2014 International Conference on Unmanned Aircraft Systems (ICUAS)
May 27-30, 2014. Orlando, FL, USA

Autonomous Navigation of UAV in Forest

Jin Qiang Cui[1], Shupeng Lai[1], Xiangxu Dong[2], Peidong Liu[3], Ben M. Chen[3], Tong H. Lee[3]

Abstract— **This paper presents a navigation system that enables small-scale unmanned aerial vehicles to navigate autonomously in foliage environment without GPS using a 2D laser range finder. The navigation framework consists of real-time onboard motion estimation and trajectory smoothing using pose graph optimization, real-time dual layer control. In particular, onboard real-time motion estimation is achieved in a Kalman filter, fusing the planar velocity measurement from scan matching of laser range finder and the acceleration measurement of inertial measurement unit. The trajectory histories from the real-time autonomous navigation together with the observed features are fed into a pose-graph optimization framework. Poses in a sliding window are optimized using GraphSLAM technique. The inner loop of a quadrotor is stabilized using a commercial autopilot while the outer loop control is implemented using robust perfect tracking. The performance of the navigation system is demonstrated on the successful autonomous navigation of a small-scale UAV in forest. Consistent mapping of the environment in indoor and outdoor scenarios are achieved by projecting all the scan measurement on the post-optimized trajectory with GraphSLAM.**

I. INTRODUCTION

Navigation of mobile robotics platforms in GPS-denied environments is being intensively studied in the research community, such as indoor offices [1][2], underwater [3] and urban canyons [4]. Researchers in GRASP Laboratory, University of Pennsylvania, have performed extraordinary research regarding indoor multi-floor navigation using a 2D scanning range finder. As mentioned by the authors [1], the strong 2.5-D environment assumption limits their UAV's navigation capability in completely unstructured environment with trees. This paper presents the autonomous navigation of small-scale UAV in foliage environment.

Navigation of ground vehicles in foliage environment has been addressed in [5][6] where a car equipped with laser range finder drove through Victoria park in Sydney, Australia. The steep terrain, thick understorey vegetation and abundant debris characteristic of many forests prohibit the deployment of an autonomous ground vehicle in such scenario. An UAV with autonomous navigation capability would be of paramount importance in forest survey [7], exploration and reconnaissance.

The idea of autonomous flight of UAV in forest has been attempted using a low-cost inertial measurement unit (IMU) and a monocular camera [8], in which an Unscented Kalman Filter (UKF) was used to estimate the locations of obstacles and the state of UAV. Experiment verification was carried out with a remote control car running in synthetic outdoor environment. More recently, Ross et al. [9] realized autonomous flight through forest by mimicking the behavior of human pilot using a novel imitation learning technique. The application of learning technique is innovative but the system suffers from relatively high failure rate which the UAV cannot afford.

To the best of our knowledge, this paper presents the first successful autonomous navigation of a small-scale UAV in foliage environment. The navigation framework consists of onboard motion estimation and pose-graph optimization based on GraphSLAM [10], real-time dual layer control and online path planning . The modular design of the framework makes it readily applicable to other mobile robotic navigation system in obstacle-strewn environment.

The paper is organized as follows. Section II presents the system structure, including the hardware and software configuration. Section III introduces the dynamics model structure of the UAV and presents the design of robust perfect tracking control law. Section IV presents the state estimation framework consisting of real-time motion estimation for autonomous control and the post pose-graph optimization based on GraphSLAM. Section V presents the experimental results regarding the autonomous flight and pose-graph optimization in outdoor environment. Section VI concludes the paper.

II. NAVIGATION SYSTEM STRUCTURE

The navigation system of UAV is depicted in Fig. 1, including functional modules such as the UAV dynamics, autonomous control, state estimation and path planning. Each module itself is such an immense topic that covering it in detail is impossible. This section reports the techniques used in each of the module to realize the autonomous flight of UAV in forest.

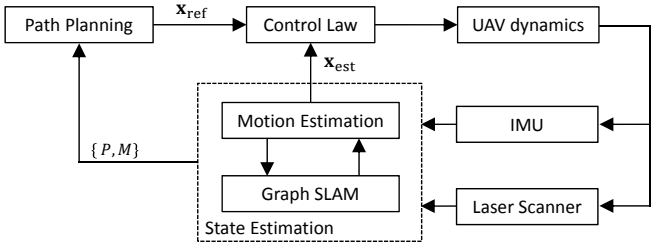


Fig. 1. System diagram of UAV navigation system.

[1] Jin Qiang Cui and Shupeng Lai are with the NUS Graduate School for Integrative Sciences & Engineering, National University of Singapore (NUS), Singapore. Email: {jinqiang, shupenglai}@nus.edu.sg

[2] Xiangxu Dong is with Temasek Laboratories, National University of Singapore, Singapore, Email: tsldngx@nus.edu.sg

[3] Ben M. Chen, Peidong Liu and Tong H. Lee are with the Department of Electrical & Computer Engineering, National University of Singapore, Singapore. Email: {bmchen, elelp, eleleeth}@nus.edu.sg

To maneuver in forest consisting of a large number of obstacles, the platform has to be compact in size and is capable of hovering in the air. Platforms like quadrotor, helicopter and coaxial rotor fulfill such requirements. In this paper, we use a quadrotor due to its extra advantages such as the symmetric structure, the easy assembly of payload and the vast availability of commercial off-the-shelf autopilot, etc. The dynamics of quadrotor consists of inner loop dynamics and outer loop dynamics. The inner loop dynamics relates the control input to the angular motion while the outer loop dynamics relates the angular motion to the linear velocity and position.

