Real-Time Visual-Inertial Localization for Aerial and Ground Robots

Helen Oleynikova, Michael Burri, Simon Lynen and Roland Siegwart
Autonomous Systems Lab, ETH Zürich

***Abstract*— Localization is essential for robots to operate autonomously, especially for extended periods of time, when estimator drift tends to destroy alignment to any global map. Though there has been extensive work in vision-based localization in recent years, including several systems that show real-time performance, none have been demonstrated running entirely on-board in closed loop on robotic platforms. We propose a fast, real-time localization system that keeps the existing local visual-inertial odometry frame consistent for controllers and collision avoidance, while correcting drift and alignment to a global coordinate frame.**

We demonstrate our localization system entirely on-board an aerial and ground robot, showing a collaboration experiment where both robots are able to localize against the same map accurately enough to allow the multicopter to land on top of the ground robot. We also perform extensive evaluations for the proposed closed-loop system on ground-truth datasets from MAV flight in an industrial setting.

I. Introduction

Robots are useful for tasks that are too difficult, repetitive, or dangerous for a human to do, such as monitoring industrial plants or mapping earthquake-damaged buildings. While the utility of robots has increased substantially in recent years, most systems still rely on external sensing or heavily structured environments.

Recent advances in robot state estimation from cameras, especially when fused with inertial-measurement unit (IMU) data, are very promising for allowing robots to operate in mostly unstructured settings [1, 2, 3]. However, these visual-inertial simultaneous localization and mapping (SLAM) algorithms accumulate errors in position and heading over time due to sensor noise and modeling errors. While the effect of such drift is often insignificant for short-term operations, reliable long-term operation requires complementing SLAM with localization against a global reference frame. Most state of the art robotics systems use external sensing for these estimates - such as GPS or motion capture systems. However, such systems heavily restrict the utility of robots in real, unstructured, GPS-denied environments. A more versatile solution to handle such situations is to localize against previously built maps from onboard sensors.

In most of the previous work, localization and map co-registration was at least partially performed off-board on a centralized server [4, 5]. In these systems, either relevant parts of the global model are sent to the robot [6], or the server directly performs the actual localization [7, 8]. However, relying on a server connection for off-board localization leads to jumps due to connectivity loss, and reduces overall robustness and reliability of the system.

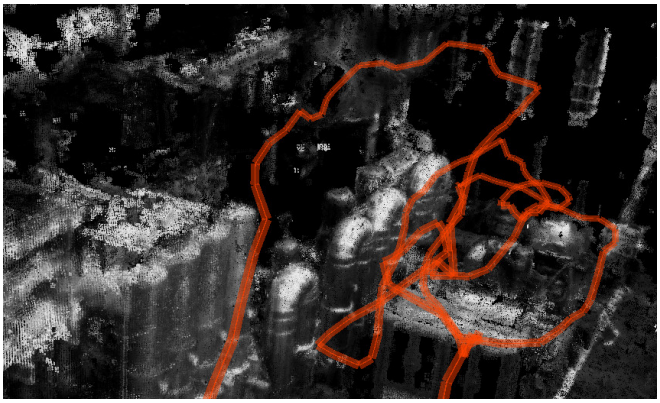


Fig. 1: A typical scenario for robot localization: performing inspection tasks with an MAV in an industrial environment. This is a 3D pointcloud reconstruction from stereo of the ETH Zürich Machine Hall, and an MAV flight trajectory from one of our evaluation datasets is shown.

Our work uses a similar approach in which we combine the visual-inertial sliding window SLAM [3] with an efficient loop-closure algorithm [9]. Both algorithms are computationally efficient enough to be run on-board the robot and thus remove the requirement of a server connection. Instead of adding constraints to the global model directly into the local SLAM formulation as proposed by Ventura *et al.*. [8] or Middelberg *et al.* [7], we aim for a decoupled approach in which we separate local pose estimation and localization to a global map. This approach allows us to keep the local map consistent with the output of the visual-inertial odometry while separately providing a current estimate of the robot's latest alignment within the global frame, called a baseframe transformation.

By only optimizing over this baseframe transformation, we are able to add new poses directly from the output of a visual-inertial odometry system. McDonald *et al.* [10] have previously used a similar approach for offline map co-registration by adjusting baseframe transformations of sub-maps. We extend their approach to run on-line and on-board robot platforms.

