running both on the embedded hardware and a laptop computer as reference.

A. VIO localization

Under the T&R problem formulation we only care about local map consistency and thus we measure relative-pose error (RPE) [22] to assess VIO accuracy. Based on a typical maximum local map size, we measure RPE over subsequences of contiguous map keyframes, of various lengths up to $5\mathrm{m}$. These measurements were performed over the MH series of EuRoC dataset sequences, given the challenging image conditions of the V series (large exposure changes, significant blur, etc.). We run experiments on both hardware platforms for the purpose of establishing the effect of limited computational power on localization accuracy (since frame dropping implies less frequent corrections and longer prediction intervals). In order to initialize the IMU we use the ground truth value of the accelerometer bias at dataset startup. On Fig. 2 we report the RPE measurements.

Analyzing the results, we can see that for the maximum trajectory length, there's a best and worst-case median RPE error of $\sim 0.05\ \mathrm{m}$ and $\sim 0.15\ \mathrm{m}$, respectively. Moreover, it can also be seen that there is not a significant difference in accuracy between both platforms, demonstrating its low computational footprint.

B. Map-based Localization

To assess the accuracy of the map-based localization, we compare the estimated map-to-VIO keyframe transform to groundtruth, as a RPE (figure 3). For this experiment, we perform the teach phase using one of the EuRoC sequences and repeat using a different one. While there is a partial overlap between sequences, there are also significant non-overlapping areas. Thus, this experiment also serves to show re-localization ability and pose prediction accuracy using VIO, when deviating from the taught trajectory. We also overlay a 3σ uncertainty region to evaluate the estimator confidence. We color in red those poses which are the result of pure predictions, and in green those ones successfully matched to the map from the teach phase. Finally, we include the longitudinal distance between the chosen map reference keyframe and current VO keyframe.

From these experiments we can see that, if map localization is successful, translation errors are around $0.1\ \mathrm{m}$ in average. On the other hand, during exploratory periods, where camera pose is only predicted using VIO w.r.t. the last successful localization, worst-case errors are between $0.2\ \mathrm{m}$ and $0.5\ \mathrm{m}$ in respective experiments (mainly in the robot forward direction x). During these periods it is also possible to see that the distance (obtained from groundtruth) between VO and map reference keyframe considerably increases. This is due to the fact that the robot in fact deviates from the known path and this reference becomes distant. Moreover, in this situation the reference keyframe is obtained using the predicted pose, instead of global localization (III-B). Thus, map localization errors are expected to increase until relocalization occurs.

Finally, we can see that, in general, the estimated pose uncertainty is consistent with the errors except for brief periods of time. It should be noted, though, that a certain degree of overconfidence is expected given the use of naïve triangulation uncertainty propagation and similar approximations.

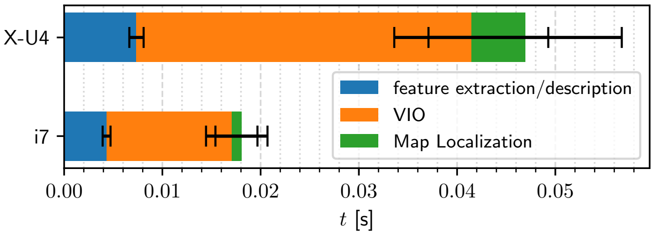


Fig. 4: Execution times of the algorithm on both hardware platforms, using `MH_01` EuRoC sequence.

