**2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011**

A Compact Exploration Strategy for Indoor Flight Vehicles

Chintasid Pravitra, Girish Chowdhary and Eric Johnson

Abstract— **This paper presents a compact exploration strategy designed to be implemented onboard indoor Miniature Air Vehicles (MAVs) operating in cluttered and confined environments. The exploration strategy uses 2D range information from a laser range scanner to generate velocity commands by blending a Sensor-based Random Tree frontier planner with a wall-following velocity field generator. The combined approach leverages the efficient exploration capabilities of frontier-based guidance and ensures that the vehicles follows a path that is free of obstacles and conducive to maintaining good scan geometry through a wall-following approach. The strategy has been successfully implemented and tested on a Quadrotor MAV and simulation results are presented. Flight test results shall be included in the final version.**

I. INTRODUCTION

A. Motivation

Indoor Miniature Air Vehicles (MAVs) can be used in various potential applications, including search and rescue, disaster assessment, reconnaissance, or other tasks that would be risky or impossible for a human to perform. These applications often require MAVs to enter unmapped buildings, and explore other cluttered and confined environments. Indoor flight in such environments pose significant challenges due to unreliability of radio links, unavailability of *a-priori* knowledge, and unavailability of external navigational aids, such as Global Positioning System (GPS). As a result, MAVs capable of exploring indoor environment should be self-contained, and capable of running all guidance, navigation, and control softwares onboard. Furthermore, indoor aerial vehicles operating in GPS denied environments often rely on Simultaneous Localization and Mapping (SLAM) algorithms to solve the navigation problem, in which local geometry is used to map the surrounding as well as bound the drifts of inertial sensors. Hence, unlike autonomous ground vehicles, the navigation and guidance problems for an MAV can be tightly coupled. Position estimate from SLAM (navigation) solution may quickly diverge if the vehicle encounters featureless scan geometries. Therefore, as part of flight safety consideration, guidance strategy must be designed such that the vehicle always "sees SLAM-favored" scan geometry [1], [2], [3], [4].

This work focuses on developing an exploration (guidance) strategy compact enough to run onboard MAV. The strategy can be superimposed with any low computational cost Simultaneous Localization and Mapping (SLAM) algorithm such as [1] or [5] to create a totally self-contained autonomous air vehicle. The developed guidance strategy is expected to meet the following requirements:

- achieve efficient exploration
- perform computations onboard using low-cost and light-weight processor
- maintain navigable scan geometry.

It should be noted that onboard SLAM, guidance, and control problems have been well-studied for ground robots. However, in extending these methods to MAVs, significant new challenges due to weight constraints, six degree of freedom dynamics, and the agility of MAVs must be overcome. In fact, many well-known optimal guidance and path planning techniques are not feasible for onboard implementation on MAVs because of limited computational power. As a result, in contrast to many published works on guidance and path planning, our approach trades-off optimality of exploration with simplicity and safety of the vehicle.

B. Previous Work

SLAM and autonomous exploration for MAV application was first demonstrated by Achtelik et. al. [6]. Achtelik et. al. uses frontier-based exploration with goal-driven dynamic programming trajectory generation. However, such navigation and guidance algorithms are computationally expensive and have to be run from ground station. The setup promotes a very effective exploration but is still subject to failure should communication link between ground station and vehicle is lost. Sobers et. al. proposed a totally self-contained MAV architecture with a very compact SLAM algorithm and a simple wall following guidance strategy[1]. Fig. 1 shows example of map created from SLAM algorithm implemented in [1]. However, behavior of wall following guidance is highly dependent on scan geometry and can be very inefficient and unpredictable in various cases. In some cases, the method may only tracks the outermost walls of the building and leaves inner rooms completely unexplored. In worse cases with unfavorable geometries of the building's entrance, the method may commands the vehicle to leave the building as soon as the vehicle had entered the building. Our goal is to improve upon efficiency of such guidance algorithms by introducing frontier-based techniques, but yet keeping the algorithm compact and SLAM-favored.

