Targeted Autonomous Indoor Flight of a Rotary-Wing MAV

Svetlana Potyagaylo
Graduate Student
Faculty of Aerospace Engineering
svetap@tx.technion.ac.il

Omri Rand
Professor
Faculty of Aerospace Engineering
omri@aerodyne.technion.ac.il

Yaron Kanza
Associate Professor
Faculty of Computer Science
kanza@cs.technion.ac.il

Technion, Israel Institute of Technology
Haifa, 32000, Israel

ABSTRACT

The paper is focused on flight missions of Autonomous Rotary-Wing Micro Aerial Vehicles (RW MAVs) in indoor GPS-denied environments and the development of algorithms for such missions. In such tasks, GPS may be unavailable and the map of the environment is a priory unknown. This paper presents a modular system consisted of the required components for flying RW MAVs in indoor environments. These components include methods for position estimation of the MAV, for planning a flight path toward a target location while taking into account detected obstacles and maneuverability limitations of the vehicle, and for calculating flight commands to fly along the planed path. The methodology is based on using a lightweight laser range finder as a sole onboard sensor. Moreover, only one of the system components requires knowledge about the dynamic model of the MAV. Our simulation results illustrate the effectiveness of the approach and of the system modules.

INTRODUCTION

In the recent years, Micro Aerial Vehicles (MAVs) received a growing attention because of their usefulness in various military and civil applications and the significant progress in onboard computational capabilities. The small size and high maneuverability of MAVs make them suitable for indoor missions, especially when compared with a fixed-wing vehicle that must maintain a minimum speed without being able to hover. However, indoor missions introduce new challenges — the operation methodology cannot rely on GPS signals for identifying the location of the vehicle, due to poor reception, or no reception at all, of the signal.

In this paper we study the problem consisted in autonomous navigation of a RW MAV from an initial known position and orientation to a definite final goal, while the map of the environment is a priori unknown. In order to safely complete the mission, the vehicle must execute different tasks during the flight. A MAV that is operating in GPS-denied environments has to be able to identify its position (e.g. two coordinates, of location, and heading angle, in a 2D setting) merely based on its surrounding. Because the map is a priori unknown, the vehicle needs to sense the environment and construct the map simultaneously, based on some estimation of its position. This task is widely known as Simultaneous Localization and Mapping (SLAM) (Ref. 1). Once the vehicle identifies its location, it can move towards the goal. Thus, the next task is determination of paths that are collision-free with respect to the obstacles detected so far. In addition, the MAV should compute the control commands necessary for flying along a desired trajectory, while taking into account limitations regarding the ability to accelerate or slow down, the ability to turn whilst maintaining its current speed, etc. The existence of system errors, disturbances, uncertainty and changes in the environment during the flight add complications to the developments of the required algorithms and to the ability to combine these algorithms.

A significant progress has been made in each of the above listed tasks for navigation of ground robots and fixed-wing aerial platforms. However, the adaptation of most of the existing algorithms to RW MAVs is not straightforward. First, RW MAVs have a considerably limited payload capability, so the use of heavy sensors or a set of lightweight sensors is usually infeasible. Second, the dynamics of flying vehicles, and especially of RW vehicles, is more complex and faster than that of ground robots and it contains many restrictions and constraints. Thus, modeling and controlling MAVs is hard, in comparison to ground autonomous vehicles. Finally, navigation commands must be provided instantly, whereas onboard computing power is limited. Although RW MAVs can interrupt the flight and

Presented at the American Helicopter Society 68th Annual Forum, Fort Worth, Texas, May 1-3, 2012.

hover (i.e. momentarily stop to allow onboard calculation to be completed), this is not desired because it decreases endurance and increases the vulnerability of the vehicle.

Thus, for an autonomous flight of a RW MAV in a GPS-denied environment, the following ingredients are essential:

- A method for estimating the position of the vehicle: In this paper, the position of the MAV is calculated in the *Position Estimation (SLAM) module* based on the *model-free SLAM method* that allows to estimate the location of the MAV simultaneously while updating the map of the environment.

- A method for path planning: In this study we developed a *Path Planner module* that employs two methods for path planning — an *A* graph search* for global planning and a *potential field method* (PFM) for local path calculating.

- A method for computing control commands: The proposed system includes the *Controller module* that calculates the control commands to fly the MAV along the planned path.

Several papers studied autonomous navigation of RW MAVs in GPS-denied environments. Shen et al. (Ref. 2) proposed a quadrotor platform for autonomous multi-floor navigation in buildings, however, differently from our system, they use several sensors. The SLAM, the path planning and the control planning tasks are carried out onboard and in real time, so the system is described as "fully autonomous". Grzonka et al. (Ref. 3) studied the control of a quadrotor system that merely uses a laser range finder. However, their system relies on the existence of a known map of the environment to estimate the position and it assumes the existence of a laser mirror to deflect some of the laser beams, to control the altitude. In their system, user interaction is allowed to control the flight altitude and heading and to prevent collision with obstacles. He et al. (Ref. 4) used a quadrotor platform as well but focused on the motion planning algorithm. In their system, during the flight, the vehicle uses a known map and a laser range finder to localize itself within the map. The algorithm ignores the vehicle dynamics and tries to find a sequence of vehicle poses (position and orientation) that can be used to move the vehicle from an initial state position to a goal position.

In contrast to the above approaches, our methodology addresses the problem of indoor navigation of a conventional helicopter MAVs (i.e. having a main-rotor tail-rotor scheme). The position estimation and mapping merely rely on laser scan data. The only onboard sensor is a lightweight laser range finder. Our approach takes into account the helicopter dynamics and limitations, however, it is model-free in the sense that it does not require estimating the position of the vehicle based on its dynamic model.

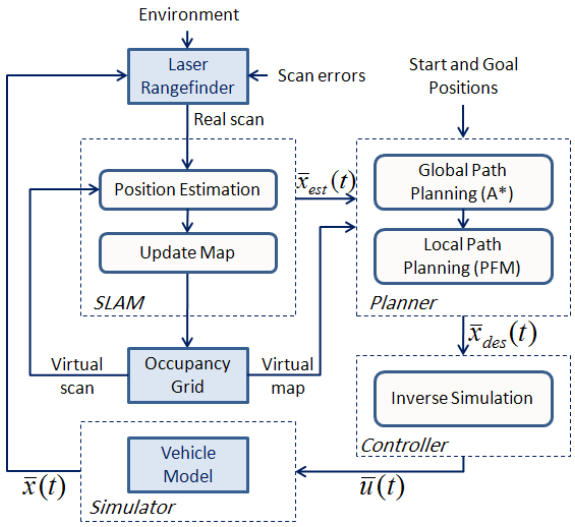


Fig. 1. The components of the system.

SYSTEM AND METHODOLOGY

We present now an overview of our methodology and of the modular system that executes it. A schematic diagram of the system and of its components is depicted in Fig. 1. The system includes a "position estimation module" (SLAM), a "path planner module" and a "controller module". The architecture is modular so that each component can be used independently of the others.

