2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV

Shaojie Shen, Nathan Michael, and Vijay Kumar

Abstract— **In this paper, we consider the problem of autonomous navigation with a micro aerial vehicle (MAV) in indoor environments. In particular, we are interested in autonomous navigation in buildings with multiple floors. To ensure that the robot is fully autonomous, we require all computation to occur on the robot without need for external infrastructure, communication, or human interaction beyond high-level commands. Therefore, we pursue a system design and methodology that enables autonomous navigation with real-time performance on a mobile processor using only onboard sensors. Specifically, we address multi-floor mapping with loop closure, localization, planning, and autonomous control, including adaptation to aerodynamic effects during traversal through spaces with low vertical clearance or strong external disturbances. We present experimental results with ground truth comparisons and performance analysis.**

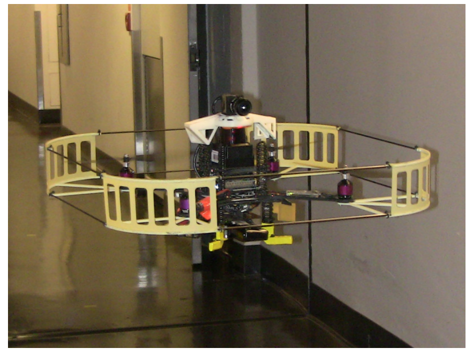


Fig. 1. The experimental platform with onboard computation (1.6 GHz Atom processor) and sensing (laser, camera, and IMU).

I. Introduction

We are interested in the problem of surveilling and exploring environments that include both indoor and outdoor settings. Aerial vehicles offer mobility and perspective advantages over ground platforms and micro aerial vehicles (MAVs) are particularly applicable to buildings with multiple floors where stairwells can be an obstacle to ground vehicles. A challenge when operating in indoor environments is the lack of an external source of localization such as GPS. For these reasons, in this work we focus on autonomous navigation in buildings with multiple floors without requiring an external source of localization or prior knowledge of the environment. To ensure that the robot is fully autonomous, we require all computation to occur on the robot without need for external infrastructure, communication, or human interaction beyond high-level commands. Therefore, we pursue a system design and methodology capable of autonomous navigation with real-time performance on a mobile processor using only onboard sensors (Fig. 1); where in this work autonomous navigation considers multi-floor mapping with loop closure, localization, planning, and control.

The paper consists of three parts: (1) a system overview that details our approach and places it into context with existing methodologies (Sect. II); (2) extensions necessary to permit operation onboard the robot in multi-floor environments and to compensate for external aerodynamic effects (Sect. III); and (3) experiment results that characterize system performance and accuracy and demonstrate application in multi-floor environments (Sect. IV).

We note that the topic of autonomous navigation with a MAV is addressed by others in the community with some similarities in approach and methodology. Relevant to this paper is the work of Bachrach et al. [1, 2], Grzonka et al. [3], and Blösch et al. [4] with results toward online autonomous navigation and exploration with an aerial vehicle. The major points of differentiation between existing results and our work are threefold. First, all the processing is done onboard requiring algorithms that lend themselves to real-time computation on a small processor. Second, we consider multi-floor operation with loop closure. Third, we design adaptive controllers to compensate for external aerodynamic effects which would otherwise prohibit operation in constrained environments.

II. Methodology and Related Literature

We are interested in real-time autonomous navigation in multi-floor indoor environments using an aerial vehicle with pragmatic constraints on available computational resources (speed and memory). Therefore, we must address the problems of mapping, localization, planning, and control given these system requirements. Each of these topics covers a breadth of literature and as such we focus here only on research that directly impacts our system design and methodology. We evaluated many strategies in the development of the system and will motivate algorithm selection based on this evaluation but will restrict any quantitative analysis to only those methods used in the experiments (Sect. IV). The discussion follows the logical flow of the system design (Fig. 2), but first we provide relevant notation.