In effort to promote efficiency of exploration, we apply the basic principle of frontier-based exploration in a novel way. Frontier-based exploration was first introduced by Yamauchi [7] as an effective way for a mobile robot to explore an unknown environment. Without frontier-based exploration,

C. Pravitra, G. Chowdhary and Assoc. Prof. E. N. Johnson are with the Daniel Guggenheim school of aerospace engineering at the Georgia Institute of Technology, Atlanta GA, `cpravitra3, Girish.Chowdhary@gatech.edu, Eric.Johnson@ae.gatech.edu`

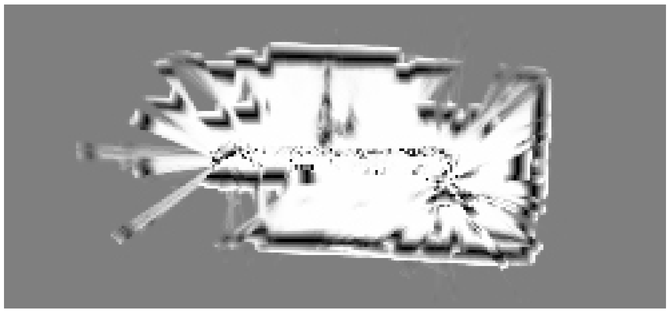


Fig. 1. SLAM map

a robot may have to explore an unknown environment randomly with some form of obstacle avoidance logic. The principle of frontier-based exploration is : *"try to get as much new information as possible by going to a boundary between explored and unexplored territory"*. Various forms of frontier-based exploration strategy have been developed, most of which require some form of global map in order to find frontiers and plan trajectories[7], [8], [9]. A global map can be grid-based, feature-based, or polygonal-based. However, such global map and its essential guidance algorithms are not computationally practical onboard a MAV.

Rather than using a global map, Freda and Oriolo applied the principle of frontier-based exploration to a data structure called Sensor-Based Random Tree (SRT) [10], [11]. In this research, we use SRT method called SRT-Star to store frontiers, store safe-regions, and sequence new waypoints [10]. The SRT-Star is blended with wall following algorithm as will be described in Section V and Section IV.

The strategy has been implemented and flown on Georgia Tech's quadrotor while participating in the 2010 International Aerial Robotics Competition (IARC) [12], [13].

The paper is organized as follows. First, we discuss the main algorithm of how frontier contribution and wall following contribution are combined. Then, wall following contribution and frontier contribution are each discussed separately in detail. Finally, we present results of the proposed strategy.

II. WALL FOLLOWING - FRONTIER GUIDANCE

This section describes overall wall following-frontier exploration strategy. Assuming near hover dynamics and semi-structured environment, we first recognize that three-dimensional guidance problem can be simplified by decoupling horizontal plane guidance and altitude guidance. Altitude guidance simply commands constant altitude above ground level while horizontal guidance commands horizontal velocity and heading.

The main idea is overall velocity command ($\vec{v}_{cmd}$) is composed of contribution from frontier velocity ($\vec{v}_{fr}$) and contribution from wall following velocity ($\vec{v}_{wf}$).

$$\vec{v}_{cmd} = \vec{v}_{wf} + \vec{v}_{fr} \tag{1}$$

Heading command (ψ_{cmd}), on the other hand, is solely generated from frontier contribution. We use raw scan data from laser range scanner to create SRT and ultimately compute velocity and heading commands. The commands are updated as soon as new scan data is available. For Hokuyo URG-04LX laser scanner [14] used in our flight vehicle, this update rate is approximately 10 Hz. Algorithm 1 illustrates sequence of commands that are executed at a particular update time step.

