Aerial and Ground-based Collaborative Mapping: An Experimental Study

Ji Zhang and Sanjiv Singh

Abstract We here present studies to enable aerial and ground-based collaborative mapping in GPS-denied environments. The work utilizes a system that incorporates a laser scanner, a camera, and a low-grade IMU in a miniature package which can be carried by a light-weight aerial vehicle. We also discuss a processing pipeline that involves multi-layer optimization to solve for 6-DOF ego-motion and build maps in real-time. If a map is available, the system can localize on the map and merge maps from separate runs for collaborative mapping. Experiments are conducted in urban and vegetated areas. Further, the work enables autonomous flights in cluttered environments through building and trees and at high speeds (up to 15m/s).

1 Introduction

The paper is aimed at solving a mapping problem. In particular, we seek for collaboration between mapping from the ground and air due to each own characteristics. Ground-based mapping is not prone to limitations of space or time. Typically, a mapping device carried by a ground vehicle is suitable for mapping in large scale and can move at a high speed. On the other hand, a tight area can be mapped in a hand-held deployment. However, ground-based mapping is limited by the sensor's altitude, difficult to realize a top-down looking configuration. As illustrated in Fig. 1, the ground-based experiment produces a detailed map of the surroundings of a building, while the roof has to be mapped from the air. If a small aerial vehicle is used, aerial mapping is limited by time due to the short lifespan of batteries. Space also needs to be open enough for aerial vehicles to operate safely. In this paper, we carry out experimental studies to pursue the advantages of both.

The collaborative mapping is based on our previous work [1–3] which develops a data processing pipeline for real-time ego-motion estimation and mapping. The

J. Zhang is with Kaarta, Inc. and S. Singh is with Kaarta, Inc. and Near Earth Autonomy, Inc.
Experiment video is available at: www.youtube.com/watch?v=nPGge78-lK8

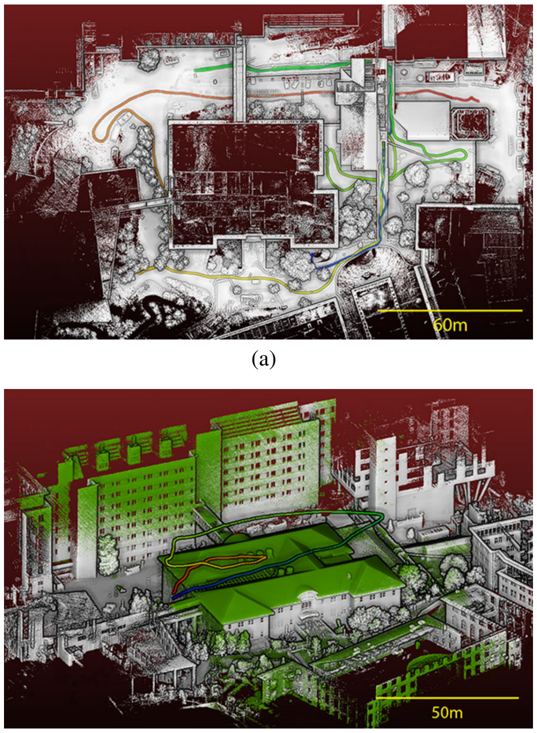


Fig. 1 Example of air-ground collaborative mapping. (a) shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) produced by an operator holding a sensor pack and walking around a building at 1-2m/s for 914m of travel. The ground-based map covers details surrounding the building except the roof. Then in (b), the same sensor pack is mounted to an aerial vehicle flying over the building at 2-3m/s for 269m. The green point cloud in (b) is the aerial map and the colored curve is the sensor trajectory in the air.

method utilizes a laser scanner, a camera, and a low-grade IMU, processes data through multi-layer optimization. The resulting motion estimates are at a high rate (200Hz) with a low drift (typically $<$0.1% of the distance travel).

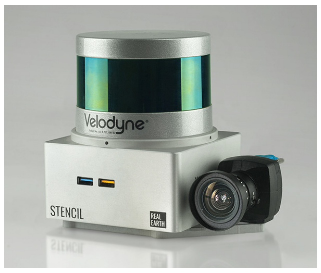
Benefit from the high-accuracy processing pipeline, the paper develops a method to merge the maps from the ground and air in real-time. This is by localization of one output w.r.t. the map from the other. In the existing literature, prior map based localization often involves particle filtering [4–9]. In addition, Nieuwenhuisen et al use multi-resolution scan matching to localize an aerial vehicle on a 3D point cloud map [10]. Also employing scan matching, our method enables both high-speed navigation (up to 15m/s) and large scale mapping (over 1km).

