Robust Autonomous Flight in Constrained and Visually Degraded Environments

Zheng Fang[1,2], Shichao Yang[1], Sezal Jain[1], Geetesh Dubey[1], Silvio Maeta[1], Stephan Roth[1], Sebastian Scherer[1], Yu Zhang[1,3] and Stephen Nuske[1]

Abstract This paper addresses the problem of autonomous navigation of a micro aerial vehicle (MAV) inside a constrained shipboard environment for inspection and damage assessment, which might be perilous or inaccessible for humans especially in emergency scenarios. The environment is GPS-denied and visually degraded, containing narrow passageways, doorways and small objects protruding from the wall. This makes existing 2D LIDAR, vision or mechanical bumper-based autonomous navigation solutions fail. To realize autonomous navigation in such challenging environments, we propose a fast and robust state estimation algorithm that fuses estimates from a direct depth odometry method and a Monte Carlo localization algorithm with other sensor information in an EKF framework. Then, an online motion planning algorithm that combines trajectory optimization with receding horizon control framework is proposed for fast obstacle avoidance. All the computations are done in real-time onboard our customized MAV platform. We validate the system by running experiments in different environmental conditions. The results of over 10 runs show that our vehicle robustly navigates 20m long corridors only 1m wide and goes through a very narrow doorway (66cm width, only 4cm clearance on each side) completely autonomously even when it is completely dark or full of light smoke.

1 Introduction

Over the past few years, micro aerial vehicles (MAVs) have gained a wide popularity in both military and civil domains. Surveillance and reconnaissance is one area where they have made a huge impact. In this paper, we aim to develop a MAV

[1]Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA.
[2]Northeastern University, Shenyang, Liaoning, 110819, China.
[3]Zhejiang University, Hangzhou, Zhejiang, 310027, China.
e-mail: {zhengf,shichaoy,sezal,gdubey,basti}@andrew.cmu.edu

that is capable of autonomously navigating through a ship to aid in fire control, damage assessment and inspection, which might be dangerous or inaccessible for humans. Such a constrained and GPS-denied environment poses various challenges for navigating though narrow corridors and doorways, especially because it might be visually degraded: potentially dark and smoke-filled. An illustrative picture is shown in Fig. 1.

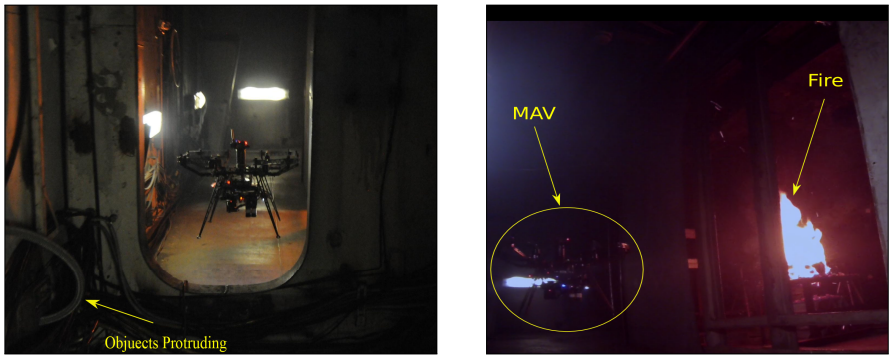


Fig. 1 Autonomous MAV for fire-detection inside a ship: The left picture shows MAV's autonomous flight through doorways. The right picture shows a testing scenario with fire.

For successful operation in such environments, we need to address several challenging problems. *First*, the MAV should be small enough to travel in the narrow corridors with narrower doorways (66cm width). Therefore, only lightweight sensors can be used, which provide limited measurement range and noisy data. *Second*, the onboard computational resources are very limited while every module should run in real-time, posing great challenges for pose estimation and motion planning. *Third*, since the practical environment is potentially a dark and smoke-filled environment, it prevents us from using state-of-the-art visual navigation methods. Though putting LED lights can give better illumination, it might not output a usable RGB image under smoky conditions. Besides, clear corridors with few geometric features or corridors with many small objects on the wall pose great difficulty for accurate pose estimation and obstacle avoidance. In addition, air turbulence from the MAV in confined spaces poses difficulty for precise control.

To address the above challenges, we build a robust and efficient autonomous navigation system with the following contributions.

- A real-time 6DoF pose estimation system that can directly recover the relative pose from a series of depth images and estimate the absolute pose of the MAV in a given 3D map.
- A data fusion framework of odometry and absolute pose with other sensors to provide fast and robust state estimation.
- An online motion planning algorithm using a modified trajectory optimization method under receding horizon control framework.

