A Mono-Camera and Scanning Laser Range Finder Based UAV Indoor Navigation System

Fei Wang, Jinqiang Cui, Swee King Phang, Ben M. Chen, Tong H. Lee

***Abstract*— This paper presents a comprehensive control and navigation scheme for an indoor UAV system. In addition to the inertial measurement unit commonly used onboard of most UAVs, the testbed quadrotor platform is also equipped with a mono-camera looking downwards and a laser range finder capable of scanning a level plane. With this setup, the UAV is able to estimate its own velocity and position robustly, while flying along the internal walls of a room without collision. The whole system does not require any remote sensory information or off-line computational power. All algorithms are self-sustained and running onboard in real time. Complete flight tests have been carried out to verify the solution.**

I. INTRODUCTION

The development of advanced indoor navigation systems for miniature UAVs has aroused worldwide interest recently because of its great potential in military and civil applications. If such a system can be pragmatically implemented, it can be used for exploration and mapping, search and rescue, and other strategic missions. However, this system has to be robust enough to face challenges caused by complicated indoor environments, such as scattered obstacles and denied reception of GPS signals, and constraints of UAV platforms, for example, unconventional aerodynamic / mechanical design and limited payload. In addition, most work on the topic of indoor navigation conducted in the last few years focuses on 2D environments and is mainly for ground robots. The development of 3D navigation systems for UAVs is still not mature.

We note that the topics of UAV autonomous control [1], [2], [3], GPS-denied navigation [4], [5], [6], and path planning [7], [8], [9] are separately addressed by researchers in areas including robotics, mechatronics, computer vision, computer science, and etc. Each of these topics covers a breadth of literature. In view of these existing theoretical works, this paper, on the other hand, focuses more on the overall system design, integration and implementation. Moreover, a lot of the existing algorithms in literature have not been realized onboard of the UAV in real time. In most cases, only simulations, off-line processing, or part of the whole system have been tested. This largely limits the possible real-life applications of the indoor UAVs, especially when the control or path planning algorithms highly relies on some remote and immobile sensors or computers. As such, this paper selects and customizes the existing algorithms and methods, which aims to realize a robust indoor UAV system with real-time processing capability solely onboard. In addition, there are also a few new ideas about system design, control law implementation, UAV state estimation, and path planning which can distinguish this paper from other complete indoor navigation solutions, such as [10], [11].

F. Wang, J. Cui and S. K. Phang are with the NUS Graduate School for Integrative Sciences & Engineering, National University of Singapore (NUS), Singapore. E-mail: {g0901899,a0033585,a0068416}@nus.edu.sg

B. M. Chen and T. H. Lee are with the Department of Electrical & Computer Engineering, NUS, Singapore. E-mail: {bmchen,eleleeth}@nus.edu.sg

Fig. 1. The custom-made quadrotor platform

The content of this paper is organized as follows: Section II gives an overall structure of the proposed indoor navigation strategy, followed by Section III, IV and V respectively expand the UAV modeling and control, state estimation, and path plan algorithms. Section VI will provide real-life implementation results to verify the overall system. Lastly, concluding remarks will be made in Section VII.

II. PLATFORM AND OVERALL SYSTEM STRUCTURE

Being mechanically simple and robust, quadrotor helicopters have been widely used these days as UAV platforms for research purposes. There are ready products, such as AscTec's Pelican, Parrot's ARDrone, and MikroKopter's Quadro, which are all capable to do stable flight semi-autonomously. However, the quadrotor platform used for this work (see Fig. 1) is completely self-customized with special considerations about weight reduction and power efficiency. Its frame is made of carbon fiber plates and rods with a durable Acrylonitrile Butadiene Styrene (ABS) landing gear. Its dimensions are 35 cm in height and 86 cm in diagonal width (tip-to-tip). This custom-made quadrotor has a maximum take-off weight of 2.9 kg (conservatively tested) and can fly at 8 m/s for about 10 to 15 mins, depending on sensor configuration and environmental factors. As the bare platform only weighs about 1.3 kg, the remaining 1.6

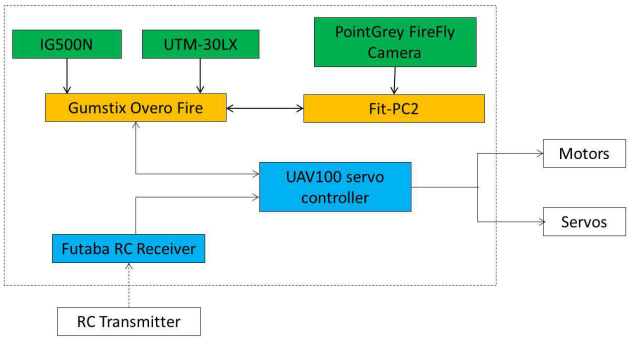


Fig. 2. Onboard avionics configuration of the quadrotor platform

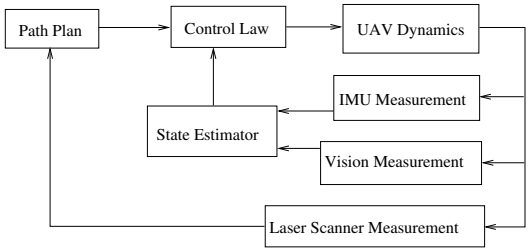


Fig. 3. Overall structure of the indoor navigation system

kg could be used as payload, including onboard avionics and battery. Currently a 4-cell 4300 mAh lithium polymer (LiPo) battery is used. The platform is also fully customizable in terms of sensor arrangement and is scalable such that additional computational boards could be mounted with a stack-based design.

