Autonomous Flight Control in Micro Air Vehicles

International Aerial Robotics Competition

(IARC)2014

Author No. 1

AbdulAhad Pakbaz Student Engineering of Islamic Azad University, Saravan Branch

Author No. 2

Mohammad Baset Dorazehi Professor of Islamic Azad University, Saravan Branch

Author No. 3

Professor doctor Naser Nosratieh Faculty member of Islamic Azad University, Saravana Branch

ABSTRACT

Recently there has been increasing research on the devel-opment of autonomous flying vehicles. Whereas most of the proposed approaches are suitable for outdoor operation, only a few techniques have been designed for indoor environments. In this paper we present a navigation system for an indoor quadrotor. Our system adapt tech-niques which have been successfully applied on ground robots to our flying platform. We validate our system with real-world experiments.

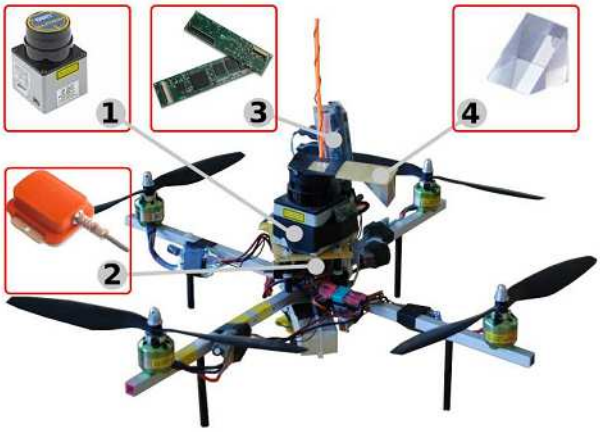
INTRODUCTION

Low-cost and small-size quadrotors are becoming broadly available. Some of these vehicles are able to lift relatively high payloads and provide an increas-ingly broad set of basic functionalities. This enables even unexperienced pilots to control these vehicles and allows them to be equipped with autonomous naviga-tion abilities. Most of the proposed approaches for autonomous flying [15, 6, 15] focus on systems for outdoor operation. Vehicles that can autonomously operate in indoor environments are envisioned to be useful for a variety of applications including surveillance and search and rescue. In such settings and compared to ground vehicles, the main advantage of flying devices is their increased mobility.As for ground vehicles, the main task for an autonomous flying robot consists in reaching a desired location in an unsupervised manner, i.e. without human interference. In the literature, this task is known as navigation. To address the general task of navigation one requires to tackle a set of problems ranging from state estimation to trajectory planning. Most of these tasks have been success- fully addressed by using ground robots. Whereas the general principles of the navigation algorithms, which have been successfully applied on ground robots, could in principle be transferred to flying vehicles,

this transfer is not straightforward for several reasons. First, due to their limited payload and size an indoor flying robot cannot carry the variety of sensors which can be easily mounted on a mobile robot. Second, the additional degrees of freedom of the vehicle prevents the direct use of well known and efficient 2D algorithms for navigation. Third the dynamics of a flying robot is substantially more complex than that of ground-based vehicles which makes them harder to control. Finally, one has to consider the increased risk of damaging the platform Several authors used vision to control or assist the control of an indoor quadrotor [1], [2], [3]. Roberts etal. [4] used ultrasound sensors for controlling a flying vehicle in a structured testing environment, while He et al. [5] presented a system for navigating a small-size quadrotor without GPS. Birds can fly in both indoor and indoor. [12:00:58 AM] GOLEYAS: It is suggested that in a vertical flying bird because if the fly of the bird in all dimentions and pivots it is better to use from two lesser scanners till the bird (robot) do scan their envoronment by to pivots of x and y to find and recognize the probable obstacles to show a three dimensional picture of the environment to the crash probability of the robot to the things under, up, back and front do decrease to have the best efficiency.