Algorithm 1 Compute Velocity Command ($\vec{v}_{cmd}$) and Heading Command (ψ_{cmd})

```
Require: $\vec{x}$, $\vec{x}_{waypoint}$, $scan$, $SRT$, and $d$
 1: $\vec{v}_{wf} \Leftarrow getWallFollowingVelocity(scan)$
 2: if $||\vec{x} - \vec{x}_{waypoint}|| < d$ then
 3:     $newFrontier \Leftarrow frontierSearch(scan)$
 4:     $SRT \Leftarrow updateSRT(SRT, newFrontier)$
 5:     if $frontierExist(SRT)$ then
 6:         $\vec{x}_{waypoint} = newWaypoint(SRT)$
 7:     else
 8:         $\vec{x}_{waypoint} = previousWaypoint(SRT)$
 9:     end if
10: end if
11: $\vec{v}_{fr} \Leftarrow getFrontierVelocity(\vec{x}, \vec{x}_{waypoint})$
12: $\psi_{cmd} \Leftarrow getHeading(\vec{x}, \vec{x}_{waypoint})$
13: $\vec{v}_{cmd} \Leftarrow \vec{v}_{fr} + \vec{v}_{wf}$
```

The main algorithm requires vehicle's current position ($\vec{x}$), position of commanded waypoint ($\vec{x}_{waypoint}$), raw scan data ($scan$), sensor based random tree structure (SRT), and threshold distance (d).

The algorithm begins by computing wall following velocity ($\vec{v}_{wf}$) directly from scan data. The algorithm then checks whether the vehicle has arrived at the commanded waypoint ($\vec{x}_{waypoint}$); if the vehicle has not arrived the commanded waypoint, the commanded waypoint will not be modified. If the vehicle has arrived at the commanded waypoint, a new commanded waypoint is generated from the frontier planner (line 3 through line 8). The vehicle is said to have arrived at the commanded waypoint when it is within distance d of the commanded waypoint. Finally, commanded waypoint and vehicle's position are used to generate heading command (ψ_{cmd}) and frontier velocity ($\vec{v}_{fr}$). Notice that although commanded waypoint may not change at a particular time step, frontier velocity and heading change at every time step because vehicle position changes.

Details of each velocity contribution and heading command are discussed in Section III and Section IV.

III. WALL FOLLOWING CONTRIBUTION

This section discusses how wall following velocity (v_{wf}) is generated. Wall following velocity is necessary because it serves two very important purposes. First, it acts as an obstacle avoidance routine, and second, it promotes navigable scan geometry. We use velocity field approach where each scan reading produces its own incremental velocity command. All incremental velocity commands are summed to produce the total velocity command.

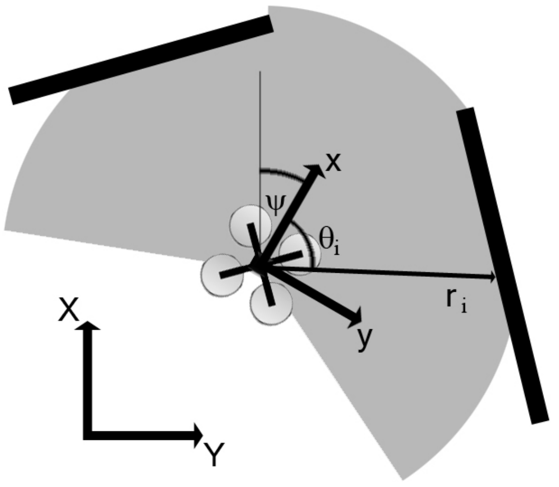


Fig. 2. Reference frame description of vehicle and scan points

Let r_i denote the i^{th} scan range reading shown in Fig.2. Incremental velocity command (v_{ri}) in radial direction due to each scan reading is obtained from:

$$v_{ri} = \begin{cases} K_{wf}(r_i - r_t) & \text{if } r_i \geq r_{safe}; \\ K_{safe}(r_i - r_t) & \text{if } r_i < r_{safe}; \end{cases} \tag{2}$$

where K_{wf} and K_{safe} are gains used for nominal and safety cases as determined by safe radius r_{safe}. r_t is a user specified parameter representing distance from walls that the vehicle tries to maintain.