For the onboard avionics, a complete attitude and heading reference system (AHRS), IG-500N from SBG Systems, is used to measure the UAV linear accelerations and also estimate Euler angles. There is also a built-in barometer which can provide air pressure measurement for height estimation. URG-30LX scanning laser range finder from Hokuyo is used to scan the surrounding objects in a frontal 270° and 30 m's range. The vision sensor used on this platform is a PointGrey FireFly camera. Two separate processors are used; one is the Gumstix Overo Fire, for control and navigation purpose, while the other one, Fit-PC2, is for vision processing. Fig. 2 shows the full onboard configurations and how the signals flow among all the components.

Zooming out, Fig. 3 shows the top-level structure of the proposed indoor navigation solution. As the UAV moves, the IMU sensor and vision sensor's measurements will be fed to the 'State Estimator' block to obtain an estimation of all state variables, and then acquired by the 'Control Law' block for feedback control. The other input to the 'Control Law' block is the UAV trajectory reference generated by the 'Path Plan' block. For path planning, the main sensor used is the scanning laser range finder. For the following sections, the 'Control Law', 'State Estimator' and 'Path Plan' blocks will be expanded and detailed respectively.

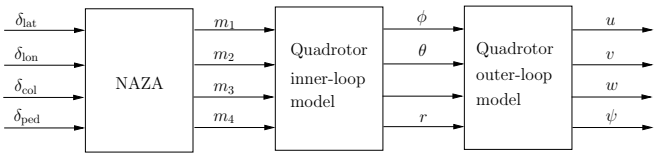


Fig. 4. Overview of quadrotor model structure

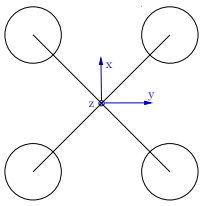


Fig. 5. Quadrotor body frame definition

III. Modeling and Control

The model structure of the quadrotor platform being used is illustrated in Fig. 4. The control inputs $(\delta_{\text{lat}}, \delta_{\text{lon}}, \delta_{\text{col}}, \delta_{\text{ped}})$ are fed into the onboard off-the-shelf NAZA controller. NAZA is an all-in-one stability controller specially designed for multi-rotor multi-axis flying platforms. With a standard quadrotor frame construction, the default control gains built in NAZA can already guarantee a good attitude stability of the quadrotor. NAZA controller outputs PWM signals to drive the four motors to generate the thrust forces, which not only lift the platform but also maintain its 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 respectively. There exists a dynamics model which maps the roll and pitch angles to the lateral and longitudinal velocities (u, v). It is mainly caused by rigid body kinematics with a bit of air resistance damping effect.

The platform operates in the so-called 'X' mode as shown in Fig. 5, where there are two frontal rotors and two rear rotors. The UAV body frame is defined as x pointing forward, y pointing rightward, and z pointing downwards, following the right-hand rule. Since the structure configuration of the platform and the design of the onboard system are highly symmetric, it is reasonable to assume that the longitudinal and lateral dynamics of this platform are exactly the same. We also assume that the model is completely decoupled among all four channels. That means the dynamic model of each channel can be identified independently. The overall system dynamics can be obtained by merging the four subsystem dynamics diagonally.

The model identification process is done with a software toolbox called CIFER. It is developed by the NASA Ames Research Center for military-based rotorcraft systems. It first converts the collected input-output data to frequency-domain responses. Then the frequency domain data are fed into NAVFIT, a low-order transfer function fitting tool.

Due to the symmetric structure of the quadrotor platform, the roll and pitch dynamics share the same model structure as well as parameters. When the platform is perturbed in the

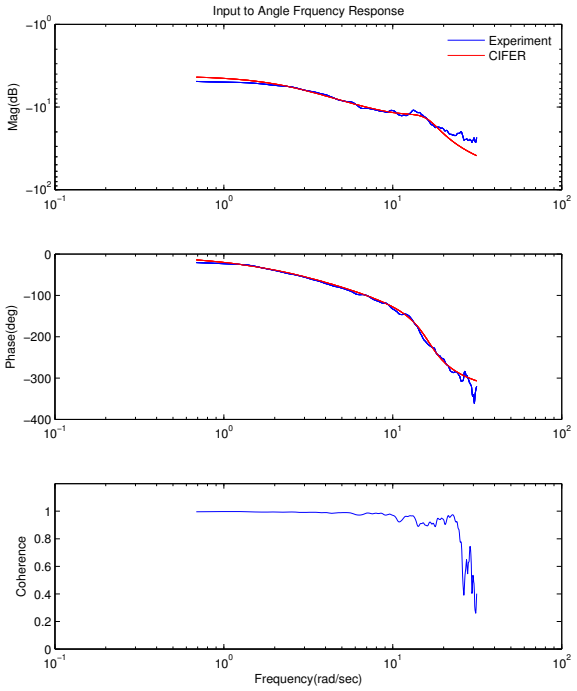


Fig. 6. Input-angle frequency response matching

roll (or pitch) direction, the onboard avionics system logs the roll (or pitch) angle and linear velocity's responses with the corresponding synchronized control input, δ_{lat} (or δ_{lon}). The ultimate goal is to identify the model from the control input δ_{lon} (or δ_{lat}) to the linear velocity u (or v). However, we can divide this task into two sub-tasks, i.e., identify the model from control input to attitude angle and identify the model from angle to velocity. Then the two models can be cascaded together to produce the overall longitudinal (or lateral) model.