For autonomous navigation of UAV, the outer loop dynamics is of more concern. Therefore, we use a commercial autopilot ('NAZA-M') to stabilize the inner loop dynamics of quadrotor. For the outer loop dynamics, we design a Robust and Perfect Tracking (RPT) control law [11] to control the UAV to track the trajectory reference from the path planning module. The RPT problem is to design a controller such that the resulting closed-loop system is asymptotically stable and the controlled output almost perfectly tracks a given reference signal in the presence of any initial conditions and external disturbances. The almost perfect tracking means the ability of a controller to track a given reference signal with arbitrarily fast settling time despite of external disturbances and initial conditions.

The foliage environment renders the GPS signal unreliable, making the state estimation in such environment the top priority. We design the state estimation to include two parts: motion estimation based on scan matching and pose-graph optimization based on GraphSLAM [10]. The motion estimation is essentially a laser odometry technique, which uses scan matching of consecutive laser scan data to generate 2D velocity estimation and heading angle estimate. Combined with the acceleration measurement in IMU, a Kalman filter (KF) is designed to estimate the position and velocity of the UAV, which are used directly for the closed-loop control. On the other hand, the position estimate from the motion estimation is prone to drift, thus GraphSLAM is used as a post optimization process to achieve the optimal trajectory and consistent map.

III. UAV Model and Control

A. Model Structure

Following the routines of traditional aircraft model, the model structure of the quadrotor platform is separated into inner loop model and outer loop model, as illustrated in Fig. 2. The 6 degrees of freedom (DOF) dynamics of quadrotor consists of the 3-DOF translation dynamics and the 3-DOF angular dynamics. The angular dynamics of quadrotor is much faster than the translation dynamics. The inner/outer loop model structure provides insightful guidelines for the model derivation and future control law design.

In the current system hardware configuration, the inner-loop attitude dynamics is stabilized using the commercial autopilot 'NAZA-M', which is an all-in-one stability controller for multi-rotor platforms. The control inputs

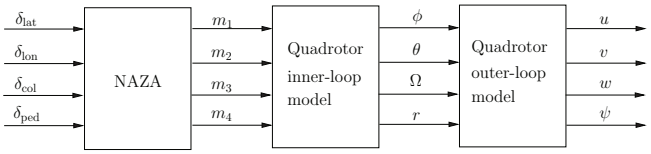


Fig. 2. Diagram of UAV model structure.

$(\delta_{\text{lat}}, \delta_{\text{lon}}, \delta_{\text{col}}, \delta_{\text{ped}})$ from the remote transmitter are fed into the onboard autopilot. Then NAZA controller generates PWM signals to drive the four rotors to generate the thrust forces, whose combined effect is to lift the platform and maintain the attitude stability. From the perspective of NAZA, the four inputs from the remote transmitter correspond to the control references for the roll angle (ϕ), pitch angle (θ), yaw angular rate (r), and average motor speed (Ω) respectively.

In the outer loop aspect, the states regarding linear velocity (u, v, w) and heading angle ψ are of concern. In the lateral and longitudinal dynamics, there is a dynamics model which maps the roll and pitch angle to the lateral and longitudinal velocity, which is governed by the rigid body kinematics with some damping effect from air resistance. The heave velocity w and heading angle ψ are governed directly by the inner loop controller.

The inner/outer model can be decoupled into four input/output pairs. Due to the symmetric system structure of the platform, we note that the dynamics model in roll and pitch direction share the same set of equations. Referring to Fig. 2, the inner/outer loop model can be separated into three groups: roll/pitch dynamics, yaw dynamics and heave dynamics.

Without knowledge of the inner loop model and the controller structure, we treat the inner loop as a black box and identify the inner loop model for the outer loop controller design. The model identification is performed in the frequency-domain. First, the platform with the inner loop controller is perturbed in roll, pitch, yaw and heave directions. The perturbed input/output are logged and processed in a software called CIFER to derive a transfer function which suites the best for the logged data [2]. The identified sub system models are listed as follows:

- Roll/Pitch dynamics: input $\delta_{\text{lat}}/\delta_{\text{lon}}$, output ϕ/θ
  Transfer function:
  $$H_1 = \frac{9688}{s^4 + 27.68\,s^3 + 485.9\,s^2 + 5691\,s + 15750}.$$
- Yaw dynamics: input δ_{ped}, output ψ
  Transfer function:
  $$H_2 = \frac{3.372}{s}.$$
- Heave dynamics: input δ_{col}, output w
  Transfer function:
  $$H_3 = \frac{-13.35}{s + 2.32}.$$

B. Outer Loop Control

In obstacle-dense environment, fast and accurate maneuvering of the UAV is required to avoid possible collision. This poses challenges for the design of outer loop controller. We need to control the UAV to follow external references including the linear position and the heading angle. In order to perform fast and precise tracking of the given references, the RPT controller is adopted from [11]. The procedures of designing an RPT controller for the state feedback case has been addressed in [11] and a case study is exemplified in [12].