While the proposed scheme fulfills collaborative mapping, it further reduces the complexity of aerial deployments. With a ground-based map, flight paths are defined and the aerial vehicle conducts mapping in autonomous missions. In experiments, the aerial vehicle is able to accomplish challenging flight tasks autonomously.

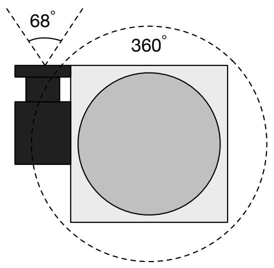
2 Method

2.1 Sensor Configuration

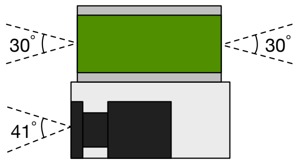
Fig. 2 presents the sensor/computer pack utilized by the paper to enable collaborative mapping. Our processing software is not limited to a particular sensor configuration. However, introducing sensors in the front helps readers understand the technology. The sensor pack (see Fig. 2(a)) consists of a Velodyne Puck laser scanner generating 0.3 million points/second, a camera at 640×360 pixels resolution and 50Hz frame rate, and a low-grade IMU at 200Hz. An onboard i7 computer processes data from the sensors in real-time for ego-motion estimation and mapping. Fig. 2(c) and Fig. 2(d) illustrate the sensor field of view. An overlap is shared by the laser and camera, with which, the processing software associates depth information from the laser to image features (more discussion in Section 2.2).



(a)



(b)



(c)

Fig. 2 (a) Sensor pack including a Velodyne Puck laser scanner, a camera, and a low-grade IMU. An onboard i7 computer processes data from the sensors to conduct real-time ego-motion estimation and mapping. (b)-(c) Horizontal and vertical field of view of the laser and camera.

2.2 *Odometry and Mapping*

The odometry and mapping method is originally proposed in [3]. For completeness, we include an overview of the method. The software processes data from a range sensor such as a laser scanner, a camera, and an inertial sensor. Instead of combining data from all sensors in a large, full-blown problem, we parse the problem as multiple small problems, solve them sequentially in a coarse-to-fine manner. Fig. 3 gives a block diagram of the software system. In such a system, modules in the front conduct light processing, ensuring high-frequency motion estimation robust to aggressive motion. Modules in the back take sufficient processing, run at low frequencies to warrant accuracy of the resulting motion estimates and maps.

The software starts with IMU data processing (orange module in Fig. 3). This module runs at the IMU frequency to predict the motion based on IMU mechanization. The result is further processed by a visual-inertial coupled method (green module in Fig. 3). The method tracks distinctive image features through the image sequence and solves for the motion in an optimization problem. Here, laser range measurements are registered on a depthmap, with which, depth information is associated to the tracked image features. Since the sensor pack contains a single camera, depth from the laser helps solve scale ambiguity during motion estimation.

The estimated motion is used to register laser scans locally. In the third module (blue module in Fig. 3), these scans are matched to further refine the motion estimates. The matched scans are registered on a map while scans are matched to the map. To accelerate the processing, scan matching utilizes multiple CPU threads in parallel. The map is stored in voxels to accelerate point query during scan matching. Because the motion is estimated at different frequencies, a fourth module in the system (gray module in Fig. 3) takes these motion estimates for integration. The output holds both high accuracy and low latency beneficial for vehicle control.

The modularized system also ensures robustness w.r.t. sensor degradation, by selecting “healthy” modes of the sensors when forming the final solution. For example, when a camera is in a low-light or texture-less environment such as pointing to a clean and white wall, or a laser is in a symmetric or extruded environment such as a long and straight corridor, processing typically fails to generate valid motion estimates. Our system automatically determines a degraded subspace in the problem state space [11]. When degradation happens, the system only solves the problem partially in the well-conditioned subspace of each module. The result is that the “healthy” parts are combined to produce the final, valid motion estimates.