Second Level Heading

action may include:
1 - Quad rotor and HXZ rotor Mini
2 - Terry copters
3 - Helicopter Model
4 - Micro birds



III. HARDWARE ARCHITECTURE quadrotor platform used to evaluate the navigation system is based on a Mikrokopter and includes a Hokuyo laser range finder (1), an XSens IMU (2), a Gumstix computer (3), and a laser mirror (4). Figure 2 shows a Mikrokopter [3] open source quadro- tor equipped with sensors and computational devices. The Mikrokopter comes with a low level controller for roll, pitch, and yaw. Our quadrotor is

similar to the one proposed by He et al. [5] and consists of the following components: an Hokuyo-URG miniature laser sensor for SLAM and obstacle avoidance (1), an XSens MTi-G MEMS inertial measurement unit (IMU) for estimating the attitude of the vehicle (2), a Linux-based Gumstix embedded PC with USB interfaces and a WiFi network card which communicates with the micro- controller on the quadrotor via an RS-232 interface (3), and amirror which is used to deflect some of the laser beams along the z direction to measure the distance to the ground (4).

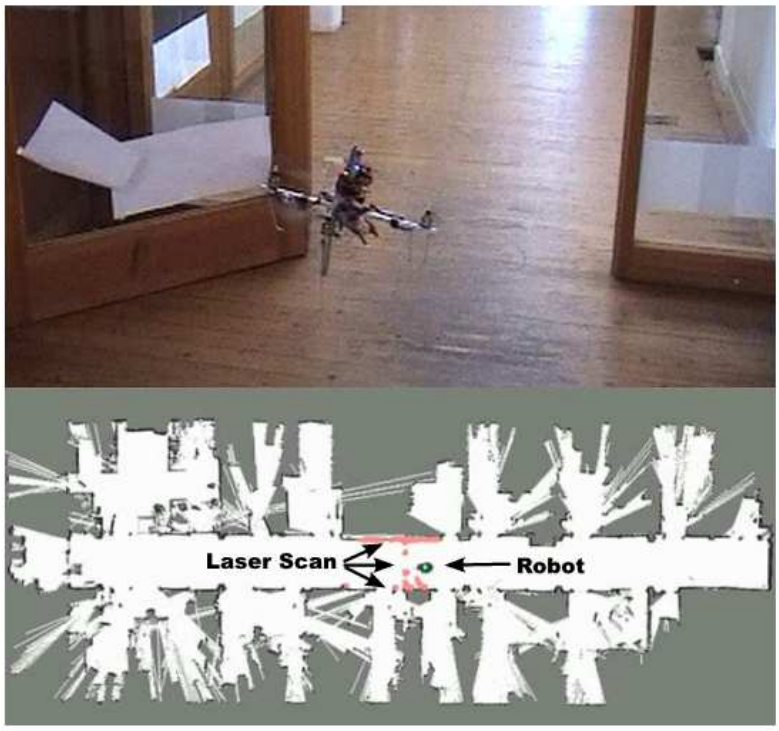


Fig. 1. Our Autonomous quadrotor during a mission (top) and position of the vehicle estimated on-line during the flight by our localization algorithm. during testing. Whereas a failure in the stabilization of the pose of a ground robot may result in a crash, the user can typically stop the system in time. In case of a flying vehicle this operation is not possible: stopping causes its fall on the ground. In this paper, we describe an autonomous quadrotor based on an open-source hardware project, namely the Mikrokopter [6]. Our system is a result of an integrated hardware/software design which meets several of the challeng-ing constraints imposed by the limited payload of the platform while preserving large degree of flexibility for future extensions. Figure 1 shows the Mikrokopter equipped with our navigation system during a mission. 2 Related Work In the last decade, flying platforms received an increasing attention from the research community. Many authors focused on the modeling and on the control of these vehicles [7, 8, 9, 10], with a particular emphasis on small helicopters. Hoffmann et al. [10] present a model-based algorithm for autonomous

flying with their STARMAC-quadrotor. Their system flies outdoors and utilizes GPS Index Terms—UAV, Quadrotor, SLAM, Navigation N recent years, the robotics community has shown an increasing interest in autonomous aerial vehicles, espe- cially quadrotors. Low-cost and small-size flying platforms are becoming broadly available and some of these platforms are able to lift relatively high payloads and provide an in- creasingly broad set of basic functionalities. This directly raises the question of how to equip them with autonomous navigation abilities. Whereas most of the proposed approaches for autonomous flying [9], [11] focus on systems for outdoor operation, vehicles that can autonomously operate in indoor environments are envisioned to be useful for a variety of applications including surveillance and search and rescue [12]. In such settings and compared to ground vehicles, the main advantage of flying devices is their increased mobility.