For localization and mapping, the system employs a *model-free SLAM* module (Ref. 5), which, differently from other SLAM modules, uses only one source of data (laser range finder) and does not use the dynamic model of the vehicle. Since the map of the environment is a priory unknown, detected obstacles are represented and stored in memory in the form of an *occupancy grid* (OG) (Ref. 6). The OG is used for performing a "virtual scan" as part of the SLAM. The virtual map is updated after each laser scan, when the position estimation is complete. The path planning module combines two algorithms — an *A* search algorithm* and a *potential field method* (PFM) (Ref. 7). The A* algorithm finds the shortest collision-free path from the current (estimated) position of the MAV toward the target position. This path provides waypoints that serve as intermediate goals for the PFM. We then apply the PFM to calculate a feasible path from the current position to the farthest waypoint which is within a line of sight from the MAV. This separation of the path planning task into two sequential algorithms is required due to the shortcoming of both global (A*) and local (PFM) planners as will be describe later on. To determine control commands that fly the MAV along the planned trajectory, a module that uses an *inverse simulation* (IS) (Ref. 8) technique was developed. The module receives the desired flight vector and interprets it to control commands that are suitable for a RW MAV.

Fig. 2 presents schematically the steps of computing a

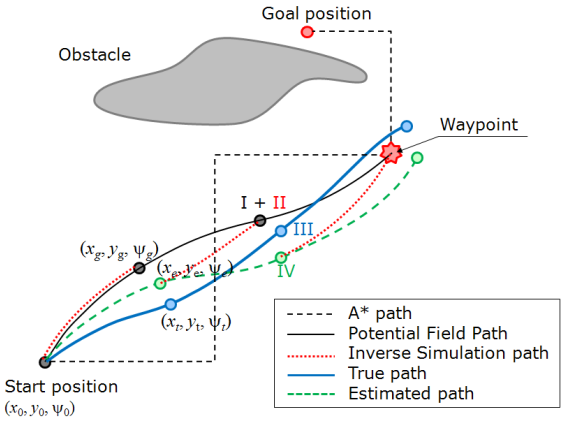


Fig. 2. The planned flight path (A* and PFM), the computed steps to fly along the planned path (IS), the estimated path (SLAM) and the actual (true) flight path.

flight path according to our methodology. It depicts the planned flight path, the actual flight path and the computed flight steps, and illustrates the process of computing the flight path. Initially, the MAV is located at a start position (x_0, y_0, ψ_0) with known coordinates. A laser scan provides initial data about the obstacles ahead. The scan results are added to the OG. Then, A* and PFM are employed consecutively to calculate the nearest waypoint and the next point (x_g, y_g, ψ_g) to arrive at (Stage I). The IS algorithm computes the control commands for arriving at this point (Stage II). This point serves as a guess for the position estimation process. After that, the control commands are executed (Stage III). The result of this stage is a point (x_t, y_t, ψ_t) that differs from the guess point due to errors in the commands and in the execution, and due to unconsidered effects of the environment. The position estimation module matches two scans — a virtual scan in the virtual map when assuming that the MAV is located at the guessed point, and a real laser scan. The matching provides an estimation of the vehicle position (x_e, y_e, ψ_e) (Stage IV). Simultaneously, the map updating process is proceeded.

Before the next iteration, the validity of the planned paths of the A* algorithm and PFM are checked. In the case of collision with newly detected obstacles or a possibility to move the waypoint forward along the A* trajectory, the A* or PFM paths are recalculated. On the next iteration, all the stages are repeated, until arriving at the goal.

The system uses two main coordinate frames — a *global coordinate system* and a *body coordinate system*. The origin of the global coordinate system is arbitrarily fixed and its axes $x_g l$ and $y_g l$ are oriented to the east and to the north, respectively. Navigation of the MAV is carried out within this frame. The body coordinate system is related to the MAV itself — the origin is located at the center of gravity (CG) of the MAV, the x_b-axis points forward and the y_b-axis is to the left side of the MAV. This coordinate system is used for computing control commands.

THE MAIN MODULES

Localization and Mapping

SLAM is one of the most fundamental problems in robotics. It deals with the task of building a map of an unknown environment along with an estimation of the robot position within this map. This problem arises since a map of the environment is a priori unavailable and the movements of the autonomous vehicle are neither perfect nor deterministic. SLAM is often referred to be a "chicken-and-egg" problem (Ref. 9): a map of the environment is required for solving a localization problem while an accurate pose estimate is needed to build a map. These two problems can be solved apart relatively easily, assuming either the presence of a map of the environment, for the localization problem, or knowing the vehicle pose (e.g. by using a GPS), for the mapping problem. However, when there is no map of the environment and there is no GPS or other device to provide the pose of the vehicle, the two problems must be solved simultaneously.

Traditionally, SLAM algorithms solve the problem using Extended Kalman filters (Ref. 1) or Particle Filters (Ref. 10). These methods use a probabilistic model to represent the observations, and a dynamic model of the vehicle. The Model-Free SLAM algorithm we propose is different from previous ones by not relying on any knowledge regarding the vehicle model.

Our algorithm uses an occupancy grid in which each cell represents a sub-area of the operation environment, and contains the number of laser "hits" registered for that sub-area. The OG is also used for performing a *virtual scan* produced by a series of ray casting operations, searching for occupied cells. The virtual scan is executed with respect to guessed position and heading of the vehicle. It is then compared with a laser scan obtained from the actual laser range finder of the MAV. The difference between the *virtual* and *real* scans is the basis for the position estimation. We refer to it as *scan matching* (Ref. 11).

We employ Algorithm 1 for computing the position estimation of the MAV, using the scan matching method. In each iteration, the algorithm "guesses" the shift between the actual location of the vehicle and the guessed position. It compares the computed virtual scan to the roto-translated actual scan in the real world. Using an adaptive direct search (Ref. 5), a new shift is computed. The process terminates when a location that minimizes the error between the actual scan and the virtual scan is discovered, and this location is returned as the estimated location of the vehicle.