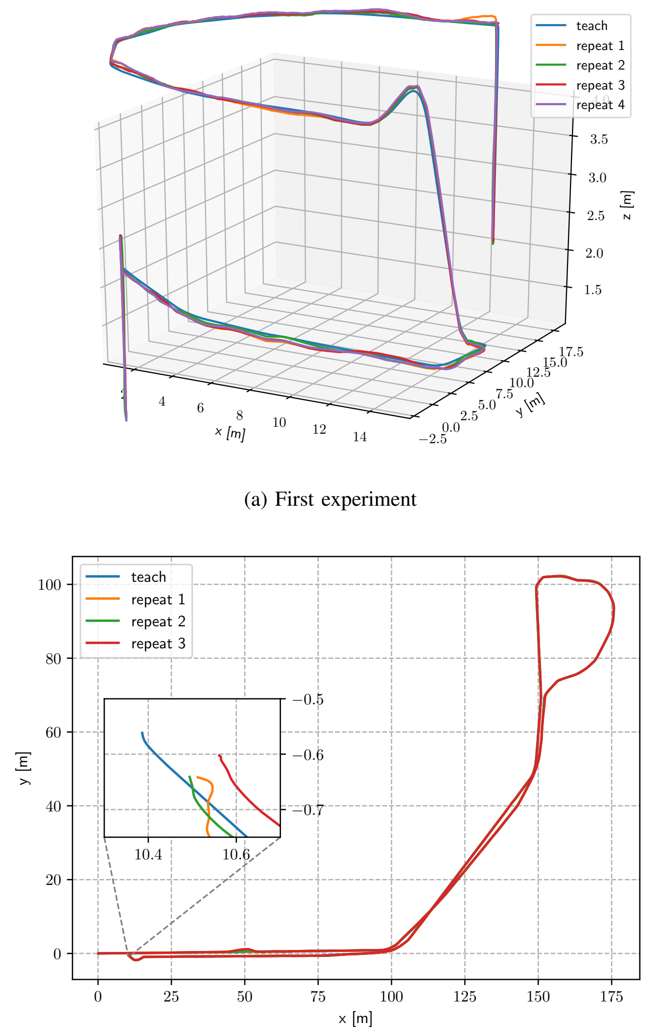
C. Computational Cost

In this section we show the mean time for feature extraction/description, VIO and map-localization tasks (Fig. 4), measured on both hardware platforms. As expected, there's a significant difference in computational capabilities of both platforms. On the Intel i7, the VIO task runs at around $50\ \mathrm{Hz}$, of which feature extraction represents near $2\ \mathrm{ms}$ of overhead. On the other hand, the computational times of the Odroid XU-4, while higher, are still within typical real-time response rates at around $25\ \mathrm{Hz}$ for the VIO task. We can also see the significant benefit of an optimized implementation of ORB/BRIEF algorithms. Finally, we can see that map-localization takes around $1\ \mathrm{ms}$ for the i7 and $2.5\ \mathrm{ms}$ on the Odroid. In our current implementation VIO and map-based localization are run sequentially but these tasks could be easily run in parallel. Finally, it should be emphasized that these processing rates correspond to state correction. Given that IMU is used for pose prediction in parallel to image-processing, states are predicted at IMU rate for high-frequency closed-loop control.

D. Navigation

In order to demonstrate how a previously taught map can smoothly be followed with the proposed method, achieving high accuracy even over a globally inconsistent map, we performed a teach phase over a simulated environment using the Gazebo/rotorS simulator followed by an autonomous repeat phase. For the experiment we simulate an AscTec Firefly hexacopter with a stereo VI sensor, configured at 20 frames per second and an IMU rate of 200 Hz. The "outdoor" rotorS simulated environment was used, consisting of a long road with houses and other structures, encompassing an area of around $100 \times 200\ \mathrm{m}$. For closed-loop control, we sample the smooth trajectory at the current time-cursor (III-C) and obtain desired position and yaw, along with their corresponding velocity and acceleration. This information is fed to a proportional controller [23] producing motor speed commands which are then sent to rotorS. In order to initialize the accelerometer bias, we obtain ground-truth value from simulator at startup. As rotorS models the corresponding random-walk process for IMU biases, drift from this initial value is also simulated.

For the first experiment, the drone was commanded to navigate around one of the houses, mimicking an inspection task, exploiting the 3D motion capability of the robot. For



(a) First experiment

(b) Second experiment

Fig. 5: Trajectories followed during teach and repeat phases running on simulator

the second experiment, a longer trajectory (around 400 m) covering the whole road was taught, demonstrating the capability of the method to return to its starting location with high accuracy. After the initial teach phase, the trajectories were repeated several times.

In figure 5 we show the trajectories followed during both stages of each experiment. In both cases, the robot is able to accurately follow the taught 3D trajectory. The average translation errors w.r.t. the reference keyframes are around 0.8 m and 0.13 m, respectively. For the second experiment in particular, the robot reaches its final goal within a 0.2 m in all instances of the repeat phase.