Heng *et al.* [11] present a robotic systems which is closest to the setup presented in this paper. Their work integrates vision-based pose estimation with building an octomap-based 3D representation of their environment. The resulting map is used by obstacle avoidance, planning, and exploration algorithms which run on-board the micro aerial vehicle (MAV), while localization to a global model is run off-board on a server. However, due to losses in connectivity to the server and jumps in the SLAM-optimized position estimate, they do not use the estimated alignment to the global model

on-board, but only for post-processing the maps.

Another similar system is from Schmid *et al.* [12], where they present a multi-camera system entirely on-board an MAV for indoor and outdoor environments. However, their lack of localization to an external reference limits the length of missions they are able to perform, as their only source of position information is a drifting reference frame. Other approaches fuse stereo, laser, and GPS on-board an MAV to mitigate drift and create more accurate estimates than with a single-sensor [13].

However, being able to localize against a reference map built with the same sensors as on-board the robot confers other advantages than just being able to mitigate drift. For example, having multiple robots registered to the same global frame allows collaboration between the two, without the robots necessarily being able to see or communicate with each other.

For example, Vidal *et al.* tackle the problem of localizing and merging maps built from helicopters flying at high altitude and ground robots [14]. Their work largely deals with creating different kinds of loop closures between multi-agent maps, such as from robots rendezvous, GPS fixes, and co-observations, but off-board and offline.

A clear application of multi-platform collaboration is given by ground and aerial robot teams for mapping earthquake-damaged buildings. Since ground robots have long battery life but very limited mobility in presence of debris, and MAVs have high mobility but very limited battery life, a ground-aerial robot team is perfect for such complex mapping tasks. Michael *et al.* demonstrate exactly such a collaboration [15]. Their experiment shows reconstructing voxel-based 3D maps by fusing the maps built by the aerial and ground robot from teleoperation. We believe that a major barrier to performing this task autonomously is getting accurate enough localization for the MAV to land back on the ground robot, which we aim to show in the experimental section of this paper.

The contributions of this work are as follows:

- An integrated visual-inertial odometry (VIO) and localization system that is computationally efficient enough to run in real-time on-board robots.
- A novel formulation of localization as a rigid baseframe alignment problem between a local map (VIO output frame) and a reference map (global coordinate frame).
- We perform evaluations of our complete system on ground truth data from a representative MAV industrial inspection scenario (Fig. 1), showing the ability of localization to significantly reduce estimator drift.
- Demonstration of autonomous ground robot and helicopter collaboration using the localization estimates from the proposed system.

II. Baseframe Based Localization

In this section, we present a novel formulation of the vision-based localization problem utilizing baseframe alignment. Our approach is well-suited for running on-board mobile robots, as it keeps the robot's local map aligned with the odometry frame, allowing the map to be built directly from VIO output. Simultaneously, we update an alignment to a global reference map, which can then be used for global planning in existing maps.

A. Map Representation

We represent our map of the robot's environment as a map consisting of several missions (sub-maps), each with a separate pose-graph. In our pose-graph representation, keyframes form vertices and IMU constraints represent edges between vertices. Vertices contain 2D keypoints (and their descriptors) as well as the 3D landmarks triangulated from keypoint tracks across keyframes. Each mission has its own baseframe which aligns it to the global coordinate frame ($_G\mathbf{T}_M$), with all vertices and landmarks in the mission represented in its local frame, as proposed in [10]. This representation allows us to easily align multiple missions together without having to change the poses of the vertices within a mission.

The starting point for our algorithm is the output of VIO, in our case the sliding window keyframe-based system proposed by [3]. The VIO system tracks 2D features through camera frames over time to triangulate 3D landmarks and establish correspondences between the frames. Using these constraints and measurements from the IMU, the system estimates the robot pose through non-linear optimization. However, due to computational constraints, only a very limited number of keyframes and landmarks can be kept in this optimization – the rest are marginalized out whenever a keyframe leaves the sliding window of poses.

In order to model the robot's environment, we insert each keyframe from the estimator as a vertex into a local map. We then attempt to establish correspondences between 2D keypoints in this *local mission* (M_L) against previously-triangulated 3D landmarks in a *reference mission* (M_R) (a pre-built, optimized map of the same structure as the local mission).