S. Shen, N. Michael, and V. Kumar are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA. {shaojie, nmichael, kumar}@grasp.upenn.edu

We gratefully acknowledge the support of NSF grants IIS-0427313 and IIP-0742304, ARO Grant W911NF-05-1-0219, ONR Grants N00014-07-1-0829 and N00014-08-1-0696, ARL Grant W911NF-08-2-0004, and Lockheed Martin.

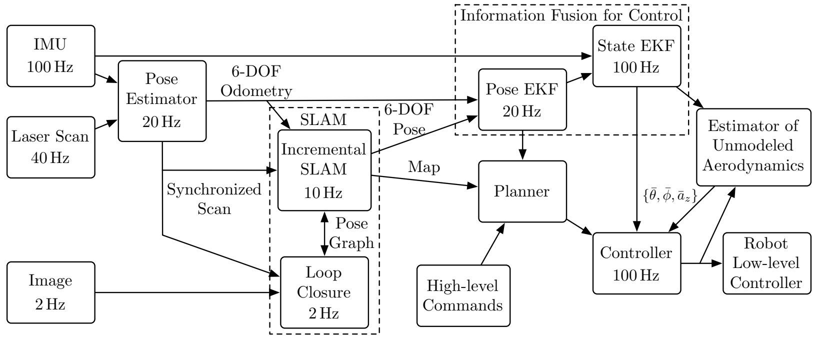


Fig. 2. Architecture diagram showing the important software modules with update rates.

A. Notation

The world frame, $\mathcal{W}$, is defined by axes x_W, y_W, and z_W, with z_W pointing upward. The body frame, $\mathcal{B}$, is attached to the center of mass of the quadrotor with x_B coinciding with the preferred forward direction and z_B perpendicular to the plane of the rotors pointing vertically up during perfect hover. We use ZYX Euler angles to model the rotation of the quadrotor in the world frame. To get from $\mathcal{W}$ to $\mathcal{B}$, we first rotate about z_W axis by the yaw angle, ψ, then rotate about the intermediate y-axis by the pitch angle, θ, and finally rotate about the x_B axis by the roll angle, ϕ. The rotation matrix that transforms coordinates from $\mathcal{B}$ to $\mathcal{W}$ is given by

$$R = R_\psi R_\theta R_\phi$$

where R_ϕ, R_θ, and R_ψ are elementary rotation matrices with respect to the x, y, and z axes. The six degree-of-freedom (DOF) pose of the center of mass in the world frame is defined by $\{x, y, z, \phi, \theta, \psi\}$.

B. Pose Estimation

A scanning laser range sensor retrofitted with mirrors for beam redirection to the floor and ceiling serves as a primary source of information for position and yaw estimation. We evaluated several laser-based methods for pose estimation such as exhaustive search [5] and feature-based approaches [6, 7]. However, as the vehicle dynamics require pose estimates with update rates of 20 Hz and our limited onboard computational resources, we chose the Iterative Closest Point (ICP) algorithm [8], which yields a robust and inexpensive continuous pose estimate. We make use of a grid based search [9] to speed up the computationally expensive closest point search in ICP. The result of the ICP algorithm is an estimate of $\{x, y, \psi\}$. The algorithm implementation is able to run at 20 Hz and requires approximately 20% of the total CPU time of the onboard processor.

A Kalman Filter (KF), similar to [3], fuses IMU data with redirected laser scans to provide altitude estimation (Sect. III-B). The remaining state variables, $\{\phi, \theta\}$, are estimated using the onboard IMU.

C. Simultaneous Localization and Mapping

We address the problems of mapping and drift compensation via an incremental simultaneous localization and mapping (SLAM) algorithm. Both particle filter-based occupancy grid [10] and feature-based [11, 12] methods perform well in practice. However, after evaluating these approaches, we found the demands too strenuous for the onboard processor. Therefore, we pursue a simplified occupancy grid-based incremental SLAM algorithm.

