iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VIII **Month of publication:** August 2021

DOI: https://doi.org/10.22214/ijraset.2021.37774

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

Path Planning and Obstacle Avoidance for UAV in an Uncharted Environment

Shubhankar Goje[1], Sumeet Singh[2], Omkar Nikhal[3]
[1]*Electronics & Telecommunication Engineering, Pune Institute of Computer Technology, Pune, Maharashtra, India*
[2, 3]*Computer Engineering, Pune Institute of Computer Technology, Pune, Maharashtra, India*

Abstract: ***The growing industry of unmanned aerial vehicles (UAV) requires an efficient and robust algorithm to decide the path of the UAV and avoid obstacles. The study of pathfinding algorithms is ongoing research not just useful in the domain of drones, but in other fields like video games (AI pathfinding), terrain traversal (mapped, unmapped, areal, underwater, land, etc.), and industries that require robots to deliver packages. This paper proposes a new pathfinding algorithm that aims to solve the problem of pathfinding in unknown 2-dimensional terrain. Based on a system of assumptions and using the help of a set of sensors aboard the UAV, the algorithm navigates the UAV from a start point to an endpoint while avoiding any shape or size of obstacles in between. To avoid multiple different types of "infinite loop" situations where the UAV gets stuck around an obstacle, a priority-based selector for intermediate destinations is created. The algorithm is found to work effectively when simulated in Gazebo on Robot Operating System (ROS).***

Keywords: Path Planning, UAV, Obstacle Avoidance, Drone Navigation, Obstacle Detection, Uncharted Environment.

I. INTRODUCTION

In today's technologically advanced world, Unmanned Aerial Vehicles (UAVs) have received a great deal of recognition and research in the military, commercial as well as in agricultural fields. The use of autonomous UAVs in diverse and complex working environments has become increasingly ubiquitous. It is vital for these vehicles to be able to produce collision-free motion plans and also to be able to modify those plans during implementation to cope with contingencies that may occur during its operation [1]. Autonomy refers to the ability of a UAV to execute a decision in dynamic environments based on current sensor-captured knowledge and theoretically encompasses all activities of the vehicle with minimum human interference. One of the most important aspects of UAVs is flight path planning which demands a high accuracy and success rate. Path planning, which allows the UAV to move from the origin to the destination under obstacle restrictions while achieving the shortest path with low energy consumption and minimum runtime, is one of the most crucial challenges. Path planning is primarily responsible for producing a space trajectory that augments the probability that a UAV executes its assigned tasks.

For computing the most optimal path for a UAV, a lot of algorithms have been proposed. Some of these algorithms are based upon achieving certain conditions or criteria like finding the minimum path, executing the task in the least amount of time, etc. Path planning can be classified into two broad categories namely global planning and local planning. Global planning is executed in a static environment consisting of obstacles that are at fixed positions. The path in global planning is decided before the actual motion of the UAV. This type of planning is not suitable for real-life scenarios where the environment is dynamic most of the time. Local planning is based on a dynamic approach that calculates the path based on data provided by the sensors at every moment. If the obstacles are introduced dynamically during its journey from origin to destination, then the UAV has to replan its path accordingly thereby generating a collision-free motion path. Motion planners with dynamic replanning prove to be of significant importance during such cases. [2] presented an approach based on mobile robots used for local path planning which not only eliminates the need for global path planning but also circumvents local minima points for finding the most optimal path.

This paper proposes a novel idea to navigate an autonomous UAV in an unknown two-dimensional space parallel to the ground. The proposed algorithm successfully overcomes cases where the UAV might get stuck in an infinite loop due to the concave shape of obstacles which in turn enables it to work in almost all practical scenarios. The proposed algorithm is designed to work on a UAV with four proximity sensors and as it turns about the yaw axis it is able to map proximity in every direction. This algorithm can easily be modified to work on UAVs with only one proximity sensor.

The rest of the paper is structured as follows: section II talks about the related background work in this area. Section III provides information related to the system model of the UAVs surrounding. Section IV describes the proposed methodology and section V presents the experimentation and obtained results. The conclusions of the paper are drawn in section VI.

II. BACKGROUND WORK

In the domain of path planning for UAVs, a variety of algorithms have been implemented to obtain the most optimal path for a given environment. These algorithms are constantly worked upon by researchers to maximize efficiency and to find the most ideal path.

Some of the works are based on optimal path planning algorithms which incorporate a high computational complexity on a large scale. It is used to ensure that the UAV travels along an efficient path in minimum time. [3] Algorithms based on mathematical programming and parameter optimization are based on optimal path planning. Traditional path planning algorithms consist of Visibility Graph [4], Artificial Potential Field [5], Simulate Anneal Arithmetic algorithm [6], and Fuzzy Logic Algorithm [7]. [25] demonstrated the convenience and efficiency of the FVF(Fuzzy Virtual Force) method in finding the path efficiently. In a convoluted environment, FVF works more efficiently than A* algorithm. Other popular obstacle avoidance algorithms for path planning are bugs algorithm [28], vector field histogram (VFH) [29], and artificial potential field (APF) [30]. Additional works like Dijkstra's algorithm [8], Floyd algorithm [9], and A* algorithm are based upon heuristic approaches which have considerable advantages in terms of efficiency. [24] demonstrated that EDA (Elevation-Based Dijkstra Algorithm) reduces calculation time considerably. [26] presented a reinforcement learning algorithm named Q-learning that eliminated obstacles until the UAV reached the destination. [27] implemented radar that provided real-time feedback on the target and determined the later motion state of the target according to its position. In addition, by integrating the feedback data and the state estimation result, it proposed a dynamic path planning. Moreover, intelligent bionic path planning algorithms, discovered through bionic research, have been encompassed by the following works. [15] makes use of a genetic algorithm that models the vehicle path as a sequence of speed and heading transitions occurring at discrete times during a dynamically changing environment. [14] Genetic algorithms have significant global searching maneuverability which instantly finds all of the solutions without falling into local optimal. Other algorithms include Ant Colony Optimization [10]-[12], Particle Swarm Optimization [13], and Neural Network Algorithm [16]. Ant-based algorithms, which are efficient in finding better solutions, have been implemented to solve many intricate problems, such as the traveling salesman problem, data mining, quadratic assignment problem, data clustering, and image retrieval. [23] implemented swarm intelligence-based method for UAVs' route optimization.