We demonstrate the effectiveness of our system through both simulation and field experiments. The field experiment is performed in a constrained shipboard environment containing a 20m long, 1m wide corridor and a 66cm wide doorway. The width of the vehicle is 58cm leaving only 4 cm clearance on both sides. We conducted more than 10 runs in various environment conditions, from normal to complete dark and smoke-filled environments to demonstrate autonomous navigation capabilities of the MAV.

2 Related Work

In recent years, a number of autonomous navigation solutions have been proposed for MAVs. Those solutions mainly differ in the sensors used for perception in the autonomous navigation problem, the amount of processing that is performed onboard/offboard and the assumptions made about the environment.

2D LIDAR has been extensively and successfully used for autonomous navigation for its accuracy and low latency [1–3]. However, those systems are usually only suitable for structured or 2.5D environments. Recently, there are also many vision-based navigation systems since cameras can provide rich information and have low weight, etc. For example, a stereo camera is used in [4] [5] and a monocular camera with IMU is used in [6–8], but vision is sensitive to illumination changes and could not work in dark or smoky environments. More recently, RGB-D cameras have become very popular for autonomous navigation of indoor MAVs [9–11] because they can provide both image and depth. For example, in [10] a RGB-D visual odometry method is proposed for real-time pose estimation of a MAV and a 3D map is created offline. In [11], a fast visual odometry method is used for pose estimation and 3D visual SLAM is used for constructing a 3D octomap in real-time.

Unfortunately, the existing autonomous navigation methods can not work in our case since our application environment is a confined, complex visually degraded 3D environment that may be very dark or filled with smoke. For example, for state estimation, vision-based methods [8, 10] could not work in our case due to that it is a potentially dark and smoky environment. Besides, for obstacle avoidance, 2D LIDAR-based methods are also unqualified for this complex environment since it only perceives planar information while there are many small objects (e.g. slim cables and pipes) protruding from the wall in our environment. In addition, many above papers' motion planning methods either compute paths offline [2] [11] or heavily rely on prior maps [1]. Some papers online generate steering angles to avoid obstacles by vector field histogram [5] or waypoints by sampling based planners (e.g. RRT*) [3]. However, steering angle is not suitable for precise control and RRT* path is usually not smooth and not fast enough.

In this paper, we present a robust autonomous navigation system that can work in challenging practical environments, which is based on our previous work [12]. However, our previous work only deals with the pose estimation problem while this paper presents all the details of the whole system. In our system, we mainly use

depth images for odometry estimation, localization and motion planning, which can work in completely dark or even light smoke-filled environments. Besides, all the components of the system run onboard on an ARM based embedded computer.

3 Approach

3.1 Real-time Pose Estimation

Pose estimation is required to allow the robot to be self aware of its placement in the surroundings and hence allows it to plan appropriate paths to maneuver around obstacles in the corridor.

3.1.1 Low-frequency Pose Estimation

Low frequency pose estimates are primarily based on the RGB-D sensor. This includes relative ego-motion of the robot calculated from depth images as well as the absolute pose of robot calculated from the point cloud and a given 3D map.

Relative Pose Estimation A direct method based on [12, 13] is used to calculate the relative pose estimation, which is much faster than state of the art ICP method [14]. Let a 3D point $R = (X,Y,Z)^T$(measured in the depth camera's coordinate system) be captured at pixel position $r = (x,y)^T$ in the depth image Z_t. This point undergoes a 3D motion $\Delta R = (\Delta X, \Delta Y, \Delta Z)^T$, which results in an image motion Δr between frames t_0 and t_1. Given that the depth of the 3D point will have moved by ΔZ, the depth value captured at this new image location $r + \Delta r$ will have consequently changed by this amount:

$$Z_1(r + \Delta r) = Z_0(r) + \Delta Z \tag{1}$$

This equation is called *range change constraint equation*.

For a pin hole camera model, any small 2D displacement Δr in the image can be related directly to the 3D displacement ΔR which gave rise to it by differentiating the perspective projection equation with respect to the components of the 3D position:

$$\frac{\partial r}{\partial R} = \frac{\Delta r}{\Delta R} = \begin{bmatrix} \frac{f_x}{Z} & 0 & -X\frac{f_x}{Z^2} \\ 0 & \frac{f_y}{Z} & -Y\frac{f_y}{Z^2} \end{bmatrix} \tag{2}$$

where f_x and f_y are the normalised focal lengths.