Fig. 2 shows a representation of the maps shown: the pose-graph in blue, M_L, is the local mission built from keyframes from the visual-inertial odometry system, and M_R is the reference mission that is previousy bundle-adjusted and considered fixed. $_G\mathbf{T}_{M_L}$ is the transformation from the local mission frame to the global coordinate frame, which is the output of our localization procedure.

B. On-Line Localization

When loading a reference mission, we add all the keypoint descriptors and their corresponding 3D triangulated positions (in the M_R frame) to a loop closure database. For each new keyframe from the VIO system, we create a new vertex in the local mission and query all keypoints against the database. Since these matches are prone to outliers and incorrect correspondences, we filter them using a perspective-*n*-point (PnP) algorithm in a RANSAC scheme, keeping only inliers matches from well supported hypothesis. We refer to these inlier matches as **structure matches** - that is, matches

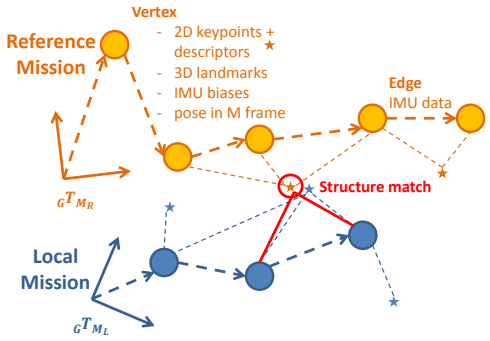


Fig. 2: This figure shows the pose-graph representation used to express the map of the environment. We use a reference map (M_R) built from the bundle-adjusted output of visual-inertial odometry (VIO), which we hold fixed in optimization. The local map (M_L) is built live on-board the robot from the output of VIO, and detected keypoints from keyframes (vertices) in this map are used to query against the reference map to establish 2D-3D correspondences between the two. The output of our localization algorithm is a baseframe transform relating the local mission to the global coordinate frame.

between 2D keypoints of frames in the local mission against 3D landmarks from the reference mission.

The PnP RANSAC algorithm not only provides inlier structure matches, but also an estimate of the vertex pose relative to the global frame (to allow matching to structure across multiple reference missions), ${}_G\mathbf{T}_V$.

We can then use this to estimate the alignment of the local mission to the global frame:

$$ {}_G\mathbf{T}_{M_L} = {}_G\mathbf{T}_V \cdot {}_{M_L}\mathbf{T}_V^{-1} \tag{1} $$

However, this pose estimate only contains information from the structure matches in the latest vertex, and is still prone to sharp jumps and susceptible to outliers. Nonetheless, using this as the initialization for the non-linear optimization that follows keeps the optimizer from getting stuck in local minima, especially from rotation offsets.

C. Optimization Over a Sliding Window

To address this shortcoming, we pose a non-linear least-squares optimization problem over the structure matches in a sliding window of N past vertices. This gives us a more refined estimate of the baseframe alignment, while also adding some degree of temporal filtering and smoothness.

We pose the optimization problem by using the reprojection errors e_i between the ith 2D keypoint $\mathbf{z}$ in the normalized image plane seen from vertices in the local mission to the 3D landmark positions ${}_G\mathbf{p}_i$ in the reference mission as error terms:

$$ \mathbf{e}_i = \mathbf{z}_i - \hat{\mathbf{z}}_i = \mathbf{z}_i - \Pi({}_{M_L}\mathbf{p}_i) = \mathbf{z}_i - \Pi({}_G\mathbf{T}_{M_L}^{-1}\ {}_G\mathbf{p}_i), \tag{2} $$

where $\hat{z}$ denotes the back-projection of the landmark position $\mathbf{p}_{M_L}$ using the projection function $\Pi(\cdot)$ which takes 3D points to the normalized image plane involving camera intrinsic and IMU to camera extrinsic calibration.

Unlike [7], where the local poses are optimized based on their alignment with global 3D landmarks, we hold all parameters except the baseframe transformation of the local frame (${}_G\mathbf{T}_{M_L}$) fixed. This makes the problem significantly faster to solve and keeps the local frame aligned to the odometry frame.