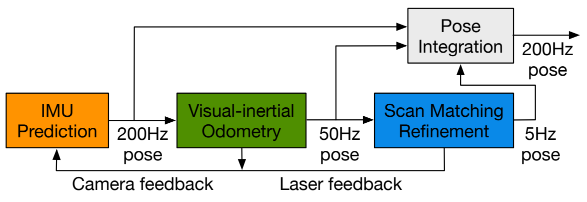


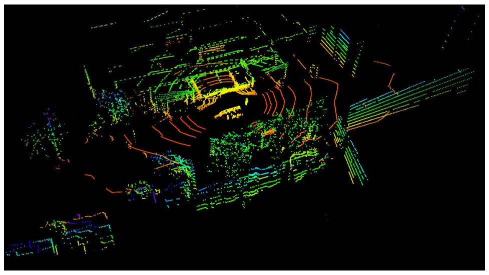
Fig. 3 Block diagram of the laser-visual-inertial odometry and mapping software system.

2.3 Localization and Map Merging

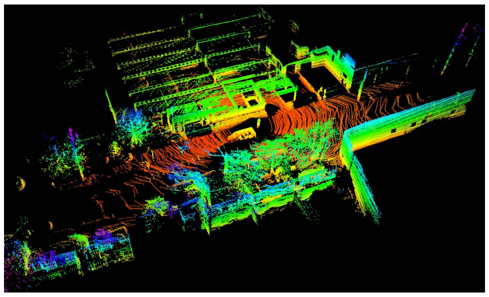
When a map is available, the method described in Section 2.2 can be extended to utilize the map for localization. This is using a scan matching method similar to the blue block in Fig. 3. The method extracts two types of geometric features – points on edges and planar surfaces, based on the curvature in local scans. Feature points are matched to the map. An edge point is matched to an edge line segment, and a planar point is matched to a local planar patch. On the map, the edge line segments and local planar patches are determined by examining the eigenvalues and eigenvectors associated with local point clusters. The map is stored in voxels to accelerate processing. The localization solves an optimization problem minimizing the overall distances between the feature points and their correspondences. Due to the fact that we use the high-accuracy odometry estimation to provide initial guess to the localization, the optimization usually converges in 2-3 iterations.

Compared to Section 2.2, the difference is that the localization does not process individual scans but stacks a number of scans for batch processing. Thanks to the high-accuracy odometry estimation, scans are registered precisely in a local coordinate frame where drift is negligible over a short period of time (a few seconds). A comparison is given in Fig. 4, where Fig. 4(a) is a single scan that is matched in Section 2.2 (scan matching executes at 5Hz), and Fig. 4(b) shows stacked scans over two seconds, which are matched during localization (scan matching runs at 0.5Hz). One can see the stacked scans contain significantly more structural details, contributing to the localization accuracy and robustness w.r.t. environmental changes. Additionally, low-frequency execution keeps the CPU usage to be minimal for onboard processing (localization consumes about 10% of a CPU thread).

Our localization is compared to a particle filter based implementation. The odometry estimation provides the motion model to the particle filter. It uses a number of 50 particles. At each update step, the particles are resampled based on low-variance resampling [12]. Comparison results are shown in Fig. 5 and Table 1. Here, errors are defined as the absolute distances from localized scans to the map. During the evaluation, we choose a number of planar surfaces and use the distances between points in localized scans to the corresponding planar patches on the map. Fig. 5 shows the error distribution. When running the particle filter at the same frequency as our method (0.5Hz), the resulting error is five times as large. While in Table 1, the CPU processing time is more than twice of ours. In another test, running the particle filter at 5Hz helps reduce the error to be slightly larger than our method. However, the corresponding CPU processing time increases to over 22x of ours. These results imply a particle filter based method does not take full advantage of the high-accuracy odometry estimation as compared to our implementation.



(a)



(b)

Fig. 4 Comparison of scans involved in odometry estimation and localization. In odometry estimation, each individual scan is processed in scan matching at 5Hz. While in localization, a number of locally registered scans are stacked and batch processed at 0.5Hz.

