Autonomous Navigation and Localization of a Quad-Rotor in an Indoor Environment

Young-Min Ahn*, Daniel J Block†, Ramavarapu S. Sreenivas‡

This paper describes an *inertial navigation* and *vanishing point* methodology that has been implemented using artificially placed landmarks and the sensors on-board a quad-rotor for autonomous navigation in a *Global Positioning System* denied, known, indoor environment. The drift in yaw is regulated using a downward facing camera and landmarks placed at strategic locations. The heading is determined by the vanishing point in the image sensed by the forward facing camera, which also helps to reduce the number of landmarks necessary for autonomous flight. The results presented in this paper suggest that data from on-board inertial measurement units, combined with the localization method described in this paper, are sufficient to achieve autonomous indoor navigation.

I. Introduction

Quad-rotors have become popular as unmanned aerial vehicle (UAV) platforms over the last decade. The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) has produced multiple quad-rotor designs that could navigate using waypoints with the help of a *Global Positioning System* (GPS) and an *inertial measurement unit* (IMU) (cf. [1, 2, 3]). Alternate, custom-made designs for quad-rotors and their flight control algorithms include those that are described in references [4, 5], the *X4-flyer* [6], the Hoverbot [7], and the meso-scale quad-rotor, the *Mesicopter* [8].

There have been several approaches to incorporate cameras into the localization and servoing tasks for UAV-control. For instance, a number of appropriately placed, ground-based cameras that track an UAV can be used to regulate its position (cf. [9, 10, 11]). References [12, 13] described the use of cameras for simultaneous localization and mapping of UAVs. Reference [14] proposes a two camera system that uses vision to stabilize and track a quad-rotor. One of these cameras is ground-based, while the other is on-board the quad-rotor. Reference [15] describes a method of tracking known objects on the ground using a downward facing camera on a UAV for purposes of navigation and control.

This paper describes experiments in navigation and localization of a quad-rotor within an indoor, GPS-denied environment with no ground-based cameras. The drift in yaw, which is commonplace with methods that rely purely

*Technical Staff, Sandia National Laboratories, 7011 East Ave, Livermore, CA 94550, yahn@sandia.gov

†Manager, College of Engineering Control Systems Laboratory, University of Illinois at Urbana-Champaign, 306 North Wright Street, Urbana IL 61801, d-block@illinois.edu

‡Associate Professor, CSL & Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, 104 S. Mathews Ave, Urbana, IL 61801,rsree@illinois.edu

on inertial navigation, is regulated using artificial landmarks and an on-board downward facing camera. The heading direction is determined by the vanishing point in the image generated by a forward facing camera. Additionally, this reduces the number of landmarks that are necessary for autonomous navigation. In a sense, these experiments can be interpreted as a combination of the concepts introduced in references [14] and [15] on the Parrot AR.Drone, which is described in the following subsection.

A. The Parrot AR.Drone

The Parrot AR.Drone quad-rotor shown in figure 1 has a 6 degrees of freedom IMU, a 2-axes gyro meter that measures roll rate and pitch rate, a 1-axis gyro meter that measures yaw rate, and a 3-axes accelerometer that senses the tilt, which also gives precise measurement of the roll and pitch attitude. However, as with all MEMS-based gyrometers, the bias offset drifts with time, and the integrated output fails to be accurate even for short time-periods. A proprietary filtering scheme compensates for the angular position error of roll and pitch by combining accelerometer and gyro measurements. However, the drift in the yaw angle cannot be compensated by this method. The vector-acceleration due to gravity is parallel to the yaw-axis of the reference coordinate frame described below, which makes it difficult to measure the absolute yaw attitude. A solution to this issue is addressed in section III.

The ultrasound sensor attached to the underside of the quad-rotor measures the altitude. There is a vertical camera (cf. figure 1c) that faces the ground with a 64° diagonal lens, and a horizontal camera (cf. figure 1d) with a 93° wide-angle diagonal lens.

a) Front View **b) Top View** **c) Bottom View: Vertical Camera & Ultrasound sensor** **d) Horizontal Camera**

Figure 1. The Parrot AR.Drone quadrotor (http://ardrone2.parrot.com).

The AR.Drone is controlled using a ground station PC via a WiFi ad hoc wireless connection, while the on-board controller stabilizes it internally. The ground station PC in our experiments was equipped with a dual 2.80 GHz processor and 3GB of RAM. A block diagram of the system along with the various components is shown in figure 2.

The rest of the paper is organized as follows. Section II presents a simplified, decoupled model for the quad-rotor for nominal indoor flight. This is followed by a validation of the system identification experiments followed by the description of a set of PI/PID controllers for the yaw, altitude, x-position, y-position, x-velocity and y-velocity of the quad-rotor. Section III describes the process of landmark detection that uses the vertical camera, which is subsequently

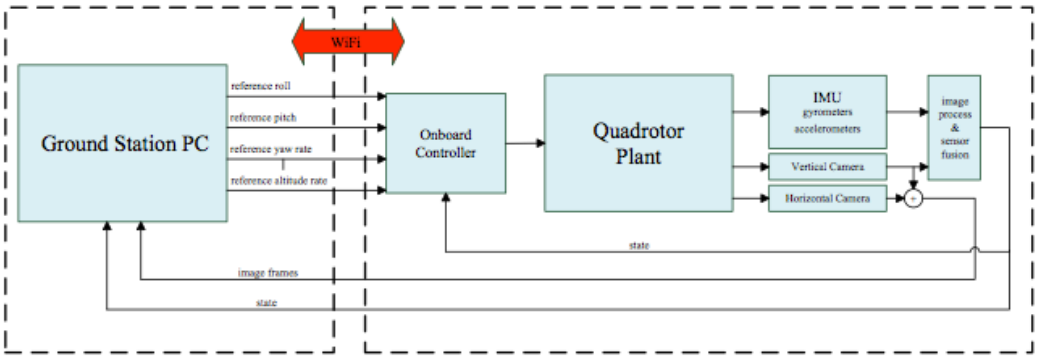


Figure 2. A block-diagram representation of the quad-rotor and ground-station. The wireless network between the quad-rotor and ground-station PC is established as an ad-hoc wireless network.

used for correcting the drift in yaw. After describing the camera calibration process, this section describes the process of navigation using vanishing points in the image captured by the horizontal camera. A summary of these experiments is presented in the concluding section.

