Journal of Korea Multimedia Society Vol. 23, No. 8, August 2020(pp. 1088-1098)
https://doi.org/10.9717/kmms.2020.23.8.1088

Obstacle Avoidance Method for UAVs using Polar Grid

Sudarshan Pant†, Sangdon Lee††

ABSTRACT

This paper proposes an obstacle avoidance method using a depth polar grid. Depth information is a crucial factor for determining the safe path for collision-free navigation of unmanned aerial vehicles (UAVs) as it can perceive the distance to the obstacles effectively. However, the existing depth-camera-based approaches for obstacle avoidance require computationally expensive path planning algorithms. We propose a simple navigation method using the polar-grid of the depth information obtained from the camera with narrow field-of-view(FOV). The effectiveness of the approach was validated by a series of experiments using software-in-the-loop simulation in a realistic outdoor environment. The experimental results show that the proposed approach successfully avoids obstacles using a single depth camera with limited FOV.

Key words: Obstacle Avoidance, Polar Grid, Unmanned Aerial Vehicle, Field of View

1. INTRODUCTION

In recent years, the use of unmanned aerial vehicles (UAVs) has been increased due to the advancement in sensor technology which makes the UAVs more stable and accessible. Increasing commercial and recreational use of UAVs has necessitated the development of reliable solutions for ensuring the safe and efficient autonomous flights. The autonomous unmanned vehicles are expected to complete the planned missions with no human intervention during the flights even in the environments with obstacles. Such autonomy can be achieved through efficient obstacle representation and path generation.

The UAVs have various applications such as, law enforcement, cargo transfer [1], and agricultural uses [3] [6]. The surveillance is another important application of UAVs. In [21], for example, the object detection technique using RGB camera has been proposed for the surveillance as an application area. The monitoring of power lines [7] [8] [9] is nowadays done using aerial vehicles with less cost, little human involvement, and no physical threat to humans. Because of the low maintenance cost, high mobility, and high maneuverability, the UAVs have huge potential in the field of the Internet of things [4]. Although the UAVs are used in various domains where human reach is either impossible or difficult, the operation of drones still requires a skilled pilot to oversee and control the vehicle remotely. Such remote operation of the UAVs is guided by the first-person view obtained from the camera attached to the vehicles as shown in Fig. 1. This requires a reliable communication link capable of transmitting the observed environment as a real-time video feed from the vehicle to the ground station. During remote operation, the

※ Corresponding Author: Sudarshan Pant, Address: (58554)Youngsan-ro 1666, Chonggye-myun, Muan-gun, Jeonnam, Korea, TEL: +82-10-4278-8412, FAX: +82-, E-mail: darshanz@mokpo.ac.kr
Receipt date: Jun. 16, 2020, Approval date: Jun. 23, 2020

† Dept. of Multimedia Eng., Graduate School, Mokpo National University
†† Dept. of Multimedia Eng., Mokpo National University

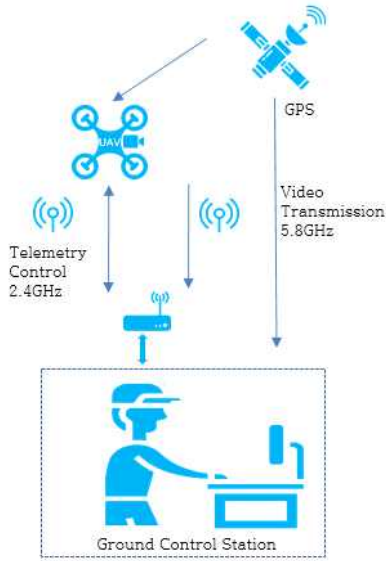


Fig. 1. A typical remote operation scenario.

flight is planned using missions. A mission refers to the set of way points with attributes such as location, speed, and other custom information that guide the vehicle during the flight. Although such missions can be accurately conducted with the help of GPS and various other onboard sensors such as accelerometer, gyroscope, magnetometer, and visual sensors, the in-air collision is a major concern during such missions.

Although the manual operation of the drones is supported by a first-person-view camera, some automated flight missions are hard to achieve without the prior knowledge of all the obstacles along the flight path. Existing obstacle avoidance methods can broadly be divided into vision-based and non-visual based methods. Visual based systems use monocular and depth-camera for sensing the obstacles while non-visual methods often require multiple sensors for an accurate representation of the environment. The reliable avoidance system needs to minimize the payload while keeping the performance of obstacle avoidance high because most of the UAVs have limited power and computation capacity.

