Autonomous Exploration and Navigation Strategies for MAVs

Sai Vemprala, Srikanth Saripalli
School of Earth and Space Exploration
Arizona State University
Tempe, Arizona, USA
svemprala@asu.edu, srikanth.saripalli@asu.edu

***Abstract*—Micro Aerial Vehicles (MAV) are becoming increasingly popular as widely applicable robotic platforms for various tasks. For effectively complementing human tasks, a strong layer of autonomy is important for the platform. Unlike ground robots, MAVs are not limited to two dimensions, but this advantage also creates additional complexity in implementing various concepts such as estimation and control as well as linking them together. This paper outlines a solution pipeline that provides an integrated and modular framework for achieving state estimation, mapping and navigation capabilities, which can be used for a complete autonomous exploration capacity on a MAV platform. The implementation was tested both in simulation and partly on a real MAV that was custom built with off-the-shelf equipment for this purpose.**

I. Introduction

Compared to ground robots, micro aerial vehicles (MAVs) offer a higher advantage in exploring unknown environments. Both by being smaller and possessing inherent advantages of speed and maneuverability, areas inaccessible by humans and ground robots can be navigated by MAVs and they are more suitable for applications such as reconnaissance, environmental sensing, disaster support etc. When such tasks are presented to MAVs, continuous human input is not an efficient approach, both because MAV control is somewhat complex to achieve with human input, and it is time consuming; which eventually demands autonomous capabilities from the MAV. For robust autonomous MAV navigation, the best course of action is accurate sensor fusion on the MAVs end for data collection and an efficient strategy on the software part to fuse higher level tasks such as mapping, trajectory planning and control.

MAVs are already being implemented on a large scale in projects involving military and civilian applications, such as surveillance, disaster support and coordination, engineering inspection and observation and such. Significant amount of work has been put into developing MAVs with a wide range of capabilities in outdoor environments, but the capacities of MAVs in indoor environments are still very limited. Estimation of its own position and condition is highly important for an MAV and whereas it is easy to obtain this information outdoors using the well-established GPS technology, which ensures accurate positioning and operation, most indoor environments remain difficult problems for MAVs to handle. The main goal of this paper is to present a systems level integration of various modules which, together, aim to achieve estimation, mapping and navigation capabilities for an MAV. Implementation has been done in simulations and in part using a real world custom built MAV platform. Ongoing work is to complete the full implementation on the MAV and test autonomous exploration capabilities in GPS-denied indoor areas.

Presented at the AHS International Specialists' Meeting on Unmanned Rotorcraft, Chandler, Arizona, 2015. Copyright ©2015 by the American Helicopter Society International, Inc. All rights reserved.

II. Related Work

Autonomous navigation using MAVs has recently been a topic of major interest. This is especially a harder problem on MAVs than ground robots because of the inherent complexity associated with control of MAVs, the lack of sensing with absolute accuracy in indoor spaces and the fact that estimation, mapping and navigation need to work together efficiently on an onboard computation module in-flight, which makes the tolerance for failure low. There were many approaches proposed in the relevant literature for the various modules of autonomous navigation, such as state estimation, environment sensing and actual navigation capabilities.

Extensive research has been conducted in the area of state estimation for MAVs. Achtelik et al [1] introduced a concept of using a monocular wide-angle camera facing downward for obtaining position estimates of the MAV in indoor environments. A similar approach was also discussed by Blosch et al [2]. But in both these approaches, owing to the fact that this downward facing camera is the only source of information, no knowledge of the environment in front of the MAV is present, which limits the possibility of path planning or obstacle avoidance. Better ways of state estimation were proposed in work such as the one by Huang et al [3], which used forward facing RGBD cameras and visual odometry approaches to compute position estimates. One possible limitation of this approach is that estimation is not very good in areas with less number of visual features, and also that high speed flights pose an issue because of motion blur. LIDAR based estimation has also been investigated in the literature, as seen in [4].