For precision control, it's desirable to include an integrator to ensure zero steady state error in case of step input. We propose a RPT controller which considers the integration of position tracking error as an augmented state. The system with state-augmentation is formulated as:

$$\Sigma_{\mathrm{AUG}}^{\mathrm{xy}}:\begin{cases}\dot{\widetilde{\mathbf{x}}}_{\mathrm{xy}}=\begin{bmatrix}0&-1&0&0&1&0\\0&0&1&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&0\\0&0&0&0&0&1\\0&0&0&0&0&0\end{bmatrix}\widetilde{\mathbf{x}}_{\mathrm{xy}}+\begin{bmatrix}0\\0\\0\\0\\0\\1\end{bmatrix}\mathbf{u}_{\mathrm{xy}}\\ \widetilde{\mathbf{y}}_{\mathrm{xy}}=\widetilde{\mathbf{x}}_{\mathrm{xy}}\\ \widetilde{\mathbf{h}}_{\mathrm{xy}}=\begin{bmatrix}1&0&0&0&0&0\end{bmatrix}\widetilde{\mathbf{x}}_{\mathrm{xy}}\end{cases} \tag{1}$$

where $\widetilde{\mathbf{x}}_{\mathrm{xy}}=[\int e\ \ r_p\ \ r_v\ \ r_a\ \ p\ \ v]^T$; r_p, r_v, r_a are the position, velocity and acceleration references; p, v are the actual position and velocity; $e=p-r_p$ is the tracking error of position. Since there is error integration $\int e$ in the augmented states, the feedback control law would contain a term of $K_i\int e$. Following the steps in [11], a linear state feedback control law of the form (2) is acquired,

$$\mathbf{u}_{\mathrm{xy}}=\mathbf{F}_{\mathrm{xy}}(\varepsilon)\,\widetilde{\mathbf{x}}_{\mathrm{xy}}\,, \tag{2}$$

where

$$\mathbf{F}_{\mathrm{xy}}(\varepsilon) = \begin{bmatrix}\frac{-k_{\mathrm{i}}\omega_{\mathrm{n}}^2}{\varepsilon^3} & \frac{\omega_{\mathrm{n}}^2+2\zeta\omega_{\mathrm{n}}k_{\mathrm{i}}}{\varepsilon^2} & \frac{2\zeta\omega_{\mathrm{n}}+k_{\mathrm{i}}}{\varepsilon}\\ 1 & -\frac{\omega_{\mathrm{n}}^2+2\zeta\omega_{\mathrm{n}}k_{\mathrm{i}}}{\varepsilon^2} & -\frac{2\zeta\omega_{\mathrm{n}}+k_{\mathrm{i}}}{\varepsilon}\end{bmatrix}, \tag{3}$$

where ε is a design parameter to adjust the settling time, ω_n, ζ, k_i are the parameters that determine the desired pole locations of the infinite zero structure of $\Sigma_{\mathrm{AUG}}^{\mathrm{xy}}$ through:

$$p(s)=(s+k_{\mathrm{i}})(s^2+2\zeta\omega_{\mathrm{n}}s+\omega_{\mathrm{n}}^2)\,. \tag{4}$$

In principle, when the design parameter ε is small enough, the RPT controller gives arbitrarily fast response. However, in practice, due to the constraints of physical system and inner loop dynamics, we would like to limit the bandwidth of the outer loop to be at least one third of the inner loop system bandwidth. The roll/pitch dynamics H_1 has a bandwidth of 3.82 rad/s. For roll/pitch outer loop controller, we select the parameters in (5) to have a bandwidth of 0.83 rad/s:

$$\omega_{\mathrm{n}}=0.4,\ \ \zeta=1.2,\ \ \varepsilon=1,\ \ k_{\mathrm{i}}=0.8\,. \tag{5}$$

For the outer loop controller in heave dynamics and yaw dynamics, the inner loop controller already controls the heave velocity w and yaw angular velocity r. We only need to design a controller to control the height z and yaw angle ψ. Similarly, the integral of tracking error is augmented to the original system and forms another augmented system $\Sigma_{\mathrm{AUG}}^{\mathrm{hy}}$:

$$\Sigma_{\mathrm{AUG}}^{\mathrm{hy}}:\begin{cases}\dot{\widetilde{\mathbf{x}}}_{\mathrm{hy}}=\begin{bmatrix}0&-1&0&1\\0&0&1&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\widetilde{\mathbf{x}}_{\mathrm{hy}}+\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\mathbf{u}_{\mathrm{hy}}\\ \widetilde{\mathbf{y}}_{\mathrm{hy}}=\widetilde{\mathbf{x}}_{\mathrm{hy}}\\ \widetilde{\mathbf{h}}_{\mathrm{hy}}=\begin{bmatrix}1&0&0&0\end{bmatrix}\widetilde{\mathbf{x}}_{\mathrm{hy}}\end{cases} \tag{6}$$

where $\widetilde{\mathbf{x}}_{\mathrm{hy}}=[\int e\ \ r_p\ \ r_v\ \ p]^T$; r_p, r_v are the position, velocity references; p is the actual height or yaw angle; $e=p-r_p$ is the tracking error of height or yaw angle.

A linear state feedback control law of the form (7) is acquired,

$$\mathbf{u}_{\mathrm{hy}}=\mathbf{F}_{\mathrm{hy}}(\varepsilon)\,\widetilde{\mathbf{x}}_{\mathrm{hy}}\,, \tag{7}$$

where

$$\mathbf{F}_{\mathrm{hy}}(\varepsilon)=\begin{bmatrix}-\frac{\omega_n^2}{\varepsilon} & \frac{2\zeta\omega_n}{\varepsilon^2} & 1 & -\frac{2\zeta\omega_n}{\varepsilon^2}\end{bmatrix}. \tag{8}$$

For height controller:

$$\omega_{\mathrm{n}}=0.5,\ \ \zeta=1.1,\ \ \varepsilon=1\,. \tag{9}$$

For yaw angle controller:

$$\omega_{\mathrm{n}}=1,\ \ \zeta=1,\ \ \varepsilon=1\,. \tag{10}$$

IV. Navigation State Estimation

A. Estimation Framework

The state of UAV includes the position, the velocity and the orientation. The orientation could be estimated by the mechanization of angular measurement from IMU with no bias nor drift. In the ideal case, the translation velocity could be derived by integrating the acceleration measurement once and the position for another integration. Since the accelerometer output is subject to bias, the double integration of acceleration will result in prohibitively large position drift, rendering it inapplicable for long time navigation of UAV. To facilitate successful autonomous navigation of UAV, we proposed a state estimation framework as shown in Fig. 3 to extract the position and velocity in 3D space, which consists of two sub-systems: front-end and back-end.