Other works are based upon sampling-based approaches which include the Probabilistic Roadmap Method (PRM) [17] and Rapidly-exploring Random Trees (RRTs) [18]-[20] which emphasizes more on performance and tractability over the complete execution of tasks. [21] introduces a new approach using the combination of different data structures and algorithms that incorporate efficient obstacle representation utilizing an efficient Quadtree data structure, using the modified Random Tree (RRT) Rapidly Exploring Algorithm and using a Dijkstra algorithm based path pruning technology, to quickly discover feasible sub-optimal. [22] integrates both sample-based motion planning techniques like Probabilistic Roadmaps and Rapidly Exploring Random Trees and presents a motion planning framework for a fully deployed autonomous unmanned aerial vehicle.

III. SYSTEM MODEL AND ASSUMPTIONS

Following are the assumptions that are necessary for understanding the algorithm.

A. *Model Assumptions*

1) The device in use is a UAV (Unmanned Aerial Vehicle) that can freely rotate in roll, pitch, and yaw axes that allow it to move in any direction on a 2-D plane, stay stationary in mid-air, and change its altitude using the throttle.
2) The UAV is equipped with four rangefinders facing front, right, back, and left that can detect any obstacle in the range of R meters.
3) The UAV is equipped with an IMU (Inertial Measurement Unit) sensor that measures and reports the UAVs' orientation. The IMU calculates three orientations: roll, pitch, and yaw. The yaw is calculated in degrees and it is 0° when the UAV is facing North, increases in a clockwise direction, and 180° when facing South.
4) The UAV is equipped with GPS sensors that calculate the UAVs' location in terms of latitude, longitude, and altitude.

B. *Environment Assumptions*

1) The environment is a 2D plane situated above ground and has a reachable target point.
2) The environment may contain a finite number of 2D obstacles that can be assumed as closed curves with finite boundaries and area.
3) The position, size, and shape of the obstacles within the environment are not known unless the UAVs' rangefinders can detect the obstacles, in which case only a single point of the whole obstacle is detected by the UAV.

C. Variable Descriptions

1) R: Maximum range of the rangefinders within which the UAV can detect obstacles.

2) $\boldsymbol{\tau}$: The tolerance, i.e., The minimum distance the UAV can be at from an obstacle without crashing into it.

3) Yaw: Angular displacement of the UAV calculated clockwise from the North axis.

4) Pitch: Angular displacement of the UAV that tells its forward inclination.

5) Roll: Angular displacement of the UAV that tells its sideways inclination.

6) S: List of points that form the UAVs' path that it takes to reach the destination.

$S = \{S_0, S_1, S_2, \ldots, S_T\}$

Each point in S is termed as a "set-point".

S_0: The first set-point of the UAVs' path. That is, the location the UAV starts its journey at.

S_T: The last set-point of the UAVs' path. That is the target location or the destination.

S_i: The i^{th} set-point the UAV calculates to reach the destination. Where i ranges from 1 to (T-1).

7) X_{Si}, Y_{Si}: X_{Si} is the latitude and Y_{Si} is the longitude of S_i.

8) θ_D: The shortest distance from the center of the UAV to the target point.

9) $I_0, I_1, I_2, \ldots, I_{15}$: These are the names of 16 equally spaced points around the UAV at 0^o, 22.5^o, 45^o, …,337.5^o in a circle of a radius of R-$\boldsymbol{\tau}$ as shown in Fig 1.

10) $D_0, D_1, \ldots, D_{15}$: These are the names of 16 equally spaced points around the UAV at 11.25^o, 33.75^o, 56.25^o, …,348.75^o in a circle of a radius of R-$\boldsymbol{\tau}$ as shown in Fig 1.

11) $L_1, L_2, \ldots, L_{15}$: L_i provides the latitude and longitude of the i^{th} point. Consider X_i as the latitude of the i^{th} point (I_i) and Y_i as the longitude of the i^{th} point I_i. S_0 their position will be $L_i = (X_i, Y_i)$.

12) $Ir_0, Ir_1, \ldots, Ir_{15}$: For $I_0, I_1, I_2, \ldots, I_{15}$, the UAV will try to detect the presence of obstacles between the UAV and the point using the rangefinders. This calculation will return the distance to the obstacle. For points that are not blocked by an obstacle, the rangefinder will return infinity, otherwise, if an obstacle is present, the rangefinder will return a value less than R (max range of detection). There will be 16 such distances for each point, so Ir_i = distance to obstacle present between the UAV and I_i (infinity if no obstacle is present).

13) $Dr_0, Dr_1, \ldots, Dr_{15}$: For $D_0, D_1, \ldots, D_{15}$, the UAV will try to detect the presence of obstacles between the UAV and the point using the rangefinders. This calculation will return the distance to the obstacle. For points that are not blocked by an obstacle, the rangefinder will return infinity, otherwise, if an obstacle is present, the rangefinder will return a value less than R (max range of detection). There will be 16 such distances for each point, so Dr_i = distance to obstacle present between the UAV and I_i (infinity if no obstacle is present).

14) $\theta_0, \theta_1, \ldots, \theta_{15}$: For the above 16 points, the distance from each point to the target point is represented by these 16 distances. S_0, θ_i = Distance from I_i to the target point.