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}:=(x,y,\psi)}{\operatorname{argmax}} \; p(\mathbf{x}_t \mid \mathbf{x}_{t-k:t-1}, \mathbf{b}_{t-k:t}). \qquad (1)$$

To solve Equation (6), we use a variant of the multi-resolution correlative scan matcher proposed by Olson [6]. The idea behind a correlative scan-matcher is to discretize the search space xt = (xt, yt, ψt) and to perform an exhaustive search

approach →	4 cm	2 cm	1 cm	weighted mean	unit
mean(x)	0.107	0.105	0.149	0.066	$[m]$
mean(y)	-0.045	0.060	-0.04	-0.05	$[m]$
std(x)	0.145	0.148	0.165	0.123	$[m]$
std(y)	0.081	0.088	0.087	0.076	$[m]$
mean($\lvert v_x \rvert$)	0.146	0.095	0.084	0.075	$[m/s]$
mean($\lvert v_y \rvert$)	0.159	0.106	0.09	0.072	$[m/s]$
std($\lvert v_x \rvert$)	0.118	0.071	0.065	0.058	$[m/s]$
std($\lvert v_y \rvert$)	0.117	0.083	0.072	0.057	$[m/s]$

TABLE I
EFFECT OF MATCHING ALGORITHM ON POSE STABILITY OF THE ROBOT.

The optimization approach is discussed in detail in [5], and an open source version is available on OpenSLAM [13].

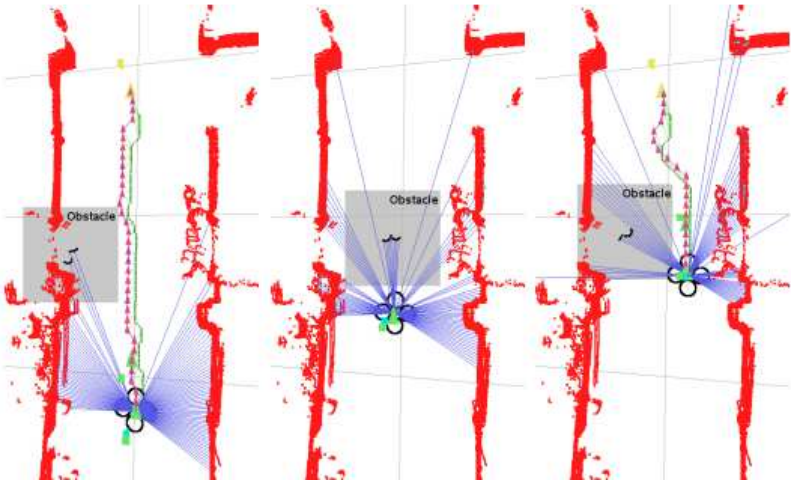


Fig. 2. Experiment for path planning and dynamic obstacle avoidance. The quadrotor is given a goal point 5m in front of it. The planned path is shown in the left image. A person enters the corridor (shaded area) and blocks the robot's path, which results in an invalid plan. The quadrotor therefore hovers around the last valid way point (second image). In the third image the person moved back leaving the quadrotor enough space for a de-tour. space to fly around the person in the second image, there is no valid plan due to the safety margins around the walls.