While the search estimates the position of the MAV, the map updating process can be proceeded. However, map updating is performed only if the matching succeeds, i.e. the result of the scan matching is below a defined threshold. This is required since the OG serves as an average of all

Algorithm 1 Estimate the position of the MAV, using Scan Matching

1: Consider an initial guessed location and heading of the MAV: $x = x_g$, $y = y_g$ and $\psi = \psi_g$, which will be denoted by the vector (x_g, y_g, ψ_g).
2: Scan the environment using a laser range finder, with respect to the actual (true) position of the MAV (x_t, y_t, ψ_t), and store the resulting data as a vector (r_r, θ_r), for the scanned angles θ in the range $-\theta_{max} \leq \theta \leq \theta_{max}$.
3: Create a new virtual scan (r_v, θ_v), based on the OG and the initial guess (x_g, y_g, ψ_g).
4: Consider the initial shift in the position and heading between the guessed position (x_g, y_g, ψ_g) and the true position (x_t, y_t, ψ_t) as: $(\Delta x_g, \Delta y_g, \Delta \psi_g)$.
5: **repeat**
6: Convert the real laser data (r_r, θ_r) so that it would look as if it was measured from the guessed position using the initial guess shift. At this step, the following equations are used:

$$x' = r_r \cdot \cos(\theta_r + \Delta\psi_g) + \Delta x_g;$$
$$y' = r_r \cdot \sin(\theta_r + \Delta\psi_g) + \Delta y_g;$$
$$r' = \sqrt{x'^2 + y'^2};$$
$$\theta' = \tan^{-1}(y'/x').$$

The result of this step is a roto-translated real scan (r', θ').
7: Apply a series of filters to the roto-translated real scan (r', θ'), to leave only valid points.
8: Calculate the norm of the error that represents the discrepancy between (r', θ') and (r_v, θ_v), only for the range covered by both.
9: Use an *adaptive direct search* for computing a new guess vector $(\Delta x_g, \Delta y_g, \Delta \psi_g)$.
10: **until** The guess vector minimizes the error norm.
11: **return** $(\Delta x_g, \Delta y_g, \Delta \psi_g)$

previous laser scans and thus, if the virtual scan is not accurate enough, the updating of the OG will not be accurate as well. This would lead to a rapidly growing error in the position estimation.

The position estimation provided by the SLAM module is used for velocity estimation as well. The velocities in the x and y directions are updated based on the displacements, in these directions, between the previous and current position estimations. The shift in the heading angle is used for updating the MAV velocities in the body coordinate system.

Path Planning

Navigating a MAV to a target location is not a simple task. First, obstacles are only discovered during the flight, so the path should be constantly updated. Secondly, the flight trajectory should be as short as possible. Thirdly, the path should comply with maneuverability limitations of the MAV, e.g. changes in the flight orientation should not be acute. The path planning module combines an A* search method and a PFM to navigate the RW MAV. The A* search algorithm is used for planning an initial global path that provides waypoints. Then in each iteration, PFM takes the farthest waypoint that is within line of sight (with respect to the location of the MAV) on the path computed by A*, as an intermediate goal, and it returns a feasible trajectory to that goal, as a reference for the computation of the control commands.

A* is a general search algorithm that traverses a graph of nodes (or grid cells). In this method, the path to the target is computed in steps. In each step every node is associated with an estimation of the distance to reach from it to the target. The method adds to the partially computed path the node for which the distance to it plus the estimated cost is minimal. The cost of the node is calculated base on a heuristic function, which estimates the path length from the current node to a goal. This process continues until the goal is reached. In this work, the Euclidean distance from the current MAV position to the target was chosen as the heuristic function, being one of the most widely used heuristic functions for such tasks.

A* is applied over an OG that has the same structure as the grid of SLAM but uses larger cells to reduce the memory requirement of the A* algorithm. When A* computes a path to the target, it allows eight directions of movement, including moving on the diagonals of cells. It produces trajectories that consist of straight segments with sharp turns in the vertices. Thus, the produced paths are irreconcilable with the MAV dynamics. Obviously, a helicopter can cope with such changes by frequent "stop and re-orient" procedures; however, this type of maneuvering is undesired. The PFM is serve to cope with this problem. It takes into account the kinematic constraints of the MAV (mainly minimum turning radius) and smooths the path, making it viable.

The PFM considers the MAV as a particle moving in a force field produced by repulsive forces from obstacles and attractive force to the goal. The approach of obstacles that "induce" forces on a vehicle has been first suggested by Khatib (Ref. 7) with respect to manipulators. The resultant force acting on a vehicle at any position in the force field can be expressed as the sum of repulsive forces from obstacles and the attractive force towards the goal (see Fig. 3):

$$\vec{F}_{rez} = \vec{F}_{goal} + \sum_{i=1}^{N} \vec{F}^{i}_{obst}, \quad (1)$$

where N is the number of obstacles; $\vec{F}^{i}_{obst}$, $i = 1 \ldots N$ are the repulsive forces; $\vec{F}_{goal}$ is the attractive force towards the goal and $\vec{F}_{rez}$ is the resultant force.

In this paper, the attractive force of the goal and the re-

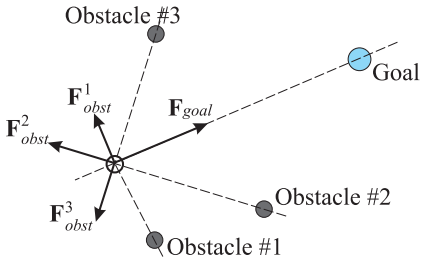


Fig. 3. Illustration of forces in the Potential-Field Method.

pulsive forces of the obstacles are defined as:

$$\begin{aligned} \overline{F}_{goal} &= I_{goal}\frac{\overline{x}_{goal}-\overline{x}_{MAV}}{d_{goal}}; \\ \overline{F}_i &= I_{obst}H_i\frac{\overline{x}_i-\overline{x}_{MAV}}{d_i}, \end{aligned} \tag{2}$$

where I_{goal} is the constant positive intensity level of the goal, $\overline{x}_{goal} = (x_{goal}, y_{goal})$ is the goal position, $\overline{x}_{MAV} = (x_{MAV}, y_{MAV})$ is the current (estimated) MAV position, d_{goal} is the distance from the current MAV position to the goal position. Similarly, I_{obst} is the negative intensity level of the obstacle, inversely proportional to the distance d_i from the cell center to the center of mass of the vehicle, H_i is the hit count of the cell, and $\overline{x}_i$ is the position of the cell center.

Yet, overlaying attractive and repulsive potential functions as described above can result in a local minima where the resulting force is equal to zero at a position other than the target. A typical scenario for this case is when the vehicle faces an obstacle and the target is behind the obstacle. This is the major drawback of the potential field approach. Thus, the main issue is escaping from the local minima. The waypoints provided by A* serve as temporary goals. The temporary goal is always in line of sight of the MAV and it precludes such cases.

The intensities of the goal and of the obstacles and the mass of the particle are "virtual" in the sense that they are used only for trajectory planning. They are chosen in a way that ensures the kinematic constraints are satisfied (mainly the turning radius of the MAV).

Inverse Simulation

Given a trajectory to the target, it is required to compute control commands to fly along it. Several general methods for motion planning and for computing a sequence of control commands are based on optimal control tools (Ref. 12) and dynamic programming (Ref. 13). Some techniques of path planning can be extended for motion planning as well, for example, the rapidly-exploring random tree algorithm (Ref. 14) and genetic algorithms (Ref. 15). These approaches are unsuitable for helicopters whose maneuverability capabilities are described using complex and highly nonlinear models. Thus, an alternative approach, namely *inverse simulation*, has been developed. While simulation is traditionally used for solving the vehicle equations of motion to find the system response to a prescribed sequence of control commands, inverse simulation reverses this process and determines the controls required to produce a given response, defined in terms of the system variables or states.