The problem of the autonomous vehicles has been studied extensively in the last decade due to easy access to the open-source hardware and software for building highly stable drones at low cost. However, the development of a robust obstacle avoidance system is still a challenge due to computation complexity in various drones. This study is aimed at simplifying the obstacle avoidance process through a geometric approach based on the depth image obtained from a camera with a limited FOV. The proposed method is implemented using the Robot Operating System (ROS)[15] and is built using open protocols for convenient porting to popular UAV flight controller platforms such as PX4 and Ardupilot.

2. RELATED WORK

Obstacle avoidance has been widely studied not only in the UAV Navigation context but also for ground robots and large vehicles. The major issues related to obstacle-free navigation of UAVs include obstacle identification in a 3D environment, finding the optimal path from take-off to the goal position.

Based on the amount of data available, the avoidance methods are classified into global or local approaches. The global algorithms use a precomputed map of the environment to find the best path among all possible trajectories. Heng et al [2] proposed a global obstacle avoidance method using a point-cloud data from the environment. Such a point-cloud is then stored as Octomap [5] which is used to calculate the search tree. Then, anytime a dynamic A* algorithm[17] is used to find the best path. Although access to the entire environment map makes it less likely to stuck in local minima, the cost of building and storing the map is higher. Conversely, the local obstacle avoidance methods do not build a map of the environment. They follow the sense and react approach by utilizing the sensor data obtained at a particular state. Baumann [10] developed a hybrid algorithm called 3DVFH*

using a 3D vector histogram and A* algorithm for pathfinding. Although this approach is able to avoid obstacles effectively, the A* algorithm makes it computationally expensive as it needs to re-calculate the whole environment or at least a part of the environment to update the path whenever the field of view is changed.

In [13], CNN-based obstacle avoidance is proposed where a depth map is predicted from a monocular image. The UAV is then steered away from obstacles by controlling the angles based on the depth-map obtained through learning. Gandhi et al. [14] collected a large dataset of drone navigation by deliberately crashing the vehicles. Thus, by labeling the data right before the collision, the flight states are labeled with binary labels and used the CNN model to conclude a yaw angle for controlling the direction of UAV. Similarly, Dronet [11] has been proved successful in navigating around the urban environment successfully. However, the major drawback of such methods is that they require a massive amount of labeled data. Also, whenever the environment changes a new dataset has to be collected and labeled for training. Similarly, reinforcement learning-based methods learn the environment with trials and errors. Shin et al [12] proposed a reinforcement learning method supported by image segmentation for obstacle avoidance in a maze environment, failing to generalize for other the real-world applications that require go-to-location or mission planning based methods which are common in numerous tasks such as infrastructure inspection and surveillance.

A collision cone based methods [18, 19] have also been used to detect the possibility of collision between a robot and an obstacle. Although these methods are simple and have low requirements for computational capacity, the velocity is not considered as an important factor in these methods. Due to constant velocity, the vehicle may collide with obstacles while flying with high velocity.

Thanh [20] proposed an efficient algorithm for obstacle avoidance using LIDAR where Lyapunov stability is used to obtain guidance law and velocity control. Although this method finds the shortest path and avoids the obstacle efficiently, it needs the distance from the obstacle in all directions in order to identify and safely avoid obstacles. Avoiding the obstacle through the sidewise movement requires depth sensors on both sides as well as on the front side. This makes the method impractical in several situations where only the front camera is available.

This study proposes a simple, way-point generation based approach guided by a polar-grid obtained from a single depth camera on the front. The depth image is converted into a polar grid which provides the direction for avoiding the obstacles. Once the way-points are obtained from the ground control station, the proposed planner inserts the way-points when the diversion from the original path is required due to the presence of the obstacles.

3. PROPOSED APPROACH

3.1 Polar Grid Representation of Depth Map

The depth image was obtained from the simulated Realsense D435 depth camera through the Gazebo[16] simulator. The camera produces a depth image of the environment with the distances to the object from the camera. The distance between the camera and the obstacle is the primary sensor information used in this approach. Using the FOV information of the camera, the depth image can be represented as a polar-grid by mapping the distances to each elevation and azimuth angles. We do not use the entire 3D space as it is redundant due to the limited FOV of the available depth cameras. Also, this approach is a purely local approach with no storage of the environment-map involved, using a polar grid for the FOV is sufficient. Each cell of the polar grid is represented by an elevation angle ε, azimuth angle ζ and holds the distance to the objects in that region. Every dis-