The front-end provides the essential estimate for the onboard autonomous control and the back-end optimizes the UAV trajectory for pose correction and consistent mapping of the environment. However, the GraphSLAM is in essence an off-line optimization technique which can not be ran in parallel with the onboard control loop. In practice, a sliding time window is applied to the trajectory history and only those poses inside the sliding window are optimized. After the whole trajectory is obtained, a global optimization in the back-end is performed to obtain the optimal trajectory and the globally consistent map.

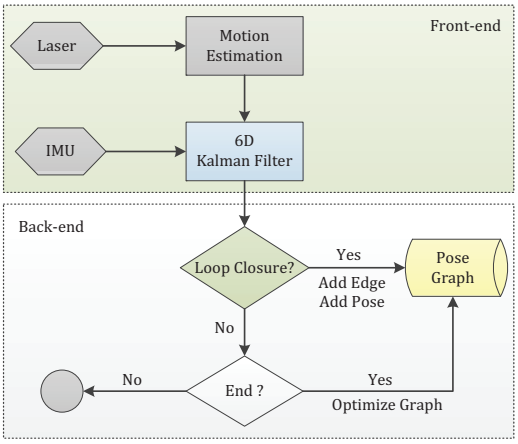


Fig. 3. System schematics illustrating front-end and back-end.

In Fig. 3, we have put the loop closure detection block to the back-end. This makes sense because we utilize the position and velocity estimates from Kalman filter as the input into the the real-time autonomous control. By putting the loop detection into the back-end, the back-end module operates in a self-contained manner which will not influence the real time performance of the UAV control.

B. Front-end Estimation

The front-end block provides high frequency state estimate for onboard autonomous control and the initial pose estimate for the back-end block. The block diagram of the front-end state estimation is shown Fig. 4, which consists of three main blocks: feature extraction, scan matching and Kalman Filter.

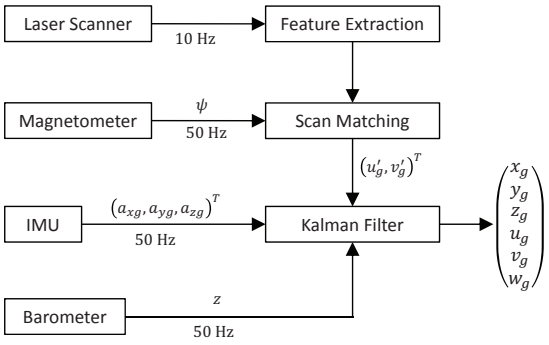


Fig. 4. The front-end state estimation diagram.

Feature extraction is the first step toward accurate motion estimation. Considering the operation environment to be forests, we choose the center of trees at flight height to be the features. A laser range finder scans the environment continuously to provide range information in 30 meter range of 270 degree field of view. To extract the validated trees, the laser scan range are processed in three steps: preprocessing, segmentation and extraction. Preprocessing the scan range is necessary since the raw laser data is noisy and corrupted with outliers. Segmentation is performed based on the jump distance between two consecutive measurements in the same scan, generating clusters of range points corresponding to candidate tree stems. The feature extraction process validates these clusters through a series of geometric descriptors, such as cluster width, number of points, producing a series of validated clusters. The estimated centers of the validate clusters are treated as the landmarks [13].

Scan matching of 2D range scans is a mature technique to estimate robot pose in unknown environment. But it suffers from local minima and large rotation errors when using only scan matching. To overcome these problem, we use feature based scan matching and incorporate the heading angle measurement from IMU in the scan matching process. The extracted features in each scan maintain a certain spatial distance to each other, making it less possible for ambiguous data association. With the heading measurement from IMU, the current scan is first projected to the previous scan using the heading angle difference. This further reduces the possibility of wrong data association induced by possible large rotation movement. Once the correct data association is established, the incremental translation and rotation could be solved in closed-form [14].

A Kalman filter is designed to fuse the sensor information: the translation measurement (u'_g, v'_g) from scan matching, the height measurement z from barometer or mirrored laser beam and the three axis acceleration from IMU.

The process model is described by,

$$\begin{bmatrix}\dot{\mathbf{P}}_\mathbf{n}\\ \dot{\mathbf{v}}_\mathbf{n}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{P}_\mathbf{n}\\ \mathbf{v}_\mathbf{n}\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{I}\end{bmatrix}\mathbf{R}_{\mathbf{n/b}}(\mathbf{a_b}+\mathbf{w_a}), \qquad (11)$$

where $\mathbf{R_{n/b}}$ is the rotation matrix from the body frame to the local north-east-down (NED) frame, $\mathbf{a_b}$ is the acceleration measurement without noise, $\mathbf{w_a}$ is the acceleration measurement noise vector with normal distribution, $\mathbf{a_b}+\mathbf{w_a}$ is the IMU acceleration measurement in body-fixed frame, $\mathbf{P_n}$ is the local NED position vector, $\mathbf{v_n}$ is the local NED velocity vector. $\mathbf{I}$ and $\mathbf{0}$ are identity matrix and zero matrix of proper dimension.

The measurement model is

$$\mathbf{y} = [\mathbf{0}_{3\times2}\ \ \mathbf{I}_{3\times3}\ \ \mathbf{0}_{3\times1}]\begin{bmatrix}\mathbf{P}_\mathbf{n}\\ \mathbf{v}_\mathbf{n}\end{bmatrix} + \mathbf{R}, \qquad (12)$$

where $\mathbf{R}$ is the measurement covariance matrix with normal distribution.