Given the incremental motion of the robot provided by ICP-based scan matching and the IMU, we correct the error in $\{x, y, \psi\}$ by aligning incoming laser scans against the existing map using a windowed exhaustive grid search. The cost-map generated from the obstacles in the existing map are approximated by an image distance transform [13]. If a stable floor transition is detected by the pose estimator, we create a new layer in a multi-layered occupancy grid. The incremental SLAM algorithm runs at 10 Hz and consumes less than 30% of the total CPU time.

Unlike particle filter-based approaches, our incremental SLAM algorithm does not embed loop closure functionality. To correct the global inconsistency caused by incremental SLAM, we employ vision-based techniques to enable robust loop closure detection that does not depend on the actual pose estimation error [14]. A fixed size visual vocabulary is constructed offline by clustering a large number of SURF features collected from different environments [15]. The detected SURF features of each incoming image are converted into the vocabulary format and matched against previous images via histogram voting. Any matched loop closure candidates are further verified by scan matching.

Loop closures add constraints to a pose graph, where each node in the graph is a sparse sample of the robot poses and their associated sensor data. Akin to the methods of [16], we apply an optimization based on an Iterative Extended Kalman Filter (IEKF) to create a globally consistent pose graph configuration. The optimization occurs in the full 6-DOF pose space with the assumption that closure only happens at the same floor level. From this optimization, we resolve a globally consistent multi-floor map and pose graph. Approximations that further speed up the optimization are detailed in Sect. III-C.

D. Information Fusion for Control

To ensure high-rate, accurate, and drift-free pose estimates for feedback control, we use two separate Extended Kalman Filters (EKF) to fuse and boost the pose estimate to 100 Hz. The first EKF combines the 20 Hz pose estimate and the

10 Hz SLAM pose correction. The process update is:

$$\mathbf{x} = f(\mathbf{x}, \mathbf{u}) = \mathbf{x} \oplus \mathbf{u}$$

where $\mathbf{x}$ is the pose estimate, $\mathbf{u}$ is the incremental motion provided by the pose estimator, and $\oplus$ is the pose update function defined in [17]. The information from the pose estimator and SLAM are not independent but the major role of this EKF is to provide smooth compensation for delay in the SLAM estimate introduced by large-scale loop closure and map corrections. We additionally employ a delayed measurement update scheme to match the timing of the different sources of information driving the pose estimate.

The second EKF combines the 20 Hz pose estimate from the first EKF and the 100 Hz IMU data to provide 100 Hz pose and linear velocity estimates in world frame. The final estimation output has an average delay of 0.01 s and feeds directly into the feedback control loop of the robot for position and velocity control.

E. Planning and Control

The focus of this work is primarily the mapping and localization required for autonomous navigation. As such, we detail preliminary approaches to planning and control here and defer discussion of ongoing efforts to Sect. V.

Given the current pose estimate of the robot and map of the environment, we pursue two path planning solutions: naive waypoint following with proportional feedback control based on a desired goal and the current state, and an incremental sampling-based method (RRT*) [18]. Either case treats the robot as a point-model kinematic system with orientation (about z_W). We employ proportional-derivative feedback control laws to transform these kinematic inputs to appropriate dynamic control inputs [19]. Tuning of control gains is accomplished via optimal control methods (LQR) based on a linearized system model [20].

III. Extensions

In the previous section we described the system design and discussed algorithm selection based on performance requirements and processor/sensor constraints. We now detail extensions to these algorithms that permit real-time operation on multiple floors. A consequence of these extensions are approximations that decrease resource demands but introduce limitations. Additionally, we discuss online estimation of aerodynamic effects that result due to wind or propeller backwash in small corridors or near walls.