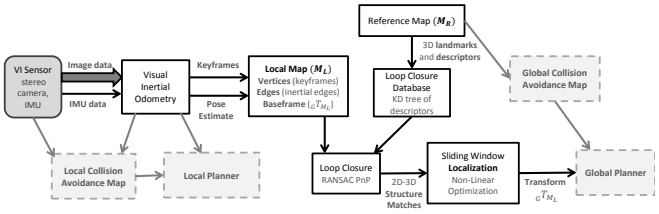


Fig. 3: An overview of our implementation of the localization system running on-board both of the robots described in Section V. The input is visual-inertial sensor data and a bundle-adjusted reference map, and the output to the rest of the robot system is a transformation describing the alignment of the local odometry frame to the global reference frame.

We also indirectly benefit from inertial data for both local and reference missions. IMU data renders roll and pitch observable, therefore allowing us to keep both missions gravity-aligned. This removes 2 degrees of freedom from the optimization, allowing is to parameterize the baseframe transformation in only 4 terms by projecting into the tangent space of the z component of the quaternion, Eq. (4). We then re-normalize the quaternion at each optimization iteration.

$$ {}_G\mathbf{T}_M = \begin{bmatrix} x & y & z & q_x & q_y & q_z & q_w \end{bmatrix} \tag{3} $$

$$ \mathrm{P}({}_G\mathbf{T}_M) = \begin{bmatrix} x & y & z & q_z \end{bmatrix} \tag{4} $$

After optimization, new estimate of the transformation aligning the local VIO frame (M_L) to the global map (G) is available for path-planning and obstacle avoidance.

III. System and Implementation

In this section, we describe the design of our on-board, real-time mapping and localization system, and how it integrates with local and global planning. All components in this section were chosen with two constraints in mind: real-time performance on-board real systems, such as payload-constrained MAVs, and keeping a consistent odometry frame for robot controllers while giving an accurate estimate of current pose within the global frame. Fig. 3 shows an overview of the system and data flow between the components.

A. Planning Considerations

The advantage of our proposed approach is the ability to separate global and local planning and trigger a global replan only when the estimate of the frame alignment changes sufficiently. The global planner plans a path in a 3D collision map (in our implementation, an octomap representation of the environment [16]), while local planning to waypoints (including obstacle avoidance) is done in the odometry frame.

B. System Components

We use a synchronized stereo and IMU camera system [17]. The images and inertial measurements are input into the VIO estimator described in [3], which tracks features across keyframes and integrates inertial data between camera frames to provide pose estimates even in environments with

small numbers of visual features. The output of this estimator is used as the odometry frame for the robots described in this paper.

The output of this system is fed into the multi-agent map distribution backend *Map API* [18] to allow map-sharing with multiple agents as well as loading the model for localization. Besides the data-sharing aspect the Map-API provides a mechanism to run algorithms asynchronously on a particular version of the map, while adding new data to it in the meantime. In our case we leverage this possibility to continously feed new map-data from the odometry into the system while running localization at the same time.

To provide higher localization accuracy, we prefer to use a previously loop-closed and bundle-adjusted reference map. This step removes most of the drift from the estimator, allowing as accurate as possible of a representation of the world through the map. In our setup, it is also possible to localize against a raw, non-bundle-adjusted map, to for example align to the local maps of other robots, but this has the disadvantage of localizing to a drifting frame.

In order to find matches between current keypoints and landmarks in the map, we use projected BRISK descriptors [9] for accelerating the Kd-Tree based nearest neighbor search. To reject implausible matches we apply covisibility graph filtering [19] which removes most of the outliers. The remaining matches are fed to a PnP RANSAC scheme which finds the optimal inlier set subsequently used for localization. In our system, we use gP3P implementation provided by OpenGV [20].

The non-linear optimization and refinement of alignment to the global map is done using the Google Ceres solver [21].

IV. Evaluation

We quantify the performance of our complete localization system on data from real flighs with an MAV platform, compared to ground truth from external sensing. We use the same system as for on-board localization for evaluations.

A. Ground Truth

We evaluate the performance of the proposed system using datasets from the European Robotics Challenge (EUROC)[1] which cover the machine hall at ETH: a typical environment to demonstrate industrial inspection using MAVs. Each dataset consists of precisely synchronized stereo images (752×480) at 20 Hz and IMU data at 200 Hz, using the sensor described in [17] mounted on an AscTec Firefly. The datasets cover a broad range of motions, ranging from slow flight to highly dynamic maneuvers.

For ground-truth we used a laser-tracker[2], which precisely measures the 3D position by tracking a prism mounted on the MAV. We run a batch optimization which also estimates orientation of the IMU, so the ground truth includes full 6 DoF pose.