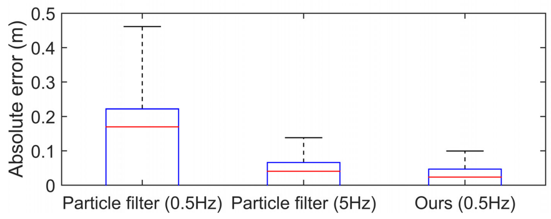


Fig. 5 Comparison of scan matching accuracy in localization. We use the absolute distances from localized scans to the corresponding local patches on the map as the metric. Points are selected from a number of planar surfaces in Fig. 4. The red lines illustrate medians, the blue boxes represent 75% of the distributions, and the black lines are the maximum errors.

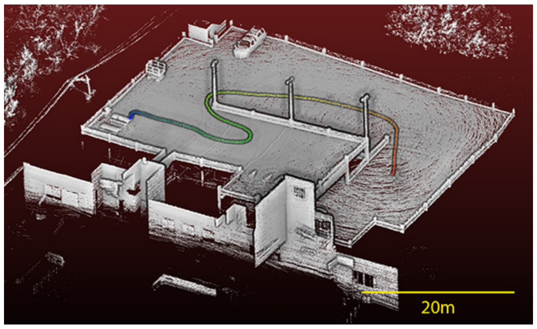
Table 1 Comparsion of CPU processing time in localization. When running the particle filter at 5Hz, sensor data is processed at 25% of real-time speed due to high CPU demand.

Method	Particle filter		Ours
Frequency	0.5Hz	5Hz	0.5Hz
Time per execution	493ms	478ms	**214ms**
Time per second	247ms	2390ms	**107ms**

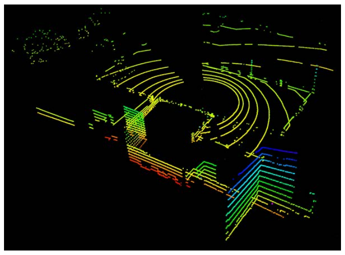
3 On Sensor Orientation

In previous work [3], we studied the system performance w.r.t. sensor degradation. We concluded that the system is robust to individual sensor failures, i.e. when the laser or camera is degraded, the corresponding module is bypassed while the rest of the system is staggered to generate the solution. In this section, we further carry out studies where the sensor pack is orientated differently. This is especially motivated by aerial mapping due to the fact that during “up and away” flights, the sensor pack has to be tilted downward in order to capture data from the ground.

The first set of study is conducted in Fig. 6 and Fig. 7. First, we carry the sensor pack horizontally in a garage building. Fig. 6(a) shows the map built and sensor trajectory. Fig. 6(b) is a single scan. In this scenario, the scan contains sufficient structural information. When bypassing the camera processing module ((green module in Fig. 3)), the system produces the same trajectory as the full pipeline. On the other hand, we run another test with the sensor pack tilted vertically down toward the ground. The results are shown in Fig. 7. In this scenario, structural information in a scan is much sparser (see Fig. 7(b)). The processing fails without usage of the camera and succeeds with the full pipeline. The results indicate the camera is critical for high-altitude flights where tilting of the sensor pack is required.

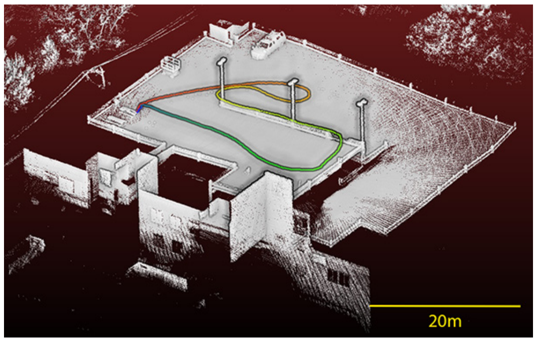


(a)

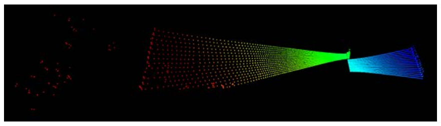


(b)

Fig. 6 Example of horizontally orientated sensor test. The processing succeeds with and without the camera, both yield the same map and sensor trajectory (colored curve starting with blue and ending with red) in (a). (b) shows a raw laser scan during the test which contains plenty of structural information for scan matching based methods to function.