A. Environment Assumptions

As the domain of interest is indoor environments and periphery, we assume 2.5 D environment models formed by collections of vertical walls and horizontal ground planes, all assumed to be piecewise constant. Let $[x_s, y_s, z_s]^{\mathrm{T}}$ be the laser scan endpoints in the body frame. We can project the laser scans to a horizontal plane by:

$$\left[x_g,\, y_g,\, z_g\right]^{\mathrm{T}} = R_\theta R_\phi \left[x_s, y_s, z_s\right]^{\mathrm{T}}$$

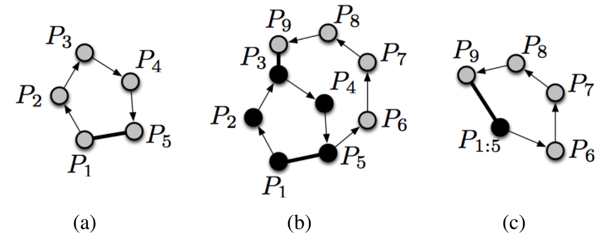


Fig. 3. Loop closure optimization of Sect. III-C. A robot transitions through pose graph nodes P_1–P_5 with loop closure detected between P_1 and P_5 (Fig. 3(a)). After continuing, loop closure occurs again between P_3 and P_9 (Fig. 3(b)), but any future optimization updates view the loop as previously closed and only consider $P_{1:5}$–P_9 (Fig. 3(c)).

We eliminate the scans that hit the floor or ceiling, which are generally not useful for scan matching, by comparing z_g with the deflected laser scans pointing upward and downward.

Although this approach largely simplifies the challenges of full 3D scan matching using only 2D laser range finders, the 2.5 D environment assumption is violated in highly clustered and outdoor environments with natural structure. In our experiments we see that partial structure in the environment satisfies this assumption and the overall performance is acceptable.

B. Altitude Measurement

We measure the variation in altitude with scans pointing to the floor and use this value to approximate the variance used in the measurement update of the KF. If the variation is too high, generally due to uneven surfaces or during a floor level change, we discard the current laser measurement and defer the measurement update until a stable measurement is obtained. An additional mirror deflects scans vertically upward toward the ceiling. These measurements provide additional robustness should there be no stable downward measurements available. When no upward or downward laser measurements are available, we use an onboard pressure sensor for short term measurement updates and force the robot to lower its height. Corrections of accumulated errors caused by frequent floor transitions are resolved through loop closure.

C. Incremental, Approximate Loop Closure

Loop closure for large loops is typically computationally expensive with significant memory requirements. While several methods exist for reducing the computational burden [16, 21, 22], we found that these methods are too demanding for a 1.6 GHz processor. With the understanding that we are more concerned with global consistency than absolute accuracy, we propose a mild extension that results in an incremental, but approximate approach.

As shown in Fig. 3, the general idea of this approach follows: a robot travels from P_1 to P_5. Loop closure is detected between P_1 and P_5; we contract this loop to $P_{1:5}$. We do not discard nodes in the pose graph as these are required for closure detection. However, we do ignore future iterations on this loop. Therefore, if the robot travels from P_5 to P_9, where a loop closure is detected between P_3 and

P_9, we do not correct any pose graph configuration inside the loop $P_{1:5}$. In this way, we only close the "new" loops.

Although the proposed method maintains a globally consistent map, it does not improve any loop structure that is already corrected, and is therefore approximate. Closing inner loops, ignoring this approximation, yields solutions consistent with traditional approaches. This will lead to locally distorted maps (i.e. the walls bend slightly), but the overall loop structure is correct. In practice, however, we find that even a slightly distorted map is still sufficient to localize the robot even in environments with multiple loops.

Our current IEKF optimizer has a complexity of $\boldsymbol{O}(n)$, where n is the total number of poses in the loop. However, it will be straight forward to adapt this approach to a more advanced optimization technique that has a complexity of $\boldsymbol{O}(\log n)$ [22].