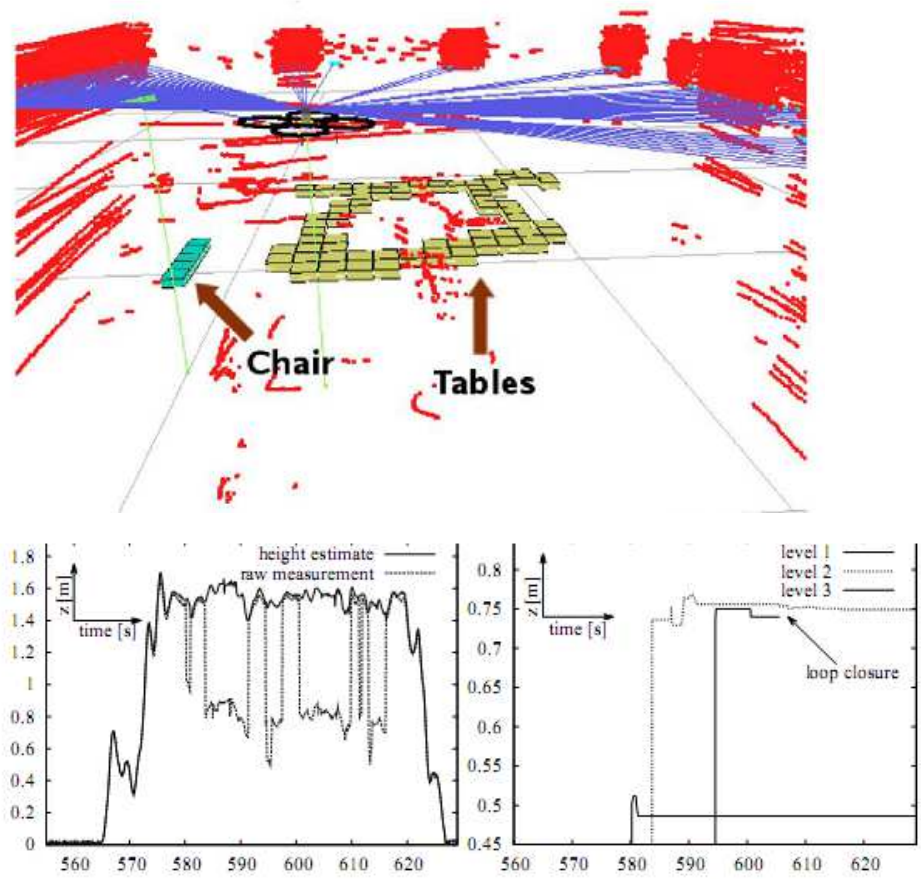


Fig. 3. Estimation of the global height of the vehicle and the underneath floor level. Whenever the quadrotor moves over a new level, the innovation is used to determine a level transition. The estimate of the height of each level is refined whenever the robot reenters that particular level. Top: the office environment. Middle: the corresponding map after autonomously flying over the vertical objects with a desired altitude of 150 cm. Bottom left: a plot showing the estimated altitude of the vehicle over time versus the raw measurement. The corresponding estimated levels are depicted in the bottom right plot. Note, that Level 3 is merged with Level 2 after the loop closure.