Inverse simulation is commonly divided into two distinct approaches: a *differentiation-based* approach (Ref. 16) and an *integration-based* approach (Ref. 17). The controller module of the system, presented in this paper, uses the integration-based approach. This approach applies repeated numerical integration of the motion equations of the helicopter. An important advantage of the integration-based method is not being model-specific. This means that it can be used for different models without changing the algorithm. One of the drawbacks of this approach is that it is considerably slower than the differentiation-based method.

In the general case, the mathematical model of the vehicle may be written as the following system of the first-order differential equations:

$$\dot{\overline{s}} = \overline{f}(\overline{s}, \overline{u}), \tag{3}$$

along with the initial conditions $\overline{s}_0$. In the above equation, $\overline{s}$ is the system state vector, $\overline{u}$ is the control vector, and $\overline{f}$ is a nonlinear function that represents the applied forces and moments, typically referred to the CG and originated from different sources such as aerodynamic, structural, gravitational and inertial sources, and from different subsystems of a helicopter.

We use a model of the helicopter with six degrees of freedom, in addition to the fuselage attitude (Euler) angles and the main rotor flapping angles:

$$\overline{s} = (u, v, w, p, q, r, \psi, \theta, \phi, \beta_0, \beta_{1c}, \beta_{1s})^T, \tag{4}$$

where u, v, w are the translational velocity components; p, q, r are the rotational velocity components; ψ, θ, ϕ are the body yaw (heading), pitch and roll attitude angles, and $\beta_0, \beta_{1c}, \beta_{1s}$ are the main rotor flapping angles. These components are given with respect to the body frame.

Having defined the model, we proceed to the inverse problem which calculates the controls required for flying the helicopter in a specific trajectory and a given speed. For that purpose, the helicopter motion equation is supplemented by an additional output equation that associates the state vector with the required quantities:

$$\overline{h} = \overline{g}(\overline{s}, \overline{u}); \tag{5}$$

where $\overline{h}$ is the output vector, $\overline{g}$ is a nonlinear function.

In this paper, the integration-based inverse simulation method initially guesses the control inputs $\overline{u}$, integrates the vehicle motion equations, to achieve the desired output vector at the next iteration. The difference between the actual

flight vector and the desired flight vector is then used to calculate the estimation of the control inputs for the next path computation step. The equations of motion at the k-th step can be written as:

$$\begin{aligned} \overline{s}(t_{k+1}) &= \overline{s}(t_k) + \int_{t_k}^{t_{k+1}} \overline{f}(\overline{s}(t_k), \overline{u}^m(t_k))\, dt; \\ \overline{h}(t_{k+1}) &= \overline{g}(\overline{s}(t_{k+1}), \overline{u}^m(t_k)), \end{aligned} \tag{6}$$

where $\overline{u}^m(t_k)$ is the m-th estimation of the control inputs. The next iteration of the control input is estimated using the residual function which represents the difference between the desired output vector and the actual one and must eventually vanish.

For simplicity and for allowing onboard implementation within the limitations of current technology, a linearized mathematical model of the helicopter, in a trim state, was developed. The assumed linear model can be expressed in the following form:

$$\begin{aligned} \delta\dot{\overline{s}} &= \mathbf{A}\delta\overline{s} + \mathbf{B}\delta\overline{u}; \\ \overline{h} &= \mathbf{C}\overline{s}, \end{aligned} \tag{7}$$

where $\delta\dot{\overline{s}}$ is the deviation of the state vector from its trim condition; $\delta\overline{u}$ is the deviation of the control vector from its trim-state vector; **A**, **B** and **C** are constant state, control, and transformation matrices.

The state, control and output vectors are:

$$\begin{aligned} \overline{s} &= (u, v, w, p, q, r, \psi, \theta, \phi)^T; \\ \overline{h} &= (\dot{x_e}, \dot{y_e}, \dot{z_e}, \psi)^T; \\ \overline{u} &= (\theta_0, \theta_{1c}, \theta_{1s}, \theta_{tr})^T, \end{aligned} \tag{8}$$

where $\dot{x_e}$, $\dot{y_e}$, $\dot{z_e}$ are the velocity components of the vehicle in the global coordinate system; θ_0 is main rotor collective pitch angle; θ_{1c}, θ_{1s} are the main rotor lateral and longitudinal cyclic pitch angles; θ_{tr} is the tail rotor collective pitch angle. In this work we assumed that the MAV is equipped with a stabilization system that provides the required balance for roll, pitch and heading directions, to overcome the high sensibility of MAVs to external disturbances, such as wind gusts, due their light weight and small size.

It should be noted that the controller module performance and accuracy highly depend on the selection of the $\overline{s}$, $\overline{u}$ and $\overline{h}$ vector components. In this model, the inverse simulation is programmed to reach certain velocity components $\dot{x_e}, \dot{y_e}, \dot{z_e}$ at the end of each step. It allows to omit the additional integration stage, compared to the version of the output vector $\overline{h}$, with only coordinate components.

The above system and control matrices **A** and **B** consist of the partial derivatives of the nonlinear functions describing the helicopter motion, and they represent the changes in the forces and moments at the trim point due to the changes in the state vector (Ref. 18). These matrices were calculated using a detailed nonlinear simulation process for the nonlinear model of Equation (3), in which for a given flight condition the trim state parameters were calculated. Then, a numerical differentiation of the derivatives of the state vector components $\left(\dot{u}, \dot{v}, \ldots, \dot{\phi}\right)$, with respect to the state vector components and the controls $\theta_0, \theta_{1c}, \theta_{1s}, \theta_{tr}$, was carried out. The matrix **C** is a transformation matrix that interprets the velocity components u, v, w from the body coordinate system to the velocity components $\dot{x_e}, \dot{y_e}, \dot{z_e}$, in the global earth coordinate system.

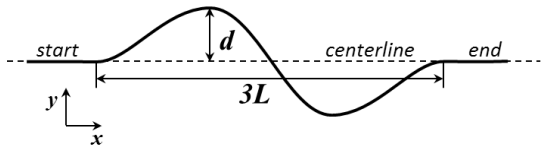
The inverse simulation module was tested to evaluate the performance of an unmanned helicopter model constructed earlier. For these purposes several mission-task-elements (MTEs) were selected and adopted, based on the performance specification ADS-33D-PRF (Ref. 19). This specification was originally designed for full-sized helicopters used in military missions, yet, the careful selection and proper adjustment of the performance parameters allow to use the specification for MAVs as well. In this paper, the following MTEs were chosen:

1. *Slalom*: The maneuver begins with a trim level flight and represents two smooth turns, up to predefined lateral shift, from the centerline to both sides, as shown in Fig. 4(a). The forward speed remains constant, and a coordinated level flight is assumed, during the maneuver. After the maneuver ends, the helicopter returns to a forward straight flight.