(a)



(b)

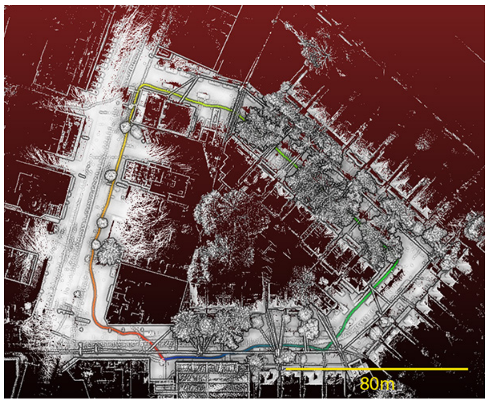
Fig. 7 Example of vertically orientated sensor test. The processing succeeds only when the camera is present, which yields the map and sensor trajectory in (a). Due to the lack of structural information in laser data (see a raw laser scan in (b)), odometry estimation requires assistance from the camera. Methods only replying on scan matching are not functional.

The second set of study compares the drift rate w.r.t. different sensor orientations. As shown in Fig. 8, the sensor pack is held by an operator walking through a circle at 1-2m/s speed with an overall traveling distance of 410m. Fig. 8(a) shows the map built and sensor trajectory with a horizontally orientated sensor configuration. The sensor is started and stopped at the same position. The test produces 0.18m of drift through the path, resulting in 0.04% of relative position error in comparison to the distance traveled. Then, the operator repeats the path with two sensor packs held at 45° and 90° angles, respectively. The resulting sensor trajectories are shown in Fig. 8(b). Clearly, tilting introduces more drift, where the relative position errors are 0.6% at 45° (blue dash curve) and 1.4% at 90° (red dash-dot curve). Finally, by localizing on the map in Fig. 8(a) using the method in Section 2.3, the drift is canceled and both configurations result in trajectories as the black solid curve.

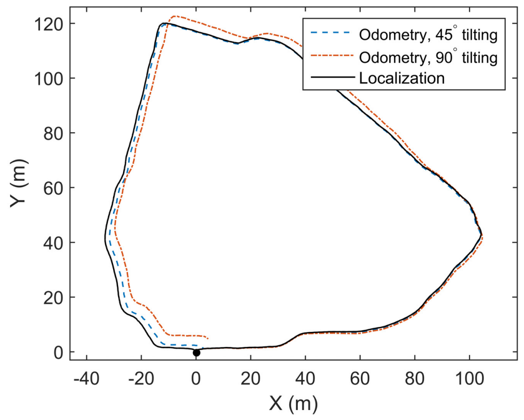
4 Flight Experiments

4.1 Drone Hardware

The drone platform is a DJI S1000 aircraft as shown Fig. 9. The aircraft weights 6.8kg (including batteries) and can carry a maximum of 4.2kg payload. The sensor/computer pack is mounted to the bottom of the aircraft, weighting 1.7kg. The bottom right of the figure shows the remote controller. During autonomous mis-



(a)



(b)

Fig. 8 Accuracy comparison between horizontally orientated and downward tilted sensor tests. (a) shows the map and sensor trajectory (colored curve starting with blue and ending with red) of a horizontally orientated setup. The sensor pack is started and stopped at the same position, and held by an operator who walks at 1-2m/s through a circle for 410m. The odometry estimation produces 0.18m of drift resulting in 0.04% of relative position error w.r.t. the distance traveled. (b) shows results of a repeated test with the operator holding two sensor packs, one at 45° (blue dash curve) and the other at 90° (red dash-dot curve). Tilting at 45° results in 2.3m of drift and correspondingly 0.6% of relative position error, while tilting at 90° generates 5.8m of drift and 1.4% of relative position error. Finally by localizing w.r.t. the map in (a), both setups produce trajectories as the black solid curve. All trajectories start at the black dot.

sions, the remote controller is operated by a safety pilot to override the autonomy if necessary. Note that the aircraft is built with a GPS receiver (on top of the aircraft). GPS data is not used in mapping or autonomous missions through the paper.