[1] http://www.euroc-project.eu/

[2] LEICA Nova MS50, http://www.leica-geosystems.com/de/Leica-Nova-MS50_103592.htm

B. Evaluation Setup

Out of the total five available datasets containing MAV flights, two pairs of trajectories cover the same parts of the environment and are thus useful to evaluate our system. A reconstructed 3D pointcloud of the environment as well as the trajectory of the MAV are shown in Fig. 1.

To evaluate the system performance we use one of the datasets per pair and build a reference map from it to be used for localization. In order to obtain exact error metrics we use ground-truth poses for the keyframe positions of these vertices and jointly optimize the poses and 3D landmarks of the model using visual-inertial bundle-adjustment.

Given the model of the environment we then run different versions of the online VIO and perform localization against the model. As a base-line we use the *VIO* output without localization or bundle-adjustment. To evaluate the influence of the limited number of keyframes in the VIO sliding window, we evaluate a second version (*BA*) which denotes the pose-estimate after running a full batch visual-inertial bundle-adjustment on the VIO output.

The last two methods evaluate the benefit from closed loop-localization: The first version denoted as *VIO-localization* combines the *VIO* estimates from one dataset with localization queries to a map built from a dataset with a similar trajectory, and baseframe optimization in a sliding window of 20 keyframes. The second version uses the bundle-adjusted VIO poses (*BA-localization*) as the input to the localization, in order to demonstrate that localization improves the estimated alignment to the global frame even when most of the drift is removed by bundle-adjustment.

To properly compare poses of the non-localized versions to the ground truth, we assume a first-pose alignment, as is often available on robotic systems: we assume that the starting pose of the robot is known exactly, and then measure the estimator drift from that position. In the case of the localized datasets, we give no initialization at all to the alignment to the global frame (which is often up to 180°s and several meters off), but do not consider the error in the first 2-3 frames before enough inliers are found to do a proper baseframe alignment.

Given the high accuracy of the VIO over a short timeframe, it is not necessary for the localization to run at every keyframe. However as shown in Table II, even performing the optimization at every frame is solvable in real-time.

C. Results

The results from all datasets are shown in Table I. Since we want to evaluate the alignment to a global frame, we only consider the absolute translation and rotation error to the global frame. We show a typical trajectory of the MAV, and compare the raw VIO estimator output with the localized version in Fig. 4. For each vertex in the localized version, we show the latest estimate of its alignment to the global frame. Though the quality of our estimator can be seen in the small amount of drift over a long, complex 3D trajectory, localization substantially improves the accuracy of positioning within the global frame and helps mitigate drift.

Step	Time [ms]
Loop Closure Detection (per vertex)	
Descriptor Matching (2 frames)	11.01
PnP + RANSAC	6.39
Optimization (per frame update)	
Optimization setup	0.25
Minimization	14.36

TABLE II: The computational cost of different sub-parts of the localization and frame alignment. Descriptor-matching and outlier rejection using PnP in a RANSAC scheme takes about 25% of the time, with the rest being spend on solving the baseframe transformation alignment problem.

In Fig. 5a, we show the influence of localization on the absolute position error. As expected, raw VIO has the highest errors and the highest variance, and bundle-adjusting the trajectory significantly decreases the error and variance. However, what is more significant is the impact of localization on both the error and the variance in both cases - running localization is better for correcting drift than just bundle-adjustment, and running localization on the bundle-adjusted trajectory reduces the error to negligible levels.

One important note about the performance of our algorithm is its reliance on 2D descriptor matches, which depend heavily on the similarity between viewpoints. Therefore, proximity of each vertex to the reference trajectory plays a major role in the quality of localization. Fig. 5b shows the relationship between detected number of structure constraints and proximity to the nearest vertex in the reference map, aggregated over all datasets presented in Table I. As can be expected, many more matches are found when the viewpoint is similar to one from the reference map.