15) $IP_0, IP_1, IP_2, \ldots, IP_{15}$: For $I_0, I_1, I_2, \ldots, I_{15}$, if there is a possibility for the UAV to go to the point directly without crashing into an obstacle, then IP for that point will be 1, otherwise, it will be 0. To calculate this, the following formula will be used:

$$IP_i = 0, if\ Ir_i < R - \tau \quad and \quad IP_i = 1, if\ Ir_i \geq R - \tau$$

16) $DP_0, DP_1, DP_2, \ldots, DP_{15}$: For $D_0, D_1, \ldots, D_{15}$, if there is a possibility for the UAV to go to the point directly without crashing into an obstacle, then DP for that point will be 1, otherwise, it will be 0. To calculate this, the following formula will be used:

$$DP_i = 0, if\ Dr_i < R - \tau \quad and \quad DP_i = 1, if\ Dr_i \geq R - \tau$$

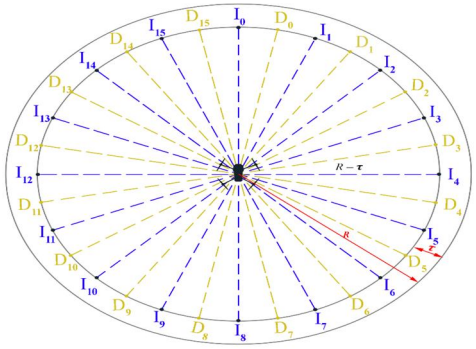


Fig. 1 Visualization of Proximity Sensor Data

IV. PROPOSED METHODOLOGY

By taking into consideration the above-mentioned assumptions and variables, the algorithm for the drone to navigate the environment and reach the target can be defined using five steps. The first step prepares the UAV for moving towards the target. The second step moves the UAV towards the target and detects obstacles. The third step is about avoiding a detected obstacle. The fourth step uses a priority loop to select the next point of course. Finally, the last step moves the UAV towards the next point of course.

A. Step 1. Getting the UAV in the initial position.

Initially, the UAV starts from position S_0. The location (Latitude and Longitude) is stored in the list S as (X_{S0}, Y_{S0}). The UAV is tasked to go to Target Location $S_T(X_{ST}, Y_{ST})$. Then the UAV is pointed towards the target by turning its yaw using the formula.

B. Step 2. Moving towards the target and detecting obstacles.

As the UAV is already pointing towards the target, it is pitched forward to move towards the target. While moving, the UAV constantly checks for obstacles in front of it using the rangefinder that is pointing ahead. If an obstacle is detected and the detected range is less than $R - \tau$ and θ_D, then the algorithm moves to Step 3. If no obstacle is detected in the entire journey and the UAV reaches the target, then the final coordinates of the target ($S_T = (X_{ST}, Y_{ST})$) are appended to list S. At this point, list S contains the path the UAV took and the UAV is at its target location, so the algorithm Ends.

C. Step 3. Avoiding obstacles and filling sensor data.

As soon as an obstacle is detected in front of the UAV, the UAV is stopped. This new position is appended to list S. The four rangefinders (front, back, left, and right) simultaneously provide the proximity of the UAV from obstacles. These sensors are used to fill and calculate data about the points I_0, I_1, I_2, …, I_{15} and D_0, D_1, ..., D_{15}. Initially, when the UAV is pointing towards the obstacle, the rangefinders (front, back, left, and right) will provide values that will be stored in Ir_0, Ir_8, Ir_{12}, and Ir_4 respectively as shown in Fig 2. The UAV starts to rotate clockwise. As the yaw reaches 11.25°, the four rangefinders (front, back, left, and right) provide the values of Dr_0, Dr_8, Dr_{12}, and Dr_4 as shown in Fig 3. At 22.5°, the rangefinders provide the values of Ir_1, Ir_9, Ir_{13}, and Ir_5. At 33.75°, the rangefinders provide the values of Dr_1, Dr_9, Dr_{13}, and Dr_5. At 45°, the rangefinders provide the values of Ir_2, Ir_{10}, Ir_{14}, and Ir_6. At 56.25°, the rangefinders provide the values of Dr_2, Dr_{10}, Dr_{14}, and Dr_6. At 67.5°, the rangefinders provide the values of Ir_3, Ir_{11}, Ir_{15}, and Ir_7. At 78.75°, the rangefinders provide the values of Dr_3, Dr_{11}, Dr_{15}, and Dr_7. At this point, it will stop rotating and we will have all the values from Ir_0 to Ir_{15} and Dr_0 to Dr_{15}. These values will tell the position of the obstacles relative to the UAV.

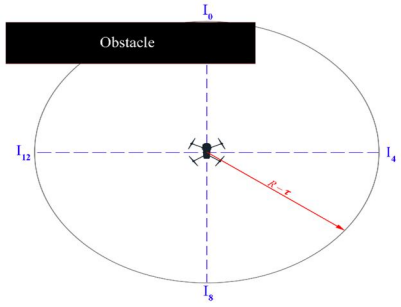


Fig. 2 Sensors Filling Information into Ir0, Ir8, Ir12, and Ir4

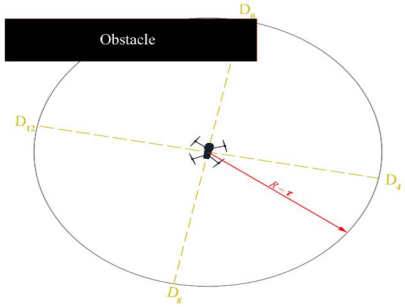


Fig. 3 Sensors Filling Information into Dr0, Dr8, Dr12, and Dr4

D. Step 4. Selecting the appropriate set-point (S_i).

The UAV will calculate the distance to the target point from its center and store it in θ_D. It will also calculate the distance from the target to each of the points I_0, I_1, I_2, …, I_{15}, and these distances will be stored in θ_0, θ_1, …,θ_{15}.

The θ_i values will be sorted in ascending order to obtain the point closest to the target.

For each point in I_0, I_1, I_2, …, I_{15} and D_0, D_1, ..., D_{15}, the possibility of reaching the point is calculated and stored in IP_0, IP_1, IP_2, ..., IP_{15} and DP_0, DP_1, DP_2, ..., DP_{15} respectively in the form of 0 or 1 as specified by the formula in the variable description.