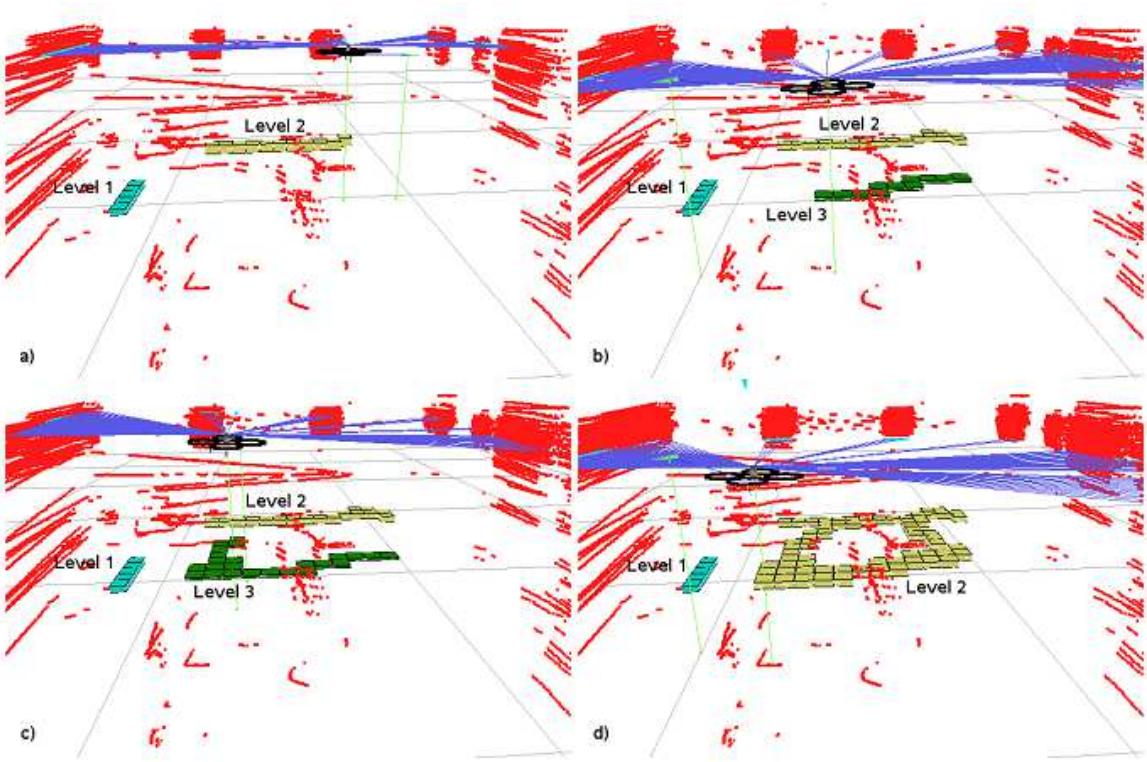


Fig. 4. Example of level joining during the estimation of the altitude of the vehicle and of the elevation of the underlying surfaces. Each level is represented

as a set of contiguous cells in the 2D grid that share the same elevation. The robot starts exploring an office environment. Initially it recognizes two levels(Level 1, and Level 2), corresponding to a chair and a table (a). Subsequently it flies away from the table, turns back and flies over a different region of the same table (a). This results in the creation of the new Level 3. Then the robot keeps on hovering over the table until it approaches the extent of Level 2 which has the same elevation of Level 3, being originated by the same table. This situation is shown in (c). Finally the robot enters Level 2 from Level 3.Our system recognizes these two Levels to have the same elevation. Accordingly it merges them and updates the common elevation estimate (d).C. Altitude Estimation Estimating the altitude of the vehicle in an indoor envi-ronment means determining the global height wrt. a fixed reference frame. Since the vehicle can move over non-flat ground, we cannot directly use the the beams h deflected by the mirror. Our approach therefore concurrently estimates the altitude of the vehicle and the elevation of the ground under the robot. In our estimation process, we assume that the (x, y, ψ) position of the robot in the environment is known from the SLAM module described above. We furthermore assume that the elevation of the surface under the robot is piecewise constant. We call each of these connected surface regions having constant altitude a “level”. The extent of each level is represented as a set of cells in a 2D grid sharing the same altitude. Since our system lacks global altitude

sensors like barome- ters or GPS to determine the altitude of the vehicle, we track the altitude of the vehicle over the ground and map different elevations by using a two-staged system of Kalman filters. Algorithm 1 describes our approach in an abstract manner. In the first stage, a Kalman filter is used to track the altitude z and the vertical velocity vz of the vehicle by combining inertial measurements, altitude measurements and already mapped levels under the robot. In the second stage, a set of Kalman filters is used to estimate the elevation of the levels currently measured by the robot. To prevent drifts in the elevation estimate, we update the altitude of a level only when the robot measures the level for the first time or whenever the robot reenters it (i.e., enters or leaves that particular level).In detail, the first Kalman filter estimates the height statez = (z, vz) and the corresponding uncertainty Σz. First, we predict the current altitude and velocity (ˆ zt) given the previous estimate, zt−1,Σzt−1 , and the acceleration measured by the IMU (see line 4 of Algorithm 1).The beams deflected by the mirror can measure more than one level simultaneously. For instance, when flying over a table it can happen that one fraction of the beams is fully reflected by the table, some beams are partially reflected by the table and partially by the floor, whereas the remaining beams are fully reflected by the floor. We therefore search in the local vicinity of the current multilevel-map for all levels which could have generated one of the measured altitudes h ∈ ht (assuming the robot's altitude is ˆ zt). This step is indicated in line 5.

$\langle h, \sigma^2 \rangle$ of the joint level has the following values:

$$\left\langle h = \frac{\sigma_k^2 h_j + \sigma_j^2 h_k}{\sigma_j^2 + \sigma_k^2}, \qquad \sigma^2 = \frac{\sigma_j^2 \sigma_k^2}{\sigma_j^2 + \sigma_k^2} \right\rangle .$$

This step is indicated in line 17 of Algorithm 1.