Under small rotation assumption, if the camera moves with instantaneous translational velocity v and instantaneous rotational velocity ω with respect to the environment, then the point R appears to move with a velocity

$$\frac{dR}{dt} = -v - \omega \times R \tag{3}$$

with respect to the sensor.

Taking the first-order Taylor expansion of the term $Z_1(r+\Delta r)$ in Eq. 1 and substituting Eq. 3 and Eq. 2 into it gives us Eq. 4 where $\nabla Z_1(r) = (Z_x, Z_y)$ are the spatial derivatives of $Z_1(r)$. This equation generates a pixel-based constraint relating the gradient of the depth image ∇Z_1 and the temporal depth difference to the unknown pixel motion and the change of depth. In practice, in order to improve the computation speed, the depth image is downsampled to 80×60 which is sufficient to get an accurate estimation. Using Eq. 4, fast odometry can be calculated from depth images.

$$\begin{bmatrix} -Y - Z_y f_y - Z_x XY\frac{f_x}{Z^2} - Z_y Y^2\frac{f_y}{Z^2} \\ X + Z_x f_x + Z_x X^2\frac{f_x}{Z^2} + Z_y XY\frac{f_y}{Z^2} \\ -Z_x Y\frac{f_x}{Z} + Z_y X\frac{f_y}{Z} \\ Z_x\frac{f_x}{Z} \\ Z_y\frac{f_y}{Z} \\ -1 - Z_x X\frac{f_x}{Z^2} - Z_y Y\frac{f_y}{Z^2} \end{bmatrix}^T \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ v_x \\ v_y \\ v_z \end{bmatrix} = Z_0(r) - Z_1(r) \tag{4}$$

where $\omega_x, \omega_y, \omega_z$ and v_x, v_y, v_z are components of the rotation and traslation vectors.

However, in environments with few geometric features, this method will suffer from the degeneration problem, for example when the camera can only see a ground plane or parallel walls. In these "ill-conditioned" cases which are really common in indoor environments, the proposed method will produce inaccurate estimates. We use the "condition number" [15] to measure the degeneration degree of Eq. 4. When severe degeneration happens, the estimation outputs a failure signal.

Absolute Pose Estimation To obtain the vehicle's absolute pose in a given 3D map, a Monte Carlo Localization (MCL) [16] algorithm is used. Though MCL has been successfully used on ground robots [16], 6DoF pose state $S = (x, y, z, \phi, \theta, \psi)$ necessary for MAVs increases the complexity of the problem. We show that by carefully designing the motion and observation model, MCL can work very well on an embedded computer. More details can be found in our previous work [12].

1) Motion Model For each subsequent frame, we propagate the previous state estimate according to the motion model $p(S_t|S_{t-1}, u_t)$. The motion command u_t is the visual odometry computed from Eq. 4. To account for unexpected motion, the prediction step adds a small amount of Gaussian noise to the motion command for each particle. The propagation equation is of the form:

$$S_t = S_{t-1} + u_t + e_t \qquad e_t \sim N(0, \sigma^2) \tag{5}$$

where u_t is the odometry and e_t is the Gaussian noise. When odometry estimation fails, we propagate the particle set using a noise-driven dynamical model

$$S_t = S_{t-1} + e'_t \qquad e'_t \sim N(0, \sigma'^2) \tag{6}$$

where σ' is much bigger than σ.

2) Observation Model The belief of vehicle's 6DoF state is updated according to three different sources of sensor information in one observation O_t, namely depth measurements d_t from depth camera, roll $\tilde{\theta}_t$ and pitch $\tilde{\phi}_t$ measurements from IMU and height measurement $\tilde{z}_t$ from ground plane detection or the point laser. The final observation model is:

$$p(O_t|S_t) = p(d_t, \tilde{z}_t, \tilde{\phi}_t, \tilde{\theta}_t|S_t) = p(d_t|S_t) \cdot p(\tilde{z}_t|S_t) \cdot p(\tilde{\phi}_t|S_t) \cdot p(\tilde{\theta}_t|S_t) \tag{7}$$

The likelihood formulation is given by a Gaussian distribution. To improve the computation efficiency, an endpoint observation model [16] is used for calculating $p(d_t|S_t)$.