Another big impact factor is the size of the sliding window used for localization: that is, how many vertices to use in the least-squares optimization. The advantage of more vertices is helping filter out outliers or low-quality matches and help smooth the baseframe alignment (more matches leads to a more accurate pose estimate). However, this comes at a price: since this is essentially a temporal filter, having too large of a sliding window will cause the frame alignment estimate to lag behind and not be as effective for correcting drift - since we do not correct the pose estimates of the past vertices within the local frame. This can be likened to estimating a drifting bias: using information that is too old can lower the accuracy of the latest alignment estimate. We show the results of evaluating different sliding window sizes (1, 5, 10, 20, 50, 100, and 200) in Fig. 6. For the other evaluations, a sliding window size of 20 was chosen, as it appears to have the most consistent performance across all datasets.

V. Experiments

In order to demonstrate the real-time performance and reliability of our algorithm, we designed an experiment to show the localization accuracy in real, unstructured environments and across two platforms with very different movement modalities. We aim to show how a ground robot and an aerial robot can localize against the same map with enough accuracy to allow the helicopter to autonomously land on the ground robot, despite imperfect sensing information and control.

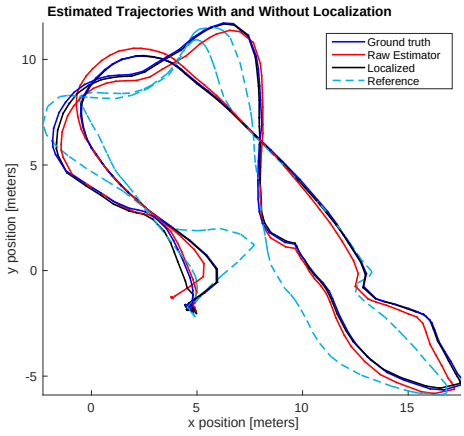
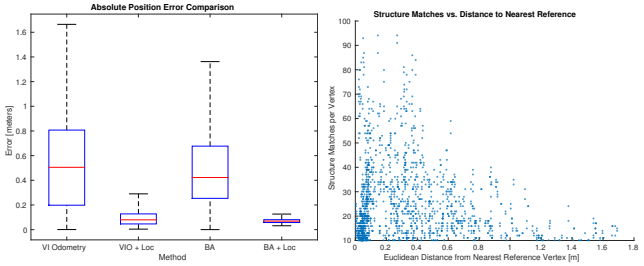


Fig. 4: A sample trajectory from evaluation dataset 3, showing how localization corrects drift in the visual odometry estimate. The reference trajectory from the dataset 4 is shown as well, and it is possible to see the quality of the localization degrade when the reference viewpoint varies substantially from the actual flown trajectory. We do not show the bundle-adjusted trajectories, as they appear overlaid on top of the ground truth and localized trajectory at this scale.



(a) (b)

Fig. 5: Left: The absolute position errors for dataset 4 based on different estimation and localization strategies. Bundle-adjusted trajectories show much lower error than raw estimator output, as expected, but localization is even more effective at reducing both the magnitude and the variance of the error. Right: Proximity to the reference path affects the number of detected structure matches per vertex (aggregated over all datasets). Being closer to the reference trajectory significantly improves the number of matches found, and also increases localization quality.

A. Platforms

We use a Pioneer 3DX[3] as the ground platform, which uses differential-drive with a 2.6 GHz dual-core Intel i5 CPU on-board, and features a state-of-the-art helicopter landing pad. The MAV is an AscTec Firefly[4], with an Intel i7 CPU, the same platform used to generate the ground truth datasets. Both platforms use the sensing and localization pipeline as described in Section III and are shown in Fig. 7a.

B. Experiment Setup

To reduce the influence of other factors (such as controller error) as much as possible, we chose a very simple experimental setup. We first build a global map using the helicopter from the on-board visual-inertial odometry. This map is then bundle-adjusted, and we project disparities from the bundle-adjusted vertex poses into 3D to generate a 3D voxel-based

[3]Adept Mobile Robots http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx

[4]http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-firefly/

Dataset	Visual-Inertial Odometry		VIO + Localization		Bundle-Adjusted		BA + Localization	
	Trans [m]	Rot [rad]	Trans [m]	Rot [rad]	Trans [m]	Rot [rad]	Trans [m]	Rot [rad]
Dataset 1 [77 m, 182 sec]	0.37±0.23	0.13±0.11	0.19±0.23	0.09±0.13	0.12±0.02	0.02±0.01	0.04±0.05	0.01±0.02
Dataset 2 [70 m, 150 sec]	0.23±0.08	0.05±0.08	0.13±0.23	0.05±0.08	0.07±0.05	0.01±0.01	0.02±0.02	0.01±0.01
Dataset 3 [90 m, 97 sec]	0.44±0.17	0.05±0.04	0.26±0.26	0.08±0.38	0.35±0.24	0.02±0.01	0.10±0.14	0.06±0.38
Dataset 4 [96 m, 108 sec]	0.26±0.14	0.01±0.01	0.09±0.06	0.01±0.01	0.18±0.14	0.02±0.01	0.07±0.02	0.01±0.01