D. Estimating and Compensating for Unmodeled Aerodynamics

In order to account for unmodeled aerodynamics we consider the second order model of the robot with additive disturbances:

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ F \end{bmatrix} - \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}$$

where F is the total force output from the rotors, and $[f_x,\, f_y,\, f_z]^T$ is the disturbance vector assumed to be a slowly varying term. While this term can clearly model changes in lift caused by changes in the aerodynamics and effects of the downwash or constant wind gusts, it can also model changes in the trim caused by a drift in the IMU or other changes in the dynamic model.

Recall that lateral robot accelerations are controlled via roll and pitch, and vertical acceleration via the thrust along z_B. To maintain hover under the effect of external disturbances, after linearizing the second order model about the near hover state ($\ddot{x} = \ddot{y} = \ddot{z} = 0$, $\phi \approx 0$, $\theta \approx 0$, $F \approx mg$), we get the z-direction acceleration and roll and pitch angles $\{\bar{a}_z,\, \bar{\phi},\, \bar{\theta}\}$ required to compensate for external disturbances:

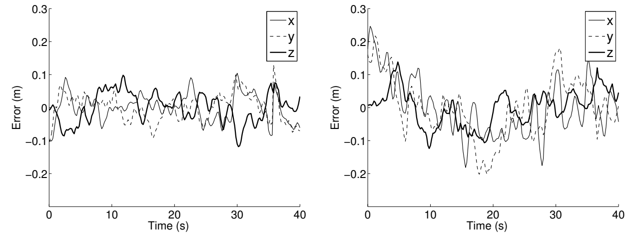
$$\begin{aligned} f_x/m &= g(\bar{\theta}\cos\psi + \bar{\phi}\sin\psi) \\ f_y/m &= g(\bar{\theta}\sin\psi - \bar{\phi}\cos\psi) \\ f_z/m &= \bar{a}_z \end{aligned}$$

We estimate these parameters, $\{\bar{a}_z,\, \bar{\phi},\, \bar{\theta}\}$, via a KF, with the state and compensated desired control defined by

$$\begin{aligned} \mathbf{x} &= \left[x,\, y,\, z,\, \dot{x},\, \dot{y},\, \dot{z},\, \bar{\phi},\, \bar{\theta},\, \bar{a}_z\right]^{\mathrm{T}} \\ \mathbf{u} &= \begin{bmatrix} \ddot{x}^{\mathrm{d}} \\ \ddot{y}^{\mathrm{d}} \\ \ddot{z}^{\mathrm{d}} \end{bmatrix} + \begin{bmatrix} g(\bar{\theta}\cos\psi + \bar{\phi}\sin\psi) \\ g(\bar{\theta}\sin\psi - \bar{\phi}\cos\psi) \\ \bar{a}_z \end{bmatrix} \end{aligned}$$

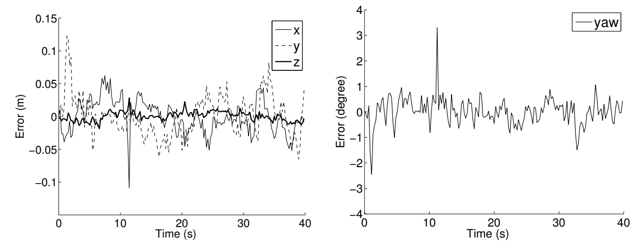
We discretize the system model and use a standard discrete KF state update equation:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{u}_t$$

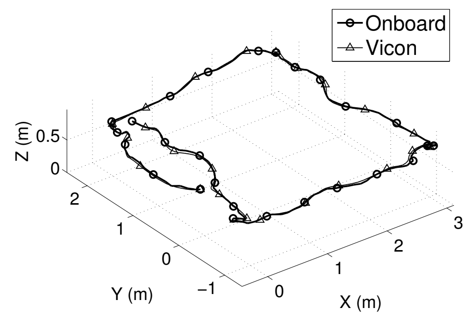


(a) Vicon (b) Onboard Estimator

Fig. 4. The robot is commanded to hover based on feedback from the Vicon (Fig. 4(a)) and onboard estimator (Fig. 4(b)).