Implementing the KF requires discretizing the continuous process model (11) and the measurement model (12) using zero-order-hold method. The following procedure is a standard KF process with interleaved time update and measurement update. One noteworthy point is that the laser motion estimation update rate is 10 Hz while the acceleration measurement refreshes at 50 Hz. When the motion estimation measurement is not available, the state is updated using only the process model (11).

C. Back-end Estimation

The position estimate from the Kalman filter is only suitable for short time navigation. This is due to the fact

that the planar position is not observable in the measurement equation and thus suffers from long term drift. The back-end of the state estimation solve the position drift problem.

Building up the graph and optimize the graph are the two main functions of the back-end block. The back-end takes in the position estimate from the front-end as the initial pose of the UAV. For building up the graph, the initial pose, together with the features seen on the pose, are fed into a loop closure detection block. The loop detection block detect whether there is sufficient overlap between the current feature set and all the previous feature sets in the time window. If yes, the current pose is added to the graph as a node and a constraint between the two overlapped nodes is added. Detecting the correct overlap is the key element for loop detection. In this paper, a feature-based scan matching is used as the data association method.

Once the graph is built, the back-end seeks to solve a nonlinear-least square problem to derive the optimal configuration of poses and landmarks. The nonlinear-least square problem is normally solved in an iterative manner: forming a linear system around the current state estimate, solving the linear system and iterating. Standard methods like Gauss-Newton, Levenberg-Marquardt, Gauss-Seidel relaxation, or variants of gradient descent algorithms are commonly used to solve this problem. We use the open source framework g^{2}o [15] to optimize the constructed pose graph.

V. Experiment Results

All the function blocks, including the RPT control law, the onboard motion estimation and the path planning, are implemented in the UAV onboard processor to form a comprehensive navigation system. The developed navigation system enables our customized UAV to perform autonomous flight through a small area of forest. First the navigation state estimation framework in verified in section V-A using simulated laser scans of circular tree stems. Then the front-end state estimation and the RPT control are verified by the flight test in section V-B. The back-end state estimation is validated using the onboard data collected in both indoor and outdoor environment in section V-C.

A. State Estimation in Simulation Environment

Building the simulation environment is essential in the sese that it can provide ground-truth measurement and trajectory, validating the algorithm developed. Random number of trees with various radius are generated in a planar plane. The UAV is modeled as a mass point and moved according to a set of predefined waypoints. Along the trajectory, the measurement of laser range finder is logged. The laser range finder is modeled according to the practical specification, including the measurement range, resolution and field of view. The ground-truth position of UAV is also logged together with the laser range finder measurement.

In order to show the effect of SLAM algorithm, we add Gaussian noise to the ground truth trajectory and treat it as the initial trajectory estimation. The ground-truth trajectory provides the benchmark for comparison. Fig 5 shows the overlapped map for the three types of trajectories: the ground-truth, the initial pose and the update pose. Fig. 6 shows other aspects of the evaluation. Fig 6(a) shows a enlarged view of a contour of one tree. From the zoom-in we could see that the measurement points projected on the initial trajectory (marked by green triangle) scatter around the ground truth contour of the tree (black dot). While the measurement points projected on the updated pose match the ground-truth perfectly. This proves that in Gaussian noise assumption, the GraphSLAM algorithm provides the optimal trajectory estimation. Fig. 6(b)-6(d) compare the difference of position and heading angle with respect to the ground truth, showing again that GraphSLAM produces the optimal trajectory estimation.

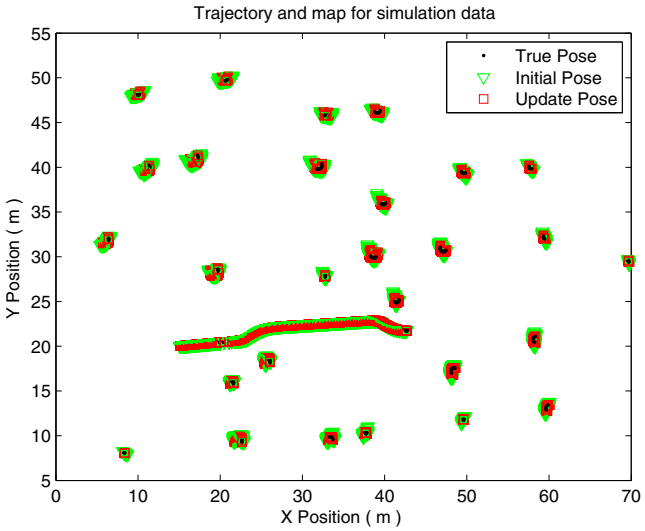


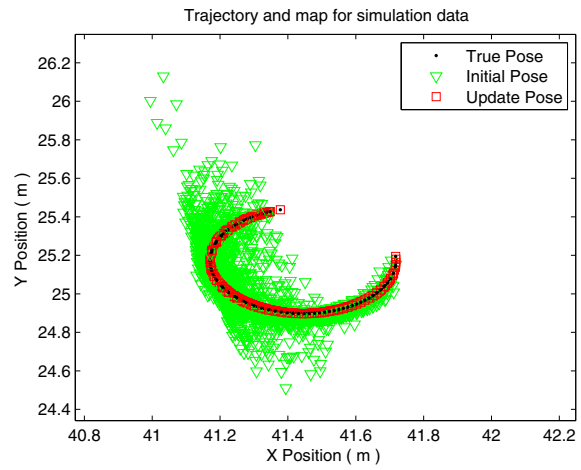
Fig. 5. Optimized map and trajectory in simulation environment.