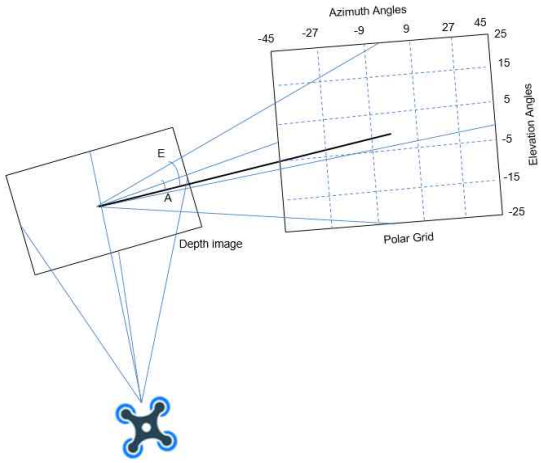


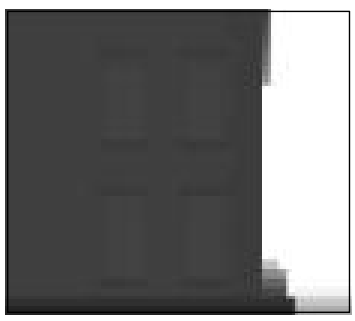
Fig. 2. Polar grid representation of a depth image.

tance value in the depth image is mapped to the grid in such a way that the angles corresponding to the image pixels range from -FOV/2 to +FOV/2. The ranges of azimuth and elevation angles can be given by:

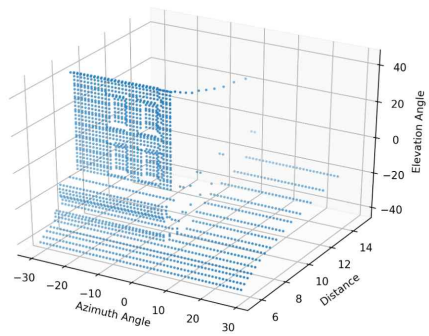
$$\zeta \in (-f_h/2,\ f_h/2)$$
$$\epsilon \in (-f_v/2,\ f_v/2) \quad (1)$$

where, f_h and f_v are the horizontal and vertical field of view of the depth camera respectively. The element of the polar grid p(ζ, ε, d) is characterized by the polar coordinates with the vehicle position as the origin., The polar gird is with size $f_h \times f_v$ is resized to 5 × 5 using min-pooling. This ensures the safe margins when the center of the free cells is used for calculation of the heading angle. The representation of the depth image into a reduced polar grid can be illustrated as shown in Fig. 2. Based on the FOV of the camera the depth values are mapped with different angles to construct the polar coordinates.

For the development of the proposed obstacle avoidance method, the depth information was analyzed for various arrangements of the obstacles. The arrangement of the multiple obstacles affects the performance of the avoidance algorithm as it becomes challenging to follow the shortest path. Since we apply the safety bubble while calculating the minimum distance to the obstacle, the collision is avoided guaranteeing the navigation to the next way-point. The most common organizations of the obstacles were analyzed to map the obstacles with the polar angles. In general, the obstacles can be either on the left or right or on both sides. Fig. 3 a), b) and c) show an example of the obstacle on the left. In the depth image shown in Fig. 3a), the building partially obstructs the path of the vehicle. The depth information in the form of the point cloud is visualized as the scattered plot in Fig. 3b), where the unoccupied space corresponding to the angles on the right direction can be seen. Similarly, Fig. 3c) shows the polar-grid representation of the depth image where the shaded region represents the unoccupied region. In such a scenario, the next



(a)



(b)

8.54	8.59	13.43	18.01	18.01
8.57	8.64	14.04	18.01	18.01
8.57	8.67	14.02	18.01	18.01
7.7	7.73	9.78	13.51	13.51
6.45	6.46	6.51	6.8	6.8

(c)

Fig. 3. An obstacle on the left represented as the grid map. (a) the depth image (b) Scatter plot of the point cloud in FOV (c) polar grid representation.