II. The Quad-Rotor Model

An earth-fixed coordinate system where the origin is located at the initial takeoff point is defined as the *inertial frame* f_i. The origin of the local frame f_l is at the center of mass of the quad-rotor, and its axes are collinear to the inertial frame axes. The origin of *body frame* f_b is at the center of mass of the quad-rotor. The local frame and body frame satisfy the relation $(x_b\ y_b\ z_b)^T = R \times (x_l\ y_l\ z_l)^T$ (cf. figure 3), where R is the roll-pitch-yaw rotation matrix

$$R = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & Cos(\Phi) & -Sin(\Phi) \\ 0 & Sin(\Phi) & Cos(\Phi) \end{pmatrix}}_{R_{x,\Phi}} \times \underbrace{\begin{pmatrix} Cos(\Theta) & 0 & Sin(\Theta) \\ 0 & 1 & 0 \\ -Sin(\Theta) & 0 & Cos(\Theta) \end{pmatrix}}_{R_{y,\Theta}} \times \underbrace{\begin{pmatrix} Cos(\Psi) & -Sin(\Psi) & 0 \\ Sin(\Psi) & Cos(\Psi) & 0) \\ 0 & 0 & 1 \end{pmatrix}}_{R_{z,\Psi}}.$$

The quad-rotor is assumed to be a single rigid body with 6 DOF, and the forces and moments are assumed to come from the four rotors and gravity. Varying the thrust of the four rotors changes the quad-rotor's state. The pair of propellers $(\#1, \#2)$ and $(\#3, \#4)$ turn in opposite directions (cf. figure 3). If $F_i(t)$ and $\tau_i(t)$ $(i = 1, \ldots, 4)$ denote the force and torque at the i-th rotor, then the force acting on the quad-rotor in body frame is $F(t) = \sum_{i=1}^{4} F_i(t)$. For an indoor flight, the roll and pitch orientations will be negligible and the angular velocity and moment of inertia in the inertial- and body-frame will be identical. The rolling- and pitching-moments are assumed to be negligible, which

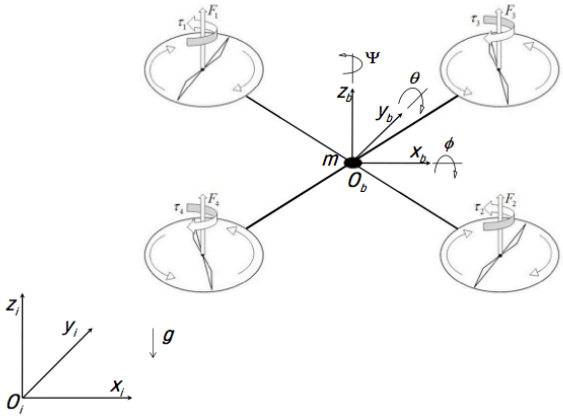


Figure 3. The coordinate system of the quad-rotor.

results in the following equations of motion for the quad-rotor (cf. [17])

$$\begin{aligned}
\frac{d^2}{dt^2}x_b(t) &= -\Theta(t)\frac{F(t)}{m} \\
\frac{d^2}{dt^2}y_b(t) &= -\Phi(t)\frac{F(t)}{m} \\
\frac{d^2}{dt^2}z_b(t) &= -g + \frac{F(t)}{m} \\
\frac{d^2}{dt^2}\Psi_b(t) &= \frac{1}{J_{zz}}((\tau_1(t)+\tau_3(t)) - (\tau_2(t)+\tau_4(t))),
\end{aligned}$$

where m is the mass of the quad-rotor, $\Phi(t)$ is the roll-angle, $\Theta(t)$ is the pitch-angle, $\Psi(t)$ is the yaw-angle, and J_{zz} is third entry in the (diagonal) angular moment of inertia matrix. Under this model, the translational state of the quad-rotor is only influenced by the thrust force, roll and pitch orientations during nominal indoor flight.

With reference to the block-diagram of figure 2, for a constant thrust $F(t) = F$, a step-change to alter the reference pitch (reference roll, respectively) will result in the quad-rotor's movement along the x-axis (y-axis, respectively) with a constant acceleration. Similarly, a step-change to the reference altitude rate, when $\Theta(t) \approx 0$ and $\Phi(t) \approx 0$, will only result in an appropriate change in the acceleration of the quad-rotor along the z-axis. The yaw rate is increased according to the above equations when a step-change to the reference yaw rate is commanded by the ground-station PC. These functionalities are built-in to the AR.Drone, and we now describe the system identification experiments that identify the transfer functions that arise from the four equations shown above.

Assuming the sensor dynamics are constant, and the AR.Drone's proprietary controller, following feedback linearization, involves a PID cntroller, the candidate transfer functions for the systems described by the four equations shown above would be of the form $T(s) = \frac{\alpha s^2+\beta s+\gamma}{s^3+\rho s^2+\eta s+\sigma}$, where the coefficients $\alpha, \beta, \gamma, \rho, \eta$ and σ have to be estimated.

A. System Identification and Validation

The decoupled linear sub-models are identified using the system identification toolbox in MATLAB (R2009a) by MathWorks. For each sub-model, multiple sets of input and output flight data are collected in a large indoor environment. The collected input-output data are processed using the *prediction error method* (PEM) (cf. [18]), This process is repeated for linear velocities of x, y and altitude to identify the transfer function models that describe these dynamics. Since there is a significant time-delay between the ground-station PC and the quad-rotor, it is necessary to add a time-delay component to the system identification experiments. The decoupled sub-model transfer functions of the system were found to be

$$T_\Psi = \frac{-29.2024s - 1.9742}{1.089s^3 + 16.74s^2 + s} e^{-0.051761s} \quad (1)$$

$$T_x = \frac{-39422s - 14.24}{1120s^3 + 9549s^2 + 3049s + 1} e^{-0.016415s} \quad (2)$$

$$T_y = \frac{-257082s - 4.3079}{7537s^3 + 46068s^2 + 33559s + 1} e^{-0.064365s} \quad (3)$$

$$T_z = \frac{22.6485s + 0.3951}{1.187s^3 + 27.3s^2 + s} e^{-0.325s} \quad (4)$$

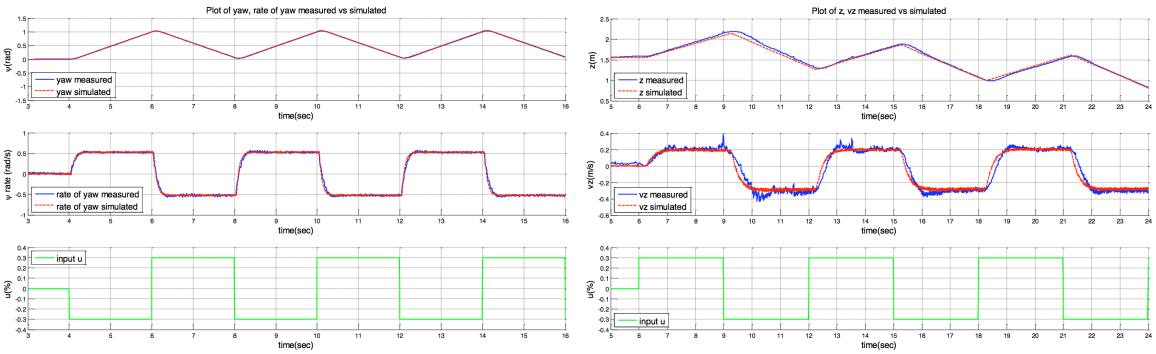
The simulation model of the quad-rotor was developed in MATLAB/Simulink, and the simulation data was compared to the recorded flight data to validate the accuracy of the simulation model. The validation plots for the yaw and yaw-rate, altitude and altitude-rate, linear x and x-velocity, and linear y and y-velocity plots are shown in figure 4. The altitude plots of figure 4d shows that gravity acts as a constant disturbance to the system.