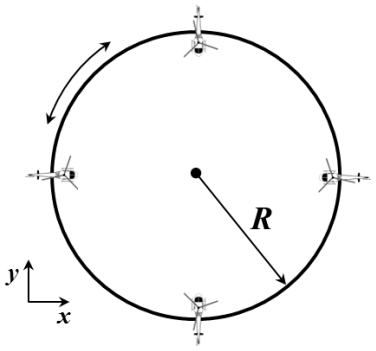
2. *Pirouette*: The maneuver is initiated while the helicopter is hovering over a point on the circumference of a circle with a given radius, having the nose of the helicopter pointing to the center of the circle. The helicopter has to accomplish a lateral translation around the circle, keeping the nose pointed to the center of the circle as depicted on Fig. 4(b). The lateral speed after the acceleration phase remains constant. The maneuver ends with an deceleration phase, to a stable hover.

For each maneuver the appropriate flight trajectory and required input for the inverse simulation algorithm were constructed. In what follows, the predefined parameters and desired input are described, for each maneuver.

1. *Slalom*: The input parameters for this maneuver are the trim forward speed $V[m/sec]$, the lateral displacement $d[m]$ and the length of the maneuver flight path along the centerline $L[m]$ (see Fig. 4(a)). The total time of the maneuver is defined according to the defined trim speed V. The equations for coordinates and velocities



(a)



(b)

Fig. 4. The maneuvers: (a) slalom; (b) pirouette.

in the x_e and y_e directions are:

$$\begin{aligned}
x_e(t) &= Vt \\
y_e(t) &= \frac{d}{8}\left(-81\xi(t)^3 + 135\xi(t)^4 - 81\xi(t)^5 + \right. \\
&\qquad \left. 21\xi(t)^6 + 2\xi(t)^7\right) \\
\dot{x}_e(t) &= V, \\
\eta(t) &= \frac{\partial y_g}{\partial \xi} = \frac{d}{8}\left(-243\xi(t)^2 + 540\xi(t)^3 - \right. \\
&\qquad \left. 405\xi(t)^4 + 126\xi(t)^5 + 14\xi(t)^6\right), \\
\dot{y}_e(t) &= \frac{V}{L}\eta(t), \\
\psi(t) &= tan^{-1}\left(\frac{1}{L}\eta(t)\right),
\end{aligned} \tag{9}$$

where $\xi(t) = Vt/L$.

2. *Pirouette*: The input parameters are the time of acceleration/deceleration phase $t_{trans}[sec]$, the lateral constant speed $V[m/sec]$, the radius of the turn $R[m]$ and the direction of the turn γ ($+1$ for a clockwise direction or -1 for a counter-clockwise direction). The total time of the maneuver is defined according to the defined trim speed V. The maneuver is performed by changing the lateral velocity of the helicopter smoothly, from a zero state to the defined value V, maintaining it constant during one revolution and that decreasing it smoothly to the zero value. The smooth change of the lateral velocity during acceleration and deceleration phases is supplied by the following polynomial functions:

acceleration phase:

$$\dot{y}_e = V\left(35\xi(t)^4 - 84\xi(t)^5 + 70\xi(t)^6 - 20\xi(t)^7\right),$$

deceleration phase:

$$\dot{y}_e = V\left(1 - 35\xi(t)^4 - 84\xi(t)^5 + 70\xi(t)^6 - 20\xi(t)^7\right), \tag{10}$$

where $\xi(t) = t/t_{trans}$.

The equations for a circular flight with a constant lateral velocity are:

$$\begin{aligned}
\psi(t) &= -\gamma\frac{2\pi R}{V}t, \\
x_e(t) &= R - R\cos(\psi(t)), \\
y_e(t) &= \gamma R\sin(\psi(t)), \\
\dot{x}_e(t) &= V\sin(\psi(t)), \\
\dot{y}_e(t) &= V\cos(\psi(t)).
\end{aligned} \tag{11}$$

For all the maneuvers it was assumed that the parameters of the desired vector, which remain unchangeable during the maneuver, are equal to zero. Additionally, the short phases of the trim forward flight or stabilized hover were embedded before and after each maneuver. The simulation results of the maneuvers, in terms of desired and obtained flight trajectory, control commands and attitude angles, are shown in Fig. 5 and in Fig. 6.

Fig. 5(a) shows that the trajectory obtained by the direct simulation of the helicopter model with the control commands calculated by the Controller module coincides with the desired trajectory for slalom maneuvers. A slight deviation in the z direction can be observed, and it is due to the presence of the trim vertical velocity. The attitude angles for the slalom maneuver, shown in Fig. 5(c), indicate that the maneuver is carried out by changing mostly the roll angle ϕ with amplitude 1° around its trim value. The amplitude of the pitch angle is twice smaller. The changes in the control commands for the slalom are shown in Fig. 5(b), where the tail rotor collective pitch command provides the desired variations in heading. The obtained trajectory for a pirouette maneuver, as depicted in Fig. 6(a), also matches the desired trajectory with high accuracy. The attitude angles and control commands remain constant during the circular flight, with contact lateral speed, according to the theory. The acceleration and deceleration for this maneuver are provided by the cyclic main rotor control, and consequently, by changes in both the pitch and roll angles.

SIMULATION RESULTS

To examine the effectiveness of the proposed system and of its components, we conducted simulations and tested the components separately and combined. We consider the simulation as a powerful and valuable tool for the system development. It gives a rich variety of possible situations and