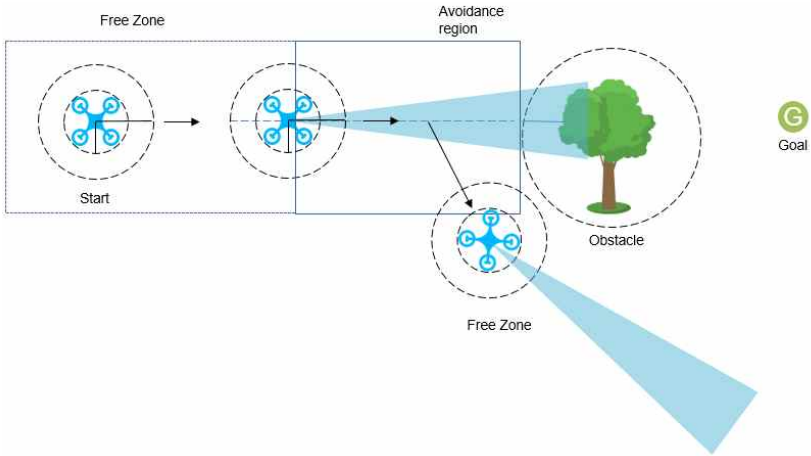


Fig. 4. Obstacle avoidance process using threshold based regions.

way-point is obtained by a short walk towards the direction of the unoccupied cell using the distance cost to the goal position.

The avoidance becomes difficult in more complex scenarios where the obstacles completely obstruct the path, for example, the long walls. In such cases, the vehicle has to head towards the edge of the grid to steer away from the wall while moving ahead to reach the end of the wall..

3.2 Polar Grid Based Obstacle Avoidance using safe distance

The polar grid discussed in the previous section is used to detect the obstacles encountered during the flight. For the safe and reliable obstacle avoidance, the flight state is classified into avoidance zone and free zone. The obstacle zone is the region where the obstacle is sensed and the vehicle needs the avoidance strategy. The region where the obstacles are not seen within the predefined threshold is called free zone as seen in Fig. 4. During the flight, a drone moves from start position to the final way-point avoiding the obstacles in between. In Fig. 4, the blue triangular area represents the FOV of the vehicle.

This approach is based on the way-point based navigation hence low-level control of the vehicle orientation is performed by the flight controller itself. The objective is to calculate the obstacle-free way-points for the vehicle based on the current position and the final goal. Several intermediate way-points need to be generated based on the occurrence of previously unseen obstacles as the vehicle approaches them. The proposed obstacle avoidance process can be summarized in a flow diagram as shown in Fig. 5.

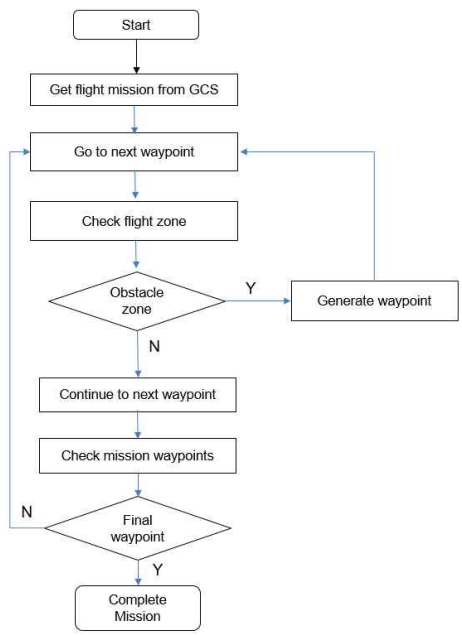


Fig. 5. Flowchart for obstacle avoidance using polar-grid method.

As the vehicle receives the mission, the next way-point in the mission is taken as the goal. To reach the goal vehicle has to use a sense-and-react strategy until it enters the avoidance zone. The avoidance zone is determined by comparing the minimum distance obtained from the depth camera and the minimum distance threshold. In this region, the velocity assigned during the mission creation is ignored and reduced speed is calculated to move towards the generated way-point. Since the speed is reduced in the avoidance zone, the threshold is set assuming low speed. As a general rule, the minimum distance to the obstacle should be defined in such a way that the vehicle can safely change direction as it approaches the goal. The safety radius is defined to form a safety bubble around the vehicle to ensure the vehicle does not collide with an obstacle by keeping the minimum distance. As the small field of view is assumed for the single forward-facing depth camera, the heading angle is constrained within a certain range of angles limited by the FOV.

Algorithm 1 shows various steps of obstacle avoidance using polar-grid based waypoint generation. The path planning step is based on the frequency of the acquisition of camera data obtained using ROS sensor messages. For each planning step, the distance to the nearest obstacle is compared with a threshold and the way-point is generated using the step-size parameter and the heading angle obtained from the polar grid. If there is no obstacle on the vehicle moves towards the goal with the speed set by GCS.