Let θ_i shown in Fig. 2 denote angle of the ith scan with respect to body frame ($\mathbf{xy}$), and let n denote number of in-range scans points. Velocity command in body frame u_{cmd} and v_{cmd} results from projecting and summing each incremental velocity.

$$u_{cmd} = \sum_{i=1}^{n} v_{ri} cos\theta_i \qquad v_{cmd} = \sum_{i=1}^{n} v_{ri} sin\theta_i \tag{3}$$

Let ψ be the heading angle referenced from an arbitrary chosen inertial frame ($\mathbf{XY}$). Velocity command in body frame can be converted to inertial frame using (4).

$$\vec{v}_{wf} = \begin{bmatrix} v_{x_{wf}} \\ v_{y_{wf}} \end{bmatrix} = \begin{bmatrix} cos\psi & -sin\psi \\ sin\psi & cos\psi \end{bmatrix} \begin{bmatrix} u_{cmd} \\ v_{cmd} \end{bmatrix} \tag{4}$$

Let us elaborate more on choice of r_t. Equation (2) can be thought of as multiple proportional controllers that try to make all scan readings equal to r_t (without commanding heading change). In practice, some incremental velocities cancel out. The overall effect is attraction to a wall if the vehicle is too far away and repulsion from a wall if the vehicle is too close. In most cases where multiple walls are visible, which wall the vehicle will follow depends on scan geometry. The vehicle is attracted to long wall segments located further away more than short wall segments located near by.

In effort to improve obstacle avoidance behavior, two separate gains are used to generate incremental velocities. The gain K_{wf} represents nominal gain for typical attraction and repulsion to walls, while the gain K_{safe} represents safety gain used only when the vehicle is dangerously close to a wall. The safety gain is selected to be orders to magnitude higher than the nominal gain. Additionally, one also has to set K_{safe} high enough to ensure that obstacle avoidance velocity overpowers frontier velocity. Note that "dangerously close to a wall" is defined by safe radius parameter (r_{safe}) and should always be set to a value smaller than r_t.

We note that this particular wall following law is designed to work together with another guidance contribution. Our strategy uses frontier velocity, but in general, this simple wall following law can be applied along with any goal-specified guidance algorithm. The wall following alone may not work by itself in all situations. It does not provide heading command. Depending on local scan geometry, it may also suffer from a local minima and the vehicle may "get stuck" at a particular location of a room.

Although one purpose of wall following is to promote navigable scan geometry, it should be reminded that wall following does not absolutely guarantee continuous navigable scan. Even with wall following, navigation may not be possible at certain unfavorable geometries. These geometries include long hallways where scan is almost identical regardless of vehicle motion, or a very large room where not enough scan readings are returned for localization.

IV. FRONTIER CONTRIBUTION

Frontier-based approach is essential for efficient exploration in an unknown indoor environment. As mentioned earlier, we seek frontier-based algorithm that requires as low computational power as possible. This work adapts and extends a very compact Sensor-Based Random Tree (SRT) frontier method called SRT-Star developed by Freda et. al. [10].

SRT-Star, outlined in line 2 to line 10 of Algorithm 1, takes advantage of 2D laser range scanners' characteristics. Upon arriving at a waypoint, SRT-Star divides laser scan beams into sectors, where each sector contains left-point, mid-point, and right-point. Mid-point is declared as a frontier if the sector is completely obstacle free while left-point and right-point are declared as frontiers if there are large discontinuities between adjacent sectors. Sectors and frontier points are illustrated in Fig. 3. If at least one frontier exists, a new commanded waypoint is generated by randomizing a point inside a sector that possesses at least one frontier. If no frontier exists, the vehicle backtracks to the previous waypoint.

Sectors and frontiers are stored in a tree-like structure with waypoints as nodes and frontiers as potential branches. Since regions inside the sectors are obstacle free by scanner's characteristic, obstacle free regions also expand as SRT evolves. Expansion of free regions implies that some existing frontiers may fall under newly acquired sectors. In that case, SRT-Star will remove the frontier in order to avoid unnecessary exploration. Details of SRT-Star are discussed in [10].