TABLE I: Comparison of the pose-estimation error of different variants of the proposed system to ground-truth with visual-inertial-odometry (VIO) as a base-line. *VIO* denotes the raw odometry estimate, aligned to the ground truth at the first pose, *VIO + Localization* the result with localization without first-pose alignment. The pose estimate of the VIO can be improved by full-batch visual-inertial bundle-adjustment *Bundle-Adjusted*, especially when combined with localization *BA + Localization*.

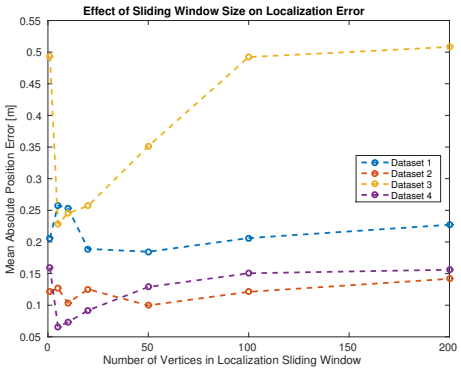
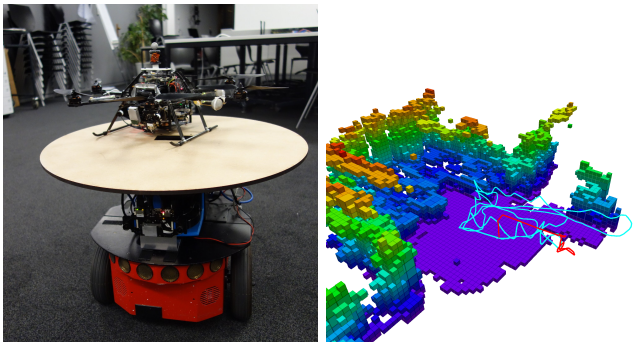


Fig. 6: Effect of sliding window size (the number of vertices that have their structure matches in the least-squares optimization) on absolute position error. The minimum number of vertices, 1, shows a high error and susceptibility to outliers. While a larger window generally deceases error, at some point keeping the constraints from a long time-period ago introduces a large lag and again increases error.



(a) (b)

Fig. 7: Left: The result of the experiment, where the helicopter autonomously landed on the ground robot based on visual-inertial localization against the same map running on-board both robots. Right: An octomap representation of the sparse map that both robots localize against. The path used to gather the reference map is shown in blue, and the helicopter's actual trajectory during the experiment is shown in red.

occupancy grid (octomap) to be used for global path planning on-board the MAV. The same reference map is used on board both robots, to show the ability of our system to localize multiple platforms against the same global reference.

We then drive the ground robot under manual control to a location, and take its reported localized alignment within the global map as the goal destination for the MAV. The helicopter then takes off and plans a path within the global map to the ground robot's localized location, and then performs a landing maneuver.

This experiment shows that the accuracy of localization running in closed-loop on board both systems is high enough that they can collaborate, even without explicitly determining their relative positions.

C. Results

In the experiment, which is available in the video accompanying this submission, we show a successful rendezvous between the two robots. Despite different viewpoints between the aerial vehicle and the ground robot, the two were able to share the same small map and very accurately locate themselves within it. Fig. 7b shows an octomap representation of the reference map, overlaid with the helicopter's actual trajectory during the experiment in red, and the trajectory used to build the reference map in blue. Note that the reference map was built on-board the MAV platform, using the complete VIO-localization system proposed in this work, with only a small number of iterations of bundle-adjustment after. Despite the difference in viewpoint between the ground and aerial robot, the ground robot was successfully able to localize against this map.