A higher level navigation/exploration framework was introduced by Bills et al [5] using a Parrot AR Drone quadrotor, but there was no actual map constructed and stored for further knowledge, for instance, by a human operator, was not shown. A more integrated set of capabilities for MAVs was presented by Heng et al [6], where the capabilities demonstrated by the MAV were obstacle avoidance, map construction and global navigation. But the state estimation in this framework was performed with the assistance of a Vicon motion capture system, thus limiting the application of this work in real world scenarios. Another approach where the MAV contained on board estimation, navigation and mapping capabilities was presented by Schmid et al [7]. The quadrotor platform described in this paper contained a forward facing stereo camera capable of performing visual odometry calculations which are then combined with the measurements of an inertial measurement unit with computations on an FPGA board. Another implementation of autonomous mapping and exploration was presented by Heng et al [8].

There have been some limitations in the approaches considered in the previous literature. It is worth noting that some of the systems presented in the literature are able to operate only within the coverage area of sophisticated external tracking systems (e.g. Vicon). It is only very recently that methods without such needs started being developed, which are capable of performing real-time, onboard computations and enabling a push towards deployable MAVs in general environments. One other limitation, which can be seen for instance in [6] and [8] is limiting the navigation approach to two dimensions, i.e., although mapping has been performed in 3D, the navigation and path planning modules only work by considering a 2D ”slice” of the 3D map by keeping the MAV at a fixed altitude, thus effectively navigating only in 2D.

Upon closer examination, it is also revealed that many solutions presented in this area for estimation, mapping, control and navigation were efficient but had a highly decentralized approach, and thus, provide answers to only a subset of the complete set of problems that need to be solved. Tackling individual issues makes it easier to solve them in a straightforward fashion, but more of a challenge is faced in integrating various capabilities to obtain a complete pipeline necessary to achieve autonomous exploration. This was the main consideration of the approach described in this paper. To improve on the problems noted in the relevant literature, the aim of the discussion in this paper is to provide a solution with three main features:

1) Should address all of the main concepts that are fundamental to 3D MAV exploration
2) Should be possible to be integrated with open source MAV platforms
3) Should focus on easy real-time implementation.

As mentioned before, these features were incorporated by presenting a systems level integration of various modules, which can be implemented with minimum extra effort or platform-specific modifications on MAVs.

III. Solution Pipeline

The main components of an autonomous exploration strategy in indoor GPS-denied environments for a MAV can be expressed to be three-fold and loosely divided as follows.

1) State estimation
2) Environmental mapping
3) Path planning in the maps

Reliable position estimates need to be obtained by the MAV through sensor fusion, and using the position data as well as well as information obtained from a forward facing camera source, dense 3D mapping needs to be performed which would then be utilized for trajectory planning.

A. MAV State Estimation

Because of the inability of an MAV to depend on precise GPS or any ground truth data in indoor spaces, the state estimation needs to be performed by a careful fusion of data from multiple sensors. Usually because of their six degrees of freedom in motion, MAVs are equipped with inertial measurement units (IMU) containing accelerometers and gyroscopes returning corresponding data along all three axes, and a double integration of the acceleration data is performed to obtain the necessary position information. Although theoretically a sufficient description of the state of the MAV, MEMS IMUs such as the ones typically found on commercial MAVs are subject to major drifts at times, which introduces an uncorrectable amount of accumulated error in position estimation.

Hence, in this approach, the main source of the position data has been chosen to be a downward facing optical flow camera fixed to the MAV. This optical flow camera sensor, when in motion along with the MAV, has the capability of computing the pixel flow between two successive frames at two instants of time using the sum of absolute differences (SAD) technique of block matching. The camera module is also provided with an ultrasonic rangefinder, which computes the distance to the ground and outputs it as height, as it is always facing the ground. The distance obtained from the rangefinder can then be used to scale the optical flow to metric velocities in the sensor frame. If the velocities in the sensor frame are v_x^s and v_y^s and the velocities in the body frame are v_x^b and v_y^b, and the height estimate from the ultrasonic rangefinder is h, the body frame velocities along X and Y axes can be computed as shown in equations 1 and 2 below.