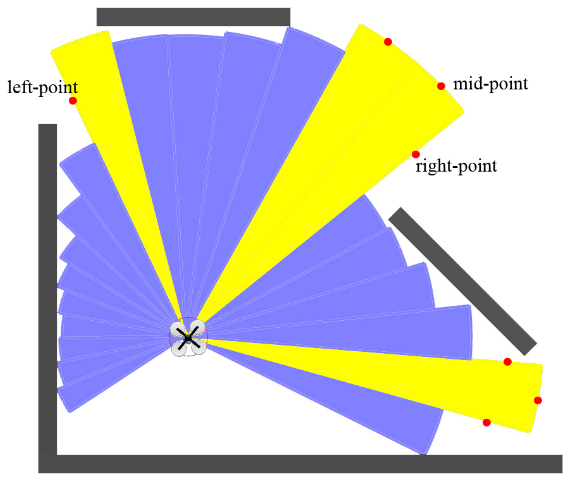


Fig. 3. SRT-Star divides laser scan into sectors

The difference between the original SRT-Star and our method is: the original SRT-Star feeds commanded waypoint directly to the controller. Our method as outlined in Algorithm 1 calculates frontier velocity and heading commands from commanded waypoint. It then blends frontier velocity command with wall following velocity, and finally feeds velocity and heading commands to the controller.

Let K_{fr} be a frontier gain specified by user. Let $x_{waypoint}$ and $y_{waypoint}$ be position of the commanded waypoint in inertial frame as obtained from SRT-Star. Let x and y be the vehicle's current position in inertial frame. Given the commanded waypoint, a simple proportional feedback law shown in (5) is used to obtain frontier velocity.

$$\vec{v}_{fr} = \begin{bmatrix} v_{x_{fr}} \\ v_{y_{fr}} \end{bmatrix} = \begin{bmatrix} K_{fr}(x_{waypoint} - x) \\ K_{fr}(y_{waypoint} - y) \end{bmatrix} \tag{5}$$

Moreover, the vehicle's heading will always be commanded to point directly to the commanded waypoint.

$$\psi_{cmd} = atan2(y_{waypoint} - y, x_{waypoint} - x) \tag{6}$$

Note that this heading command also provides another benefit. It steers the scan scope to see more walls in neighborhood of the commanded waypoint. Wall following velocity will command attraction to these walls. Therefore, wall following velocity also implicitly brings the vehicle towards the commanded waypoint.

V. PRACTICAL CONSIDERATIONS

This section addresses additional practical considerations required for successful flight. Besides from wall following-frontier strategy explained in Section , III, and IV, the following items are also implemented to ensure vehicle's safety:

- *Speed limit*: Speed limit is necessary as our approach relies on summing several incremental velocities. The total velocity magnitude is very difficult to predict. It depends upon scan geometry and distance to the commanded waypoint. Without speed limit, the vehicle may unsafely command high speed as it observes distant walls and waypoints.
- *Yaw rate limit*: When a new commanded waypoint is acquired, yaw rate limit ensures that the vehicle does not turn too fast. This behavior is most profound when vehicle recognizes that no local frontier exist and starts to backtrack. High yaw rate should be avoided as it may reduce quality of SLAM.
- *Waypoint-to-waypoint time limit*: We impose traveling time limit between one waypoint to another. It should be reminded that SRT-Star assumes perfect navigation. However, real operations may suffer from slow drift of inertial frame due to imperfections, or worse, temporary loss of SLAM due to unfavorable geometry. In such cases, commanded waypoint as erroneously interpreted by SLAM may no longer be obstacle free. Time limit ensures that the vehicle will not "get stuck" trying to reach an unreachable waypoint. If time limit is exceeded, the SRT is erased and a new exploration is initiated.