During this experiment, we also demonstrate the real-time on-board capability of the system. The runtimes of each component of the localization system, per keyframe and per frame alignment update, are shown in Table II. Per keyframe, descriptor matching is the slower step and depends on the size of the reference map. However, our usage of a hash-table based lookups [22] allows us to scale to larger maps without degrading look-up times significantly. The alignment optimization step does not need to run at every keyframe, but is nonetheless fast enough to do so. Additionally, since we use only a sliding-window of keyframes, this sets an upper limit on the number of residuals and the 4-term parameterization of the optimization terms allows this stage to scale.

VI. Conclusions

In this work, we show a real-time, entirely on-board system integrating visual-inertial odometry and localization against a previously-built map that aims to combat the effects of estimator drift.

We rely on rigidly aligning the baseframes of the local map and reference map, in order to keep the local map consistent with the robot's visual-inertial odometry frame. This allows one map to always be consistent with the frame used for the robot's current state estimate and local planning, while the other is always aligned with the global map. We also propose a scheme in which the global planner uses this frame alignment estimate to trigger re-planning and re-projecting the global plan into the local coordinate frame

when necessary.

To demonstrate the performance of our system, we run evaluations of the complete closed-loop system on a series of datasets from MAV flights in a realistic industrial survey environment, compared with ground truth from external sensing. Our evaluations show substantial reduction in estimator drift and overall accuracy of position estimation in a global frame.

Finally, to show our system running on-board real robots, we design an experiment where an MAV and a ground robot localize in real-time against the same map. The MAV then uses these localization estimates to autonomously land on top of the ground robot.

One possible extension to this work includes using a particle filter for initialization, to increase robustness in the rare case of wrong initial alignment estimates. Another improvement would be to increase robustness of the descriptor matching from different viewpoints, which would especially help in the case of aerial-ground robot collaboration.

REFERENCES

[1] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Robotics and Automation, IEEE International Conference on*, IEEE, 2007.

[2] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium," *Journal of Field Robotics*, 2013.

[3] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization," *International Journal of Robotics Research (IJRR)*, 2014.

[4] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular slam with multiple micro aerial vehicles," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, 2013.

[5] C. Forster, M. Pizzoli, and D. Scaramuzza, "Air-ground localization and map augmentation using monocular dense reconstruction," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, 2013.

[6] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg, "Real-Time Self-Localization from Panoramic Images on Mobile Devices," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[7] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt, "Scalable 6-DOF Localization on Mobile Devices," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[8] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg, "Global Localization from Monocular SLAM on a Mobile Phone," *IEEE Transactions on Visualization and Computer Graphics*, 2014.

[9] S. Lynen, M. Bosse, P. Furgale, and R. Siegwart, "Placeless place-recognition," in *3D Vision (3DV), 2nd International Conference on*, 2014.

[10] J. McDonald, M. Kaess, C. D. C. Lerma, J. Neira, and J. J. Leonard, "6-dof multi-session visual slam using anchor nodes.," in *ECMR*, 2011.

[11] L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous visual mapping and exploration with a micro aerial vehicle," *Journal of Field Robotics*, 2014.

[12] K. Schmid, T. Tomic, F. Ruess, H. Hirschmuller, and M. Suppa, "Stereo vision based indoor/outdoor navigation for flying robots," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE, 2013.

[13] 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 *Robotics and Automation (ICRA), IEEE International Conference on*, IEEE, 2014.

[14] T. A. Vidal-Calleja, C. Berger, J. Solà, and S. Lacroix, "Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain," *Robotics and Autonomous Systems*, vol. 59, no. 9, 2011.

[15] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, *et al.*, "Collaborative mapping of an earthquake-damaged building via ground and aerial robots," *Journal of Field Robotics*, 2012.

[16] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, 2013.

[17] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart, "A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time slam," in *Robotics and Automation (ICRA), IEEE International Conference on*, IEEE, 2014.

[18] T. Cieslewski, S. Lynen, M. Dymczyk, S. Magnenat, and R. Siegwart, "Map api - scalable decentralized map building for robots," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 2015.

[19] T. Sattler, B. Leibe, and L. Kobbelt, "Improving Image-Based Localization by Active Correspondence Search," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

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

[21] S. Agarwal, K. Mierle, and Others, "Ceres solver." `https://code.google.com/p/ceres-solver/`.

[22] S. Lynen, T. Sattler, M. Bosse, J. Hesch, M. Pollefeys, and R. Siegwart, "Get out of my lab: Large-scale, real-time visual-inertial localization.," in *Robotics: Science and Systems*, 2015.