1) Step 4a: Avoid already visited points and avoid near-miss of obstacle

In the sorted list of θ, the smallest value is selected and the corresponding I is checked for possibility using the variable IP. For I_i, if IP_i is 1, then DP_i and DP_{i+1} (i+1=0 if i = 15) are checked, if they are 1 as well, then it is checked if I_i has not been visited before using the list S. If all the conditions are met, then the UAV selects I_i as the next set-point and moves on to Step 5. If it doesn't satisfy any of these criteria, the next smallest point is checked for the same criteria. If by the end of the list θ, no point is selected, the algorithm moves on to Step 4b.

2) Step 4b: Avoid already visited and allow near-miss of obstacle

In the sorted list of θ, the smallest value is selected and the corresponding I is checked for possibility using the variable IP. For I_i, if IP_i is 1, then it is checked if I_i has not been visited before using the list S. If all the conditions are met, then the UAV selects I_i as the next set-point and moves on to Step 5. If it doesn't satisfy any of these criteria, the next smallest point is checked for the same criteria. If by the end of the list θ, no point is selected, the algorithm moves on to Step 4c.

3) Step 4c: Allow already visited and avoid near-miss of obstacle

In the sorted list of θ, the smallest value is selected and the corresponding I is checked for possibility using the variable IP. For I_i, if IP_i is 1, then DP_i and DP_{i+1} (i+1=0 if i = 15) are checked, if they are 1 as well, then all the conditions are met and the UAV selects I_i as the next set-point and moves on to Step 5. If it doesn't satisfy any of these criteria, the next smallest point is checked for the same criteria. If by the end of the list θ, no point is selected, the algorithm moves on to Step 4d.

4) Step 4d: Allow already visited and allow near-miss of obstacle

In the sorted list of θ, the smallest value is selected and the corresponding I is checked for possibility using the variable IP. For I_i, if IP_i is 1, then the UAV selects I_i as the next set-point and moves on to Step 5. If it doesn't satisfy any of these criteria, the next smallest point is checked for the same criteria. If by the end of the list θ, no point is selected, the UAV is stuck in a confined space and the target is unreachable.

E. Step 5: Moving to the next set-point.

At this point, the UAV has selected a set-point that it can move to. This point is added to list S as the next point in the path.

If the selected point is I_0, then the UAV needs to move towards the target and has successfully avoided the current obstacle, so the algorithm moves to step 2 to continue the UAVs' journey.

If the selected point is anything other than I_0, then the UAV moves to the selected point, then stops when it reaches the selected point, and finally rotates about the yaw axis to face the target. The algorithm moves to step 3 to continue the journey of the UAV.

V. EXPERIMENTATION AND RESULTS

The proposed 2D path planning algorithm for UAVs is implemented and simulated. The simulation environment consists of Gazebo Simulator with Robot Operating System (ROS). "ROS is a flexible framework for writing robot software. In addition to the core middleware components, ROS provides common robot-specific libraries and tools" that helped us communicate with our UAV by sending it information like propeller speeds and receiving information like proximity sensor data, GPS location, and its camera data for computer vision applications. Gazebo is an open-source 3D robotic simulator that enabled the rendering of the UAV model in a virtual city. The simulator has a reliable physics engine that is able to simulate forces like gravity, inertia, and thrust. The models are compatible with Python 2 which was used to program the UAV.

As the drone starts navigating, it attains an elevation of 10 meters more than the greater of the source and destination elevations. The UAV now has to navigate from source to destination avoiding obstacles like buildings, vehicles, billboards, street lights, flyovers, antennas, towers, etc.

The following figure (Fig 4) is the top view of a virtual city showing obstacles, source, and destination.

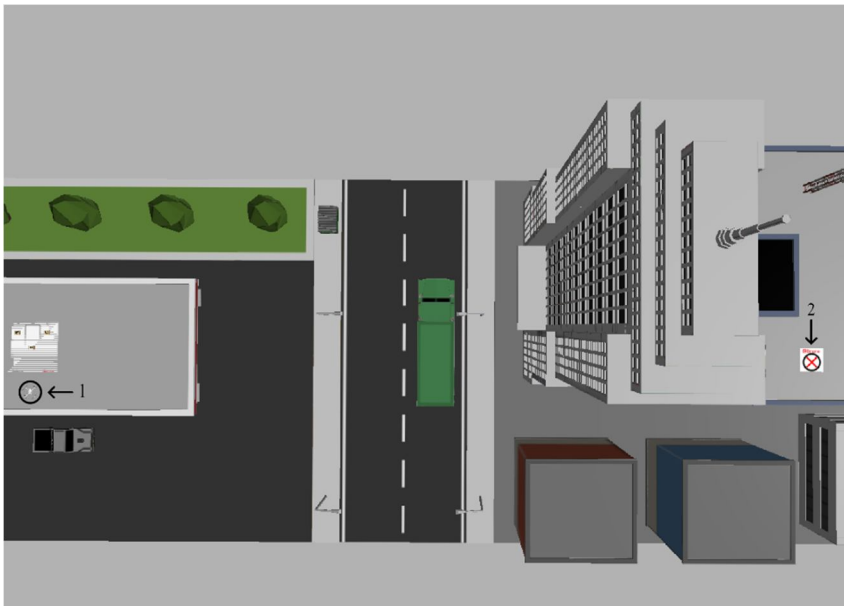


Fig. 4 Top view of simulation representing the UAV as (1) and destination as (2)

The drone begins by moving towards the destination. Fig. 5 shows a close up of the drone while it moves towards the destination, and Fig. 6 is a wide angle side view of the same with the obstacle (building).