VI. SIMULATION RESULTS

We apply wall following-frontier guidance strategy to a quadrotor shown in Fig. 4 developed at Georgia Tech UAV Research Facility (GTUAVRF). The vehicle is designed to accomplish the 6^{th} mission of the International Aerial Robotics Competition (IARC)[12], [13]. The vehicle is based on AscTec Pelican quadrotor from Ascending Technologies GmbH. The vehicle structure, motors, and propellers of AscTec Pelican are used without modification. GTUAVRF then supplements the quadrotor with processors and sensors for autonomous operation. The sensors include Hokuyo URG-04LX laser range scanner [14], sonar range finder for altitude measurement, and inertial measurement unit. The processors include Gumstix Overo Fire running guidance, navigation, and control software, and ATMega128 running stability augmentation software. Details of the architecture is discussed in [12].

Our exploration strategy has been implemented and tested in a simulation software developed in-house at Georgia Tech UAV Research Facility. The software simulates full nonlinear vehicle dynamics, sensors with noise characteristics, and indoor structure. Fig. 5 shows a simulated flight of the proposed strategy. The quadrotor, quadrotor's trajectory, commanded waypoint, laser scan, active SRT, and the building are shown on Fig. 5. The vehicle is able to avoid obstacles while creating SRT and explores the building. In Fig. 5(c), the quadrotor is able to get pass through a small gap on the northwest corner without collision. The quadrotor then explores towards the east of the building but later recognizes that there is an unexplored room in the middle of the building. Fig. 5(f) shows result when the entire building is explored.