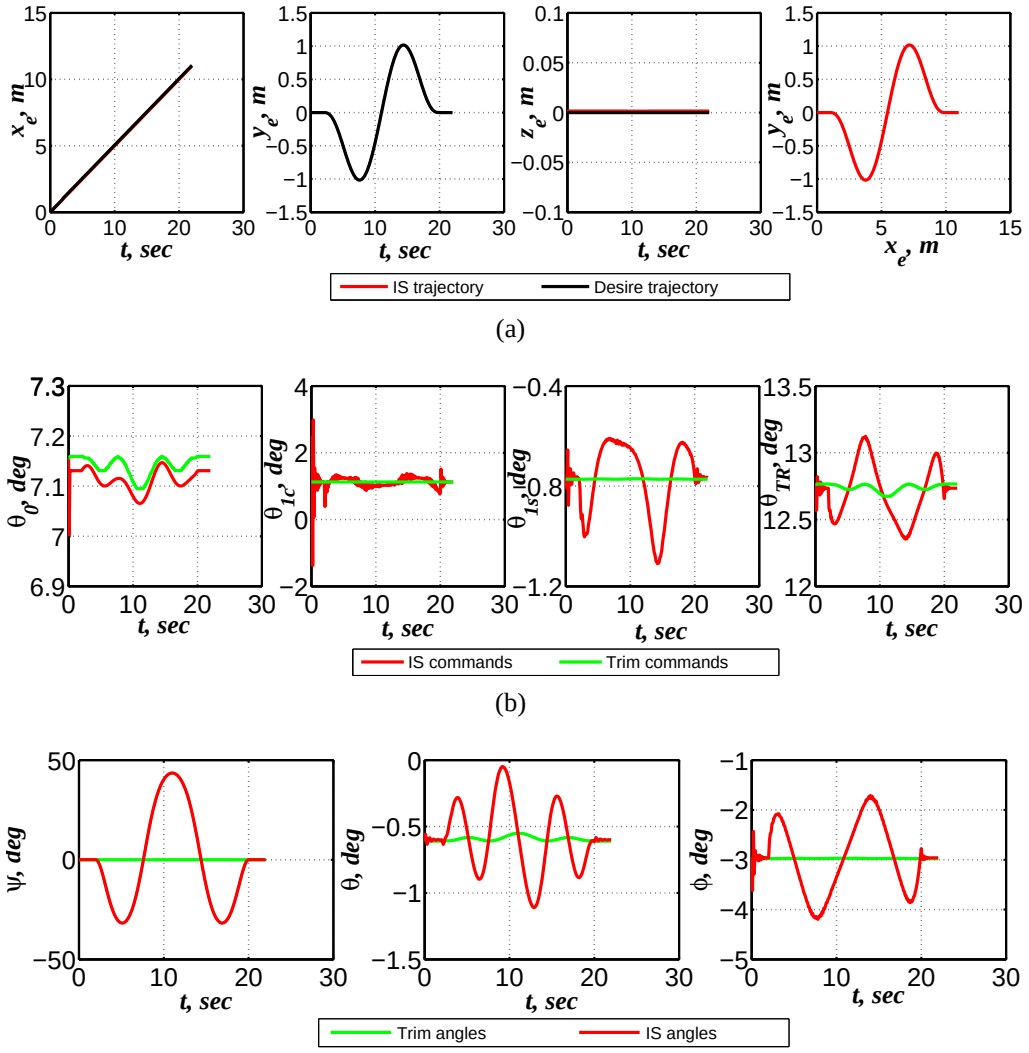


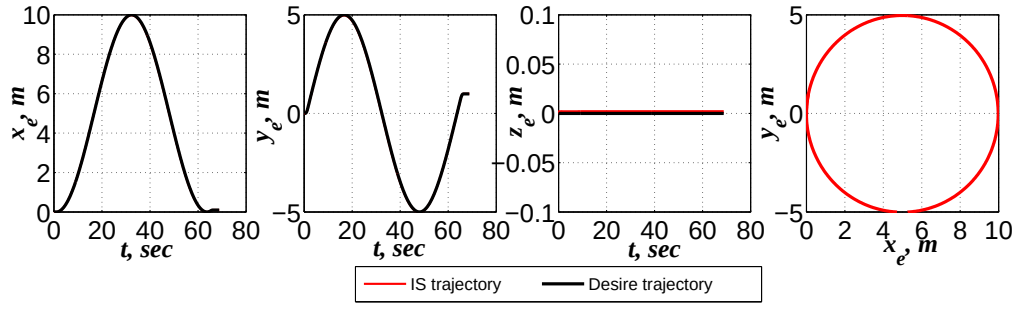
Fig. 5. The results of the Inverse Simulation for the slalom maneuver: (a) the desired and obtained trajectory; (b) the control commands calculated by the Controller module; (c) the attitude angles of the MAV.

scenarios for testing the system, estimating its performance and examining the collaborative work of its modules in different regimes and scales. The task considered in this paper belongs to the class of problems for which the simulation is highly effective. A numerous unrealistic environments may be constructed and verified; the operating regimes and scales may be selected to generate test cases that cannot be examined in real experiments. Moreover, the simulation allows to measure the accuracy of the system with very high precision. This is almost impossible in real life, where the full information about the vehicle state and the environment properties are often unavailable.

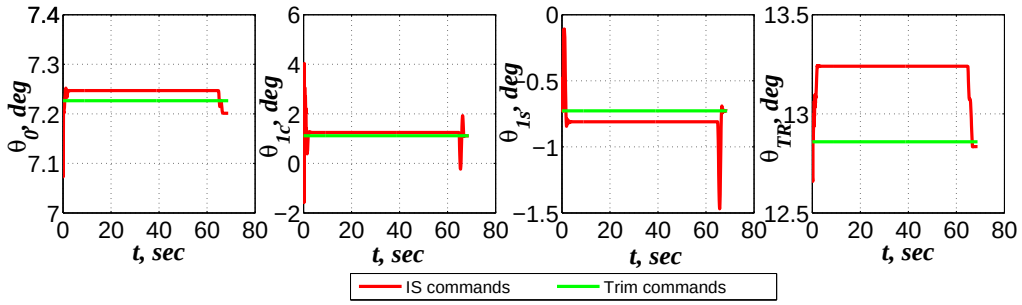
Although we used specific parameters in our simulations, the parameters can be changed and the approach can be applied to navigation at different scales. We can do so by taking the length of the OG cell as a reference unit, replace this unit by some real unit, and change the other parameters proportionally.

Simulation Setup

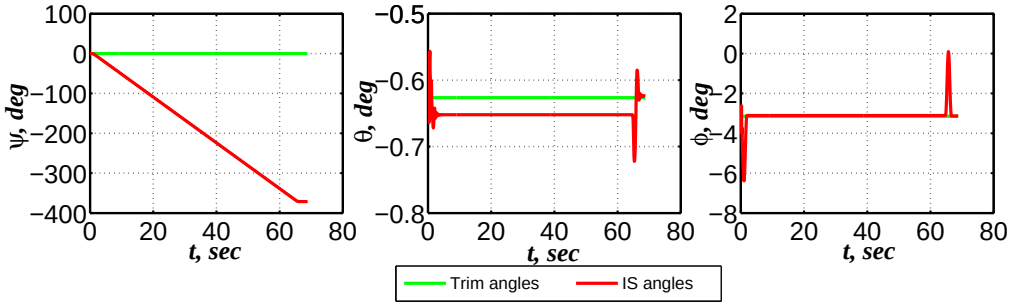
In the simulation we assumed the use of the following components. As a single laser range finder we consider the Hokuyo 2D laser scanner (Ref. 20). This scanner has a maximum range of approximately 30m, field of view of 270° and an angular resolution of 0.25°. According to the specifications, the distance accuracy of the laser range finder is below 1% of the measured distance, at the worst case. Thus, in all the simulations we present, the noise level of "real" scans is 1.5%. The helicopter we considered is SR RTF, manufactured by Blade, Horizon Hobby, Inc. (Ref. 21). It was chosen as a prototype for modeling the controller and simulator modules. The main parameters of the helicopter



(a)



(b)



(c)

Fig. 6. The results of the Inverse Simulation for the pirouette maneuver: (a) the desired and obtained trajectory; (b) the control commands calculated by the Controller module; (c) the attitude angles of the MAV.

are: main rotor diameter of 552*mm*, tail rotor diameter of 82*mm*, weight of 340*g*, length of 485*mm*.