To validate the claim that cross-coupling effects are negligible during nominal indoor flight, multiple inputs of square waves with nominal amplitudes are simultaneously sent to the quad-rotor, and the response is shown in figure 5. From these responses it is clear that the position and velocity responses are well predicted by the decoupled models identified earlier.

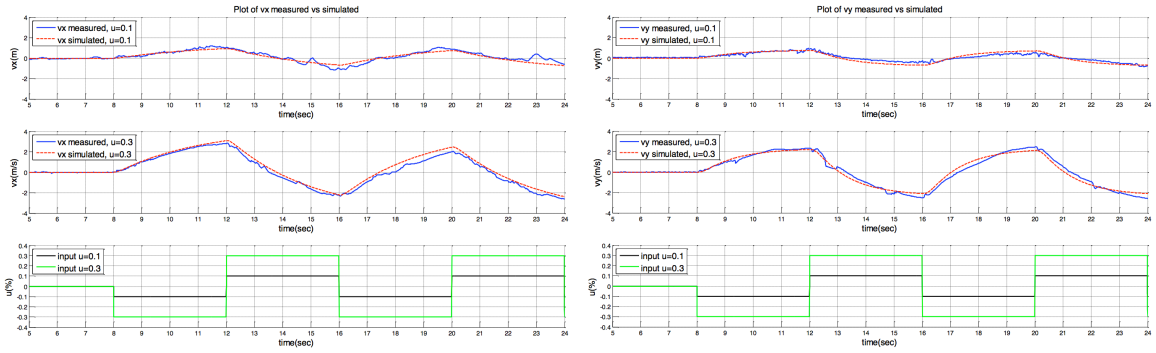
B. Quad-Rotor Controller Design

The quad-rotor's position and velocity are controlled using a PID controller in the outer-loop that uses the decoupled sub-system models derived and validated earlier. The PID gains were picked using the rltool in MATLAB such that the resulting system had (1) a (10%-90%) rise-time of no more than ≈ 1 second, (2) settling time (5%) of no more than 2 seconds, (3) overshoot of no more than $\approx 5\%$, and (4) the magnitude of the control effort is below the saturation limit. The flight controller PID gains are shown in table 1. Due to the highly noisy readings from the acceleration sensors, just a PI controller is implemented for velocity regulation.

Figure 6a (6b, respectively) shows a plot of the measured and simulated yaw (altitude, respectively) step response. Figures 7a and 7b show the measured and simulated response of the quad-rotor to step input for x-velocity and y-velocity, respectively. Although the rise time is within the predicted range, the overshoot is greater than anticipated.

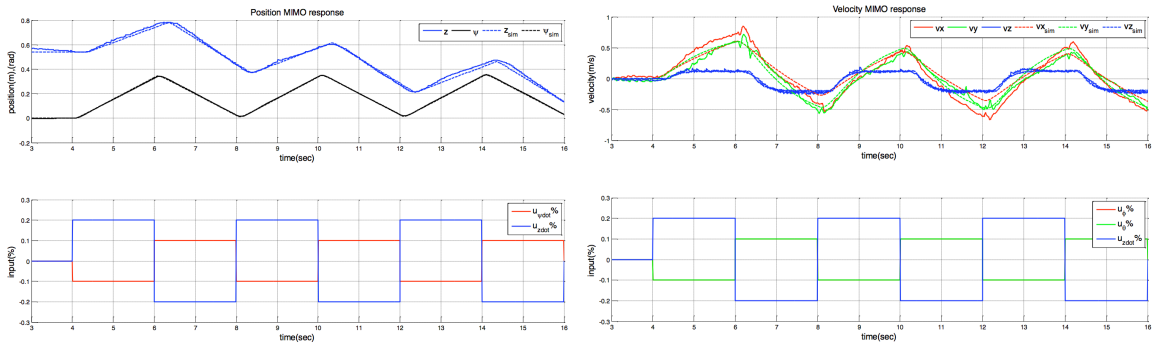


a) Yaw and Yaw-rate validation **b) Altitude and Altitude-rate validation**



c) Linear x and x-velocity validation **d) Linear y and y-velocity validation**

Figure 4. Validation of the simulation model.

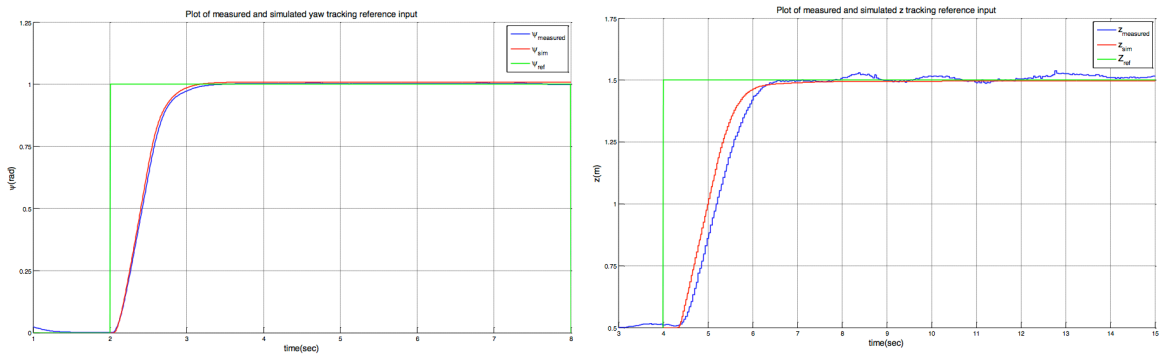


a) Cross-Coupling Validation 1 **b) Cross-Coupling Validation 2**

Figure 5. Comparison plot of multiple, measured, and simulated input/output response of the quad-rotor.

	Yaw	Altitude	x-velocity	y-velocity
K_p	3	1.5	0.4	0.25
K_I	0.03	0.025	0.18	0.2
K_d	0.3	0.1	N/A	N/A

Table 1. Flight controller PID gains.