Using NAVFIT in CIFER, the dynamics from the longitudinal (or lateral) control input to the pitch (or roll) angle can be well fitted by the following 4th order transfer function:

$$H_1 = \frac{9688}{s^4 + 27.68\,s^3 + 485.9\,s^2 + 5691\,s + 15750}. \quad (1)$$

The frequency response comparison between the identified model and the real data is shown in Fig. 6. The third sub-plot in the figure shows the coherence value of the frequency domain matching. At frequencies below 20 rad/s, the coherence value remains above 0.8, indicating the system can be well characterized by a linear process.

Similarly, the transfer function from roll (or pitch) angle to the lateral (or longitudinal) velocity can be identified as a first order transfer function shown below:

$$H_2 = \frac{8.661}{s + 0.09508}. \quad (2)$$

As a result, the cascaded dynamics from the control input to the UAV's linear velocity is a 5th-order system whose transfer function is the product of H_1 and H_2. It is observed that H_1 has its poles at $-4.0977 + 15.8272i$, $-4.0977 - 15.8272i$, -15.7414, -3.7433, while H_2 has its pole at -0.0958. Obviously, the pole of H_2 is much closer to the imaginary axis than any poles from H_1. Hence, the dynamics of the cascaded system from input to velocity is dominated by the pole of H_2. To simplify the control law design, we remove the poles from H_1 and keep the overall steady state gain the same. As a result, the simplified model from the control input to the linear velocity is

$$H_3 = \frac{5.3275}{s + 0.09508}. \quad (3)$$

Note that this simplified model will only be used for control law design purpose. The full 5th-order complete model will still be used for high-confidence simulation or control law verification.

For the yaw channel, since the most inner-loop dynamics controlled by NAZA is extraordinary fast, we can safely assume that the control input directly corresponds to the yaw angular rate with a proportional gain. Thus, the transfer function from yaw control input to the yaw angle is an integration of a constant, shown as below:

$$H_4 = \frac{3.372}{s}. \quad (4)$$

The transfer function from the collective input δ_{col} to the z-axis velocity w is identified to be

$$H_5 = \frac{-13.35}{s + 2.32}. \quad (5)$$

The negative coefficient on the numerator of H_5 is due to opposite definition of the positive control direction and the positive z-axis velocity direction.

The above sub-system models are identified with the presence of the most inner-loop NAZA controller. The next step is to design a model based outer-loop control law to track a reference 3D position and heading. Again, since all channels' dynamics are assumed to be decoupled, the outer-loop control laws can be designed separately. The control structure for each channel can be found in Fig. 7 – 9 (the longitudinal and the lateral channels share the same model and control law). Except for the conventional state feedback and reference feed-forward components, one special component of this control structure is that the derivative of the reference is also used. In this way, the UAV responds faster to reference changes and the time factor is emphasized. This is obviously an advantage for flying vehicle's mission of trajectory tracking, but the position reference needs to be generated smooth enough for pragmatic implementation. By tuning the controlled sub-system's bandwidth and damping ratio, the final gains are shown in Table I. The control performance will be shown in section VI.

IV. State Estimation with Sensor Fusion

While angle and angular rate measurements required by the inner-loop control law can be directly provided by the onboard IMU, the outer loop's position and velocity measurements need to be estimated indirectly. Since dead reckoning using IMU acceleration will face severe drifting

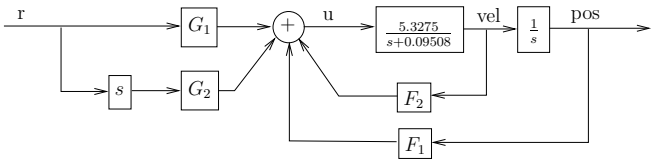


Fig. 7. x, y channel control structure

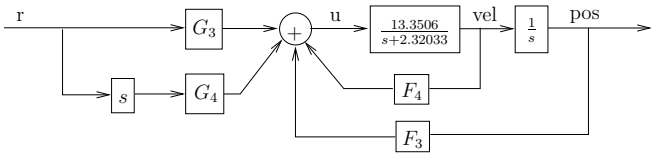


Fig. 8. z channel control structure

problems, information from other sensors must be fused in to cure the divergence and at the same time, to reduce noises. The solution goes to the onboard camera and barometer. Moreover, a Kalman Filter is formulated to fuse the information from all sensors and provide a complete and smooth UAV state estimation for control purposes.

A. Velocity Estimation by Vision Optical Flow

For many times, optical flow based vision algorithms have been proposed by researchers to solve UAV GPS-denied navigation problems. However, the camera mounting direction is never unified. Some researchers mount the UAV onboard camera pointing forward to do obstacle avoidance, while some other researchers mount the camera pointing sideward to navigate in indoor corridors or outdoor canyons. For our case, since there is also information from the onboard laser scanner, the frontal and sideward obstacles are already detectable. It is preferable to mount the camera pointing downward instead, so that the UAV body axis x, y direction velocity can be better estimated (optical flow estimates velocity more accurately in the direction perpendicular to the camera axis). This configuration also takes great advantage from the fact that the floor of an indoor environment is usually flat and vision algorithms involving feature points on the same plane can be largely simplified, thus easier to be implemented onboard in real time.