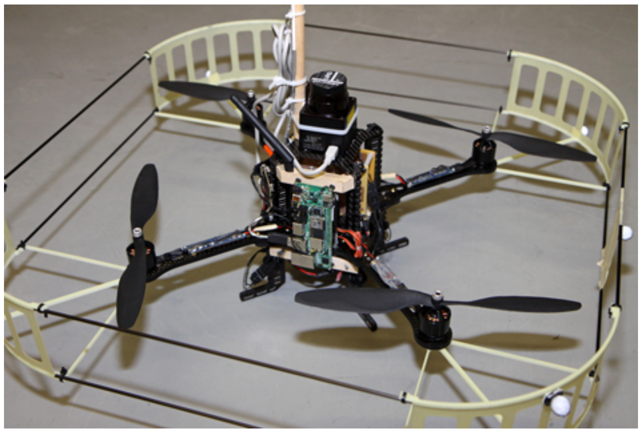


Fig. 4. Georgia Tech's quadrotor

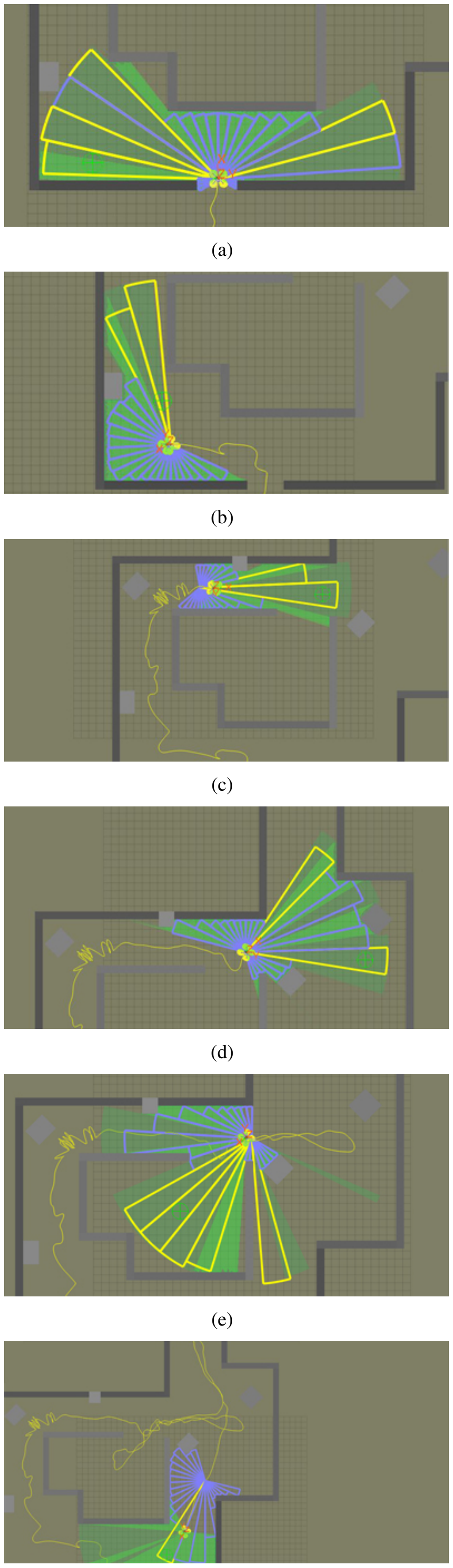
VII. Flight Test Results

A. Flight Test Results

In this section we present flight test results of the GTQ exploring an indoor cluttered environment fully autonomously without any external sensing aids (such as GPS). The onboard GNC algorithm does not assume any a-priori knowledge of the indoor environment. Navigation is performed by solving a SLAM problem online using techniques presented in [15]. Guidance is achieved by frontier-guidance type method coupled with wall-following guidance as described in this paper, and control is achieved using a dynamic inversion based linear control architecture. The flight test begins with the aircraft hovering at about 2.8 ft above the ground. The onboard guidance logic then commands waypoints that take the aircraft towards unexplored frontiers, the onboard navigation logic provides the aircraft with its pose information and simultaneously builds a map of the environment in real-time. The map information is fed back into the guidance logic to explore new frontiers. Note that all computation, including SLAM is performed online using onboard avionics. The GNC algorithms were optimized to execute completely onboard the embedded computer (Gumstix Overo Fire) used by trading-off map accuracy with guaranteeing a reliable instantaneous position fix for pose estimation. This trade-off results in a slight skew in the onboard generated map. However, the position accuracy was found acceptable for exploring indoor areas reliably.

VIII. Conclusions

We presented a compact and efficient exploration strategy intended for running onboard indoor Miniature Air Vehicles. Efficient exploration was achieved by augmenting a frontier based guidance strategy with wall-following logic. This approach ensured that the vehicle explores unknown indoor environments efficiently using a path that is free of obstacles and conducive to maintaining good scan geometry. We demonstrated the feasibility of the strategy onboard a low-cost off-the-shelf embedded computer on a Quadrotor MAV running six degree of freedom SLAM algorithm. While significant work remains to be done in guaranteeing



(a)

(b)

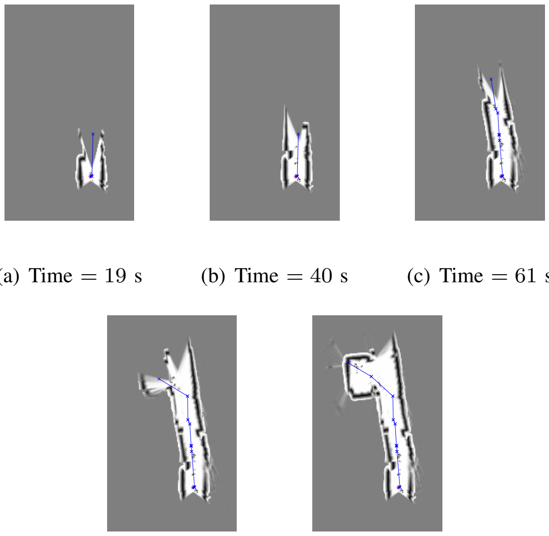
(c)

(d)

(e)

(f)

Fig. 5. Simulated Exploration in Indoor Environment



(a) Time = 19 s (b) Time = 40 s (c) Time = 61 s

(d) Time = 92 s (e) Time = 124 s

Fig. 6. The GTQ autonomously explores an unknown cluttered indoor environment without any external sensing or computational aids. The figures show the map of the unknown indoor environment generated by the onboard navigation algorithm as the GTQ explores the indoor environment. Stars mark the waypoints commanded by the guidance strategy, the guidance strategy ensures the GTQ explores unexplored areas of the indoor environment. The pictured area is approximately 50 by 80 feet. Note that all computation, including solving the SLAM problem, is performed onboard.

optimal performance, we conclude that this strategy can be used to create a totally self-contained miniature flight vehicle running guidance, navigation, and control algorithms onboard for autonomously exploring indoor environment. This conclusion is supported from simulation and flight test results.