Algorithm 1 Multilevel-SLAM

Input: beams deflected by mirror at time t: $\mathbf{h}_t$
Input: previous multilevel map: $\hat{\mathbf{M}}$
Input: elapsed time: Δt
Input: current pose: $\mathbf{x}_t = (x_t, y_t)$ // output of SLAM module
Input: previous height state $\mathbf{z}_{t-1} = (z_{t-1}, v_{z_{t-1}})$
Input: previous height state uncertainty $\Sigma_{z_{t-1}}$
Input: z-acceleration and uncertainty: a_z, σ_z // from IMU
Output: current height state: $\mathbf{z}_t, \Sigma_{z_t}$
Output: current multilevel map: $\mathbf{M}$

1: **function** Multilevel-SLAM
2: // ———— 1st stage: update height estimate ————
3: // KF is short for Kalman Filter
4: $(\hat{\mathbf{z}}_t, \hat{\Sigma}_{z_t}) = \text{KF}(\mathbf{z}_{t-1}, \Sigma_{z_{t-1}}).\text{predictionStep}(\Delta t, a_z, \sigma_z)$
5: $\mathbf{E} = \hat{\mathbf{M}}.\text{at}(\mathbf{x}_t \pm \Delta\mathbf{x}).\text{getExistingLevelsMatching}(\mathbf{h}_t, \hat{\mathbf{z}}_t)$
6: **if** $\mathbf{E} \neq \emptyset$ **then**
7: $(\tilde{m}, \tilde{\sigma}_m) = \text{createVirtualHeightMeasurement}(\mathbf{h}_t, \mathbf{E})$
8: $(\mathbf{z}_t, \Sigma_{z_t}) = \text{KF}(\hat{\mathbf{z}}_t, \hat{\Sigma}_{z_t}).\text{measurementUpdate}(\tilde{m}, \tilde{\sigma}_m)$
9: **else**
10: $(\mathbf{z}_t, \Sigma_{z_t}) = (\hat{\mathbf{z}}_t, \hat{\Sigma}_{z_t})$
11: **end if**
12: // ———— 2nd stage: update map ————
13: $\mathbf{L} = \text{estimateLevels}(\mathbf{h}_t, \mathbf{z}_t)$
14: $\mathbf{M} = \hat{\mathbf{M}}.\text{addNewLevels}(\mathbf{L}, \mathbf{x}_t)$
15: $\mathbf{M} = \mathbf{M}.\text{updateExistingLevels}(\mathbf{L}, \mathbf{x}_t)$
16: $\mathbf{M} = \mathbf{M}.\text{extendExistingLevels}(\mathbf{L}, \mathbf{x}_t)$
17: $\mathbf{M} = \mathbf{M}.\text{searchForLoopClosures}(\mathbf{x}_t)$
18: return $\mathbf{z}_t, \Sigma_{z_t}, \mathbf{M}$
19: **end function**

To summarize, we store a level as a set of 2D grid cells representing the area covered by the corresponding object. First, we estimate the current height of the robot given the known levels in the multi-level map. In a second step we update the map, given the estimated altitude of the robot. Here, a level is constantly re-estimated whenever the vehicle enters or leaves this specific level, and the data association is resolved by the known (x, y, ψ) position of the vehicle. Finally, measurements not explained by any level present in the map are assumed to be generated by new levels which are then included in the map. D. High-Level Control for Pose and Altitude The high level control algorithm is used to keep the vehicle in the current position. The output of the control algorithm are variations in the roll, pitch, yaw, and thrust, denoted respectively as uφ, uθ, uψ and uz. The input are the position and the velocity estimates coming from incremental scan- matching. A variation of the roll translates in a motion along As for ground vehicles, the main task for an autonomous flying robot consists in reaching a desired location in an unsupervised manner, i.e., without human interaction. In the literature, this task is known as navigation or guidance. To address the general task of navigation one is required to tackle a set of problems ranging from state estimation to trajectory planning. Several effective systems for indoor and outdoor navigation of ground vehicles are nowadays available [14], [2]. E.

Path Planning and Obstacle Avoidance The goal of the path planning module is to compute a path from the current location to a user specified goal location which satisfies one or more optimality criteria and is safe enough to prevent collisions even in the case of small disturbances. Safety is usually enforced by choosing a path that is sufficiently distant from the obstacles in the map. Finally, due to the increased degrees of freedom of a flying vehicle compared to a ground robot, the path should be planned in 4DOF space instead of 3DOF. In our system we use D* lite [15], a variant of the A∗ algorithm that can reuse previous solutions to correct an invalid plan rather than recomputing it from scratch. Since directly planning in 4DOF is too expensive for our system, we compute the path in two consecutive steps. First, we use D* lite to compute a path in the x − y − z space, but we only consider actions that move the robot in the 2D space x − y. For each (x, y) location we know from the multi-level map the elevation of the surface underneath the robot. This known elevation is used to Recently, Celik et al. [16] presented their system for indoor simlutaneous localization and mapping (SLAM) using a monocular camera and ultrasound. Our work is orthog- onal to a recent work of Bachrach et al, [17] where the authors present a system for performing autonomous exploration and map acquisition in indoor environments. They extend the 2D robot navigation toolkit CARMEN [18] by adding a Rao- Blackwellized particle filter for SLAM and an algorithm for frontier-based autonomous exploration