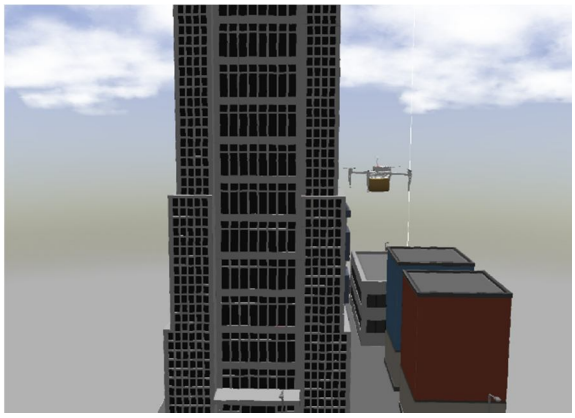


Fig. 5 Close up image of UAV moving towards destination

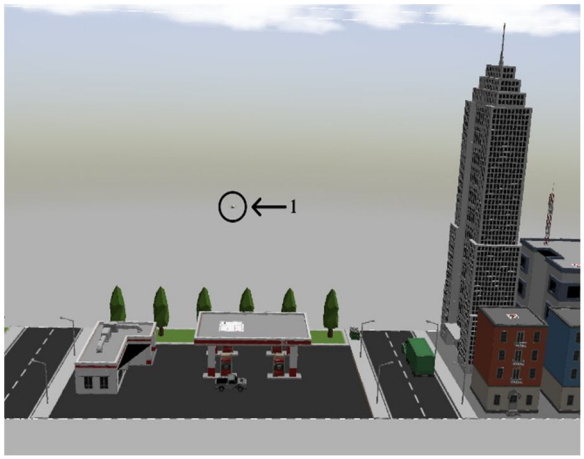


Fig. 6 Side View of UAV (1) moving towards destination.

It then detects the obstacle and stops as shown in Fig. 7. The UAV comes to a halt in front of the obstacle and starts scanning the environment around itself.

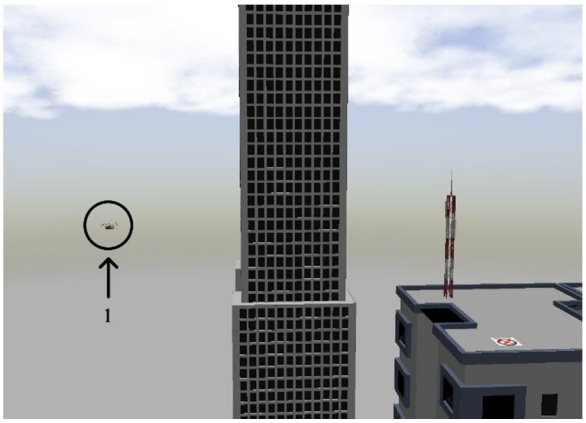


Fig. 7 Side view of UAV (1) scanning the environment

Fig. 8 shows the UAV scanning the environment around itself after stopping in front of the obstacle. Initially the UAV scans proximity at 0^o, 90^o, 180^o, and 270^o using the four perpendicular scanners. After this, the UAV rotates by 11.25^o and scans in the four directions again. This repeats for 22.5^o, 33.75^o, 45^o (Fig. 9), 56.25^o, 67.5^o, 78.75^o, 90^o (Fig. 10). The UAV is able to scan in 32 equally spaced directions using this method.

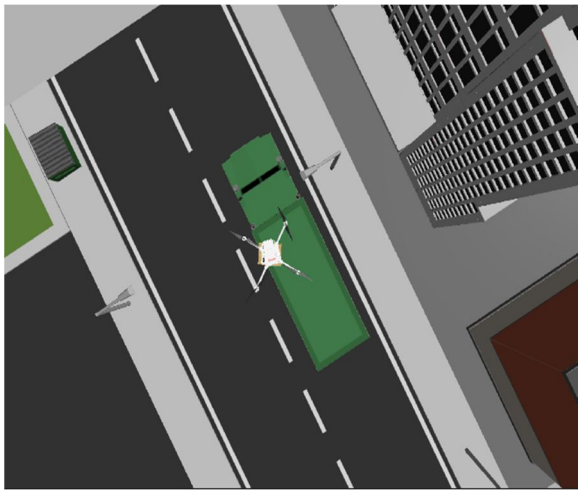


Fig. 8 Top view of UAV scanning proximity in four directions

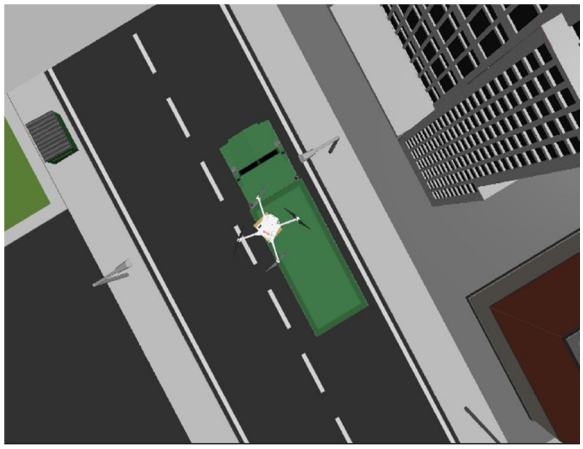


Fig. 9 Top view of UAV rotated 40 degrees about yaw scanning proximity in four other directions

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

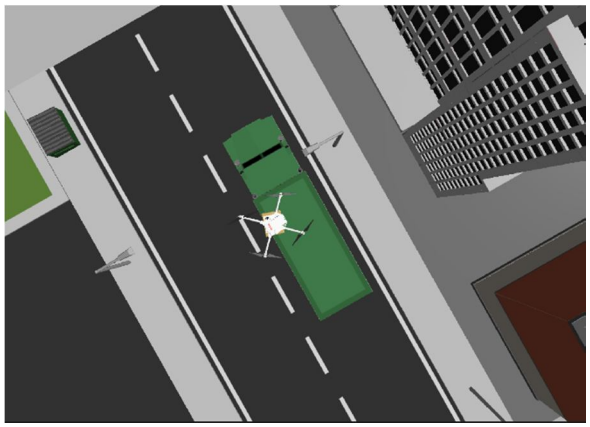


Fig. 10 Top view of UAV rotated 90 degrees about yaw scanning proximity in four more directions

The UAV then starts moving towards the next set point as exhibited by Fig. 11. Fig. 12 is an arbitrary position of the UAV as it moves towards the set point. Fig. 13 shows the UAV decelerating as it reaches the set-point. It scans the location again and finds that the obstacle still exists between itself and the destination, so it moves towards the next best set point. After that, it doesn't find any obstacle between itself and destination and it starts moving towards the destination.