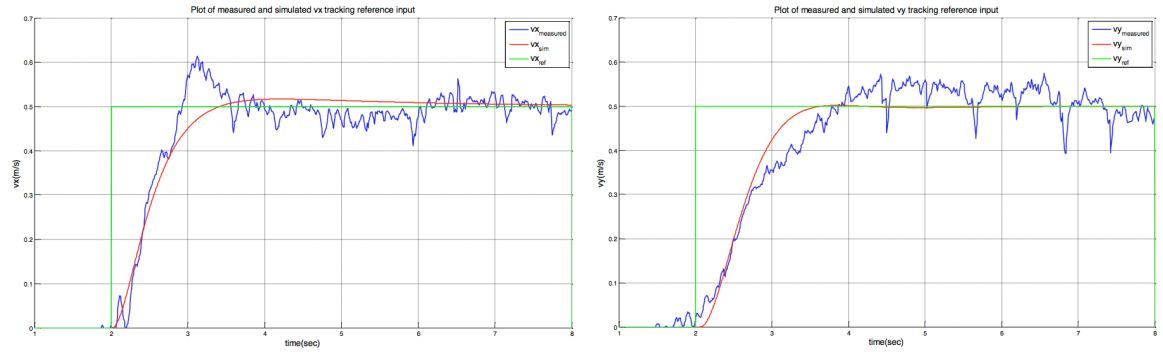


a) Yaw Step response **b) Altitude Step response**

Figure 6. The measured and simulated quad-rotor response for step reference for the yaw and altitude.

The main reason for this could be due to the noisy measurement of velocity, which is calculated on-board using a combination of IMU and an optical flow algorithm. However, the steady state response is satisfactory and the overall performance was found to be acceptable. Additional details on the system identification and control experiments can be found in reference [17].

The next section describes the process of incorporating the images captured by the two on-board cameras on the AR.Drone for navigation within an indoor environment.



a) x-velocity Step response **b) y-velocity Step response**

Figure 7. The measured and simulated quad-rotor response for step reference for the x- and y-velocities.

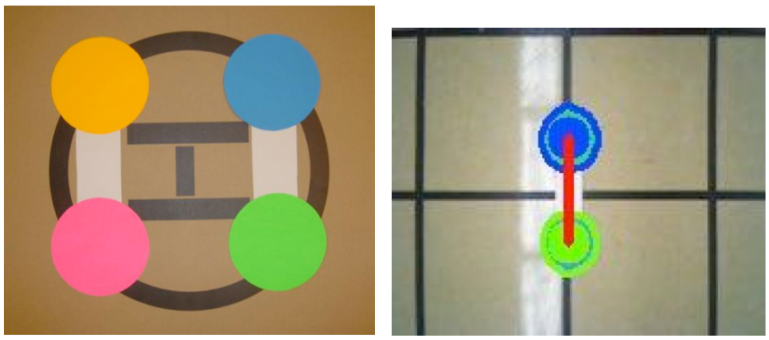
III. Vision

In this section we discuss implementations of yaw drifting correction, localization by landmarks, and autonomous flight by using vanishing points that uses computer vision techniques.

A. Landmark Detection

Artificial landmarks are constructed from a pair of circular color objects as shown in figure 8a. The image data sent from the quad-rotor is in Red-Green-Blue (RGB) color space. To threshold each image, the RGB color space is transformed into Hue-Saturation-Value (HSV) color space for convenience. The HSV image is further processed by applying 2-D Gaussian filter of the form $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$. This smoothing kernel creates a weighted average that weights pixels at its center much more than at its boundaries. A kernel window of 5×5 is chosen with $\sigma = 1.25$. The filter parameters are empirically determined by comparing suppression of high frequency noise in the binary threshold image.

The absolute orientation of the landmark is obtained by pairing two circular shapes as one. This is accomplished by the computation of the moments and centroid of the binary threshold image of each marker. Knowing the centroids of each circular object, the orientation of the landmark is obtained by connecting the centroids. The detected landmarks are shown in figure 9.



a) Artificial Landmarks b) Landmark picture by vertical camera

Figure 8. (a) The circular shapes in the artificial landmarks have a radius of 6 centimeters. (b) A picture of the detected landmark taken by the vertical camera in hovering flight.

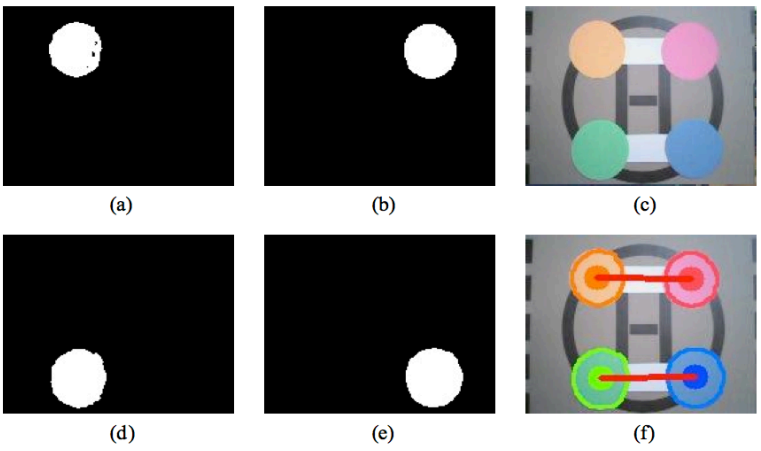


Figure 9. (176×144)-resolution pictures of detected landmarks taken by vertical camera. Each circular color object is smoothed by Gaussian filter, which is followed by the application of the threshold for each object in HSV color space. The result of threshold image is displayed in binary image where picture (a) is orange, (b) is pink, (d) is green, and (e) is blue. Picture (c) is the original image before processing, and picture (f) shows the centroids for each object and orientation line.

B. Yaw Drift Correction by Landmarks

The figure 8b shows a picture of the landmark on the floor captured by the quad-rotor while hovering over it. The quad-rotor's heading is referenced at zero degree and regulated by the PID controller discussed earlier. The comparison between compensated and uncompensated yaw attitude is shown in figure 10. As seen in this figure, in just over 14 minutes of hovering, the uncompensated yaw attitude has drifted by 110 degrees.

Landmarks on the floor can provide a precise correction of yaw attitude when they are positioned perpendicular to the yaw axis of the reference coordinate system. Following this, the yaw drift can be compensated by constantly updating the bias offset when landmarks are visible to the quad-rotor's vertical camera.

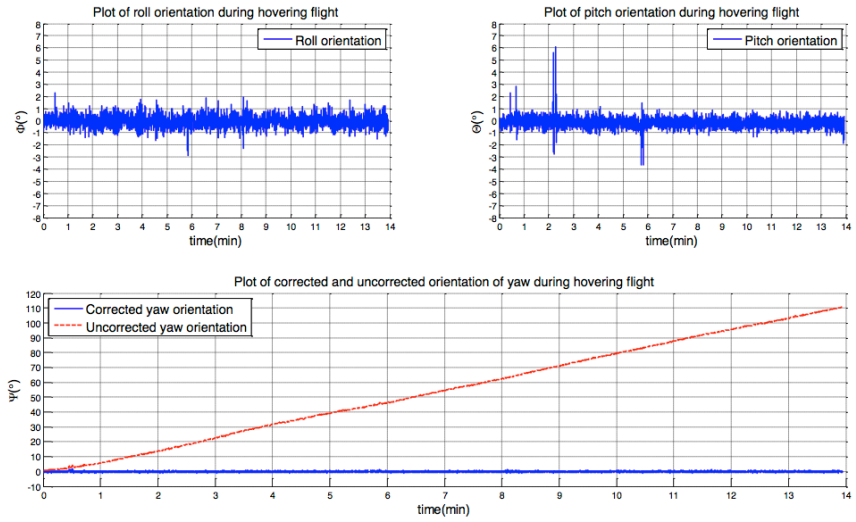


Figure 10. Plot of roll, pitch, corrected and uncorrected yaw attitude during hovering flight. The heading is fixed at zero degree by the yaw PID controller. The uncorrected yaw attitude drifts from its true yaw attitude of 0 to 110 degrees within 15 minutes.