3.1.2 High-frequency Pose Estimation

For real-time control, low latency, accurate, fast and robust estimate of the position and orientation is required. We fuse data from all of the sources providing motion information to output high frequency state estimate as shown in the Fig. 2. We run a high rate attitude estimator at 250Hz on the flight controller unit (FCU) to stabilize the robot's angular motion. We also designed a robust full state with 9DoF position estimator capable of fusing data from optical flow (downward facing camera), odometry and localization (running onboard computer), height measurement and inertial sensors using an EKF running on the FCU at 100Hz. Such a setup allows to maintain a reliable estimate of full current pose even when data from a sensor/estimator degrades due to change in environment e.g. if optical flow fails to find enough features on the floor to generate odometry, other sensors/estimators provide enough information to estimate the current pose, therefore maintaining system redundancy and allowing smooth operation of the motion controller. Also, the input signals to the position estimator is pre-processed to produce a smooth input and reject any outliers eg. a moving average with outlier rejection is used for sonar. All these techniques together ensure the filter running on pixhawk FCU doesn't diverge due to outliers.

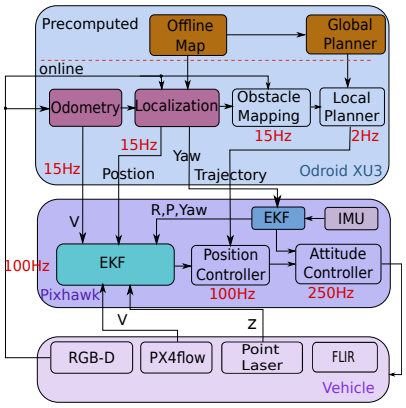


Fig. 2 Software architecture showing main modules with update rates

3.2 Online Motion Planning

Online motion planning is needed to keep MAV safe by quickly avoiding the obstacles which are represented by an online updated 3D occupancy grid [17]. Global mission points for motion planning are specified by a human or a high level mission planner. Here, we focus on local motion planning to generate collision-free trajectories, which is divided into two steps: *path planning* to generate optimal waypoints and *spline fitting* to generate optimal polynomial trajectories through waypoints.

3.2.1 Path Planning

We first search an optimal path, containing a series of safe waypoints to avoid the obstacles. We adopt the receding horizon control (RHC) framework, which searches the best path among an offline library [18]. In order to get a good path for different environments, the library is usually dense with large amounts of paths which is time consuming to check. Instead, we combine RHC with a modified CHOMP optimization method [19]. RHC serves to provide a good initial guess and CHOMP further optimizes it. Through the comparison in Section 4.3, this method is faster and better than RRT* in corridor environments.

Each waypoint in the path contains 4 DOF $\{x, y, z, \psi(yaw)\}$, namely the flat output space of quadrotor [20]. Let the path be $\xi(s) : [0,1] \mapsto R^4$ mapping from arc length s to 4 DOF ($\xi(0)$ is starting point, $\xi(1)$ is ending point) such that:

$$\begin{aligned} \min_{\xi} \quad & J(\xi) = w_1 f_{obst}(\xi) + w_2 f_{smooth}(\xi) + f_{goal}(\xi) \\ \text{s.t.} \quad & \xi(0) = \xi_0 \end{aligned} \tag{8}$$

where w_1, w_2 are the weighting parameters of different cost functions. $f_{obst}(\xi)$ is the obstacle cost and $f_{smooth}(\xi)$ is the path smoothness cost as defined in CHOMP [19]:

$$f_{obst}(\xi) = \int_0^1 c_{obs}(\xi(s)) \| \frac{d}{dt} \xi(s) \| ds \tag{9}$$

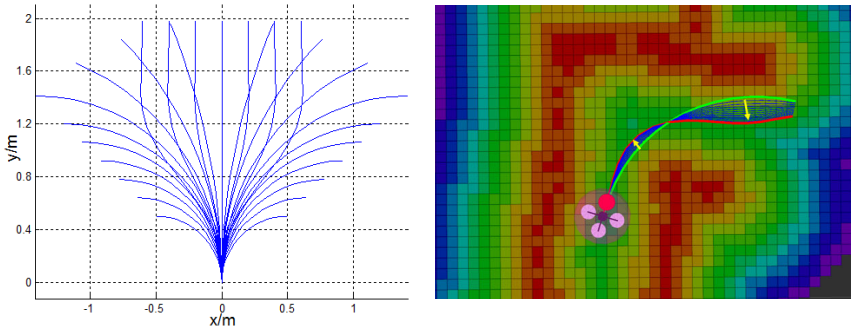
$$f_{smooth}(\xi) = \frac{1}{2} \int_0^1 \| \frac{d}{ds} \xi(s) \|^2 ds \tag{10}$$

$f_{goal}(\xi)$ is the cost-to-go heuristic measuring distance between path endpoint $\xi(1)$ to global mission point ξ_g. We add this heuristic to free the endpoint for optimization, while CHOMP doesn't.

$$f_{goal}(\xi) = \| \xi(1) - \xi_g \|^2 \tag{11}$$

As mentioned before, we create an offline path library L containing 27 specifically designed paths shown in Fig. 3(a). It is based on the structure property of corridor, where obstacles usually lie on two sides of walls. So it is easy and fast to find a safe path from the library.