B. Autonomous Flight in Forest

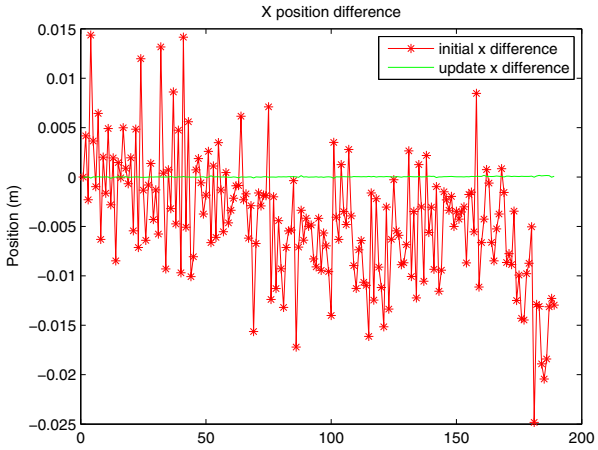
The test field is set in a small forest with sparse trees as shown in Fig. 7. The distances among trees are sparse enough to provide an obstacle-free trajectory for the UAV to fly. At flight height, some trees have thick branches instead of a single tree trunk. This poses challenges for the onboard feature extraction and data association. The obstacle-free trajectory is predefined by assuming knowledge of the tree positions in the environment. This simulates the case where the onboard obstacle avoidance is properly functional. It should be noted that the map information is not used during the process of motion estimation and autonomous control.

Fig. 7 shows the test platform flying in the test field. The UAV platform is being autonomously controlled while following a predefined trajectory. The predefined trajectory is loaded into the system during the system power up. The front-end state estimation provides the position and velocity estimates which are fed back to the RPT control law to control the UAV.

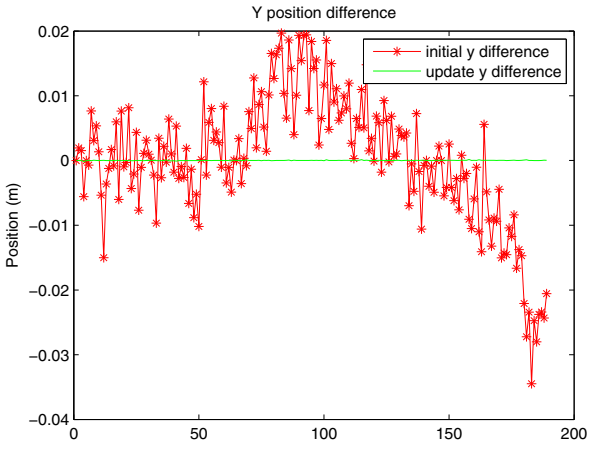
Fig. 8 shows the position tracking performance together with the tracking error. In the top two sub-figures, the red dashed lines are the position references and the blue solid



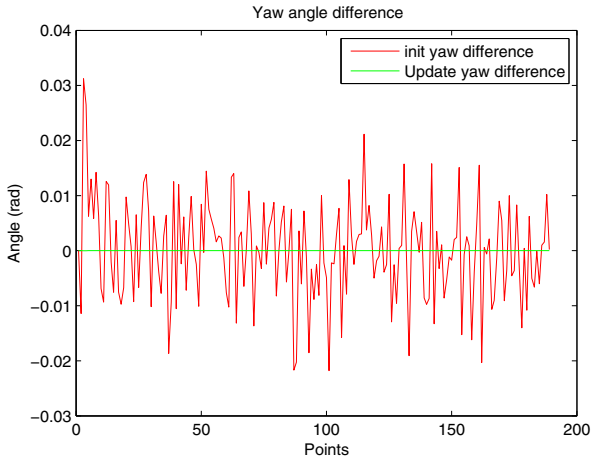
(a) Optimized tree contour



(b) X position difference



(c) Y position difference



(d) Heading angle difference

Fig. 6. Position and heading angle difference to ground truth.

Fig. 7. The outdoor testing scenario with the flying quadrotor.

lines are the state estimates from the UAV. As can be seen from the top two sub-figures, the RPT controller controls the UAV to track the position reference in both x and y direction. The tracking error in x direction is 0.2 meter and 0.5 meter in y direction. The maximum tracking errors occur at the beginning of the trajectory when there is step acceleration reference.

The flight test demonstrates the accuracy of the UAV dynamics model and the efficiency of onboard motion estimation in a computation-limited processing unit. It also shows that the front-end state estimate can provide reliable state estimate for the RPT controller.

C. Back-end Optimization

The back-end state estimation seeks to optimize the initial trajectory to produce a consistent map of the environment. To better verify the effect of trajectory optimization, we test the GraphSLAM algorithm using data collected in two environments: the indoor forest environment (Fig. 9) and the outdoor real forest (Fig. 7).