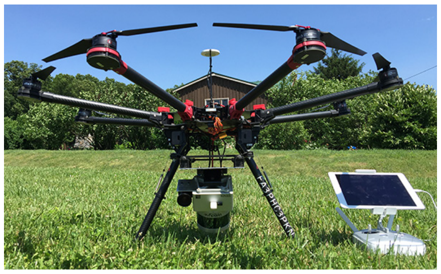


Fig. 9 DJI S1000 aircraft with sensor pack. The aircraft is built with a GPS receiver (on top of the aircraft). GPS data is not used in mapping or autonomous missions as in this paper.

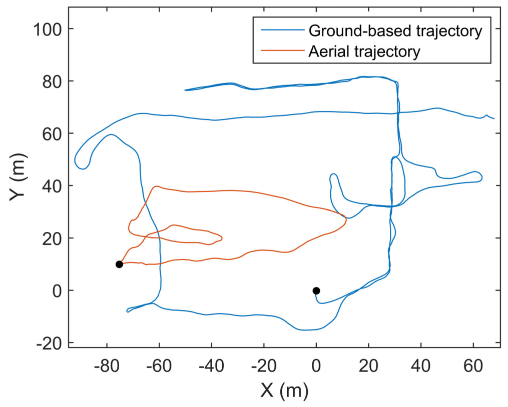
4.2 Teleoperated Flight Results

In the first collaborative mapping experiment, an operator holds the sensor pack and walks around a building. Results are shown in Fig. 1. In Fig. 1(a), the ground-based mapping covers surroundings of the building in detail, conducted at 1-2m/s over 914m of travel. As expected, the roof of the building is empty on the map. Second, the drone is teleoperated to fly over the building. In Fig. 1(b), the flight is conducted at 2-3m/s with a traveling distance of 269m. The processing uses localization w.r.t. the map in Fig. 1(a). That way, the aerial map (green points) is merged with the ground-based map (white points). After the ground-based map is built, the take-off position of the drone is determined on the map. The sensor starting pose for the aerial mapping is known, and from which, the localization starts. Fig. 10 presents the aerial and ground-based sensor trajectories, in top-down and side views.

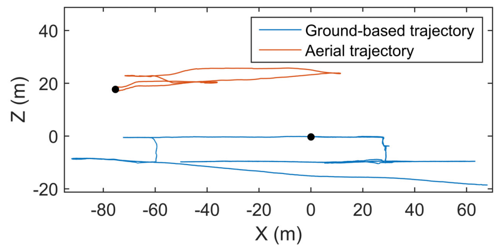
4.3 Autonomous Flight Results

Further, we conduct autonomous flights to realize aerial mapping. In Fig. 11, we first build a ground-based map by hand-held mapping at 1-2m/s for 672m of travel around the flight area. The map and sensor trajectory are shown in Fig. 11(a). Then based on the map, way-points are defined and the drone follows the way-points to conduct aerial mapping. As shown in Fig. 11(b), the colored curve is the flight path, the large colored points on the curve are the way-points, and the green points form the aerial map. In this experiment, the drone takes off inside a shed on the left side of the figure, flies across the site and passes through another shed on the right side, then returns to the first shed to land. The speed is 4m/s crossing the site and 2m/s passing through the shed. Fig. 11(c) and Fig. 11(d) are two images taken by an onboard camera when the drone flies toward the shed on the right and is about to enter the shed. Fig. 11(e) shows the estimated speed during the mission.

Finally, we conduct another experiment over a longer distance. As shown in Fig. 12, the ground-based mapping involves an off-road vehicle driven at 10m/s from the left end to the right end, over 1463m of travel. With the ground-based map and way-points, the autonomous flight crosses the site. Upon take-off, the drone as-



(a)



(b)

Fig. 10 Sensor trajectories corresponding to Fig. 1. The ground-based mapping is at 1-2m/s with an overall distance of 914m. The aerial mapping is at 2-3m/s with 269m of travel. The trajectories start at the black dots.

cends to 20m high above the ground at 15m/s. Then, it descends to 2m above the ground to fly through a line of trees at 10m/s. The flight path is 1118m long as the colored curve in Fig. 12(b). Two images are taken as the drone flies high above the trees (see Fig. 12(c)) and low underneath the trees (see Fig. 12(d)).