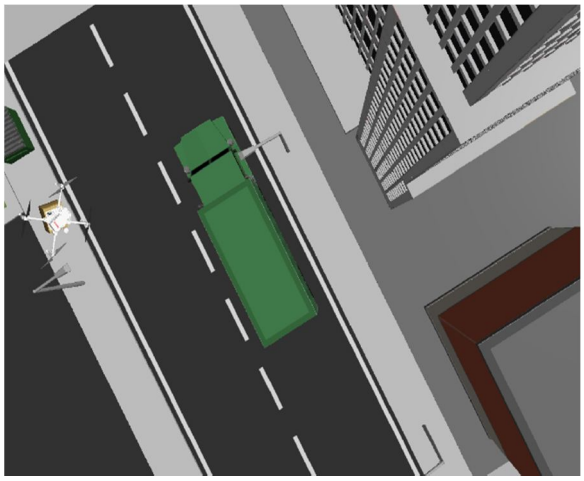


Fig. 11 Top view of UAV starting to move in the direction of the next set point

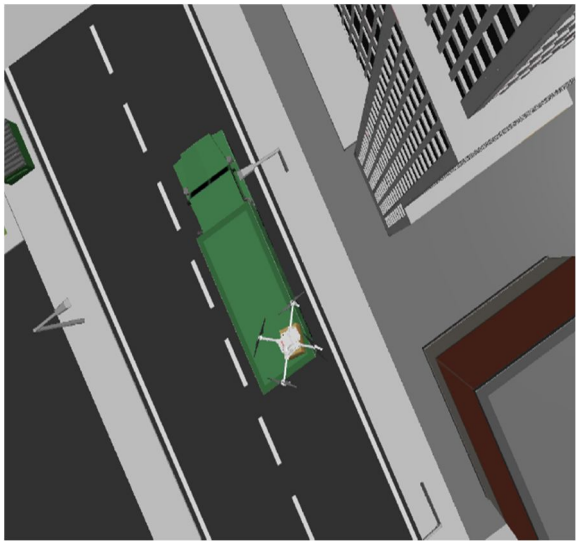


Fig. 12 Top view of UAV moving in the direction of the next set point

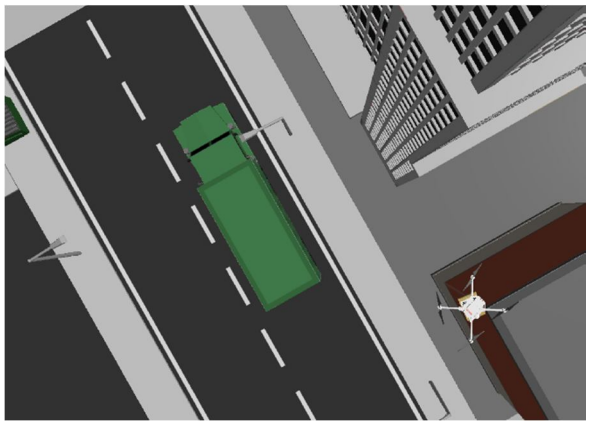


Fig. 13 Top view of UAV decelerating to stop at the next set point

Upon reaching the destination, it scans for the landing pad and lands on it as visible in Fig. 13.

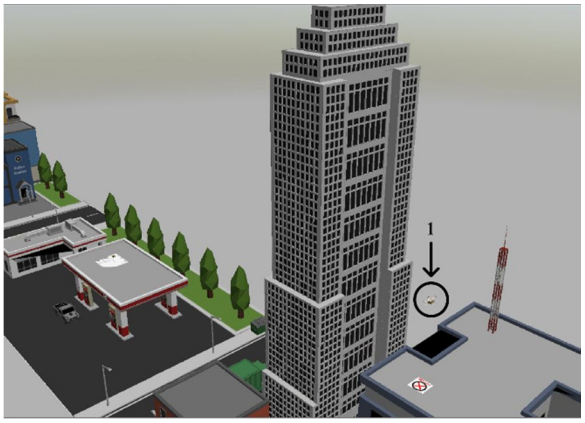


Fig. 14 Side view of UAV (1) at a halt on top of the destination

Fig. 15 is an illustration of the path the UAV took during the simulation. S_0 is the starting point of the UAV. As the simulation begins, the UAV knows the coordinates of the destination, so it starts to move towards it. This movement is represented by the line S_0S_1. During the movement the UAV detects the presence of an obstacle and starts to decelerate. It comes to a halt at position S_1. After scanning the environment, the algorithm provides the coordinates of S_2 as the next set point to move to and the UAV proceeds to do so. Upon reaching S_2 the UAV detects the obstacle between itself and the destination, so the algorithm provides the coordinates of S_3 as the next set point. The UAV moves to S_3 and stops. Finally, the UAV is able to detect no obstacles within the range of R meters between itself and the destination. It starts to move towards the destination and keeps scanning if an obstacle comes in range in front of the UAV. No obstacle is detected during the entire path and the UAV is able to reach the destination.