4. SIMULATION AND EXPERIMENTS

4.1 ROS based path planner implementation

In small and medium-sized drones, the ability to mount sensors is limited because of weight as well as the computing ability of the companion hardware. We performed the simulation using realistic simulation settings using the various simulation tools. Gazebo is the most commonly used simulation environment used for simulation of Robots. The Gazebo based simulation was used due to the ease of environment and model creation. Gazebo can be integrated into various robotic platforms through the Robot Operating System (ROS). Gazebo provides the physics engine with the graphical simulation of different environments. Such a visual environment is referred to as a world that includes various static and dynamic objects known as models. Such models and the world are configured using an XML based format known as SDF (Simulation Description Format) and interaction with the middleware platforms like ROS is done through plug-ins. The communication between the avoidance algorithm and gazebo-based quadcopter and depth camera models is done through ROS. ROS is an open-source framework for robotic software development. It standardizes various software components for the collaborative development of robot software. ROS performs purely peer-to-peer communication between various components referred to as ROS nodes. The ROS nodes are managed by a master ROS node called ROS core through unique URI (Uniform Resource Identifier) which is based on the IP address of ROS nodes. Various nodes communicate through ROS

Algorithm 1. Obstacle Avoidance based on Polar-Grid

1. Loop over the available way-points in the way-point manager
2. Generate the polar-grid from depth image using FOV information
3. Based on the obstacle distance, generate next way-point.
4. Move towards the generated way-point with reduced speed.
5. Move towards the goal with normal speed if no intermediate way-points are available

messages, services, or actions through the publish/subscribe model. The ROS nodes setup in this experiment are as shown in Fig. 6. The figure shows various ROS nodes communicating in the implementation of the proposed system. The ROS nodes are shown as ellipses while the ROS topics are represented by the rectangles. We created an */autoflight_node* implementing the proposed avoidance technique. The proposed waypoint generator obtains the depth camera data with the gazebo simulator node through ROS messages published with topic */camera/depth/image_raw.* The */mavros* node is used for controlling the vehicle and getting the vehicle status through various topics as shown in Fig. 6.

The simulation for the experiments was designed using the commonly used open-source flight controller platform, Ardupilot. The communication between the computer and the flight controller simulator was done using the MAVLink protocol.

We developed a custom way-point management system to assign way-points and manage the state of the completion of a mission during the flight. The main functions of the waypoint management system include the creation of a mission with multiple waypoints and communication with the flight controller and the companion computer running the path planning algorithm. The proposed approach was tested in a computer with Intel(R) Core(TM) i5-3470 3.2 GHz CPU and 8 GB memory. The experiments were run on 64-bit Ubuntu 16.08 distribution with ROS Melodic.

Two different arrangements of the obstacles were prepared in the Gazebo world to simulate different obstacle scenarios. The experiments were carried out in following different scenarios. An empty gazebo world was created with the ground pane only with no obstacles above the ground. This step is important to ensure that the drone follows the straight path without any deviation. This scenario verifies the proper functions of the way-point manager developed for the experiments. The way-point manager intercepts the mission loaded from the ground control station, and plans the mission based on the initial way-points and the way-points generated during avoidance.

For the experiments, the default flight control parameters were used. The flight controller parameters were confirmed to be optimal by conducting the mission-based flight simulation in the environment with no obstacle without connecting to the proposed path planner. The way-points used are based on the global positioning system(GPS) coordinates which were obtained from the embedded GPS module. The Depth camera was simulated using the Realsense plugin for the gazebo. With the help of the gazebo model descriptor, the plugin can be configured to simulate the Realsense D435 depth camera. Other parameters specific to the proposed algorithm are listed in Table 1.

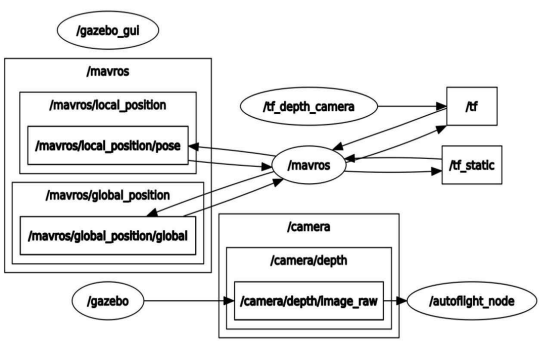
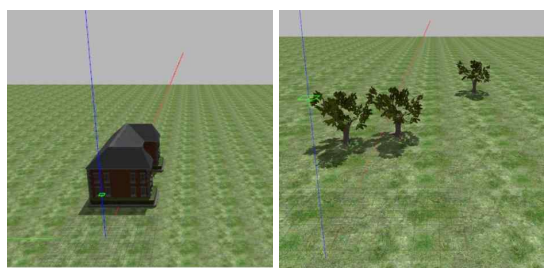


Fig. 6. Communication between the ROS nodes setup in this experiment.