We align the library with current pose then select $\xi^* = \arg\min_{\xi \in L} J(\xi)$ as the initial guess and optimize it through modified CHOMP. We keep discrete waypoint parametrization $\xi_0, ...\xi_n$ of the path as in [19] instead of a continuous path to speed up the optimization. An optimization example during turning is shown in Fig. 3(b), where the gradient pushes the path away from obstacles. Note that the end point is freed for optimization, different from standard CHOMP algorithm because our method is planning within a horizon and doesn't directly search a path from start to goal. A short horizon makes the optimization faster and more reactive.



(a) (b)

Fig. 3 (a) Initial path library. It is manually designed for the corridor environment where obstacles usually lie on two sides. It includes straight line, turning arcs with different curvatures and lane changing curves with parallel ending direction, corresponding to the three common flight patterns in the corridor. (b) Path optimization in turning. The color grid represents the distance map, computed from online 3D occupancy map [17]. The green curve represents the initial best path, blue curves are the paths during optimization based on the gradient (yellow). The final optimized path is in red.

3.2.2 Spline fitting

After getting path waypoints $\xi_0, ...\xi_n$, we fit a continuous spline $\xi(t)$ through them. It specifies the pose MAV should be at each time. The polynomial spline allows us to analytically compute feedforward control input for quadrotor [20], which guarantees exponential tracking stability of the controller while waypoint following or steering angle methods cannot.

We represent the spline as 5 segments of 6^{th} order polynomials. To find the optimal polynomial coefficients, we formulate it as a quadratic programming (QP) problem similar to [20, 21]. The cost function is to minimize the integration of L_2 norm of snap, namely the 4^{th} order derivative (wrt. time). The constraints are passing through waypoints and keeping derivative $c^1, ..., c^4$ continuous. A closed form solution of QP with equality constraint could be found using Lagrange multipliers. Tikhonov regularization [22] is used in case of QP matrix ill-condition problem.

4 Experiments

4.1 System Setup

The platform we use for our experiment is a customized MAV as shown in Fig. 4.1. It's mainly composed of two computation units. One unit is an ARM-based Quadcore embedded computer (Odroid XU3), responsible for high-level task processing, such as odometry estimation, localization and motion planning, etc. The other one is the Pixhawk FCU which is used for multi-sensor data fusion and real-time control. A forward-looking RGB-D camera is used for pose estimation and motion planning. A downward-looking optical flow camera is used for velocity estimation and a point laser is used for height estimation. Besides, a FLIR camera is used for fire detection.

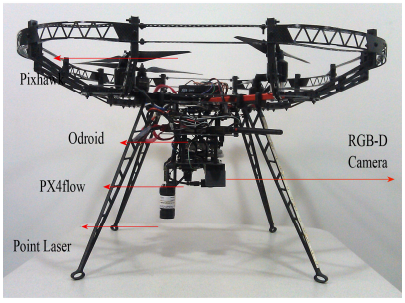


Fig. 4 Micro Air Vehicle platform

We first conduct some experiments to validate the performance of our state estimation and motion planning algorithms using the datasets recorded by carrying the robot in the ship. Then, field experiments were performed on the ex-USS shadwell to test the performance of the whole system. In the experiments, the RGB-D images are streamed at frame rate of 15Hz with QVGA resolution. We create the offline 3D maps by using LOAM system [23] and the map resolution is set to 4cm.

4.2 Pose Estimation Experiments

We test the odometry and localization algorithms by manually carrying our customized MAV system. The experiment is conducted in a constrained and visually degraded shipboard environment, which has a size of 16m $\times$ 25.6m $\times$ 4.04m. In this environments, most of the time the RGB images are very dark as shown in Fig. 5, while the depth images are still very good. There are also some challenging locations where the depth camera can only see the ground plane, one wall or two parallel walls, or even nothing when it is very close to the wall (minimum range $>$0.5m). In such situations, the depth-based odometry will suffer from the degeneration problem. In our algorithm, we monitor the degeneration status. If the degeneration is too severe, the odometry estimation method will not output motion estimation results, but a failure indicator. Then, our localization algorithm will use the noise-driven motion model to propagate the MCL particle set. In our experiment, we find that if the odometry failure is relatively short in duration (less than 3 seconds), it is possible for the localization algorithm to overcome this failure entirely. The localization

result is shown in Fig. 5. From the experimental result, we can see that our robot can robustly localize its self even the odometry is not good.