The actual model of the MAV includes the different sources of uncertainness and noises, such as wind gusts, and noises in the control commands. The wind model implies an injection of the wind velocity components normally distributed with zero mean and standard deviation of $0.2m/sec$ as a disturbance input to the vehicle respective channels. The command noise level is 2%.

The resolution of the OG was set so that cells have a size of $10 \times 10mm$. The cells of the A* OG are at size $500 \times 500mm$. The two constant parameters the PFM uses (the intensity of the goal force and fictitious mass of the vehicle) where chosen in a way that ensures the required smoothness of the flight trajectory. In addition, forces in PFM are computed only with respect to cells of the OG which are in the fore third of the field of view, i.e. $\pm 60^\circ$ with respect to the heading vector. The algorithms are implemented in MATLAB combined with C++ mex-files, so the computational costs of the system modules were evaluated in terms of their percentage of the total runtime.

The simulations were carried out using the RAPiD/RaTE rotorcraft analysis software package (Ref. 22). This package is designed to model and analyze general rotorcraft and rotary-wing based configurations. It is capable of modeling, analyzing and simulating general conventional helicopters (i.e. helicopters with a main and a tail rotor), tandem helicopters, coaxial and tiltrotor configurations. RAPiD/RaTE is also capable of analyzing propellers and various types of wind turbines. The RAPiD/RaTE solution methodology is based on direct time domain integration and post-processed frequency analysis. RAPiD/RaTE has been review and display extensively (Ref. 23), (Ref. 24).

Results and Discussion

To demonstrate the effectiveness of our modular system, Fig. 7 illustrates the results of the SLAM, Path Planner and the Controller modules. This figure depicts three successive snapshots of one of the simulated environments. Note that not all the obstacles are detected immediately due to the limitations of the laser range finder. Hence, the computed flight path must be adapted during the flight. The estimated A* and Potential Field paths are unremittingly checked for their validity according to the incoming data about newly detected obstacles. In the case of a collision between a planned path and a detected obstacle, the planned paths are recalculated.

The error plot demonstrates that the SLAM module provides highly accurate results for position estimation and heading angle and shows that no additional errors are accumulated through time. The true trajectory and estimated trajectory depicted in Fig. 7(a) also uphold the accuracy of the position estimation module. The control commands calculated in the controller module and the true controls that are passed to the simulator are shown on Fig. 7(b). The true commands are received from the control commands by adding a noise with Gaussian distribution with deviation of 0.5° and zero mean. The controller module does not take into account all the limitations of the control commands. Consequently, the commands can exceed their allowed margins. Therefore, a *limiter filter* adjusts the control to feasible values within the limits, before applying these commands in the simulator.

The control commands are generated at frequency of $10Hz$ but the frequency of the SLAM and Path Planner modules is lower and is about $0.5-1Hz$. To adjust the differences in the operation frequency of the different system modules, for each time step of the SLAM and Path Planner modules, the system creates a polynomial function that smoothly modifies the velocity components and heading of the MAV, from the current estimation to the desired values. The desired velocity and heading values are computed based on the position coordinates at the next time step provided by the Path Planner, and current flight regime of the vehicle. It was assumed that there are five flight regimes: (1) normal flight with constant forward speed; (2) transition regime from a forward flight to a hover state; (3) hover; (4) turn while hovering; (5) flight resumption from hover. The current flight regime is selected relying on the information from all system modules. The nominal flight regime of the MAV is a forward flight with constant speed. The slight velocity deviations can be occurred during a single step of the flight, due the guarantee to cross the required distance in a two-dimensional plane.

Switching from one flight regime to another flight regime is conducted using the strong links between the modules. For example, in the case where the SLAM module failed to estimate the current position of the MAV, at some point along the path, the algorithm autonomously cause the MAV to slow down to a hover state. This hover regime may provide enough time for additional laser scans, their capturing and improving the position estimation. This case is illustrated in the rightmost drawing of Fig. 7(a). Another example of a collaborative work of the modules is presented in the middle snapshot of Fig. 7(a), where large changes in the heading are required. In this case, the algorithm causes the MAV to hover at the point where the heading should change, to accomplish the turn while in a hover state and then to resume the flight.

CONCLUSIONS

We presented a modular system for indoor navigation of RW MAVs toward a given target. The problem is studied for a vehicle that senses the environment using merely a laser range finder. The main challenges in such task are the following: (1) Position estimation — in the absence of a GPS signals, the location of the vehicle must be estimated according to the measurement of the environment. (2) Map updating — in the case of a priory unknown environment the map must be constructed based on the accurate information regarding the vehicle position. (3) Obstacle avoidance — the path planning should prevent collision of the MAV with obstacles. (4) Keeping the flight to be as short as possible — the planned path should not be much longer than what is necessary for completing the task. (5) Considering the dynamics of the vehicle — the control command for flying along the planned path should be in accordance with the flight abilities of the MAV.

The proposed system consists of independent modules for position estimation, path planning, and computation of control commands. Our simulation results show that:

1. Accurate location estimation is a crucial, yet difficult, task for navigating a MAV in a GPS-denied environment. Our results show that the SLAM module is accurate without relying on any particular knowledge regarding the model of the vehicle.

2. The absence of a map, i.e. the lack of a priori known information about the environment and about the obstacles in it, prevents using many existing path-planning algorithms that relays on the existence of a fully known map, and it reduces the effectiveness of others. The path planning module proposed in this study provides smooth flight trajectories while dealing with unexpected dead-ends, avoiding obstacles and taking into consideration the maneuvering limitations of the vehicle.

3. The dynamics constraints of the vehicle must be taken into account to prevent computation of non-executable paths. The inverse-simulation module translates the planned flight path to actual commands accurately enough for the SLAM to cope with execution errors.

4. The proposed methods were tested for different environments. The results illustrate the potential of the algorithms and the viability of the combination of these modules for indoor navigation of RW MAVs .

REFERENCES

[1]H. Durrant-Whyte and T. Bailey. Simultaneous Localisation and Mapping (SLAM): Part I. The Essential Algorithms. *Robotics and Automation Magazine*, 13, 2006.

[2]S. Shen, N. Michael and V. Kumar. Autonomous Multi-Floor Indoor Navigation with a Computationally Contrained MAV. In *IEEE Int. Conference on Robotics and Automation*, pages 20–25, Shangai, China, May 2011.

[3]S. Grzonka, G. Grisetti and W. Burgard. Towards a Navigation System for Autonomous Indoor Flying. In *IEEE Int. Conference on Robotics and Automation*, pages 2878–2883, Kobe, Japan, May 2009.