(a) (b)

Fig. 7. Simulation environments with (a) single obstacle and (b) multiple obstacles.

Table 1. The parameters for path planning

Parameter	Description	Value
dist_min_obs	Minimum allowed distance to the obstacle	4.0
r_bubble	Radius of safety bubble around the vehicle	2.0
v_max	Maximum velocity during avoidance	2.5
avoid_step_size	Step size for safe walk within the safety bubble	2.0

5. RESULT AND DISCUSSION

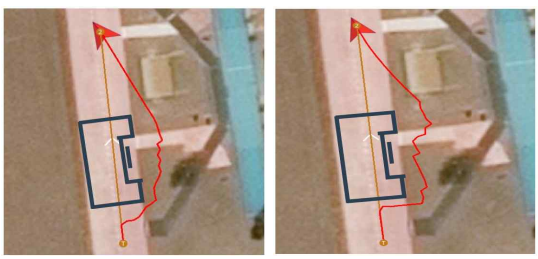
To validate the proposed method, the simulations were carried out in various obstacle arrangements. The Evaluation was done using path visualization, minimum clearance distance to the obstacle, and total traveling distance.

In one of the experiments, we placed a building model in between the takeoff position and the goal position. The vehicle was able to navigate around the building as shown in Fig. 8. The figure shows the flight path from the GCS screen. The black polygon was drawn on the image to display the position of the building in the simulation world. The dark-yellow line represents the planned path from the takeoff point to the target. The red curve is the trajectory generated by the proposed avoidance technique. Fig. 8 (a) and (b) show the path generated by the proposed method and the method presented in [20].

Fig. 9 shows the obstacle arrangement and the generated path for multiple obstacle scenario. Three oak tree models were placed between the take-off and goal position. The red line shows the path generated by the proposed technique avoiding the trees on the way.

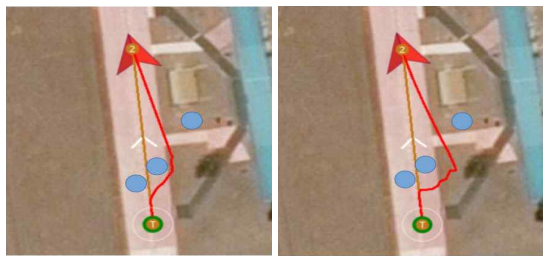
As seen in Fig. 8(a) and 9(a), the proposed technique can efficiently avoid the obstacle with the use of a single low FOV depth camera. In [20] the LIDAR sensor with FOV of 270 degrees. Similarly, the implementation of [20] in a similar environment resulted in the trajectory shown in Fig. 8(b) and 9(b). Although the trajectory in Fig. 8(b) and 9(b) shows a wider margin with the obstacles, our approach avoided the obstacles with the shorter path. Since our method does not have the peripheral sensors, only the obstacles within the horizontal FOV of 85 degrees were visible at a given time. Unlike in [20], this method can not move sideways because of the limited FOV, the yaw has to be adjusted towards generated way-point. Constraining the motion of the vehicle within the FOV and the safety bubble ensures that the vehicle does not jump into unseen obstacles. Fig. 10. shows the change in yaw during the flight. In our approach, the heading angle frequently changes in the presence of the obstacles because of the low FOV.

Fig 11 compares the change in speed during the



(a) (b)

Fig. 8. Avoiding Single Obstacle using (a) the proposed approach and (b) the method proposed by Thanh [20].



(a) (b)

Fig. 9. Avoiding Multiple Obstacle using (a) the proposed approach and (b) the method proposed by Thanh [20].

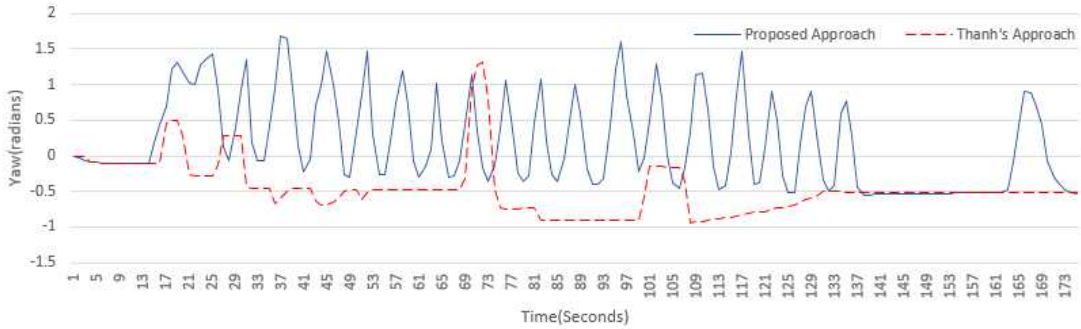


Fig. 10. Change in yaw during obstacle avoidance.