(a) (b)



(c)

Fig. 5. Error between estimates from Vicon and onboard estimator (Figs. 5(a)–5(b)) while the robot controls along a specified trajectory (Fig. 5(c)).

where

$$A = \begin{bmatrix} I_3 & \Delta t I_3 & -\frac{g\Delta t^2}{2}M \\ 0_3 & I_3 & -g\Delta t M \\ 0_3 & 0_3 & I_3 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{\Delta t^2}{2} I_3 \\ \Delta t I_3 \\ 0_3 \end{bmatrix}$$

$$M = \begin{bmatrix} \sin\psi & \cos\psi & 0 \\ -\cos\psi & \sin\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

With slow varying parameters, the estimator will converge if the robot stays in a specific environment for a period of time. Of course there are no guarantee if the robot operates in environments with rapidly changing wind conditions or sudden change in ceiling/terrain heights.

IV. Experimental Results

A. Experiment Design and Implementation Details

Four experiments are presented: (1) a study of the estimator performance compared to ground truth data; (2) navigation across two floors in an indoor environment; (3) navigation starting in the same environment and transitioning

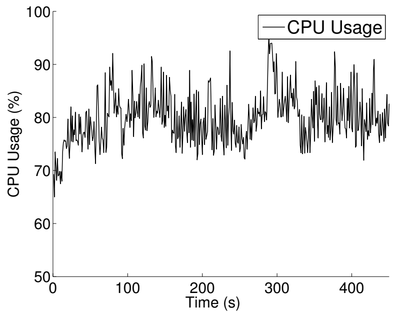


Fig. 6. Percentage of CPU time used by the system (including host OS) while following a trajectory similar to Fig. 5(c).

to the outdoor periphery; and (4) a large two-floor loop. The motivation for each study is stated in the respective discussions.

The robot platform is sold by Ascending Technologies, GmbH [23] and is equipped with an IMU (accelerometer, gyroscope, magnetometer) and pressure sensor. We developed custom firmware to run at the embedded level to address feedback control and estimation requirements. The other computation unit onboard is a 1.6 GHz Atom processor with 1 GB of RAM. The sensors on the robot include a Hokuyo UTM-30LX (laser), and a uEye 1220SE with a Kowa 3.5 mm f1.4 lens (camera). A custom 3D printed mount is attached to the laser that houses mirrors pointing upward and downward (along z_B). Communication with the robot for monitoring experiment progress is via 802.11n or 802.11s networking. All algorithm development is in C++ using ROS [24] as the interfacing robotics middleware. The experiment environment includes three buildings and a courtyard in the School of Engineering and Applied Science at the University of Pennsylvania.

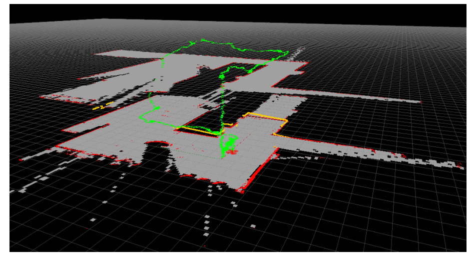
B. Evaluating Estimator Performance

We wish to study the performance of the onboard estimator as compared to ground truth, where ground truth is defined by a sub-millimeter accurate Vicon motion tracking system [25]. Two studies are presented. The first compares feedback control to maintain a single position using ground truth data for feedback as compared to the same scenario but using the pose resulting from the onboard estimator (Fig. 4). The standard deviations of the errors in the Vicon case are $\{\sigma_x, \sigma_y, \sigma_z\} = \{4.16, 4.16, 4.54\}$, while using the onboard estimator, the standard deviations of the hover performance are $\{\sigma_x, \sigma_y, \sigma_z\} = \{8.49, 9.11, 5.54\}$ (units in cm).