$$v_x^b = h(\lambda_x v_x^s - \Delta\theta) \tag{1}$$

$$v_y^b = h(\lambda_y v_y^s - \Delta\phi) \tag{2}$$

$\Delta\theta$ and $\Delta\phi$ are the pitch and roll angles obtained from the IMU equipped on the MAV and are used for compensating for the rotations of the whole frame around the coordinate frame. Hence, orientation data obtained from the IMU can be used to convert the velocities from the sensor frame to the body frame of the MAV. Once the body frame velocities of the MAV are computed, integrating the linear components of the velocities over time results in an estimate of the position in X and Y coordinates, whereas position in the Z axis is measured by the ultrasonic sensor. A loosely coupled filtering approach is implemented on top of the camera/rangefinder data and has been found to successfully enable autonomous hover and flight capabilities on the MAV through position/attitude estimation and the accompanying control techniques. This complete onboard implementation has been found to successfully eliminate the need for sophisticated motion tracking systems.

B. *Mapping of surroundings*

Once the MAV is able to localize and keep track of itself with sufficient accuracy guaranteed, the focus is then moved to the capability of mapping the surroundings. Mapping can usually performed in two perspectives: one, to enable the MAV to understand its surroundings and plan accordingly when trying to provide a solution for a planning problem, and two, to provide more accurate data to a human operator in the background and to aid in high level problem solving/task delegation. Because of the nature of its utilization, the former type of map can be lower in resolution and features than the latter.

To satisfy the knowledge prerequisites for 3D navigation, the main focus was on providing a map that has sufficient description of the surroundings in three dimensions so as to allow efficient planning and obstacle avoidance through the environment for the MAV. Prior conventional approaches included solutions such as implementing a 2.5D map: i.e., storing the 2D map at different heights and assembling them to construct an approximated version of the whole environment. Although this can be an efficient solution when some assumptions are made about the knowledge, these representations cannot store the full volumetric information of free/occupied spaces of the environment, hence, cannot be a sufficient representation to achieve full 3D planning capabilities. Because a full 3D map is required, conventional grid-based representations also will not be a good solution because they tend to be far more resource intensive than what a generic MAVs onboard computational capabilities can handle.

The solution considered here was an Octree-based 3D likelihood description of the environment. In the scenario being considered in exploration related applications, because of MAV navigation taking place in unexplored environments, the amount of area unexplored would always be more than the amount of area explored: i.e., the map would be always sparsely populated. This causes grid based approaches to take up an unbearably large amount of memory because of their tendency to store both explored and unexplored areas of the map. Unlike those, an octree representation has the capability of only storing those parts of the map that have been explored.

In order to implement the octree representation in software, the Octomap approach [9] has been utilized. To maintain the same focus of simplicity throughout the framework, the MAV considered here is equipped with a simple 3D RGBD sensor. The RGB and depth images received from the camera are then processed as point clouds and transferred through the Octomap module to obtain a probabilistic occupancy representation of that area of the map. Free/unknown ambiguity, which is common in grid based representations is solved in the octree representation by explicitly representing free nodes as free, whereas unknown cells do not occupy any storage space. An estimate of the amount of confidence in the occupancy of a given cell is also maintained by Octomap, and once the confidence reaches the minimum or maximum thresholds set beforehand, the cell is treated as a stable cell. By denoting the confidence levels of the cells in the map, a possibility of compression of the map can be achieved which can result in smaller sizes and memory overhead.

As per the implementation of Octomap, there is no upper bound on the size of the occupancy map, which may result in a high amount of memory use as well as processing burden. To counter this, it has been considered to be a good trade-off to use an Octomap of the range 0.2 to 0.3 m resolution in the navigation framework. Using a slightly lower resolution map provides a compromise between mapping detail and size/processing speed. If a higher resolution map is desired for further analysis in the background, the voxels in the Octomap can be used to directly represent the color image data instead of occupancy probabilities and stored to disk, or can be extended for more types of representations.