Fig. 5 Localization in degraded visual environment: Pink: Odometry, Red: Localization. The center plot shows the odometry, localization results with the 3D octomap. Pictures on both sides show the RGB and depth images from onboard RGB-D camera.

4.3 Planning Experiments

A simulated depth camera based on 3D point cloud map is used to create an occupancy grid. The mission planner then provides some goal points based on the prior map, about 5m away from each other and local planner keeps replanning to reach them. The pose history during simulation is shown as red curve in Fig. 6.

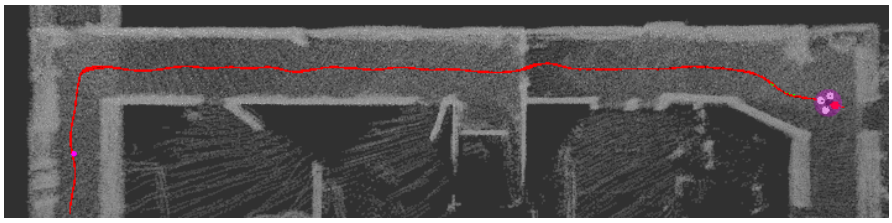


Fig. 6 An example trajectory calculated using path optimization with receding horizon control through a simulated shipboard environment.

To demonstrate the quality of our method, we compare our path planning method with RRT* and keep spline fitting part (minimizing snap) as the same. To bias RRT*, the local goal points are set closer to each other ($\sim$ 2 m) to greatly decrease the search space. The comparison is implemented on the embedded computer and the result is shown in Table .1. With bias, RRT* still needs more time than our method to generate a valid path and the quality in terms of obstacle cost and snap cost is higher than ours. This is mostly due to the fact that the corridor is a structured environment where obstacles usually lie on two sides. So our path set method could quickly find a smooth and safe path while RRT* needs many random samples in order to get a valid and smooth path.

Table 1 Path planning comparison with RRT*. Dist stands for vehicle distance to the obstacle.

Methods	Time(ms)	Mean dist(m)	Min dist	Mean snap(m/s^4)	Max snap
RRT*	70	0.46	0.16	1.46	14.02
Our	30	0.47	0.18	0.58	2.50

An end-to-end offline path could also be computed from the prior map but blindingly following it tends to cause a collision if there is big state estimation error. Instead, the proposed online obstacle mapping and motion planning can guarantee the safety. The goal points in our planner should be set properly so as to avoid being trapped in local dead-ends. Though offline path with online modification could relieve the problem, it is not applicable in other unknown environments.

4.4 Autonomous Flight Test Results

The mission of the completely autonomous flights is to search, detect and locate fire using only onboard sensors and computation resources. In our tests, the MAV needs to operate in a variety of environments:

1. Narrow passageways and doorways: The most common shipboard environment. The space constraints limit the vehicles size.
2. Areas with poor or no lighting: Become visually degraded. Performance of optical flow sensor decreases.
3. Areas filled with smoke and fire: Smoke density varies with fuel source. It strongly affects the depth image and optical flow sensor.

Fig.7 shows the created offline point cloud map of the testing area and typical sensor images in each environment. MAV is launched around the 'start point' and flies autonomously in the 1 m wide, 20 m long corridor, with a tight doorway (66cm wide, 8 cm clearance) and reaches the 'end point', while detecting fires.

We performed 20 experiments in this testing area under the three environment conditions. The vehicle pose of one experimental run is shown in Fig. 8. The success ratio of 20 runs is shown in Table 2. Failure cases are usually due to quadrotors being slightly rotated and stuck in the tight doorway. It is difficult to cross the door in dense smoke because the depth image is corrupted by smoke making it difficult for state estimation and obstacle detection. Results show that our robot can work very well in all the conditions except very dense smoke.