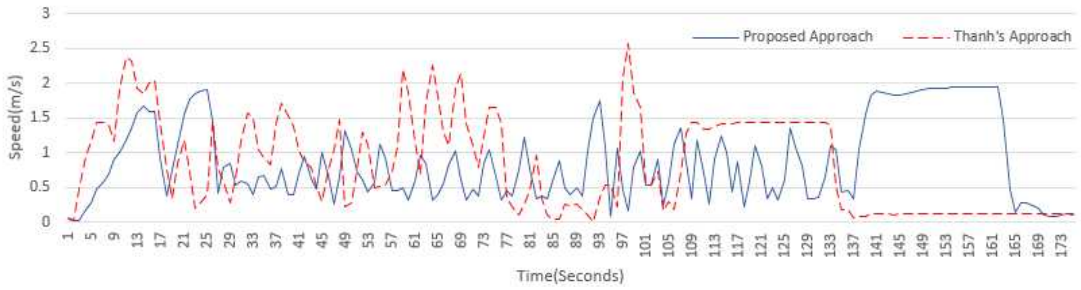


Fig. 11. Speed adaptation during obstacle avoidance.

flight using the proposed approach and approach proposed by [20] for the same mission in a single obstacle scenario. In both cases, the velocity is reduced during avoidance. Once the obstacle is detected the vehicle slows down gradually to avoid gliding towards the obstacle during the avoidance maneuver. The default speed set using QGC is adapted during the avoidance process, and after the avoidance is completed the speed set during mission creation is applied to reach the goal. Although our approach has a lower speed throughout the flight, the overall flight time is similar because of different trajectories.

The velocity during the avoidance was kept low for safe navigation. as this approach makes the restrictive assumptions in terms of speed and field of view. However, the velocity can be adjusted at the cost of the path length as faster velocity would require the drone to react as early as possible. Thus, the required velocity can be achieved by increasing the minimum obstacle distance and increasing the safety bubble radius. From the experiment, it can be concluded the proposed approach is able to avoid the obstacles using a single low FOV camera with comparable speed.

5. CONCLUSION

In this paper, an obstacle avoidance method for UAV is proposed. The presented approach is based on a way-point generation using a polar grid through movement within the safety bubble of the vehicle. The waypoint based navigation is implemented for the Ardupilot-based quadcopter using MAVLink protocol. The depth information is obtained using a depth camera in the form of raw depth image through a ROS message. It is then represented as a polar grid with distances in selected vertical and horizontal angles according to the FOV of the camera. A ROS based simulation environment is implemented with varying arrangements for the obstacles.

The simulated experiments show that the small-sized polar grid representation is effective to obtain safe waypoints for obstacle-free navigation. It also shows that our approach successfully avoids obstacles using a single depth camera with limited FOV.

REFERENCE

[1] S. Zhao, Z. Hu, M. Yin, K.G.Y. Ang, P. Liu, F. Wang, et al., "A Robust Real-time Vision System for Autonomous Cargo Transfer by an Unmanned Helicopter," *IEEE Transactions on Industrial Electronics,* Vol. 62, No. 2, pp. 1210-1219, 2015.

[2] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous Obstacle Avoidance and Maneuvering on a Vision-guided MAV Using On-board Processing," *Proceeding of IEEE International Conference on Robotics and Automation*, pp. 2472-2477, 2011.

[3] Y. Huang, W.C. Hoffman, Y. Lan, W. Wu, and B.K. Fritz, "Development of a Spray System for an Unmanned Aerial Vehicle Platform," *Applied Engineering in Agriculture*, Vol. 26, No. 6, pp. 803-809, 2009.

[4] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint," *IEEE Communications Surveys and Tutorials*, Vol. 18, No. 4, pp. 2624-2661, 2016.

[5] J. Kuffner and S. LaValle, "RRT-connect: An Efficient Approach to Single-query Path Planning," *Proceeding of IEEE International Conference on Robotics and Automation*, pp. 995-1001, 2000.