C. *Navigation through 3D environment*

The final part of the strategy is the autonomous navigation capability through indoor environments. This can be implemented by usage of a robust path planner that allows movement through the areas that have been explored, and also an accompanying strategy that controls how the MAV ventures into unknown areas.

Many of the previous approaches used to solve this problem tend to consider a 2D representation of the areas. Although the map obtained is a three dimensional map, it is effectively projected into a two dimensional representation, by for instance, considering a 2D slice of the 3D map which is capable of marking areas as explored or unexplored on one plane. The MAV then can be commanded to explore unknown areas by marking and moving into the areas unexplored, i.e., frontiers, which is the reason it is known as frontier based exploration. Although an effective means of exploration, only 2D frontier based exploration suffers from two main issues:

1) As the size of the map increases, there could be many frontiers to explore and it will be hard for the MAV or the high level algorithm to prioritize the areas for exploration.
2) 2D frontier based exploration assumes that the MAV is flying at a fixed altitude, and does not take into account the three dimensional functionality that is expected from a MAV.

The exploration methodology of frontiers is improved upon in this approach by considering what can be called a pyramidal approach. The MAV's mapping is initialized at a certain height and instead of immediately trying to navigate directly to the frontiers of the map, more importance is given to maximize the features of the surroundings from the same spot by gaining more height until a certain threshold. This threshold can either be preset or can be dynamically computed from the RGBD sensors streaming data. By exploiting the third dimension, the inherent advantage of an MAV is used to sense more areas and achieve more mapping efficiency without expending too much power.

After the map is built with enough detail in 3D from the current position, the MAV can then be commanded to navigate to certain areas of interest, or try to "explore" the map by itself. If it is desired to let the MAV explore the unknown area by itself, this is achieved by providing incremental waypoints to the MAV at the borders of the known area. This is similar to the principle of frontier-based exploration, but the MAV can maximize information gain by adjusting the altitude accordingly as it moves. Depending on the application, there could also be various constraints that can be provided to the MAV so as to determine certain "areas of interest". They could be areas of scientific interest determined on-the-go using other sensors the MAV might be equipped with, predefined areas by a human operator for gathering more information etc. The framework utilized for navigation is flexible and allows integration of various ways to convey the desired goals to the MAV.

Once the knowledge of a certain area or a point (goal) is determined by the MAV, the final step would involve actually constructing a plan that takes the MAV there while avoiding any obstacles in the map. This capability is achieved by the navigation strategy by utilizing sampling based motion planners. Incremental sampling based motion planners such as rapidly exploring random trees (RRTs) [10] can be utilized effectively in solving high dimensional state space search problems with reasonable time and computational effort. The algorithm used here is an extension of an RRT known as RRT* [11], which introduces the concept of guaranteed asymptotic optimality to improve upon the original RRT. Hence, RRT* is definitely capable of returning an optimal solution as long as one exists; where the optimality is with respect to a distance function of the state space where the MAV belongs. Because of the sampling-based approach, RRT* is very well suited for navigating through large and densely populated 3D spaces.

One important limitation of the normal RRT* algorithm is that it works well in systems with simple dynamics, for instance, holonomic robots, because RRT* has the tendency to build a set of direct connections multiple possible states of the robot with an optimal trajectory. This approach works well with simple robots, such as ground robots, because they are limited to one plane, and connecting one state of the robot to another would usually result in a straight line. But for differentially constrained systems like quadrotors, finding a feasible trajectory between two "states" is a non-trivial problem and can be quite complex to solve for the RRT* algorithm. To overcome this issue in the framework here, the RRT* algorithm has been used only to sample the world and not the state space, and to return a valid "path" in the space. This path, which is a collection of points, is then broken down by a separate module and the waypoints are incrementally assigned to the MAV's flight control module and the low level control is made responsible for actually generating the commands to move the MAV from point to point.