V. Conclusions

We have presented a simple and efficient visual-inertial T&R navigation method suitable for small unmanned aerial robots in applications such as 3D inspection tasks. The method was tested on challenging EuRoC dataset and demonstrated closed-loop behavior in simulation. Localization accuracy when running on embedded hardware is comparable to that obtained with a modern laptop computer. The resulting performance allows for accurate trajectory repetition of long trajectories and thus demonstrates the feasibility of the approach.

For future work we wish to precisely establish the cost and accuracy of full state estimation (map and past poses) compared to our reduced formulation. Finally, we will further examine the performance of our method when running onboard an aerial robot.

References

[1] R. Brooks, “Visual map making for a mobile robot,” in *ICRA*, 1985.

[2] C. Mei *et al.*, “RSLAM: A system for large-scale mapping in constant-time using stereo,” *IJCV*, vol. 94, no. 2, pp. 198–214, 2011.

[3] P. Krüsi *et al.*, “Lighting-invariant Adaptive Route Following Using Iterative Closest Point Matching,” *JFR*, vol. 32, no. 4, pp. 534–564, 2015.

[4] M. Paton *et al.*, “Bridging the appearance gap: Multi-experience localization for long-term visual teach and repeat,” in *IROS*, 2016.

[5] A. Pfrunder, A. P. Schoellig, and T. D. Barfoot, “A Proof-of-Concept Demonstration of Visual Teach and Repeat on a Quadrocopter Using an Altitude Sensor and a Monocular Camera,” in *CRV*, 2014.

[6] M. Warren *et al.*, “Towards Visual Teach and Repeat for GPS-Denied Flight of a Fixed-Wing UAV,” in *Field and Service Robotics*, M. Hutter and R. Siegwart, Eds., 2018, pp. 481–498.

[7] M. Fehr *et al.*, “Visual-Inertial Teach and Repeat for Aerial Inspection,” p. arXiv:1803.09650, Mar 2018.

[8] S. Leutenegger *et al.*, “Keyframe-based visual–inertial odometry using nonlinear optimization,” *IJRR*, vol. 34, no. 3, pp. 314–334, 2015.

[9] N. De Palézieux, T. Nägeli, and O. Hilliges, “Duo-VIO: Fast, light-weight, stereo Inertial Odometry,” *IROS*, 2016.

[10] N. Keivan, A. Patron-Perez, and G. Sibley, “Asynchronous adaptive conditioning for visual-inertial SLAM,” *STAR*, vol. 109, pp. 309–321, 2016.

[11] T. Qin and S. Shen, “Robust initialization of monocular visual-inertial estimation on aerial robots,” in *IROS*. IEEE, 2017.

[12] G. Sibley, “Relative Bundle Adjustment,” *Electronic Notes in Theoretical Computer Science*, vol. 220, no. 3, pp. 1–26, dec 2009.

[13] P. Batista, C. Silvestre, and P. Oliveira, “Necessary and sufficient conditions for the observability of linear motion quantities in strapdown navigation systems,” in *ACC*. IEEE, 2009.

[14] H. Strasdat, J. M. Montiel, and A. J. Davison, “Visual SLAM: Why filter?” *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[15] C. Forster *et al.*, “On-manifold preintegration for real-time visual–inertial odometry,” *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[16] R. Mur-Artal and J. D. Tardós, “Fast relocalisation and loop closing in keyframe-based slam,” in *ICRA*, 2014.

[17] C. Richter, A. Bry, and N. Roy, “Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments,” in *Springer Tracts in Advanced Robotics*, vol. 114, 2016, pp. 649–666.

[18] S. Agarwal, K. Mierle, and Others, “Ceres solver,” http://ceres-solver.org.

[19] C. Chatfield, “Real-time feature extraction on the Raspberry Pi and other ARM processors supporting NEON,” https://github.com/0xfaded/pislam, 2018.

[20] Burri *et al.*, “The euroc micro aerial vehicle datasets,” *Int. Journ. of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[21] F. Furrer *et al.*, “RotorS—A modular gazebo MAV simulator framework,” *Studies in Computational Intelligence*, vol. 625, pp. 595–625, 2016.

[22] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in *CVPR*. IEEE, jun 2012, pp. 3354–3361.

[23] T. Lee, M. Leok, and N. McClamroch, “Control of complex maneuvers for a quadrotor UAV using geometric methods on SE (3),” *arXiv preprint arXiv:1003.2005*, no. 3, pp. 1–32, 2010.