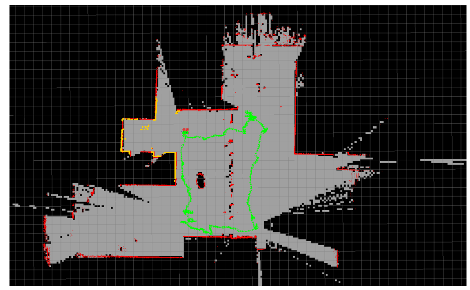
The second study considers the accuracy of the onboard estimate compared to the Vicon estimate while the robot controls along a specified trajectory (Fig. 5). In this case, the Vicon and onboard estimate compare well with a standard deviation of $\{\sigma_x, \sigma_y, \sigma_z\} = \{2.47, 3.23, 0.70\}$ and $\sigma_\psi = 0.55$ (units in cm and deg, respectively).

C. Indoor Navigation across Multiple Floors and Outdoors

We now consider autonomous navigation between multiple floors in an indoor environment. Additionally, we consider the case when a robot exits the indoor environment into the surrounding area. While the quadrotor is autonomous, the planned trajectory is determined by a user specifying goal locations with respect to the map.



(a) Indoor



(b) Indoor/Outdoor

Fig. 7. Maps generated while flying across multiple floors (Fig. 7(a)) and into the peripheral outdoor environment (Fig. 7(b)).

In Fig. 7(a), we see the map generated by the robot navigating through an indoor two-floor environment. Corresponding images of the robot flying are shown in Figs. 8(a)-8(b). While we do not have ground truth for these cases, we observe that there is minimal error or drift in the map. The robot is commanded to return and hover at its starting location. The error was observed to be on the order of a couple of centimeters.

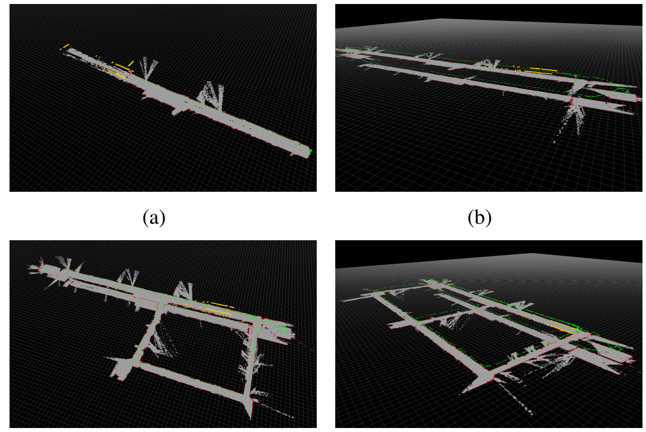
The second case leads the robot out of the indoor environment and into a courtyard (Fig. 7(b)). We see the removal of the 2.5-D assumption and observe that the robot is able to localize when observing multiple walls. However, further testing suggests that these methods do not extend to completely unstructured areas with trees. Another observation, and motivation for Sect. III-D, is that aerodynamic effects due to changes in varying ceiling height and distance to the floor are non-trivial when entering and exiting the building. Figures 8(c)-8(d) show corresponding images.

D. Closing Large Multi-floor Loops

The final experiment seeks to push the limits of the onboard processing and demands non-trivial loop closure across multiple floors. To pursue a large scale experiment with a length that exceeds feasible flight time, we carry the vehicle such that it emulates flight. Maps at multiple stages of the experiment are shown in Fig. 9. It is worth noting that even with loop closure, the onboard CPU usage remained on par with that shown for the trajectory following case in Fig. 6.

(a) (b) (c) (d)

Fig. 8. Navigation between multiple floors (Figs. 8(a)-8(b)) and indoor/outdoor environments (Figs. 8(c)-8(d)). See the attached video or http://mrsl.grasp.upenn.edu/icra2011/movie.



(a) (b) (c) (d)

Fig. 9. Map with multiple large loops over two floors.