The indoor forest scenario consists synthetic trees with perfect cylindrical shapes. Besides the poles, square concrete pillars and the interior of the wall will also be measured by the laser range finder. We manipulate the distance among the trees to make sure there are enough number of features in each scan. An autonomous flight is performed to collect all the onboard data including the trajectory history and the

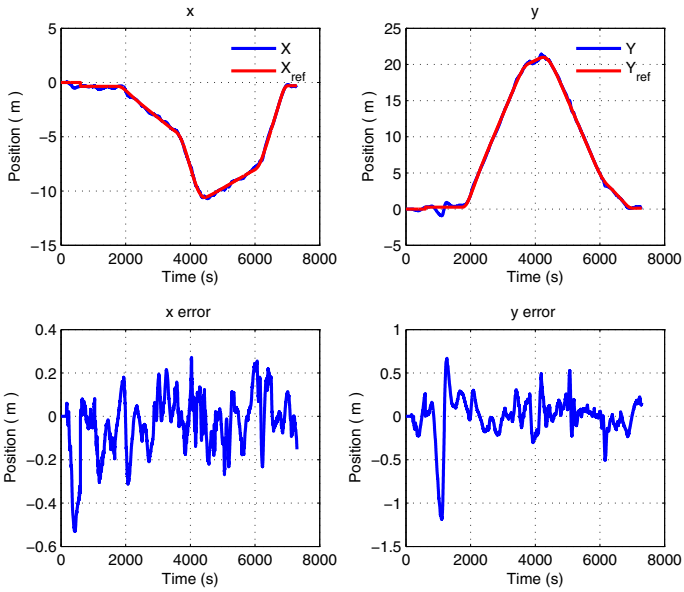


Fig. 8. Position and velocity tracking in X-Y direction.

Fig. 9. Indoor test scenario for SLAM verification.

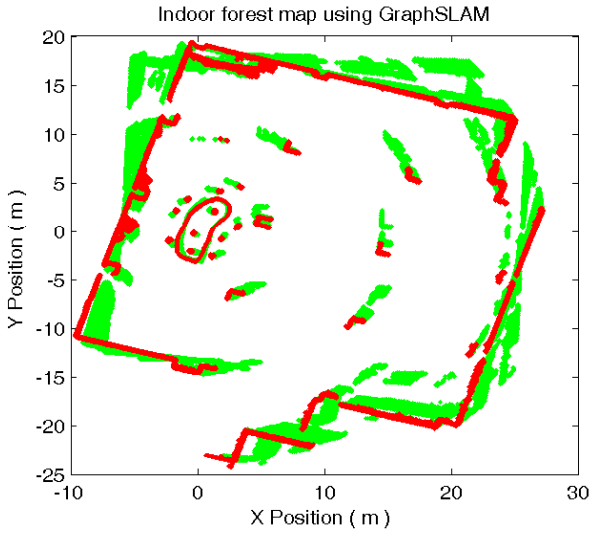


Fig. 10. Map comparison between initial and optimized trajectories.

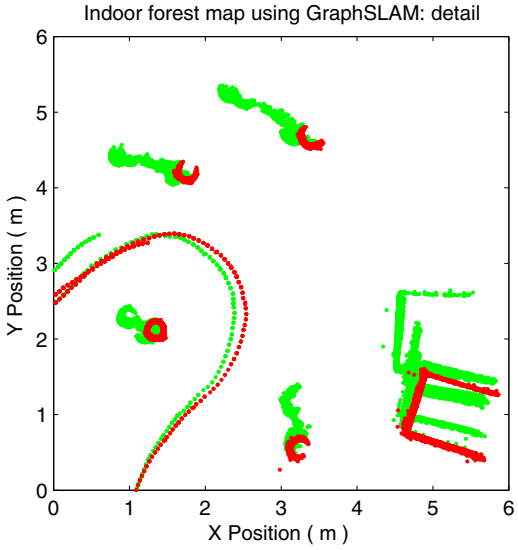


Fig. 11. Optimized map and trajectory detail.

laser range data.

Fig. 10 compares the map projected on the initial trajectory and the optimized trajectory respectively. Since there is no ground truth available in the indoor forest dataset, we can not quantitatively analyze the effect of trajectory optimization. We consider the consistency of the projected map as the criteria to evaluate the back-end state estimation. The initial map and trajectory are the results of motion estimation based on scan matching, marked by green dots plot. It can be seen that when all the measurements are projected on the initial trajectory, the overall map is not consistent. The interior walls and position of the square pillars drift away. The red-dot plot is the optimized map and trajectory. By visual checking of the map we could see the optimized map is more consistent than the initial map. Fig. 11 shows close-view of one part of the map, indicating that the red pillar retains its rectangular shape while the green pillars deform to an incorrect way. Fig. 11 also clearly manifests the perfect circular contour of the landmarks while the initial green contours scatter around.

After the evaluation of the GraphSLAM algorithm in indoor environment, we apply the algorithm to the dataset from in a real small forest. Fig. 12 and Fig. 13 show the comparison between the initial map and the optimized map for a small forest as shown in Fig. 7. The initial map is generated by projecting all the measurement on the initial poses from kalman filter. The red one in Fig. 13 is the optimized trajectory and map. After close inspection of the two figures, it can be seen that the optimized map is more consistent than the initial one. For example, in the location near $(-2, -5)$, there are two neighboring clusters of green plots in Fig. 12 while only one cluster of red points is seen in Fig. 13. Due to the complexity of the environment, there is no hollow tree contours extracted. There are still large clusters of objects which do not correspond to trees in the environment.

The effect of trajectory optimization is less optimal than the indoor forest dataset. This is expected since the real environment exhibits several challenges: first, the uneven terrain produces undesired ground strikes of the laser range

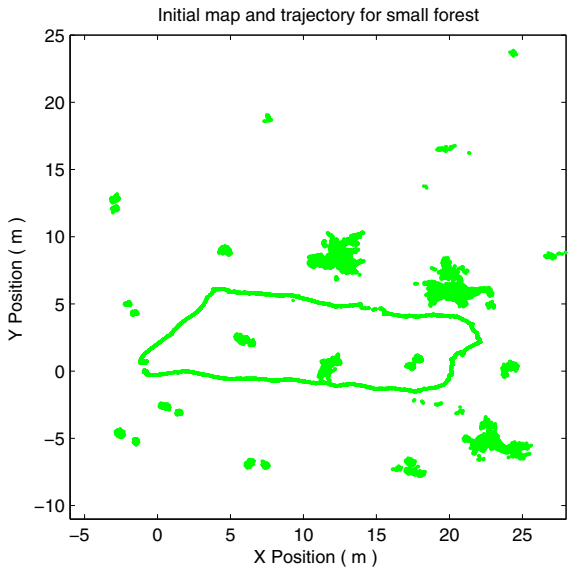


Fig. 12. Initial map and trajectory in small forest.