The running time of an RRT* algorithm can always be derived to be a constant factor of the running time of a normal RRT algorithm. Considering the memory efficiency of an Octomap representation and the planning effectiveness as well as reliability of RRT*, quick and efficient planning and avoidance in full 3D environment descriptions can be achieved using this combination. RRT* has the capability to quickly and efficiently navigate large and even highly populated 3D spaces and return valid plans successfully.

IV. Implementation Details

The approach described in the previous section has been implemented in two scenarios: in simulation as well as a hexacopter built with off-the-shelf equipment.

A. Simulation

The code, simulation and algorithms were all built and run within the environment of Robot Operating System (ROS) [12] with Gazebo as the main simulation tool. The model of the MAV used in simulations was a generic quadrotor model presented by Meyer et al [13]. The quadrotor model also contains a model of a Microsoft Kinect sensor that returns RGB and depth image data, which is then processed as point clouds and integrated into the physics description of the world, thus enabling collision detection and avoidance. Octomap is available as an open source implementation in ROS. For the path planning implementation, RRT* has been utilized through an open source library named Open Motion Planning Library (OMPL) [14], which is in turn bundled in ROS as part of a framework known as MoveIt. Whereas MoveIt is usually geared towards manipulation-centric applications and algorithms, necessary changes and custom built code were developed to apply the same underlying framework to an MAV model. Both mapping and navigation were successfully tested in the simulations, but state estimation was not necessary because it was both directly possible to obtain ground truth from the Gazebo simulator and hard to simulate optical flow camera data inside the simulator.

B. Real platform

The same approach has also been implemented on a custom built hexacopter. The hexacopter is based on a commercially available DJI F550 frame and utilizes the 3DRobotics Pixhawk [15] as its main flight control module, which contains a six-axis gyroscope, accelerometer and magnetometer forming the core IMU modules and a code stack running on an ARM processor which takes care of low level control. A 3DRobotics PX4Flow camera has been used as the optical flow camera sensor [16]. This module contains a 16mm lens capable of outputting image data in 752x480 resolution, an associated ultrasonic sensor to measure height data and an ARM Cortex M4F processor which is capable of computing velocities from pixel flow using the Lucas-Kanade estimation method. Utilizing this data, the Pixhawk module executes state estimation as well as position and attitude control on its microcontroller in real-time. An ASUS Xtion RGBD sensor has been utilized to serve the purpose of the front facing camera and a 3rd generation Intel NUC with a Core i5 processor is used as the onboard computation module which runs Linux and ROS for all the software modules, which perform the same tasks as they would with the MAV in simulation. The complete MAV setup can be seen in figure 1.

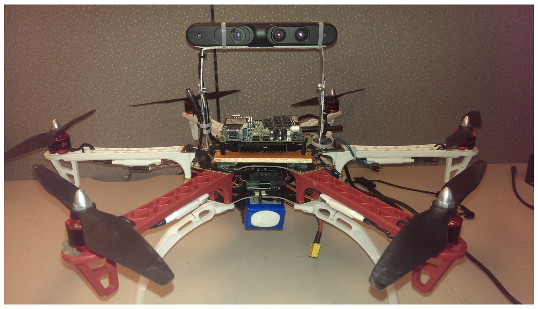
Fig. 1. Custom built hexacopter frame and equipment

V. Results and Discussion

This section presents the results obtained through application of the navigation strategy both in simulation and real applications.

The framework was successfully tested in the Gazebo simulations where environments were generated beforehand.