Table 2 Autonomous Flight Results

Environment	Total run	Succeeded	Rate
Normal	4	4	100%
Dark	7	5	71.4%
Smoky	9	5	55.5 %

Runtime performance is also very important for MAVs since the onboard computation abilities are limited. We record the performance including CPU usages of

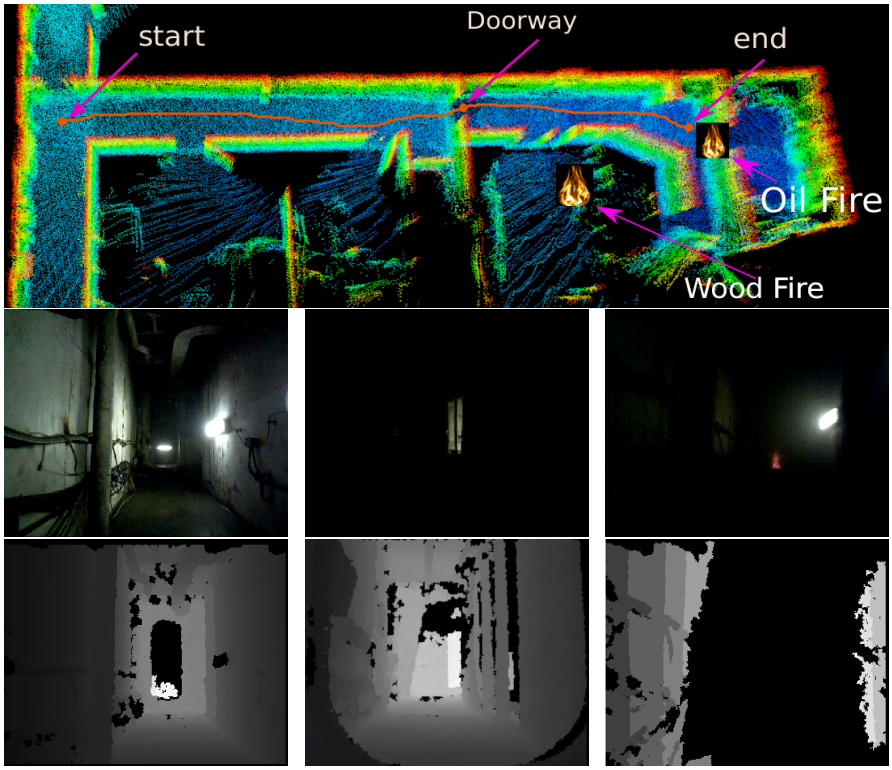


Fig. 7 Map and RGB and depth images of each environment condition from onboard RGB-D cameras. From left to right: with lights on, with lights off, with fire and dense smoke.

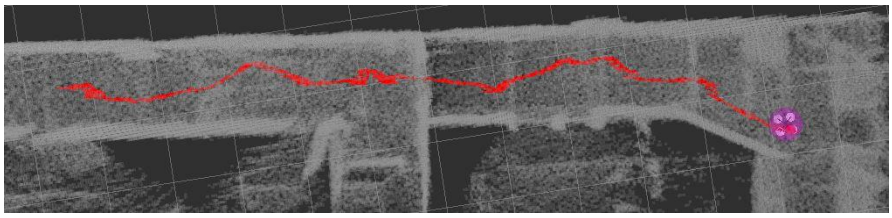


Fig. 8 Localization result from one autonomous flight.

some key algorithms on the Odroid system shown in Table. 3. We use 300 particles for MCL localization. When all the system modules are running, the total CPU usage is between $60 \sim 65\%$. The experiment result shows our navigation system can run in real-time by only using the onboard computation resources.

For fire detection, we use a lightweight FLIR-tau thermal camera to measure the temperature of the environment. We segment the appropriate range of temperature for fire, people etc. based on the thermal images. Anything over 100°C is considered to have a high probability of being fire or close to fire. Similarly, segmented blobs

Table 3 Per-Frame Runtime Performance on the Embedded Computer

Name	Algorithm Runtime			
	Mean	Min	Max	StdDev
Odometry	18.3ms	8ms	25.8ms	5.2ms
Localization	65.8ms	45.8ms	97ms	16.5ms
Local Planning	29.2 ms	15.2 ms	37.8 ms	6.7 ms

with temperature close to 30°C is considered to belong to a human being. The video of a field experiment at Shadwell in Nov 2014 can be found at `https://www.youtube.com/watch?v=g3dWQCECwlY`.

5 Conclusion

In this paper we have shown the feasibility of an autonomous fire detection MAV system in a GPS denied environment with tough visibility conditions. This was achieved without the need of any additional infrastructure on the ship. We achieved autonomous flight with fully online and onboard state estimation and planning through 1m wide passages while crossing doorways with only 8cm clearance. We demonstrated 10 consecutive runs where the vehicle crossed completely dark, light smoky passageway respectively and ended by detecting wood and diesel fires.