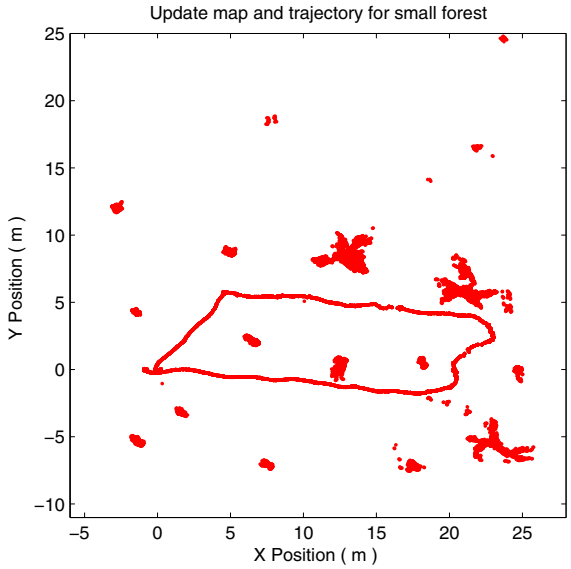


Fig. 13. Optimized map and trajectory in small forest.

finder, making the feature extraction a challenging task. Second, the real trees in the environment are relatively smaller than the ones in indoor forest and have thick branches at the flight height. Wrong data association may happen at a certain time step. Third, the distance among trees are very large, leaving each segment in the clustered range scan with limited number of points. Estimating the tree centers with less points produces large position error. Last, the cross section of tree trunk at the scanned plane may not follow a circular shape. The estimated position of tree centers may be different when the trees are scanned at different view angle. All these factors make the optimized map less consistent than the one in indoor forest. But with proper feature extraction and data association, the back-end optimization demonstrates its positive effect in correcting the trajectory and generating more consistent map.

VI. Conclusion

We have presented the navigation system for UAV in foliage environment. The model of a quadrotor UAV is first identified as inner/outer loop model before a robust perfect tracking outer loop control law is designed. The outer loop control law demonstrates perfect tracking of given position reference in the autonomous flight of UAV in forest. The state estimation consists of front-end and back-end. The front-end achieves real-time motion estimation in a Kalman filter framework, providing the essential state estimate for the outer loop control. The back-end takes the accumulated trajectory from the front-end and build a pose graph. The pose graph is optimized using GraphSLAM technique to produce consistent map of the environment. Successful autonomous navigation of UAV in forest is realized to demonstrate the performance of the navigation system.

References

[1] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained mav," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2011, pp. 20–25.

[2] F. Wang, J. Q. Cui, S. K. Phang, B. M. Chen, and T. H. Lee, "A mono-camera and scanning laser range finder based UAV indoor navigation system," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, May 2013, pp. 694–701.

[3] L. Paull, S. Saeedi, M. Seto, and H. Li, "AUV Navigation and Localization: A Review," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 131–149, Janunary 2014.

[4] A. Georgiev and P. Allen, "Localization methods for a mobile robot in urban environments," *IEEE Transactions on Robotics*, vol. 20, no. 5, pp. 851–864, October 2004.

[5] J. Guivant, E. Nebot, and S. Baiker, "Localization and map building using laser range sensors in outdoor applications," *Journal of Robotic Systems*, vol. 17, no. 10, pp. 565–583, 2000.

[6] J. Guivant, F. Masson, and E. Nebot, "Simultaneous localization and map building using natural features and absolute information," *Robotics and Autonomous Systems*, vol. 40, no. 2-3, pp. 79–90, 2002.

[7] R. A. Chisholm, J. Q. Cui, S. K. Y. Lum, and B. M. Chen, "UAV LiDAR for below-canopy forest surveys," *Journal of Unmanned Vehicle Systems*, vol. 01, no. 01, pp. 67–68, 2013.

[8] J. W. Langelaan, "State estimation for autonomous flight in cluttered environments," Ph.D. dissertation, Stanford University, March 2006.

[9] S. Ross, N. Melik-Barkhudarov, K. Shankar, A. Wendel, D. Dey, J. Bagnell, and M. Hebert, "Learning monocular reactive UAV control in cluttered natural environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, pp. 1765–1772.

[10] S. Thrun and M. Montemerlo, "The GraphSLAM algorithm with applications to large-scale mapping of urban structures," *International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, May-June 2006.

[11] K. Liu, B. M. Chen, and Z. Lin, "On the problem of robust and perfect tracking for linear systems with external disturbances," *International Journal of Control*, vol. 74, no. 2, pp. 158–174, Janurary 2001.

[12] B. Wang, B. M. Chen, and T. H. Lee, "An RPT approach to time-critical path following of an unmanned helicopter," in *8th Asian Control Conference (ASCC)*, May 2011, pp. 211–216.

[13] J. Q. Cui, F. Wang, X. Dong, K. Ang Zong Yao, B. M. Chen, and T. H. Lee, "Landmark extraction and state estimation for UAV operation in forest," in *32nd Chinese Control Conference (CCC)*, Xi'an, China, July 2013, pp. 5210–5215.

[14] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 249–275, 1997.

[15] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g^2o: A general framework for graph optimization," in *Proceddings of the IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 3607–3613.