References

Physics Consulting : Mojtaba Mahmoodzadeh Pirwahshi Faculty member, Islamic Azad University, Saravan Branch

Structural Consultant : Manzoor Ahmed Dhvary Dehaki Faculty member, Islamic Azad University, Saravan Branch

Structural Consultant : Hamed Makhdomi Darmian Faculty member, Islamic Azad University, Saravan Branch

Mathematics Consultant : Professor doctor Azizola Nosrat Faculty member, Islamic Azad University, Saravan Branch

[1] Carmen, http://carmen.sourceforge.net/.

1- S. Ahrens, D. Levine, G. Andrews, and J.P. How. Vision-based

guidance and control of a hovering vehicle in unknown, GPS-denied

environments. In Proceedings of the IEEE international conference on

robotics and automation, pages 2643–2648, 2009.

2- N.G. Johnson. Vision-assisted control of a hovering air vehicle in an

indoor setting. Master thesis, Brigham Young University, 2008.

3- C. Kemp. Visual control of a miniature quad-rotor helicopter. PhD, Churchill College University of Cambridge, 2005.

4- 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 (EMAV), 2007.

5- Carmen, http://carmen.sourceforge.net/.

6. Mikrokopter, http://www.mikrokopter.de/.

7- P. Pounds, R. Mahony, and P. Corke. Modelling and Control of a Quad-Rotor Robot. Proceedings of the Australasian Conference on Robotics and Automation (ACRA), 2006

8- A. Tayebi and S. McGilvray. Attitude stabilization of a four-rotor aerial robot. 43rd IEEE Conference on Decision and Control (CDC), 2, 2004.

9- A. Tayebi and S. McGilvray. Attitude stabilization of a VTOL quadrotor aircraft. Control Systems Technology, IEEE Transactions on, 14(3):562–571, 2006.

10-E. Altug, J.P. Ostrowski, and R. Mahony. Control of a quadrotor helicopter using visual feedback. In Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), 2002.

11- S. Thrun, M. Diel, and D. Hahnel. Scan Alignment and 3-D Surface Modeling with a Helicopter Platform. Field and Service Robotics (STAR Springer Tracts in Advanced Robotics), 24:287–297, 2006.

12- S. Bouabdallah, C. Bermes, S. Grzonka, C. Gimkiewicz, A. Brenzikofer, R. Hahn, D. Schafroth, G. Grisetti, W. Burgard, and R. Siegwart.

13- OpenSLAM - Open Source Navigation Software Repository, http://www.openslam.org.

14- A. Coates, P. Abbeel, and A.Y. Ng. Learning for Control from Multiple Demonstrations. Proceedings of the International Conference on Machine Learning (ICML), 2008.

14- S. Grzonka, G. Grisetti, and W. Burgard. Towards a navigation system for autonomous indoor flying. In Proc. IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, 2009.

15- S. Koenig and M. Likhachev. Fast replanning for navigation in unknown terrain. IEEE Transactions on Robotics, 21(3):354–363, 2005.

16- K. Celik, S.J. Chung, M. Clausman, and AK Somani. Monocular vision SLAM for indoor aerial vehicles. In Intelligent Robots and Systems (IROS), 2009., pages 1566–1573. IEEE, 2009.

17- A. Bachrach, R. He, and N. Roy. Autonomous Flight in Unknown Indoor Environments. Intl. Journal of Micro Air Vehicles, 1(4):217–228, 2009.

18- N. Roy, M. Montemerlo, and S. Thrun. Perspectives on standardization in mobile robot programming. In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 2003.

19- Towards palm-size autonomous helicopters. In International Conference and Exhibition on Unmanned Areal Vehicles (UAV), Dubai, UAE, 2010.

20- ROS - Robot Open Source, http://www.ros.org.