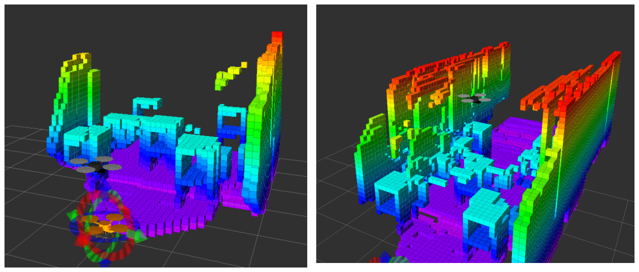
Fig. 2. Demonstration of pyramidal approach increasing mapping efficiency

Figure 2 shows how exploiting the third dimension enhances the MAV's knowledge of the world around it. Initially at a lower altitude, the MAV senses the environment to be obstructed by obstacles, which in turn makes it form a conclusion that the areas beyond are either inaccessible or require traversal of a longer path to get to those areas. If the MAV tries to maximize the possibility of improving the existing map (which in this case is through gaining altitude) instead of directly trying to plan paths towards the frontiers of the map, it can be seen that the area above the obstacles is clear and can be used for successful planning. The following figure show how 3D path planning works in simulation. The green outline of the quadrotor in the center is the initial state and the orange one is the goal, beyond a wall. A trail of positions shown in the map represents a valid path that can be traversed by the quadrotor.

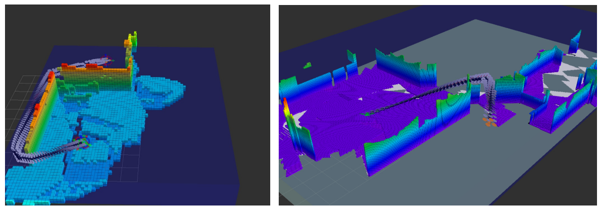
Fig. 3. Path planning using RRT* within the simulator

On the real MAV flights, the main capabilities tested were estimation and mapping. Estimation was performed onboard the Pixhawk control module by integrating the velocity data returned by the optical flow camera mounted on the bottom, and the position data is in transformed in frames and in turn communicated to the mapping/navigation modules running on ROS. Mapping was tested indoors during autonomous flights of the MAV with offboard velocity commands being sent, but the same mapping approach is applicable to autonomous exploration as well.

Planning was performed offline on this stored map data. Once a sufficiently accurate map is constructed, the occupancy map can be stored and the same planning module can be executed which returns path plans with obstacle avoidance capabilities. In the real world application, a PID controller running on the Pixhawk flight control module has the capability of using the position estimates and performing autonomous navigation depending on the details of the goal points received. Once the trajectory is constructed by the planner, based on the way the state space was sampled, a controller coded within ROS translates the sampled points belonging to the plan into control commands for the MAV. Work is ongoing so as to deploy this controller framework and the exploration strategy directly on the MAV and thereby control the navigation system. Figures 4 and 5 show some results of the mapping and offline planning stages of the implementation on the hexacopter in long corridor-like spaces.

VI. Conclusion

This paper mainly presents a systems level integration of solutions for a set of problems which is essential to achieve autonomous exploration capabilities on MAVs. The approach described here performs all of the main tasks, i.e., state estimation, mapping and navigation with a focus on simplicity and ease of implementation on widely available