In the robotics community, when a robot moves in an indoor environment, simultaneous localization and mapping

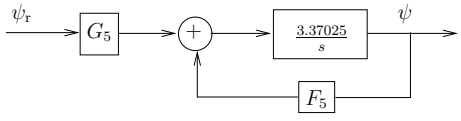


Fig. 9. Yaw channel control structure

TABLE I

CONTROL GAINS USED ON THE QUADROTOR UAV

F_1	F_2	F_3	F_4	F_5
-0.3059	-0.4614	-0.0509	0.0749	-0.4943
G_1	G_2	G_3	G_4	G_5
0.3059	0.4614	0.0509	-0.0749	0.4943

(SLAM) aims to build the map of the environment and localize the robot by analyzing information from various sensors. In the computer vision society, this is largely known as the problem of structure from motion (SFM), in which the camera translation and rotation can be calculated by given consecutive views of a scene. There are two fundamental sub-problems from SFM:

1) Correspondence (2D motion of the image): Which elements in this frame correspond to which elements in the previous frame?
2) Reconstruction (3D motion of the camera): Given a number of correspondences, and possibly the knowledge of camera's intrinsic parameters, how to recover the 3D motion and structure of the observed world?

The first problem is commonly known as the optical flow problem. It requires the knowledge of spatial and temporal image intensity derivatives. Let the grey-level intensity of a pixel on the image be $E(x,y,t)$. It is a continuous and differentiable function of space and time. Suppose the brightness pattern is locally displaced by a distance dx, dy over time period dt, the intensity of the displaced pixel should be the same as the original pixel:

$$E(x,y,t) = E(x+dx, y+dy, t+dt). \tag{6}$$

In other words, the total derivative of E w.r.t. time is zero.

$$\frac{\delta E}{\delta x}\frac{dx}{dt} + \frac{\delta E}{\delta y}\frac{dy}{dt} + \frac{\delta E}{\delta t} = 0. \tag{7}$$

If we denote

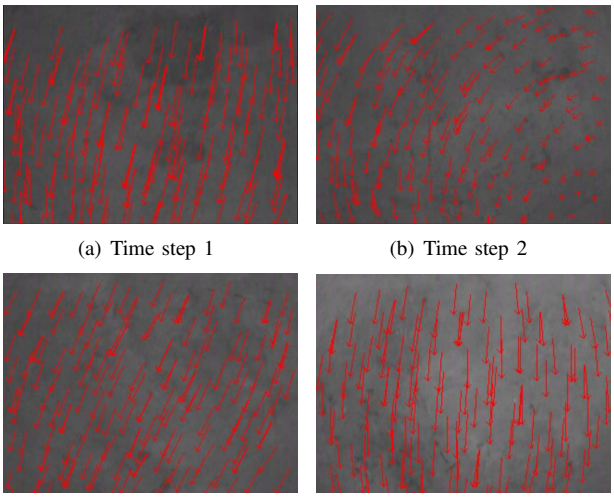
$$E_x = \frac{\delta E}{\delta x},\ E_y = \frac{\delta E}{\delta y},\ E_t = \frac{\delta E}{\delta t},\ v_x = \frac{dx}{dt},\ v_y = \frac{dy}{dt},$$

then the equation can be rewritten as:

$$E_x v_x + E_y v_y + E_t = 0. \tag{8}$$

The above is known as the Brightness Constancy Equation (BCE). In BCE, E_x, E_y and E_t are measurable, while v_x and v_y are the unknown 2D flows. One individual BCE is not enough to estimate the 2D flows. Solutions to this normally involve considering a small window of adjacent pixels. Among all these solutions, the Lucas & Kanade algorithm is the most classical and elegant one. It assumes that the motion field at a given time is constant over a block of pixels. For that particular block of n number of pixels, there are n BCEs. They form a linear over-determined equation set. Least square estimation can be used to obtain a reliable v_x, v_y over that region.

The above optical flow calculation can be carried out at any pixel locations on the image. However, some locations are more stable and reliable for the calculation, while others are not. First of all, if an image region is almost homogeneous, then E_x, E_y, E_t are near zero. This leads to numerical problems and will even result in an under-determined equation set. Second, even if the region is not homogeneous, there is the well-known aperture problem. 2D motion flow is ambiguous if only one-directional intensity change can be observed. In general, corner-like feature points



(a) Time step 1 (b) Time step 2

(c) Time step 3 (d) Time step 4

Fig. 10. The 2D optical flow implementation result

are more suitable to be set as the center of optical flow calculation. Therefore, high quality feature detection usually needs to be done before optical flow calculations are carried out at these feature locations.

To implement this 2D optical flow algorithm, the well-known OpenCV library from Intel can be used. OpenCV library also provides the 'goodFeaturesToTrack' function which is able to find predefined number of feature points on an image and these feature points are selected particularly suitable for the later 'calcOpticalFlowPyrLK' function, which calculates the optical flow for a sparse feature set using the iterative Lucas-Kanade method with pyramids. The implementation result can be found in Fig. 10(a) - 10(d).