[6] S. Islam, Q. Huang, F. Afghah, P. Fule, and A. Razi, "Fire Frontline Monitoring by Enabling Uav-based Virtual Reality with Adaptive Imaging Rate," *Asilomar Conference on Signals Systems and Computers*, pp. 1-6, 2019.

[7] C. Martinez, C. Sampedro, A. Chauhan, and P. Campoy , "Towards Autonomous Detection and Tracking of Electric Towers for Aerial Power Line Inspection," *Proceeding of International Conference on Unmanned Aircraft Systems*, pp. 284-295, 2014.

[8] Z. Zhou, C. Zhang, C. Xu, F. Xiong, Y. Zhang, and T. Umer, "Energy-efficient Industrial Internet of UAVs for Power Line Inspection in Smart Grid," *IEEE Transactions on Industrial Informatics,* Vol. 14, No. 6, pp. 2705-2714, 2018.

[9] L. Wang and C. Zhang, "Automatic Detection of Wind Turbine Blade Surface Cracks Based on UAV-taken Images," *IEEE Transactions on Industrial Electronics,* Vol. 64, No. 9, pp. 7293-7303, 2017.

[10] T. Bauman, *Obstacle Avoidance for Drones Using a 3DVFH**, Master's Thesis of Swiss Federal Institute of Technology, 2018.

[11] A. Loquercio, A.I. Maqueda, C.R. del-Blanco, and D. Scaramuzza, "DroNet: Learning to Fly by Driving," *IEEE Robotics and Automation Letters*, Vol. 3, No. 2, pp. 1088-1095, 2018.

[12] S. Shin, Y Kang, and Y. Kim, "Reward-driven U-Net Training for Obstacle Avoidance Drone," *Expert Systems with Applications,* Vol. 143, 113064, 2020.

[13] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L.V. Eycken, "CNN-based Single Image Obstacle Avoidance on a Quadrotor," *Proceeding of IEEE International Conference on Robotics and Automation*, pp. 6369-6374, 2017.

[14] D. Gandhi, L. Pinto, and A. Gupta, "Learning to Fly by Crashing," *IEEE/RSJ International Conference on Intelligent Robots and Systems,* pp. 3948-3955, 2017.

[15] M. Quigley, K. Conley, B. Gerkey, J. Faust, T.B. Foote, J. Leibs, et al., "ROS: An Open-source Robot Operating System," *Proceedings of International Conference on Robotics and*

Automation Workshop on Open Source Software, Vol. 3, No. 3.2, pp. 5, 2009.

[16] N. Koenig and A. Howard, "Design and Use Paradigms for Gazebo and Open-source Multi-robot Simulator," *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems,* pp. 2149-2154, 2004.

[17] P.E. Hart, N.J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernatics*, Vol. 4, No. 2, pp. 100-107, 1968.

[18] A. Chakravarthy and D. Ghose, "Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 28, No. 5, pp. 562-574, 1998.

[19] C. Wang, A.V. Savkin, and M. Garratt, "A Strategy for Safe 3D Navigation of Non-holonomic Robots among Moving Obstacles," Robotica, Vol. 36, No. 5, pp. 275-297, 1998.

[20] H.L Thanh, *An Effective Trajectory Determination for Collision Avoidance of Unmanned Vehicles*, Doctoral Thesis of Sejong University, 2018.

[21] D.H. Kim, J.E. Kim, J.H. Song, Y.J. Shin, and S.S. Hwang, "Image-based Intelligent Surveillance System Using Unmanned Aircraft," *Journal of Korea Multimedia Society,* Vol. 20, No. 3, pp. 437-445, 2017.

Sudarshan Pant

received the B.S., and M.S., in Multimedia Engineering in Mokpo National University, Korea in 2010, and 2012, respectively. He worked at Arihant Technologies, Nepal and TekTak Nepal Pvt Ltd, Nepal as Software Engineer and Mobile Application Architect respectively from 2012 to 2016. He is currently a Ph.D. student in department of Multimedia Engineering at Mokpo National University. His research interests include Computer Vision, Sensors, Robotics, data analysis and Machine Learning.

Sangdon Lee

received his B.S., M.S., and Ph. D. degrees in Computer Engineering from the department of Computer Engineering, Seoul National University, Korea in 1984, 1986 and 1996 respectively. He worked at Research and Development Group in Korea Telecom, Seoul, Korea for 10 years before joining Mokpo National University. He is currently a professor of the Department of Multimedia Engineering. His research interests include big data management, UAVs, multimedia services and intelligent data processing.