The next challenges are to increase the robustness and safety of the vehicle while increasing flight time. This will involve improvements in both hardware and software. The current size of vehicle is a little large, resulting in a very tight fit through the ship doorways. In future, we intend to move from a quadrotor design to a single/coaxial ducted rotor design to decrease size but increase flight time efficiency. Currently, our sensor suite loses reliability in dense smoke conditions. We plan on adding sensors which extend the range of environments our robot can successfully navigate and inspect. On the software side, one important goal is to decrease the dependency on a prior map for state estimation to make the system more adaptable to changing or damaged environments. Pursuing exploration and mapping in a damaged environment poses many interesting research challenges.

References

1. Slawomir Grzonka, Giorgio Grisetti, and Wolfram Burgard. A fully autonomous indoor quadrotor. *IEEE Transactions on Robotics*, 28(1):90–100, 2012.
2. Ivan Dryanovski, Roberto G. Valenti, and Jizhong Xiao. An open-source navigation system for micro aerial vehicles. *Autonomous Robots*, 34(3):177–188, March 2013.
3. S. Shen, N. Michael, and V. Kumar. Autonomous multi-floor indoor navigation with a computationally constrained mav. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 20–25. IEEE, 2011.

4. Konstantin Schauwecker and Andreas Zell. On-board dual-stereo-vision for the navigation of an autonomous MAV. *J. Intell. Robot. Syst. Theory Appl.*, 74:1–16, 2014.
5. F. Fraundorfer, L. Heng, and D. Honegger. Vision-based autonomous mapping and exploration using a quadrotor MAV. In *IEEE Int. Conf. Intell. Robot. Syst.*, pages 4557–4564, 2012.
6. A.D. Wu, E.N. Johnson, M. Kaess, and et al. Autonomous Flight in GPS-Denied Environments Using Monocular Vision and Inertial Sensors. *J. Aerosp. Inf. Syst.*, 10:172–186, 2013.
7. D Scaramuzza, M Achtelik, L Doitsidis, and et al. Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in gps-denied environments. pages 26–40, 2014.
8. S. Weiss, D. Scaramuzza, and R. Siegwart. Monocular-slam–based navigation for autonomous micro helicopters in gps-denied environments. *Journal of Field Robotics*, 28(6):854–874, 2011.
9. G. Flores, S. Zhou, R. Lozano, and P. Castillo. A vision and gps-based real-time trajectory planning for a mav in unknown and low-sunlight environments. *Journal of Intelligent & Robotic Systems*, 74(1-2):59–67, 2014.
10. AS Huang and A. Bachrach. Visual odometry and mapping for autonomous flight using an RGB-D camera. *Int. Symp. Robot. Res.*, pages 1–16, 2011.
11. R. G Valenti, I. Dryanovski, and C. Jaramillo. Autonomous quadrotor flight using onboard rgb-d visual odometry. In *2014 IEEE Int. Conf. Robot. Autom.*, pages 5233–5238. IEEE, 2014.
12. Z. Fang and S. Scherer. Real-time Onboard 6DoF Localization of an Indoor MAV in Degraded Visual Environments Using a RGB-D Camera. *2015 IEEE Int. Conf. Robot. Autom.*, May 2015.
13. Berthold K.P. Horn and John G. Harris. Rigid body motion from range image sequences. *CVGIP Image Underst.*, 53(1):1–13, January 1991.
14. F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat. Comparing ICP variants on real-world data sets. *Auton. Robots*, 34(3):133–148, February 2013.
15. K Callaghan and J Chen. Revisiting the Collinear Data Problem: An Assessment of Estimator Ill-Conditioning in Linear Regression. *Practical Assessment, Research & Evaluation*, 13(5):5, 2008.
16. S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.
17. S. Scherer, J. Rehder, and et al Achar, S. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214, 2012.
18. C.J. Green and A. Kelly. Optimal sampling in the space of paths: Preliminary results. 2006.
19. N. Ratliff, M. Zucker, and et al Bagnell, J.A. Chomp: Gradient optimization techniques for efficient motion planning. In *2009 IEEE Int. Conf. Robot. Autom.*, pages 489–494, 2009.
20. D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE Int. Conf. Robot. Autom.*, pages 2520–2525, 2011.
21. C. Richter, A. Bry, and N. Roy. Polynomial trajectory planning for quadrotor flight. In *International Conference on Robotics and Automation*, 2013.
22. G.H. Golub, P.C. Hansen, and D.P. O'Leary. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications*, 21(1):185–194, 1999.
23. J. Zhang and S. Singh. LOAM : Lidar Odometry and Mapping in Real-time. *Robotics: Science and Systems Conference (RSS)*, 2014.