[4]R. He, S. Prentice and N. Roy. Planning in information space for a quadrotor helicopter in a GPS-denied environment. In *IEEE Int. Conference on Robotics and Automation*, pages 1814–1820, Pasadena, California, USA, May 2008.

[5]C. Friedman, I. Chopra, S. Potyagaylo, O.Rand and Y. Kanza. Towards Model-Free SLAM Using a Single Laser Range Scanner for Helicopter MAV. In *AIAA Guidance, Navigation, and Control Conference*, Portland, USA, August 2011.

[6]S. Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous robots*, 15(2):111–127, 2003.

[7]Khatib O. "Real-time obstacle avoidance for manipulators and mobile robots". *International Journal of Robotics Research*, 5(1), 1986.

[8]D. G. Thomson and R. Bradley. Inverse simulation as a tool for flight dynamics research – principles and application. *Progress in Aerospace Sciences*, 42:174–210, 2006.

[9]S. Thrun, W. Burgard and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[10]D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An Efficient FastSlam Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV, USA, October 2003.

[11]A. Diosi and L. Kleeman. Laser Scan Matching in Polar Coordinates with Application to SLAM. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[12]A. E. Bryson and Y. C. Ho. *Applied Optimal Control*. Prentice-Hall, Hemisphere, New York, 1975.

[13]D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1-2. Belmont, Mass.: Athena Scientific, 2005-2007.

[14]E. Frazzoli, M. A. Dahleh and E. Feron. Real-time Motion Planning for Agile Autonomous Vehicles. *J. of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[15]V. Shaferman and T. Shima. Unmanned Aerial Vehicles Cooperative Tracking of Moving Ground Target in Urban Environment. *J. of Guidance, Control and Dynamics*, 31(5):1360–1370, 2008.

[16]D. G. Thomson and R. Bradley. Development and Verification of an Algorithm for Helicopter Inverse Simulations. *Vertica*, 14(2):185–200, 1990.

[17]R. A. Hess and C. Gao. A Generalized Algorithm for Inverse Simulation Applied to Helicopter Maneuvering Flight. *J. of the American Helicopter Society*, 38(4):3–15, 1993.

[18]G. D. Padfield. *Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modelling*. Oxford: Blackwell, 2007.

[19]ADS-33D-PRF. Aeronautical Design Standard Performance Specification Handling Qualities Requirements for Military Rotorcraft. Technical report, U.S. Army Aviation and Troop Command, 1996.

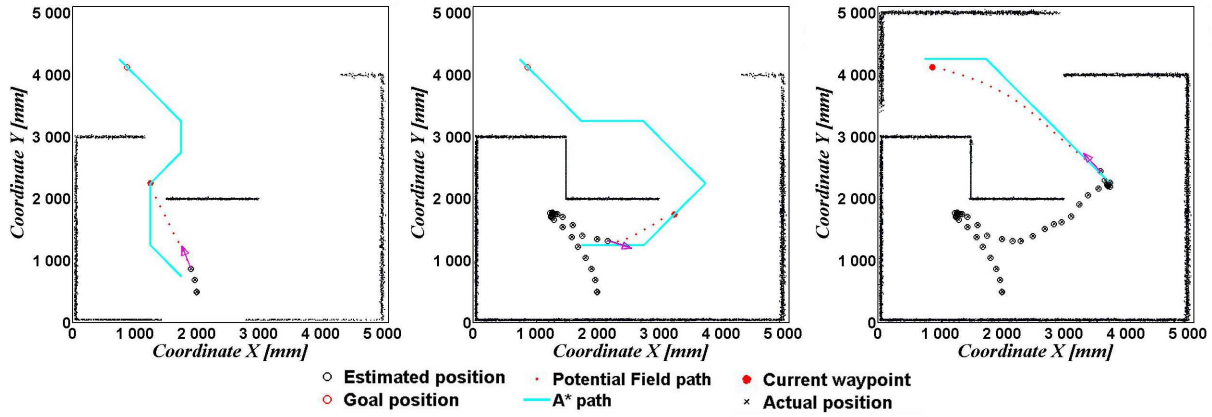
[20]Hokuyo. UTM-30LX. Technical report, http://www.hokuyo-aut.jp, 2009.

[21]Blade, Horizon Hobby, Inc. http://www.bladehelis.com/.

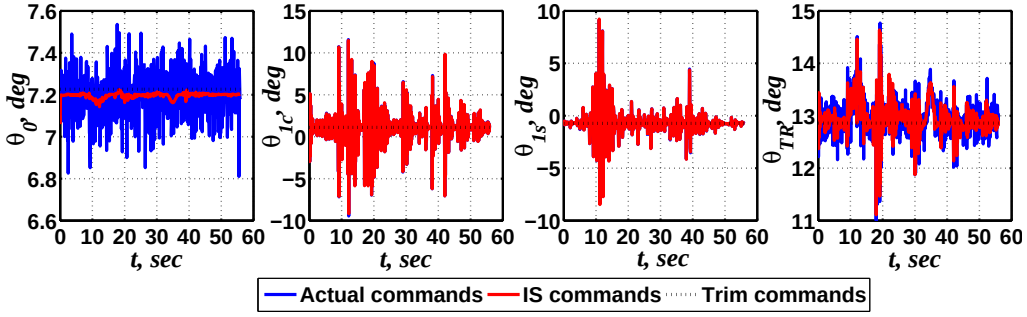
[22]O. Rand and S.M. Barkai. Numerical Evaluation of the Equations of Motion of Helicopter Blades and Symbolic Exactness. *J. of the American Helicopter Society*, 40(1):59–71, 1995.

[23]R. E. Hansford. Review of RAPID. *Vertiflite*, 42(5), September/October 1996.

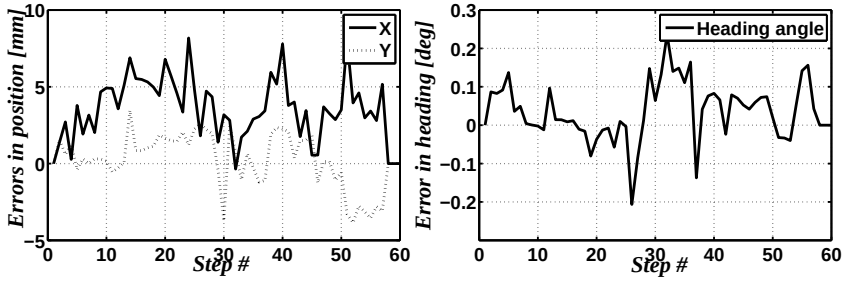
[24]O. Rand. Technology Display and Exhibition of the American Helicopter Society 56th Annual Forum, May 2000.



(a)



(b)



(c)

Fig. 7. Flying through an environment whose map is unknown: (a) Left: Initial planned trajectory; Middle: An abrupt heading change due to a dead-end recognition; Right: The actual flight trajectory. (b) The tail rotor collective pitch command and heading angle of the MAV. (c) The differences (errors) between estimated and actual position and orientation of the MAV.