V. Conclusion and Future Work

In this paper, we considered the problem of autonomous navigation with a micro aerial vehicle in indoor environments. In particular, we discuss autonomous navigation in buildings with multiple floors. To ensure that the robot is fully autonomous, we require all computation to occur on the robot without need for external infrastructure, communication, or human interaction beyond high-level commands. We presented a system design and methodology that enables autonomous navigation with real-time performance on a mobile processor using only onboard sensors We reviewed experimental results with performance analysis and address the adaptation to changing aerodynamic conditions induced by wind disturbances or changes in airflow in constrained environments.

We are interested in pursuing autonomous exploration across multiple floors and are currently working on improved control strategies using alternate parameterizations of the system dynamics and robust control methods. We also wish to further optimize our implementation to reduce unnecessary computational complexity. Finally, we are extending this work to multiple robots toward cooperative exploration.

References

[1] A. G. Bachrach, "Autonomous flight in unstructured and unknown indoor environments," Master's thesis, MIT, Cambridge, MA, Sept. 2009.

[2] A. Bachrach, A. Garamifard, D. Gurdan, R. He, S. Prentice, J. Stumpf, and N. Roy, "Co-ordinated tracking and planning using air and ground vehicles," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics. Springer Berlin, 2009, vol. 54, pp. 137–146.

[3] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Kobe, Japan, May 2009, pp. 2878–2883.

[4] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Anchorage, AK, May 2010.

[5] E. B. Olson, "Robust and efficient robotic mapping," Ph.D. dissertation, MIT, Cambridge, MA, June 2008.

[6] G. D. Tipaldi and K. O. Arras, "FLIRT - interest regions for 2D range data," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Anchorage, AK, May 2010, pp. 3616–3622.

[7] F. Ramos, D. Fox, and H. Durrant-Whyte, "CRF-matching: Conditional random fields for feature-based scan matching," in *Proc. of Robot.: Sci. and Syst.*, Atlanta, GA, June 2007.

[8] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proc. of Intl. Conf. on 3-D Digital Imaging and Modeling*, Quebec City, Quebec, May 2001, pp. 145–152.

[9] D. Chetverikov, "Fast neighborhood search in planar point sets," *Pattern Recognition Letters*, vol. 12, no. 7, pp. 409–412, July 1991.

[10] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Barcelona, Spain, Apr. 2005, pp. 2432–2437.

[11] J. Civera, A. J. Davison, and J. Montiel, "Inverse depth parameterization for monocular SLAM," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 932–945, Oct. 2008.

[12] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *Intl. J. Robot. Research*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006.

[13] "The OpenCV library," http://opencv.willowgarage.com/wiki/.

[14] M. Cummins and P. Newman, "Probabilistic appearance based navigation and loop closing," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Rome, Italy, Apr. 2007, pp. 2042–2048.

[15] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. of the European Conf. on Computer Vision*, Graz, Austria, May 2006.

[16] C. Estrada, J. Neira, and J. D. Tardos, "Hierarchical SLAM: Real-time accurate mapping of large environments," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 588–596, Aug. 2005.

[17] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Rayleigh, NC, Mar. 1987, p. 850.

[18] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," in *Proc. of Robot.: Sci. and Syst.*, Zaragoza, Spain, June 2010.

[19] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro UAV testbed," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, Sept. 2010.

[20] A. Weinmann, *Uncertain Models and Robust Control*. New York: Springer-Verlag, 1991.

[21] C. Hertzberg, "A framework for sparse, non-linear least squares problems on manifolds," Ph.D. dissertation, University of Bremen, Bremen, Germany, Nov. 2008.

[22] G. Grisetti, C. Stachniss, and W. Burgard, "Nonlinear constraint network optimization for efficient map learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 428–439, Sept. 2009.

[23] "Ascending Technologies, GmbH," http://www.asctec.de/.

[24] "Robot Operating System," http://pr.willowgarage.com/wiki/ROS/.

[25] "Motion capture systems from Vicon," http://www.vicon.com/.