IX. ACKNOWLEDGMENTS

The authors would like to thank the members of Georgia Tech Aerial Robotics team and the members of the Georgia Tech UAV Research Facility including D. Michael Sobers, Allen Wu, H. Claus Christmann, Hiroyuki Hashimoto, Chester Ong, Roshan Kalghatgi, Suresh Kannan, Nimrod Rooz, Jeong Hur, Shusaku Yamaura, Erwan Salaun, M. Scott Kimbrell, and John Ottander.

REFERENCES

[1] D. M. Sobers, S. Yamaura, and E. N. Johnson, "Laser-aided inertial navigation for self-contained autonomous indoor flight," in *Proc. AIAA Guidance, Navigation, and Control Conference (GNC'10)*, Toronto, Ontario, Canada, Aug. 2010.

[2] S. Yamaura, "Development of guidacne algorithm for unmanned indoor flight vehicle by using 2d laser scanner," unpublished.

[3] L. Romero, E. Morales, and E. Sucar, "An exploration and navigation approach for indoor mobile robots considering sensors perceptual limitations," in *Proc. IEEE International Conference on Robotics and Automation (ICRA01)*, Seoul, Korea, May 2001.

[4] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a gps-denied environment," in *Proc. IEEE International Conference on Robotics and Automation (ICRA08)*, Pasadena, California, May 2008.

[5] B. Steux and O. E. Hamzaoui, "Coreslam : a slam algorithm in less than 200 lines of c code," Mines ParisTech, Center for Robotics, Paris, France, Tech. Rep., 2009.

[6] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Autonomous navigation and exploration of a quadrotor helicopter in gps-denied indoor environments," in *Proc. of the 1st Symposium on Indoor Flight, International Aerial Robotics Competition*, 2009.

[7] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA97)*, Monterey, CA, July 1997, pp. 146–151.

[8] H. Gonzalez-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *Int. J. Robotics Research*, vol. 21, no. 10, pp. 829–848, 2002.

[9] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS02)*, 2002.

[10] L. Freda and G. Oriolo, "Frontier-based probabilistic strategies for sensor-based exploration," in *Proc. IEEE International Conference on Robotics and Automation (ICRA05)*, Barcelona, Spain, Apr. 2005.

[11] L. Freda, F. Loiudice, and G. Oriolo, "A randomized method for integrated exploration," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS06)*, Beijing, China, Oct. 2006.

[12] G. Chowdhary, D. M. Sobers, C. Pravitra, H. C. Christmann, A. D. Wu, H. Hashimoto, C. Ong, R. Kalghatgik, and E. N. Johnson, "Georgia tech team entry for the 2010 auvsi international aerial robotics competition," 2010.

[13] (2010) International aerial robotics competition. [Online]. Available: http://iarc.angel-strike.com/

[14] Y. Okubo, C. Ye, and J. Borenstein, "Characterization of the hokuyo urg-04lx laser rangefinder for mobile robot obstacle negotiation," in *SPIE*, vol. 7332, Orlando,FL, Apr. 2009, p. 10.

[15] G. Chowdhary, D. M. Sobers, C. Pravitra, C. Christmann, A. Wu, H. Hashimoto, C. Ong, R. Kalghatgi, and E. N. Johnson, "Integrated guidance navigation and control for a fully autonomous indoor uas," in *Guidance Navigation and Control Conference*. Portland, OR: AIAA, August 2011, submitted.