5 Conclusion

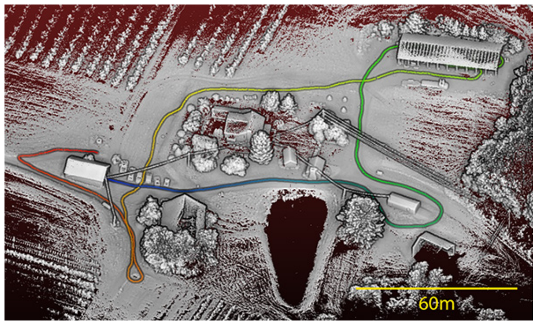
The paper presents experimental studies on collaborative mapping from the ground and air. Utilizing a miniature sensor pack consisted of a laser scanner, a camera, and a low-grade IMU, we demonstrate aerial and ground-based mapping while the maps are merged in real-time during the flights. This benefits from a data processing pipeline with multi-layer optimization – modules in the front execute at high frequencies to handle aggressive motion and produce high-rate motion estimates, while modules in the back run at low frequencies and take sufficient computation to warrant accuracy. We evaluate the system in both teleoperated and autonomous flights, through cluttered environments and at high speeds (up to 15m/s).

Acknowledgment

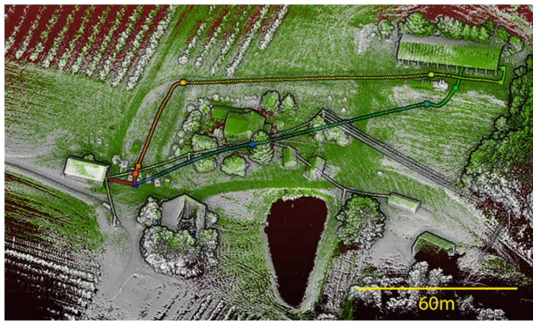
Special thanks are given to R. Chadha, D. Murphy, S. Spiker, K. Rugg, D. Duggins, K. Dowling, and M. Bergerman for their insightful inputs and invaluable help.

References

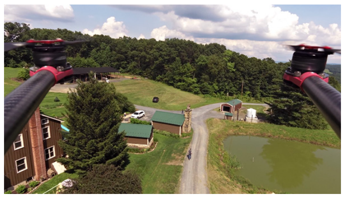
1. J. Zhang and S. Singh, “LOAM: Lidar odometry and mapping in real-time,” in *Robotics: Science and Systems Conference (RSS)*, Berkeley, CA, July 2014.
2. J. Zhang, M. Kaess, and S. Singh, “Real-time depth enhanced monocular odometry,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, IL, Sept. 2014.
3. J. Zhang and S. Singh, “Enabling aggressive motion estimation at low-drift and accurate mapping in real-time,” in *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, May 2017.
4. M. Fallon, H. Johannsson, and J. Leonard, “Efficient scene simulation for robust monte carlo localization using an rgb-d camera,” in *IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, May 2012.
5. N. Ganganath and H. Leung, “Mobile robot localization using odometry and kinect sensor,” in *IEEE International Conference on Emerging Signal Processing Applications (ESPA)*, Las Vegas, Nevada, Jan. 2012.
6. G. Tipaldi, D. Meyer-Delius, and W. Burgard, “Lifelong localization in changing environments,” *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1662–1678, 2013.
7. H. Nurminen, A. Ristimäki, S. Ali-Löytty, and R. Piché, “Particle filter and smoother for indoor localization,” in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Montbliard, France, Oct. 2013.
8. A. Redondi, M. Chirico, L. Borsani, M. Cesana, and M. Tagliasacchi, “An integrated system based on wireless sensor networks for patient monitoring, localization and tracking,” *Ad Hoc Networks*, vol. 11, no. 1, pp. 39–53, 2013.
9. A. Bry, C. Richter, A. Bachrach, and N. Roy, “Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments,” *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 969–1002, 2015.
10. M. Nieuwenhuisen, D. Droeschel, M. Beul, , and S. Behnke, “Autonomous navigation for micro aerial vehicles in complex gnss-denied environments,” *Journal of Intelligent and Robotic Systems*, 2015.
11. J. Zhang, M. Kaess, and S. Singh, “On degeneracy of optimization-based state estimation problems,” in *IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016.
12. S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, The MIT Press, 2005.



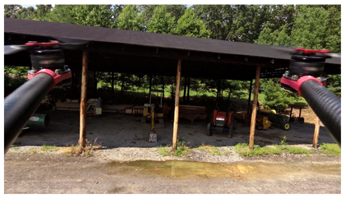
(a)



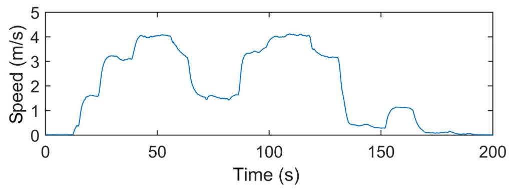
(b)



(c)

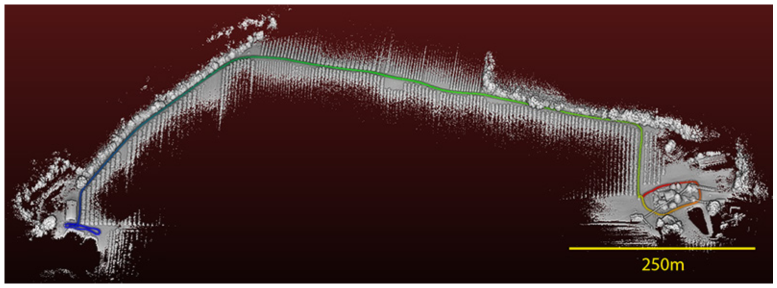


(d)

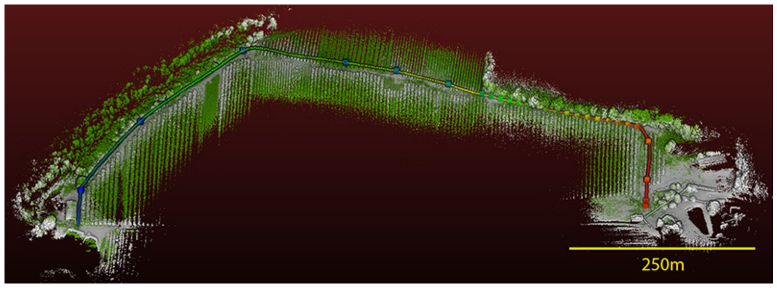


(e)

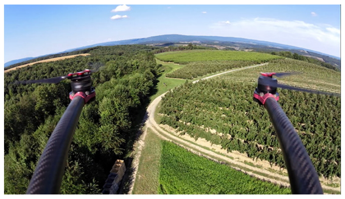
Fig. 11 Autonomous flight result through a shed. (a) shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) built by an operator holding the sensor pack and walking at 1-2m/s for 672m. With the ground-based map, way-points are defined and the drone executes an autonomous mission. It takes off inside a shed on the left side of the figure, navigates across the site at 4m/s, passes through another shed on the right side at 2m/s, and returns to the first shed over 390m of travel. In (b), the green point cloud is the aerial map, the colored curve is the sensor trajectory in the air, and the large points on the curve are the way-points. (c) and (d) are two images taken by an onboard camera when the drone flies between the sheds and is about to enter the shed on the right. (e) shows the estimated speed through the route.



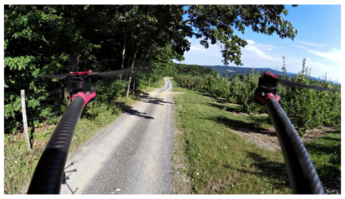
(a)



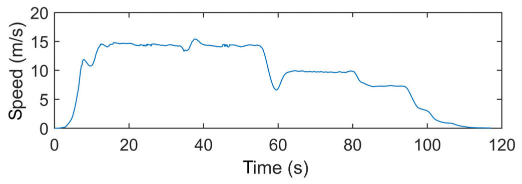
(b)



(c)



(d)



(e)

Fig. 12 Autonomous flight result over a long-run. (a) shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) generated by the sensor pack mounted to an off-road vehicle, driven at 10m/s for 1463m. (b) shows the result of the autonomous mission from an 1118m flight. During the mission, the drone first flies at 20m above the ground at 15m/s and then descents to 2m above the ground through a line of trees at 10m/s. The green point cloud is the aerial map, the colored curve is the flight trajectory, and the large points on the curve are the way-points. (c) and (d) are from an onboard camera when the drone flies high above the trees and low underneath the trees. (e) shows the estimated speed through the route.