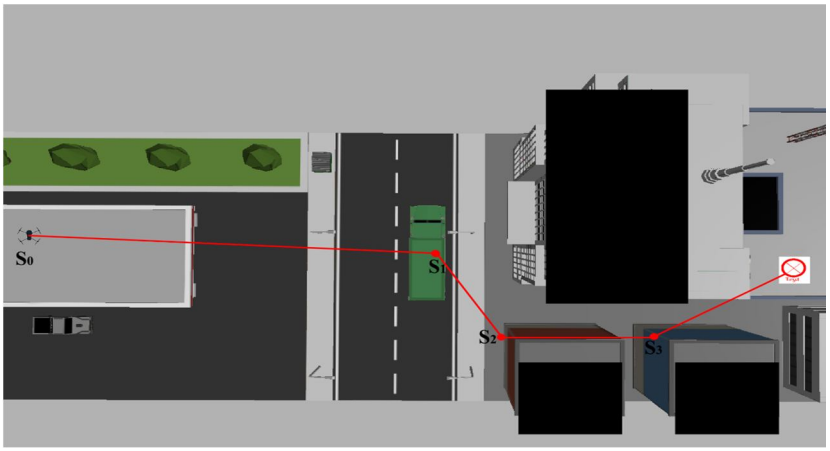


Fig. 15 Illustration of the path taken by the UAV from first set-point to destination

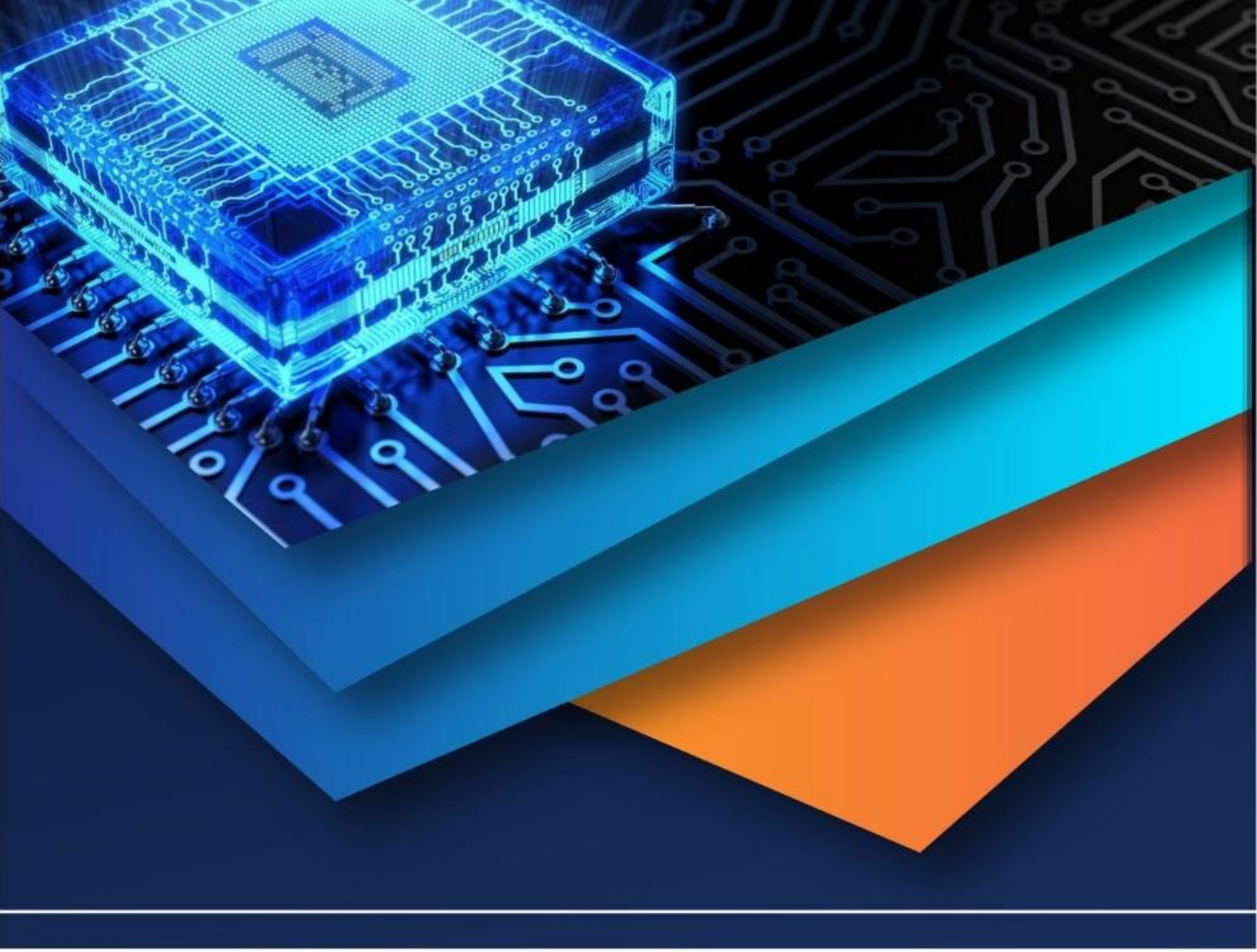
VI. CONCLUSIONS

The application of Unmanned Aerial Vehicles (UAVs) is worldwide and is implemented in a diversity of fields. The proficiency to plan a path for the UAV without encountering any obstacles is of utmost importance for it to function efficiently and with competence. The UAV should be capable of revising its path in accordance with the alterations made in its environment. This paper presents an algorithm that detects obstacles and navigates the UAV around the obstacles to reach its target destination. The algorithm also proves to be successful in an environment containing concave obstacles thereby solving the problem of the UAV getting stuck in an endless loop. The proposed algorithm is applied to a UAV and the implementation has been performed in a simulation environment.