Similarly, the x and y position of the quad-rotor can be corrected. The ultrasound sensor attached to the bottom of the quad-rotor consequently estimates the altitude of the quad-rotor the altitude readings do not suffer from any drift issues. In effect, the localization problem is simplified to a 2-D problem where the coordinates of x- and y-values for quad-rotor in the world coordinates of frame are to be estimated. Having a priori knowledge of the 2-D coordinates of landmarks in the world frame accomplishes this objective. To solve this problem, one must be able to relate the landmark coordinates in the world frame to pixel coordinates in the image frame and to the quad-rotor's body coordinates, and then back to the world frame. This is facilitated by the intrinsic and extrinsic parameters of the camera discussed in the following subsection.

C. Camera Calibration

The intrinsic, and extrinsic parameters of the vertical camera on-board the AR.Drone must be known to infer any coordinate information from landmarks on the floor. A linear calibration model suffices for the low-resolution cameras on the AR.Drone.

We use the method described in reference [20] to model the intrinsic matrix of the vertical camera. A test using a calibration grid yielded an aspect ratio that was approximately unity. Assuming zero skew, and modeling the camera as a pinhole lens, we obtain the equations $u = u_0 - f \times \frac{x}{z}$ and $v = v_0 - f \times \frac{y}{z}$, where u and v are the coordinates in the image plane, f is the focal length of the camera, u_0, v_0 are the origins of the image plane that is collinear with the camera frame's origin through the optical axis, and x, y and z are coordinates of a point in the world frame described in the camera reference frame (cf. figure 11). The intrinsic parameters of the camera can be expressed in matrix form as

$$\mathbf{K} = \begin{pmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Three mutually orthogonal *vanishing points*, $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k)$, in the image can be used to solve for the three unknowns in $\mathbf{K}$. A vanishing point is a point in a perspective image to which parallel lines, that are not parallel to the image plane, appear to converge. This requires the following equations to the satisfied simultaneously: $\mathbf{p}_i^T(\mathbf{K}^{-1})^T\mathbf{K}^{-1}\mathbf{p}_j = 0$, $\mathbf{p}_j^T(\mathbf{K}^{-1})^T\mathbf{K}^{-1}\mathbf{p}_k = 0$, and $\mathbf{p}_k^T(\mathbf{K}^{-1})^T\mathbf{K}^{-1}\mathbf{p}_k = 0$.

The three vanishing points expressed in homogeneous coordinates are obtained by capturing a photograph of different orientations of a paper box with gridlines (cf. figure 12). The vanishing point that minimizes the square distance error from all intersection points was found in each direction. The intrinsic parameters, obtained as an average of several different orientations, are $u_0 = 88.003, v_0 = 69.66$ and $f = 203.533$, expressed in units of pixel length. If x, y are the coordinate points of the centroid of a detected landmark, which is expressed in the camera coordinate frame, and u, v is its position in the image, we have $x = (u - u_0)\frac{z}{f}$ and $y = (v - v_0)\frac{z}{f}$, where z is the altitude of the quad-rotor. The x, y coordinates of the quad-rotor in the world frame can be computed using the expression $\mathbf{P}^w - \mathbf{R}_c^w\rho^c$, where ρ^c is the x, y position of the centroid of the detected landmark in the camera coordinate frame; $\mathbf{R}_c^w$ is the rotational matrix associated with the world frame, that can be computed using the roll, pitch and yaw readings from the on-board sensors; and $\mathbf{p}^w$ is the location of the centroid of the landmark in the world frame.

To test the accuracy of intrinsic model, the quad-rotor is attached to a fixture above ground where the vertical camera's optical axis is perpendicular to the surface plane. The centroid of an object, which is placed on the plane detected by the camera, is calculated using the above-mentioned method. This value was compared with the ultrasound sensor readings, and the detected object coordinates had approximately $\pm 15\ mm$ along the x and y-axis. Consequently, observing the known landmarks in an indoor environment can localize the quad-rotor.

D. Navigation using Vanishing Points

When flying down a long corridor the flight path controller developed in previous section has limitations. The IMU suffers from drifting, and the accumulated errors in heading direction will inevitably fly the quad-rotor towards the wall. Providing multiple landmarks along the track to correct the heading can negate this issue, but the number of

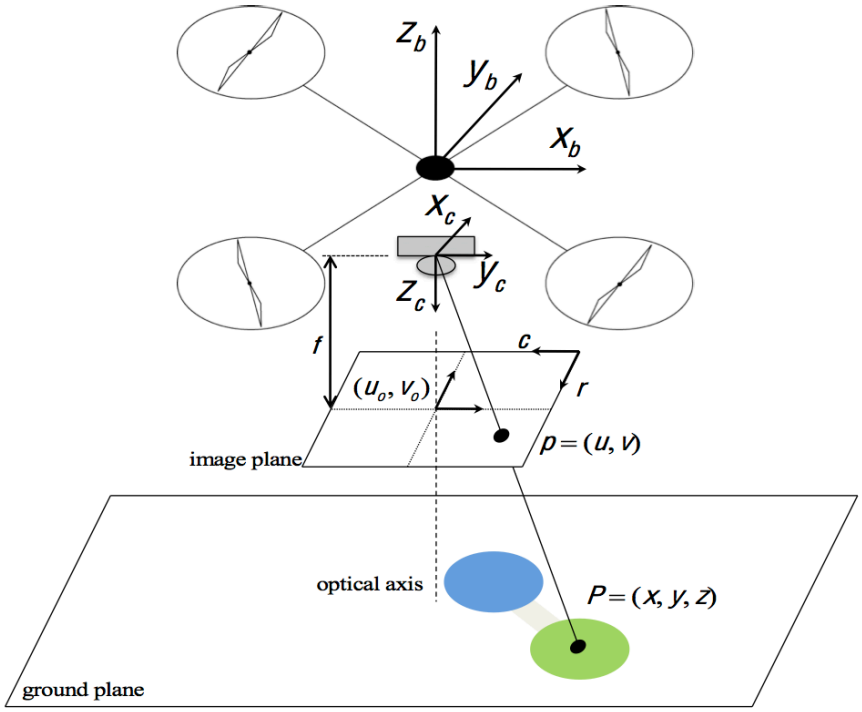


Figure 11. A schematic of the vertical camera system.