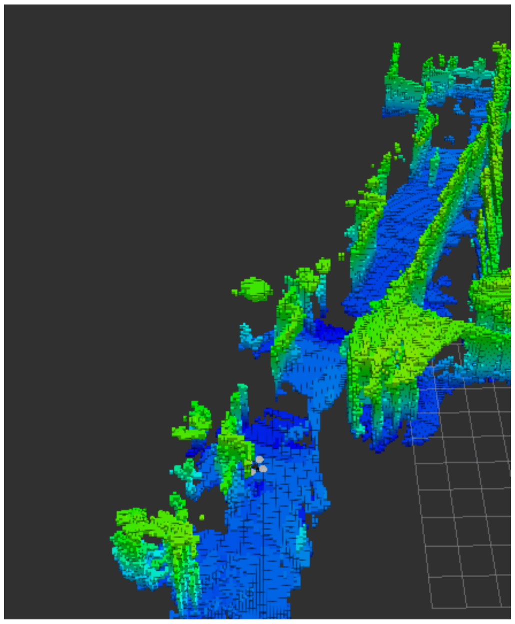

Fig. 4. Map of a corridor built through actual flight

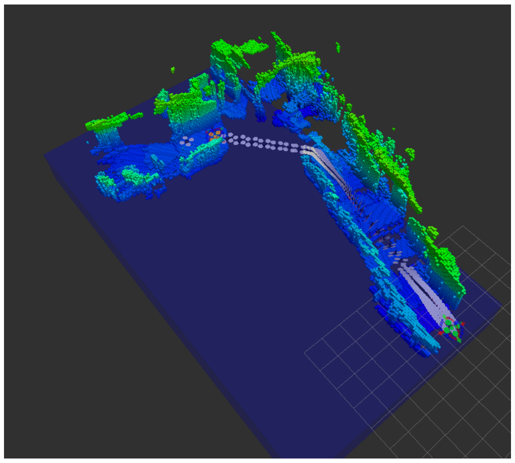

Fig. 5. Path planning performed offline on map data obtained through flights

MAV platforms. The navigation framework can be extended to work with any types of constraints without necessitating a complete rehaul of the foundations, and hence is possible to be utilized in different scenarios such as exploration of unknown areas for reconnaissance, or identification of spots of interest in disaster management etc. Simulation results have been presented to show the capabilities of the framework and it has been partly tested in real-time applications. Further work is currently ongoing to test the complete pipeline in challenging real world situations for achieving autonomous exploration capability.

References

[1] Achtelik .M, Weiss .S, and Siegwart .R, "Onboard imu and monocular vision based control for mavs in unknown in- and outdoor environments," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 3056–3063.

[2] Blosch .M, Weiss S., Scaramuzza D., and Siegwart R., "Vision based mav navigation in unknown and unstructured environments," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, May 2010, pp. 21–28.

[3] Huang A.S., Bachrach A., Henry P., Krainin M., Maturana D., Fox D., and Roy N., "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *International Symposium on Robotics Research (ISRR)*, 2011, pp. 1–16.

[4] Zhang J. and Singh S., "Loam: Lidar odometry and mapping in real-time."

[5] Bills C., Chen J., and Saxena A., "Autonomous mav flight in indoor environments using single image perspective cues," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 5776–5783.

[6] Heng L., Meier L., Tanskanen P., Fraundorfer F., and Pollefeys M., "Autonomous obstacle avoidance and maneuvering on a vision-guided mav using on-board processing," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 2472–2477.

[7] Schmid K., Tomic T., Ruess F., Hirschmuller H., and Suppa M., "Stereo vision based indoor/outdoor navigation for flying robots," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2013, pp. 3955–3962.

[8] Heng L., Honegger D., Lee G.H, Meier L., Tanskanen P., Fraundorfer F., and Pollefeys M., "Autonomous visual mapping and exploration with a micro aerial vehicle," *Journal of Field Robotics*, vol. 31, no. 4, pp. 654–675, 2014.

[9] Hornung A., Wurm K.M., Bennewitz M., Stachniss C., and Burgard W., "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[10] LaValle S.M., "Rapidly-exploring random trees a ew tool for path planning," 1998.

[11] Karaman S. and Frazzoli E., "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[12] Quigley M., Faust J., Foote T., and Leibs J., "Ros: an open-source robot operating system."

[13] Meyer J., Sendobry A., Kohlbrecher S., Klingauf U., and von Stryk O., "Comprehensive simulation of quadrotor uavs using ros and gazebo," in *Simulation, Modeling, and Programming for Autonomous Robots*. Springer, 2012, pp. 400–411.

[14] A. Şucan I., Moll M., and Kavraki L.E., "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, http://ompl.kavrakilab.org.

[15] Meier L., Tanskanen P., Fraundorfer F., and Pollefeys M., "Pixhawk: A system for autonomous flight using onboard computer vision," in *2011 IEEE international conference on Robotics and automation (ICRA)*. IEEE, 2011, pp. 2992–2997.

[16] Honegger D., Meier L., Tanskanen P., and Pollefeys M., "An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 1736–1741.