The next step is to estimate the camera 3D motion from the 2D optical flow mentioned above. One of the most elegant methods relies on an important concept called 'homography'. 'Homography' is a well-known term in the computer vision society in describing the linear position relationship between feature points on two different (image) planes. Suppose a downward-looking camera on the UAV takes image of the ground during flight. Given two consecutive images taken at time t_1 and t_2, the corresponding visual features' pixel positions in the 1st and 2nd images are related by a 3 by 3 matrix $\mathbf{H}$, provided that the ground scene is within a level plane [12]. This $\mathbf{H}$ is called the 'homography' matrix. The level plane assumption is quite reasonable for any indoor environment as the floor is usually manmade, thus flat. $\mathbf{H}$ carries useful information about the UAV motion from t_1 to t_2. If $\mathbf{R}$ and $\mathbf{T}$ are the inter-frame rotation and translation of the UAV from t_1 to t_2, $\mathbf{N}$ is the unit-length normal vector of the ground plane resolved in the camera frame at t_1, and d is the UAV altitude with respect to the ground plane, then the homography matrix $\mathbf{H}$ can be expresses as [13]:

$$\mathbf{H} = \mathbf{R} + \frac{1}{d}\mathbf{T}\mathbf{N}^{\mathrm{T}}. \tag{9}$$

The decomposition of $\mathbf{H}$ into $\mathbf{R}$, $\mathbf{T}$ and $\mathbf{N}$ is doable but quite complicated and it usually results in large numerical errors in practice. Fortunately, $\mathbf{R}$ and $\mathbf{N}$ are known in our case since the IMU will give Euler angle information at every moment. If we have UAV attitude angles ϕ_1, θ_1, ψ_1 at t_1 and ϕ_2, θ_2, ψ_2 at t_2, then

$$\mathbf{N} = \begin{bmatrix} -\sin\theta_1 \\ \sin\phi_1\cos\theta_1 \\ \cos\phi_1\cos\theta_1 \end{bmatrix}, \tag{10}$$

and

$$\mathbf{R} = \mathbf{R}_{\mathrm{b/n}}(t_2)\mathbf{R}_{\mathrm{n/b}}(t_1), \tag{11}$$

where $\mathbf{R}_{\mathrm{b/n}}$ is the rotational matrix to convert 3D points from the inertia frame to the UAV body frame and $\mathbf{R}_{\mathrm{n/b}}$ is vice versa and they are transpose of each other. Hence, with $\mathbf{H}$, $\mathbf{R}$, d and $\mathbf{N}$ known, the translational motion, $\mathbf{T}$ can be calculated as:

$$\mathbf{T} = d(\mathbf{H} - \mathbf{R})\mathbf{N}. \tag{12}$$

B. Height Measurement from Barometer

As mentioned previously, indoor UAV height measurement can be from various sensors. Using range sensor is one of most direct ways to measure the distance from the UAV body to the ground. However, to obtain the UAV absolute height with respect to its initial position, this range sensor approach has to assume that the ground is without any altitude variations. Moreover, if the UAV is required to go through windows, this measurement will be inaccurate when the UAV is directly above the windowsill. In view of this, a more robust choice would be the barometer, as its measurement is independent of the environment's physical shape. If a reference pressure (usually the pressure right before taking off), P_0, is selected, then the z direction position of the UAV at any moment after can be calculated by the following equation:

$$z_g = -44307\left[1 - \left(\frac{P}{P_0}\right)^{0.1902}\right], \tag{13}$$

with P being the present pressure sensor output in Pascal.

C. Data Fusion by Kalman Filter

The Kalman filter framework describes a discrete-time linear system as follows,

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}(\mathbf{u}(k) + \mathbf{w}(k)), \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{v}(k), \end{aligned} \tag{14}$$

where $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{y}$ are the state, input and measurement vectors respectively. $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are system matrices with appropriate dimensions. $\mathbf{w}$ and $\mathbf{v}$ are input and measurement noises, which are assumed to be Gaussian with zero means. The main objective of Kalman filter is to estimate $\hat{\mathbf{x}}(k|k)$ at the time step k with the measurement $\mathbf{y}(k)$, input $\mathbf{u}(k-1)$ and the previously estimated state $\hat{\mathbf{x}}(k|k-1)$. If system 14 is observable, then the statistically optimal estimator is given as follows,

Time Update:

$$\begin{aligned}\hat{\mathbf{x}}(k|k-1) &= \mathbf{A}\hat{\mathbf{x}}(k-1)+\mathbf{B}\mathbf{u}(k-1),\\ \mathbf{P}(k|k-1) &= \mathbf{A}\mathbf{P}(k-1)\mathbf{A}^{\mathrm{T}}+\mathbf{B}\mathbf{Q}\mathbf{B}^{\mathrm{T}},\end{aligned}$$

Measurement Update:

$$\begin{aligned}\mathbf{H}(k) &= \mathbf{P}(k|k-1)\mathbf{C}^{\mathrm{T}}(\mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}^{\mathrm{T}}+\mathbf{R})^{-1},\\ \hat{\mathbf{x}}(k) &= \hat{\mathbf{x}}(k|k-1)+\mathbf{H}(k)(\mathbf{y}(k)-\mathbf{C}\hat{\mathbf{x}}(k|k-1),\\ \mathbf{P}(k) &= (\mathbf{I}-\mathbf{H}(k)\mathbf{C})\mathbf{P}(k|k-1),\end{aligned}$$

To apply Kalman filter in this indoor UAV state estimation problem, the motion model and the measurement model need to be first defined. For the motion model, the simple point mass kinematics model can be used, that is, in the NED frame, position can be integrated by velocity and velocity can be integrated by acceleration. The discrete representation of this motion model with a sampling frequency of 50 Hz is shown below:

$$\begin{aligned}\mathbf{x} &= \begin{pmatrix}x_g & y_g & z_g & u_g & v_g & w_g\end{pmatrix}^{\mathrm{T}},\\ \mathbf{u} &= \begin{pmatrix}ax_g & ay_g & az_g\end{pmatrix}^{\mathrm{T}},\\ \mathbf{y} &= \begin{pmatrix}z_g & u_g & v_g\end{pmatrix}^{\mathrm{T}},\\ \mathbf{A} &= \begin{bmatrix}1 & 0 & 0 & 0.02 & 0 & 0\\ 0 & 1 & 0 & 0 & 0.02 & 0\\ 0 & 0 & 1 & 0 & 0 & 0.02\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix},\\ \mathbf{B} &= \begin{bmatrix}0.0002 & 0 & 0\\ 0 & 0.0002 & 0\\ 0 & 0 & 0.0002\\ 0.02 & 0 & 0\\ 0 & 0.02 & 0\\ 0 & 0 & 0.02\end{bmatrix},\\ \mathbf{C} &= \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix}.\end{aligned}$$

While the above system matrices have been numerically defined, the input noise and measurement noise matrices $\mathbf{Q}$ and $\mathbf{R}$, being both diagonal, can be selected by logging real flight test data; the non-zero elements in $\mathbf{Q}$ represent the acceleration measurement noises and the non-zero elements in $\mathbf{R}$ represent the noise of z direction position measurement from barometer and velocity measurements from vision optical flow.

V. The Wall Following Path Planning Strategy

Classical UAV path planning algorithms require the UAV to know the global map, i.e., its own position, the target's location, and all obstacles' positions presented in the global frame. However, the environment is usually unknown in practice and the exploration mission may not involve a well defined target. In these cases, two approaches are usually used. One is to build the unknown map through SLAM, so that all the conventional path planning algorithms can be applied afterwards. However, this kind of methods suffer from the problem of huge computational burden, and it is so far difficult to be implemented onboard and executed in real time on a miniature indoor UAV. The second approach is to use UAV local information only and dynamically calculate the tracking reference of the UAV, that means the reference generation is contemporary and only one step ahead. In this paper, the path planning strategy proposed follow the second approach, i.e. no global map or self-location information is needed. It is a fairly universal strategy for any indoor enclosed environment with regular walls. First of all, this algorithm utilizes the concept of artificial potential field. All measurements from the laser scanner are treated as obstacles and they exert a repulsive force on the UAV body. Besides, to let the UAV keeps moving forward, there is a constant attractive force coming from a virtual target two meters ahead from the UAV body. If the UAV is ordered to follow the wall on its left, then this virtual target is placed at the left-front of the UAV heading. If the the UAV is order to follow the right side wall, then the target is placed at the right-front of the UAV heading. The followings are the formulation:

$$\mathbf{F} = \mathbf{F}_{\text{att}} - \mathbf{F}_{\text{rep}}, \tag{15}$$

where

$$\mathbf{F}_{\text{att}} = e^{-\frac{|\mathbf{T}|^2}{2\sigma_1^2}}\mathbf{T}, \tag{16}$$

and

$$\mathbf{F}_{\text{rep}} = \sum_{k=n_1}^{n_2} e^{-\frac{|\mathbf{W}(k)|^2}{2\sigma_2^2}}\frac{\mathbf{W}(k)}{\sigma_2 K} \tag{17}$$

Above, $\mathbf{F}$ is the resultant force of the artificial potential field, $\mathbf{F}_{\text{att}}$ is the attractive force coming from the virtual target and $\mathbf{F}_{\text{rep}}$ is the repulsive force generated by the wall obstacles. $\mathbf{T}$ and $\mathbf{W}(k)$ are the vectors pointing towards the virtual target and the points measured by the laser scanner. σ_1 and σ_2 represent the stiffness of the Gaussian-like potential fields which can be tuned for different indoor situations. In Eqn. 17, all laser scanner measurements indexed from n_1 to n_2 will be examined one by one. Invalid measurements (in case of out of range) will be dropped, and K is the total number of valid measurements being used.

Although we call $\mathbf{F}$ a 'force', it can be interpreted as other physical entities in practice. In this implementation, we let the UAV 2D velocity reference be proportional to $\mathbf{F}$, and by integration, it also forms the 2D position reference. For the z direction, the UAV is ordered to maintain a predefined height with respect to the flat indoor floor. In addition, to determine the UAV heading reference, another algorithm is running at the same time to determine the UAV yaw angle reference at every scan moment, and it runs as follows:

1) If the UAV is to fly along the wall on its left, then omit the scanned points on the right side. If the UAV is to fly along the right side wall, then omit the scanned points on the left side.

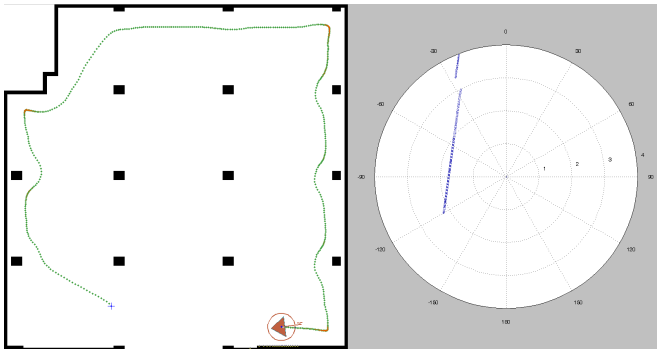


Fig. 11. Wall following strategy simulation result

2) For all remaining scanned points, calculate the best straight line fit using the least square optimization method.
3) The UAV heading reference deviates from the UAV current heading direction by a fraction ($a\%$) of the difference between the fitted line gradient and the current UAV heading (a can be tuned for different indoor environments).

By combining the potential field algorithm with the line fitting algorithm, the UAV outer-loop references can be completely set up. Before any real implementations or actual flight tests, a Matlab program is written to simulate the capability of the aforementioned wall-following algorithm. A virtual map and a virtual laser scanner sensor, both to real life scale, are established for the purpose of navigation simulation. In Fig. 11, the sub-figure on the left is the global view of the indoor environment, which includes the walls, pillars and the UAV position and heading information. The sub-figure on the right shows the laser scanner measurements in the UAV body frame. The control reference to the UAV are purely calculated from the information in the right sub-figure, while the contents in the left sub-figure are only for display purpose and not available to the navigation algorithm. The result of one run of the simulation is shown. This time, the UAV is ordered to follow the wall on its left. The configurations of this simulation is: ($\sigma_1 = 2$, $\sigma_2 = 1$, $a = 5$). The green dotted line is the flight path of the UAV.

VI. The Implementation Results

Real-life flight tests have been carried out to verify the performance of the UAV position and velocity state estimator as well as the control law. An autonomous hover test via the estimated position and velocity feedback is carried out. For the whole time duration, from $t = 210$ to $t = 250$, the UAV is at autonomous position hold mode, while one intentional push is given to the UAV as disturbances to see how it recovers (see Fig. 16(a) - 16(d)). The outer-loop position and velocity tracking performance is illustrated in Fig. VI. Position (x_g, y_g, z_g) and heading angle (ψ) signals are all controlled within small errors with respect to their respective references, which means control law is appropriate. Besides, the on-site position hold performance according to several observers at different viewing angles

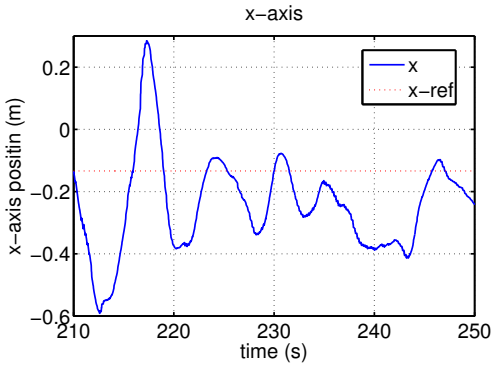


Fig. 12. Indoor autonomous hovering result – x

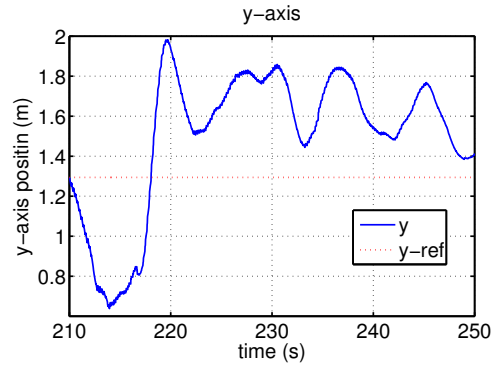


Fig. 13. Indoor autonomous hovering result – y

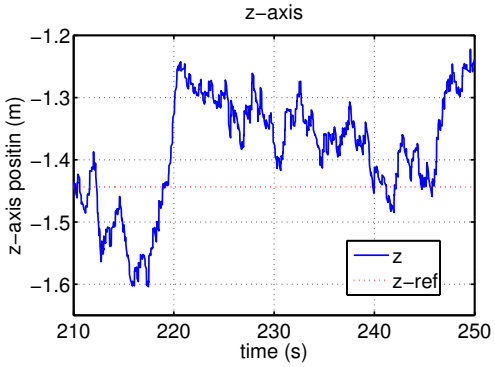


Fig. 14. Indoor autonomous hovering result – z

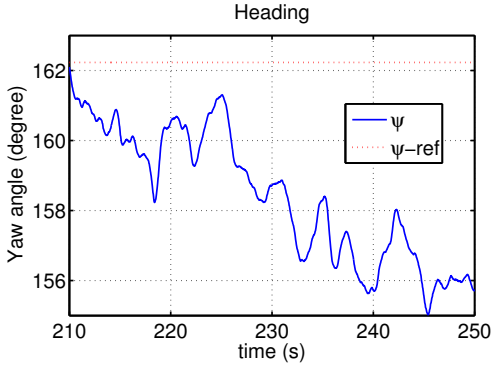


Fig. 15. Indoor autonomous hovering result – ψ

(a) Moment 1 (b) Moment 2

(c) Moment 3 (d) Moment 4

Fig. 16. Indoor autonomous hover with disturbances

are all superior. That means the state estimator is working properly, and surprisingly, the position estimation is with very little drift. In addition, the wall following algorithm mentioned in section V has also been implemented onboard. Verification flight tests have been carried out. Our quadrotor platform, equipped with the 30 m's Hokuyo laser scanner, has performed an autonomous wall following flight in an indoor hall successfully. Fig. 17 sequentially show the 8 instances of the flight, with the left sub-figure showing the real-life flying condition and the right sub-figure showing the body-frame laser scanner data at that instant.

VII. CONCLUSIONS

In conclusion, this paper has proposed a complete control and navigation scheme for an indoor quadrotor UAV system by selecting, customizing, and combining some suitable existing algorithms. Innovations have been focussing on improving algorithm efficiency and system integration. Real-life flight tests have been carried out to verify the applicability and performance of the overall system. The greatest advantage of this scheme is its minimal requirement on the computational power. Hence, pure onboard processing can be realized. The UAV, after being issued the main navigation command, does not need to maintain any wireless link to the Ground Control Station. Furthermore, although the testbed platform is a quadrotor UAV, the proposed scheme is also applicable to other miniature UAV platforms as long as they have IMU, mono-camera and scanning laser range finder mounted onboard.

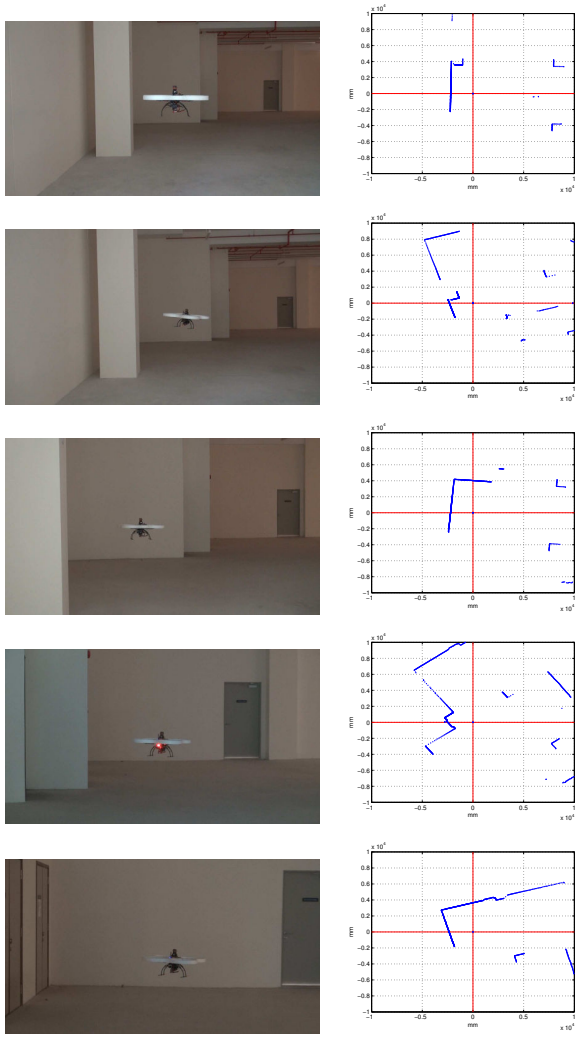


Fig. 17. Instances of the wall following flight test

REFERENCES

[1] P. Pounds, R. Mahony, and P. Corke, "Modelling and Control of a Quad-Rotor Robot," *Proceedings of the Australasian Conference on Robotics and Automation*, 2006.

[2] J. F. Roberts, T. Stirling, J. C. Zufferey, and D. Floreano, "Quadrotor Using Minimal Sensing For Autonomous Indoor Flight," *European Micro Air Vehicle Conference and Flight Competition*, 2007.

[3] A. Tayebi and S. McGilvray, "Attitude stabilization of a VTOL quadrotor aircraft," *IEEE Transactions on Control Systems Technology*, vol. 14(3), pp. 562-571, 2006.

[4] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti, and S. Longhi, "Vision-based Autonomous Navigation and Landing of an Unmanned Aerial Vehicle Using Natural Landmarks," *17th Mediterranean Conference on Control and Automation*, pp. 910-915, 2009.

[5] J. Kim and I. S. Kweon, "Vision-based Autonomous Navigation Based on Motion Estimation," *International Conference on Control, Automation and Systems*, pp. 1738-1743, 2008.

[6] Y. Liu and Q. Dai, "Vision Aided Unmanned Aerial Vehicle Autonomy: An Overview," *3rd International Congress on Image and Signal Processing*, pp. 417-421, 2010.

[7] S. Mittal and K. Deb, "Three-dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms," *IEEE Congress on Evolutionary Computation*, pp. 3195-3202, 2007.

[8] Y. Qu, Q. Pan, and J. Yan, "Flight Path Planning of UAV Based on Heuristically Search and Genetic Algorithms," *31st Annual Conference of IEEE on Industrial Electronics Society*, 2005.

[9] Y. Chen, Y. Zhao, and H. Wang, "Real Time Path Planning for UAV Based on Focused D," *2011 Fourth International Workshop on Advanced Computational Intelligence*, pp. 80-85, 2011.

[10] A. G. Bachrach, "Autonomous Flight in Unstructured and Unknown Indoor Environments," Master's thesis, MIT, Cambridge, MA, 2009.

[11] S. Shen, N. Michael, and V. Kumar, "Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV," *2011 IEEE International Conference on Robotics and Automation*, pp. 20-25, 2011.

[12] Y. Ma, S. Soatto, J. Kosecka and S. Sastry, *An Invitaion to 3D Vision*, Springer, New York, 2004.

[13] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, ISBN: 0521540518, 2nd ed., 2004.