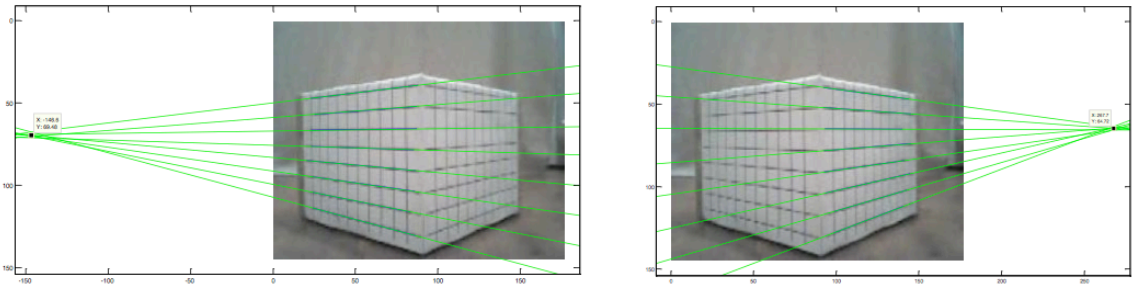


Figure 12. An illustration of the calculation of the vanishing point performed in MATLAB.

landmarks required will increase in direct proportion to the length of the corridor. Additionally, the limited load capacity of the AR.Drone does not permit the use of additional proximity sensors, as it would reduce the flight time.

The straight lines in the corridor have been used to navigate a robot by detecting vanishing points in the scene in the literature (cf. [21]). In theory, vanishing points in the corridor should be located at the center, but because of noisy response from the image, further processing is required to estimate the true vanishing point in the scene.

The edges in the scene are obtained by using *Canny Edge Detection* algorithm [22]. Following this, straight lines in the detected edge image are found by using the *Hough transform* [23]. This method uses a voting scheme to find a line in edge image. The detected edge image is in binary form, so each pixel in the binary image can be part of some set of possible lines. For each point in the binary image, two parameters that describe the point are plotted in the accumulator plane also known as the Hough space, and appropriate bin sizes are chosen along with a threshold value that is then used to obtain lines in the binary image. There are many tunable parameters to consider when working with Canny Edge Detection and Hough transform. These values are obtained empirically by running captured video frames of the corridor off-line and tuning the parameters prior to implementation on the quad-rotor. The relevant details of this procedure can be found in reference [17].

After eliminating lines in the edge-detected frame that are close to vertical or horizontal, the intersection of the remaining lines in the current image frame F_t are computed using the cross-product. A weighted average of the median of the ensemble of intersection points of the current image frame F_t and the previous image frame F_t is presented as the present estimate of the vanishing point. This is illustrated in figure 13. Unlike ground robots, when the quad-rotor is in flight, the image plane will rotate due to pitching and rolling motions. Consequently, the image plane has to be rotated back into its original frame using the rotational matrices introduced previously. The result of this operation is shown in figure 14.

1. Vanishing Point Navigation

The vanishing point in the image frame can generate an input signal to the quad-rotor to enable autonomous navigation through the corridor. The heading of the quad-rotor is regulated by a PI controller, and the error signal $e_v = (0 - v_u)$, where v_u is the normalized pixel distance between the vanishing point and image center (cf. figure 15). The altitude and velocities are regulated using the method described in section B, where the x-velocity and altitude is user-defined, while the y-velocity is regulated to zero.

The combinations of vanishing point navigation controller with object detection ability, the quad-rotor is able to navigate through a corridor and localize itself. The experiment is done at Mechanical Engineering Laboratory (MEL) corridor at University of Illinois at Urbana-Champaign. As shown in figure 16, two landmarks are placed in the corridor. When the horizontal camera detects the landmark, the quad-rotor tracks the landmark by generating an error signal, and a simple proportional controller is implemented. When the tracked landmark goes out of sight, the camera is immediately switched to the vertical camera. The quad-rotor sweeps over the landmark and is able to localize given

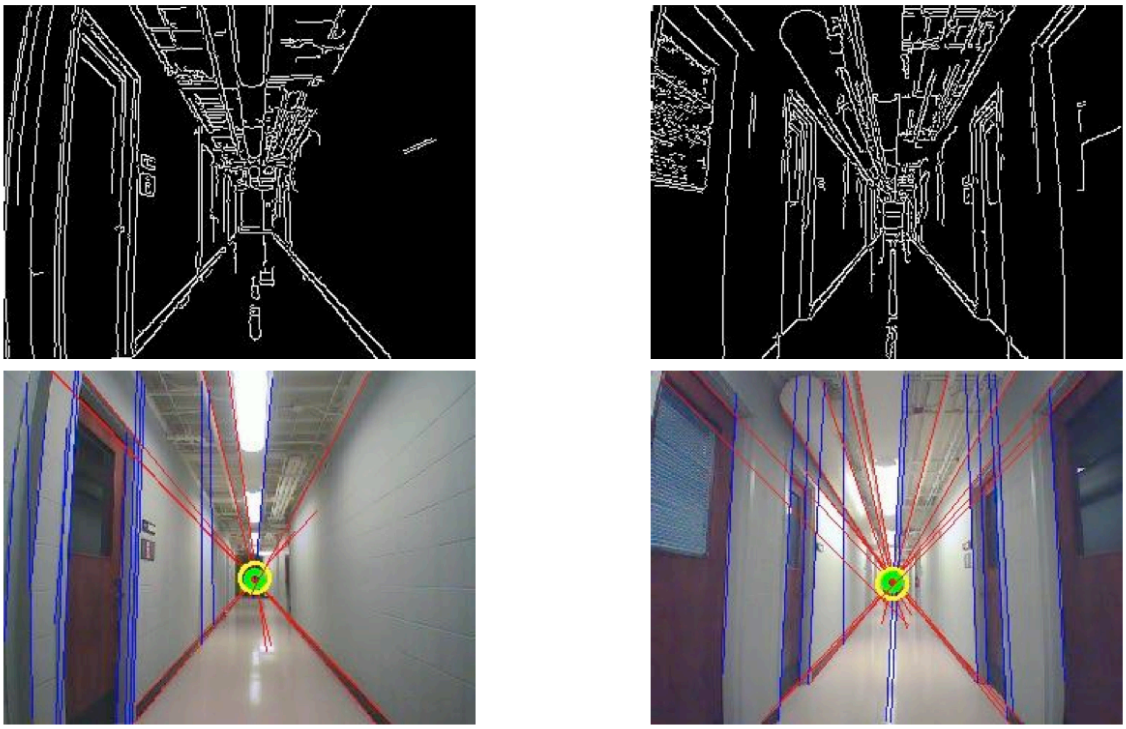


Figure 13. Images of edge detection and vanishing point detection in the corridor during real-time flight. The yellow circle indicates result of median value in the current frame, and the green circle indicates the weighted average of the vanishing point with the previous frame's vanishing point.

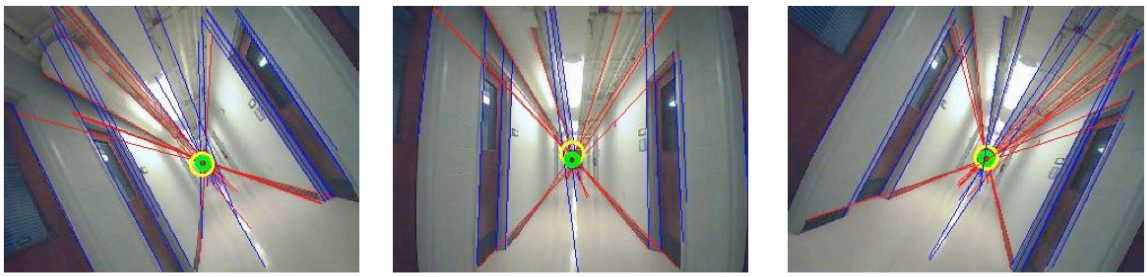


Figure 14. Result of applying rotational matrix to image plane. The blue vertical line in image (b) represents true vertical lines in the world frame. The rotated images (a) and (c) from positive and negative roll orientation, the true vertical lines are ignored. Only the red lines are used for vanishing point calculation.

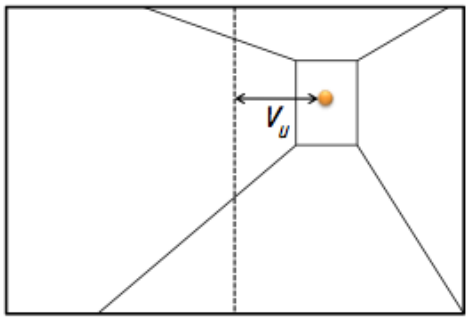


Figure 15. The centerline represents optical center and the v_u is the pixel distance of vanishing point in image plane to the optical center along the horizontal direction.

the landmark's coordinate and orientation in the world frame. The vanishing point disappears as the quad-rotor gets close to the end of the corridor. Vanishing point navigation alone cannot control the quad-rotor to turn the corners in the corridor. By combining the flight path controller developed previously with the vanishing point navigation controller, corner turn is achieved.

Because the dead reckoning position estimate is unreliable, at least one landmark per 10 meters of travel is required for reasonable performance. To navigate through two corridors that span over 40 meters of distance, four landmarks are used (cf. figure 17). Two landmarks per corridor are used where one of them is placed at near the corner turn. The flight path controller relies on inertial sensor data to turn the corner and proceed with vanishing point navigation. The result is shown in figure 18 with each landmark locations plotted. Prior to take off, the yaw orientation is drifted to -9.17 degrees, which propagates the position estimation error until it is corrected. This demonstrates the sensitivity of the dead reckoning method to initial conditions. In addition, there is a discrepancy between true ending position and assumed ending position, which is due to the absence of landmarks in the 3rd corridor. Nevertheless, the experiment of corner turn navigation is successfully executed by combining two controllers with minimal number of landmarks placed. The landmarks could be placed further apart on the ground for wider corridors. However, the AR.Drone could drift significantly from the middle of the path between landmarks.

Figure 16. Landmarks in the MEL corridor.

A video of a different instance of a successful autonomous flight along the path identified in figure 18 can be found at the following link. The initial drift in the yaw orientation for this flight, before the first landmark corrects the position estimate, is smaller than the drift in the instance shown in figure 18. This was the main difference between what was observed in the thirteen runs, each of which was successful, on this flight-path. The average absolute value of the position-error, with landmarks, was 0.38 meters, with a standard-deviation of 0.37 meters for this flight-path.

The vanishing point method described in this paper will perform poorly in an environment where there is an insufficient number of straight-lines in the edge-detected image of the forward facing camera (cf. figure 13). This is the case when the quad-rotor has to turn a corner, for example. The presence of landmarks around these locations is

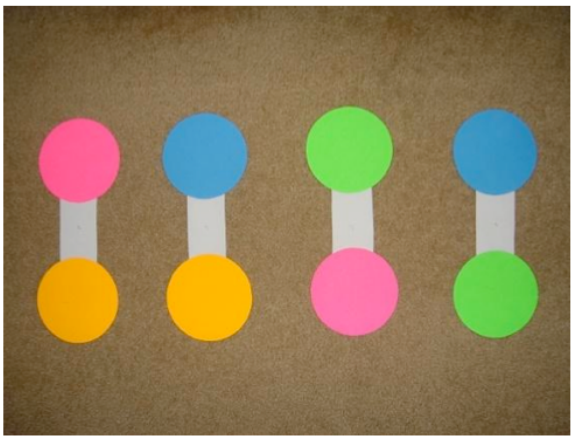


Figure 17. Four different landmarks used for corner turn navigation and localization.

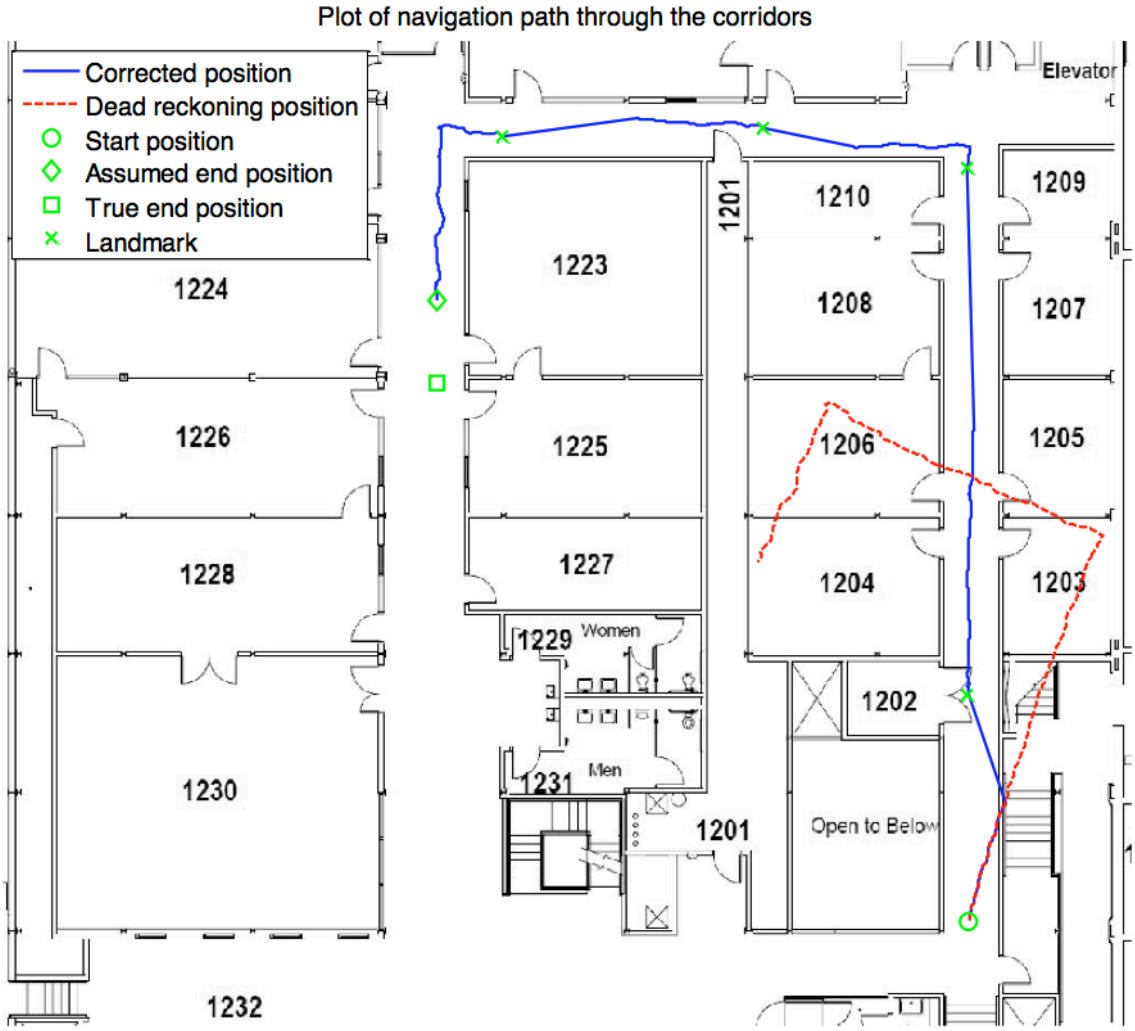


Figure 18. An instance of successful autonomous navigation and localization through multiple corridors. The dead reckoning position and uncorrected yaw orientation accumulates significant amount of error throughout the course of this experiment.

mandatory for satisfactory flight performance.

IV. Conclusions

In this paper we considered the problem of autonomous navigation of a quad-rotor in a *Global Positioning System* denied, known, indoor environment that uses only the on-board sensors. We obtained a decoupled model for the quad-rotor using a gray-box modeling approach and standard system identification tools. This model was subsequently validated for indoor flight. The drift in yaw, which is commonplace to any method that relies solely on inertial navigation, was regulated using artificial landmarks and a vertical on-board camera. To reduce the number of landmarks, and to manage the heading direction computation of the quad-rotor, we used a method that relied on the vanishing point in the image sensed by the forward facing camera. These methods enabled the quad-rotor to successfully navigate autonomously and localize itself in an indoor environment.

References

[1] Hoffmann, G., Rajnarayan, D. G., Waslander, S. L., Dostal, D., Jang, J. S., and Tomlin, C. J., "The Stanford Testbed of Autonomous Rotorcraft for Multi Agent Control (STARMAC)," *In Proceedings of the 23rd Digital Avionics Systems Conference*, Salt Lake City, UT, November 2004.

[2] Hoffmann, G. M., Waslander, S. L., and Tomlin, C. J., "Quadrotor helicopter trajectory tracking control," *AIAA Guidance, Navigation and Control Conference and Exhibit*, American Institute for Aeronautics and Astronautics, Honolulu, 2008.

[3] Hoffmann, G. M., Huang, H., Waslander, S. L., and Tomlin, C. J., "Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment," *AIAA Guidance, Navigation and Control Conference and Exhibit*, American Institute for Aeronautics and Astronautics, Hilton Head, South Carolina, August 2007.

[4] Castillo, P., Dzul, A., and Lozano, R., "Real-time stabilization and tracking of a four-rotor mini rotorcraft," *IEEE Transactions on Control Systems Techology*, Vol. 12, No. 4, 2004, pp. 510–516.

[5] Bouabdallah, S., Murrieri, P., and Siegwart, R., "Design and Control of an Indoor Micro Quadrotor," *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA-04)*, April-May 2004, pp. 4393–4398.

[6] Hamel, T., Mahony, R., and Chriette, A., "Visual Servo trajectory tracking for a four rotor VTOL aerial vehicle," *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA-02)*, May 2002, pp. 2781–2786.

[7] "Hoverbot," http://www-personal.umich.edu/~johannb/hoverbot.htm, 2014, [Online; accessed 24-February-2014].

[8] "Mesicopter," http://aero.stanford.edu/mesicopter/FinalReport.pdf, 2014, [Online; accessed 24-February-2014].

[9] Altug, E., Ostrowski, J. P., and Taylor, C. J., "Control of a quadrotor helicopter using visual feedback," *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA-02)*, 2002, pp. 72–77.

[10] S.Park, Won, D. H., Kang, M. S., Kim, T. J., Lee, H. G., and Kwon, S. J., "Ric (robust internal-loop compensator) based flight control of a quad-rotor type UAV," *In Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 3542–3547.

[11] Lupashin, S., Schöllig, A., Sherback, M., and DAndrea, R., "A simple learning strategy for high-speed quadrocopter multi-flips," *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA-10)*, May 2002, pp. 1642–1648.

[12] Ahrens, S., Levine, D., Andrews, G., and How, J. P., "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA-09)*, 2009, pp. 2643–2648.

[13] Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R., "Vision Based MAV Navigation in Unknown and Unstructured Environments," *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA-10)*, May 2010, pp. 21–28.

[14] Altug, E., Ostrowski, J. P., and Taylor, C. J., "Quadrotor Control Using Dual Camera Visual Feedback," *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA-03)*, September 2003, pp. 4294–4299.

[15] Ludington, B., Johnson, E., and Vachtsevanos, G., "Augmenting UAV autonomy," *IEEE Robotics and Automation Magazine*, Vol. 13, No. 3, 2006, pp. 63–71.

[16] "OpenCV," http://opencv.org/documentation.html, 2014, [Online; accessed 24-February-2014].

[17] Ahn, Y.-M., *Autonomous Navigation and Localization of a Quadrotor Mini-UAV by Landmarks in Indoor Environments*, MS thesis, University of Illinois at Urbana-Champaign, 2011, Department of Mechanical Science and Engineering.

[18] "Control System Toolbox," The MathWorks, Inc., User's Guide.

[19] Wendel, A., Irschara, A., and Bischof, H., "Natural Landmark-based Monocular Localization for MAVs," *Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, May 2011, pp. 5792–5799.

[20] Caprile, B. and Torre, V., "Using vanishing points for camera calibration," *Internal Journal of Computer Vision*, Vol. 4, No. 2, 1990, pp. 127–140.

[21] Shi, W. and Samarabandu, J., "Corridor line detection for vision based indoor robot navigation," *Proceedings of the 2006 Canadian Conference on Electrical and Computer Engineering (CCECE -06)*, Ottowa, Ontario, May 2007, pp. 1988–1991.

[22] Canny, J., "Computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 6, 1986, pp. 679–698.

[23] Duda, R. O. and Hart, P. E., "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Communications of Association for Computing Machinery*, Vol. 15, No. 1, 1972, pp. 11–15.