REFERENCES

[1] Mariusz Wzorek and Patrick Doherty. 2006. Reconfigurable Path Planning for an Autonomous Unmanned Aerial Vehicle. In Proceedings of the 2006 International Conference on Hybrid Information Technology - Volume 02 (ICHIT '06). IEEE Computer Society, USA, 242–249.
[2] Hoc, Nguyen & Le, Hai. (2016). Path planning and Obstacle avoidance approaches for Mobile robot.
[3] Zhangjie Fu, Jingnan Yu, Guowu Xie, Yiming Chen, Yuanhang Mao, "A Heuristic Evolutionary Algorithm of UAV Path Planning", Wireless Communications and Mobile Computing, vol. 2018, Article ID 2851964, 11 pages, 2018.
[4] C. Chen, J. Tang, and Z. Jin, “A Path Planning Algorithm for Seeing Eye Robots Based on V-Graph,” Mechanical Science and Technology for Aerospace Engineering, 2014.
[5] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” International Journal of Robotics Research, vol. 5, no. 1, pp. 90–98, 1986.
[6] A. E.-S. Ezugwu, A. O. Adewumi, and M. E. Frˆıncu, “Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem,” Expert Systems with Applications, vol. 77, pp. 189–210, 2017.
[7] D. Adhikari, E. Kim, and H. Reza, “A fuzzy adaptive differential evolution for multi-objective 3D UAV path optimization,” in Proceedings of the 2017 IEEE Congress on Evolutionary Computation, CEC 2017, pp. 2258–2265, Spain, June 2017.
[8] S. Wang X, X. Wu Z, and X. Wang S XWu Z, “Improved Dijkstra shortest path algorithm and its application,” Computer Science, vol. 39, no. 5, pp. 223-222, 2012.
[9] Z. He and L. Zhao, “The comparison of four UAV path planning algorithms based on geometry search algorithm,” in Proceedings of the 9th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2017, pp. 33– 36, China, August 2017.
[10] X. F Wan, W. Hu, W. F. Fang et al., “Research on path planning of robot based on improved ant colony algorithm,” Computer Engineering and Applications, vol. 50, no. 18, pp. 63–66, 2014.
[11] X. Chen, Y. Kong, X. Fang, and Q. Wu, “A fast two-stage ACO algorithm for robotic path planning,” Neural Computing and Applications, vol. 22, no. 2, pp. 313–319, 2013.
[12] T. Zhao, X. Pan, and Q. He, “Application of dynamic ant colony algorithm in route planning for UAV,” in Proceedings of the 7th International Conference on Information Science and Technology, ICIST 2017, pp. 433–437, Viet Nam, April 2017.
[13] M. D. Phung, C. H. Quach, T. H. Dinh, and Q. Ha, “Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection,” Automation in Construction, vol. 81, pp. 25–33, 2017.
[14] G. Ji, “A survey of Genetic Algorithm,” Computer Applications and Software, vol. 21, no. 2, pp. 69–73, 2004.
[15] GAO, XIAO-GUANG & FU, XIAO-WEI & Chen, Daqing. (2005). A Genetic-Algorithm-Based Approach to UAV Path Planning Problem.
[16] W. Wang, S. M. Wei, Y. Q. Yang, Y. F. Jiang, and L. I. DuanLing, “Path planning for a mobile robot using neural networks,” Journal of Beijing University of Technology, vol. 45, no. 10, pp. 221–225, 2009.
[17] Kavraki, L. E.; Svestka, P.; Latombe, J.; and Overmars, ˇ M. H. 1996. Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces. Proc. of the IEEE Transactions on Robotics and Automation 12(4):566–580.
[18] Martin, S. R., Wright, S. E. and Sheppard, J. W. “Offline and Online Evolutionary Bi–Directional RRT Algorithms for Efficient Re–Planning in Dynamic Environments”, Proceedings of the IEEE International Conference on Automation Science and Engineering, Scottsdale, AZ, 22–25 Sep. 2007, Vol. 2, pp. 1131–1136
[19] Zhu, Q., Wu, Y., Wu, G. and Wang, X. “An improved anytime RRTs algorithm”, International Conference on Artificial Intelligence and Computational Intelligence, Shanghai, China, 7–8 Nov. 2009, pp. 268–272.
[20] Clifton, M., Paul, G., Kwok, N., Liu, D. and Wang, D. “Evaluating Performance of Multiple RRTs”, IEEE conference on Mechatronic and Embedded Systems and Application, Beijing, China, 12–15 Oct. 2009, pp. 564–569.
[21] Amin, Jayesh & Boskovic, Jovan & Mehra, Raman. (2006). A Fast and Efficient Approach to Path Planning for Unmanned Vehicles. Collection of Technical Papers - AIAA Guidance, Navigation, and Control Conference 2006.
[22] Wzorek, Mariusz & Doherty, Patrick. (2006). Reconfigurable Path Planning for an Autonomous Unmanned Aerial Vehicle. 438-441.
[23] Jevtić, Aleksandar & Andina, Diego & Jaimes, Aldo & Gomez, Jose & Jamshidi, Mo. (2010). Unmanned Aerial Vehicle route optimization using ant system algorithm. 2010 5th International Conference on System of Systems Engineering, SoSE 2010. 1 - 6.
[24] Felipe Leonardo Lobo Medeiros and Jose Demisio Simoes da Silva. (2010). A Dijkstra Algorithm for FixedWing UAV Motion Planning Based on Terrain Elevation. Springer-Verlag Berlin Heidelberg, 213-222.
[25] Dong Zhuoninga, Zhang Rulin, Chen b Zongjia and Zhou Ruia. (2010). Study on UAV Path Planning Approach Based on Fuzzy Virtual Force. Chinese Journal of Aeronautics23, 341-350.
[26] Jinbae Kim, Saebyuk Shin, Juan Wu, Shin-Dug Kim and Cheong-Ghil Kim. (2017). Obstacle Avoidance Path Planning For Uav Using Reinforcement Learning Under Simulated Environment. IASER 3rd International Conference.
[27] BO Wang, Jianwei Bao, Li Zhang and Qinghong Sheng. (2018). UAV autonomous path optimization simulation based on radar tracking prediction. EURASIP Journal on Wireless Communications and network.

[28] J. Ng and T. Bräunl, "Performance comparison of bug navigation algorithms," J. Intell. Robot. Syst., vol. 50, no. 1, pp. 73–84, 2007.
[29] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," IEEE Trans. Robot. Autom., vol. 7, no. 3, pp. 278–288, 1991.
[30] J. Guldner and V. I. Utkin, "Sliding mode control for gradient tracking and robot navigation using artificial potential fields," IEEE Trans. Robot. Autom., vol. 11, no. 2, pp. 247–254, 1995.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY