RRT* based path planning for Quadrotors

Pradeep Gopal[1] , Arjun Srinivasan Ambalam[2] and Ashwin Kumar Prabhakaran[3]
A James Clark School of Engineering,
University Of Maryland, MD-20742

Abstract— **The aim of our project is to implement the path planning algorithm presented in the paper "RRT-based 3D Path Planning for Formation Landing of Quadrotor UAVs" and upgrade the path planning algorithm to an optimised RRT* algorithm and simulate the Quadrotor motion in Gazebo using ROS (Option 1)**

I. INTRODUCTION

Quadrotor UAV's are considered to be reliable, simple and easy to manipulate aerial vehicles. But highly stable automated path planning and navigation for quadrotors are still a topic under research. In literature many researchers have proposed many techniques for path planning for vehicles in 2D space[1][2]. But there is a necessity for a robust path planning algorithm for quad rotors in 3D space while considering all constraints. Path-planning is an important primitive for autonomous mobile robots that lets robots find the shortest – or otherwise optimal – path between two points. The geometry of Robot and the obstacles is defined in a 2D or 3D configuration workspace and the motion is represented by a path in the configuration space. The field of path planning and navigation has found a wide range of interest among researchers since it started finding its application in different fields such autonomy, automation, video games and architectural designs [3]. Many systematic methods and probabilistic methods in path planning methods have been introduced in the past few decades [4-7]. Sampling based path planning are among the latest and popular path planning algorithms. These algorithms work well for high-dimensional configuration spaces, because unlike combinatorial algorithms, their running time is not (explicitly) exponentially dependent on the dimension of the map. These algorithms are generally considered computationally less complex, works well without any knowledge about the obstacle space, probabilistically complete, meaning the probability that they will produce a solution approaches 1 as more time is spent. However, they cannot determine if no solution exists. Many researchers have presented an extensive comparative study and analysis of a number of sampling based techniques over the years [8], [9]. Rapidly Exploring Random Tree (RRT) [10], is one of the quickest algorithms in finding the first path but it does not ensure asymptotic optimality [11]. Although RRT achieves probabilistic convergence, it does not give an optimal path. A recent upgrade to this technique called RRT* proves to gives optimal path while maintaining probabilistic completeness, but at the cost of slow convergence rate. Research to increase the efficiency of RRT has been in place for a long time and various researchers have presented different methods including potential function planner [12], density avoided sampling [13] including variations of such planners as well. This paper addresses these issues by introducing path optimisation and early stopping techniques to RRT*. Moreover, it uses intelligent sampling to give an optimum or near optimum path at a very fast rate of convergence and reduced execution time. The solution obtained also facilitates the robot to track the trajectory as it is straighter and with less way points. Thus, it gives a more efficient path planning solution as compared to the normal RRT* without optimization. The algorithm proposed, in this manuscript, demonstrates these distinguishing features through experimental results and performance analysis. Since the RRT* gives us an optimal solution with probabilistic completeness, we have chosen to implement on a 3D space for a quadrotor. Robot Operating System (ROS) and Gazebo tool have been employed to test and analyze the robustness of the algorithm.

This paper is organized as follows: Section II explains about the methodology that has been involved to obtain the optimized path. The simulation results and other experimental results are discussed in section III. The final section consists of the conclusion and future scope of this project.

II. METHOD

A. See the Goal Point and Maximum Step Propagation:

In the general RRT$*$ algorithm, the exploration of nodes takes place by finding the new nodes only in the unexplored areas randomly. The expansion for searching of new nodes continues unless there is a definite space which has been defined around the goal point. Only when this space is reached the exploration will stop. In the paper considered, a different type of RRT algorithm is implemented to stop the exploration of nodes. For each new node that has been added, a straight line is connected to the goal node. If there is no obstacle space which is hindering this straight line from the new node then the algorithm stops right away. If an obstacle is found, the exploration continues till it finds a point which can be directly connected with the goal point. The Fig.1 gives a clear idea on how the algorithm works when the goal node is directly connected and terminates the exploration algorithm.

The step propagation during the direct connect to goal node is set to some predefined value and will not exceed that. As a feasible path is already found, the time taken for the exploration depends on the step size. If the step size is

Algorithm 1 Algorithm for finding optimal path using RRT*

```
1:  Add start point; Assign start point -> Root node
2:  Repeat
3:  if (Goal is visible from the new-node) then
4:      1. Connect the node to the goal using maximum step
    propagation
5:      2. Break repeat
6:  else
7:      Generate Random seed within the working space
8:      for Nodes already existing in the tree do
9:          1. Calculate Euclidean Distance between the
10:         nodes in the tree and the new node.
11:         2. Add this distance with the euclidean distance
12:         from each node to the root of the tree
13:         3. Find the parent node with the minimum
14:         total distance
15:     end for
16:     do
17:         Navigate tree from parent node to the new node
18:     while
19:       Navigation goes out of the working space or
20:       Navigation collides with obstacles or
21:       Maximum length is met or
22:       Navigation reaches the new node
23:     end while
24:     Add the navigation stop point to tree as new node
25:     if Rewiring nodes within a radius decreases cost then
26:         Rewire nodes via the newly added node
27:     end if
28:     Store:
29:       1.Node distance between this node and tree root
30:       2.Travelling distance between new node and root
31:       3.Parent node number
32: end if
33: End Repeat
34: Trace back from finally added node to the tree root to
35: generate the feasible path.
36: Optimize the final path by pruning and grafting tech-
    niques
```

small then the time taken for the exploration will be longer. In this the step size is set as the same for that of the vehicle movement.

B. Seed generation and parent node selection:

Many researchers have come up with techniques to manipulate seed generation to get a feasible path in less amount of time [15][16]. These techniques are more suitable for specific environment and conditions, making them work for generalised workspace is not possible. Hence, for this project, the seeds are randomly generated in the workspace.For the parent node selection the Euclidean distance between the seed and nodes on the tree are calculated; by adding this distance to the stored distance in each tree node (distance the node has to cover before reaching the tree root), total distance between the seed and tree root can be obtained; the node via which

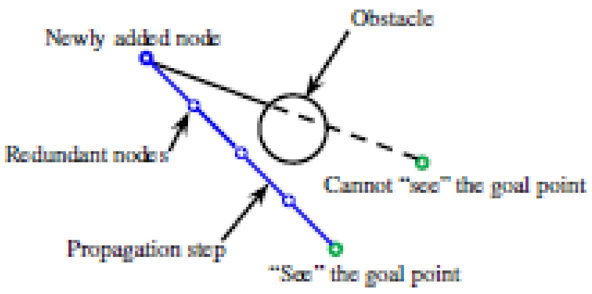

Fig. 1. Early stop technique-(Source Y.Dong et al.[23])

the random seed could approach tree root at a lowest distance is selected as the parent node, which is explained in (lines 7-14) of the algorithm

C. Navigation and Collision detection

Once the newly sampled node and its parent node are found, the navigation process takes place from the parent node till the newly sampled node until certain stop conditions are met. The first step is to check and stop if the navigation reaches into certain predefined neighbourhood region of the random new node. The next stop condition occurs when the sampled point lies outside the work space, in such cases navigation stops before the line segment connecting the parent to the new node goes out of the work space.

The dimension of the map is set at 20 x 15 x 10 meters. A maximum step propagation of 3 (meter) is considered a balance between working space exploration and computation. In Fig.3, if the randomly sampled node is sampled at a point having euclidean distance more than 3 metre from its parent node, the navigation stops at a distance of 3 metre from the parent node. The point at which the navigation stops is added to the tree instead of the previously sampled point.

For simulation purposes, the quadrotor dynamics are modelled as a point mass and the quadrotor navigation is modeled as a straight line connection. The algorithm is fed with the information of the work-space before the exploration starts. Information such as workspace dimension, obstacle locations, start and stop coordinate for the robot are prefed. But for real test conditions, indoor positioning system (OptiTrack/ Vicon) or image processing based Simultaneous Localization and Mapping(SLAM) techniques can be used to get the localization information for the quadrotor [17][18]. Fig.2 shows the 3D obstacle environment created to test the algorithm.

At certain cases, the maximum step length of 3 metre can be too large, it might poke into an obstacle space or go out of the workspace as shown in Fig.3. In such cases, a propagation method is used to propagate with a smaller shift step of 0.2 meters until it either goes out of the workspace or pokes into an obstacle.

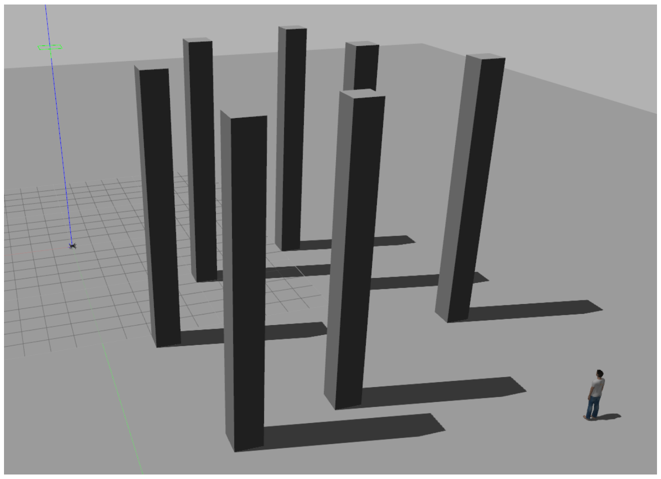

Fig. 2. Gazebo World

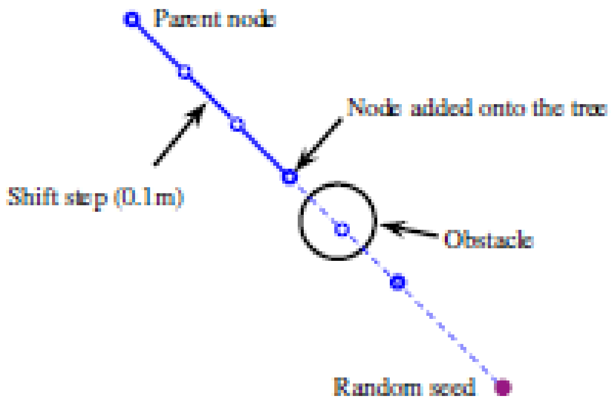

Fig. 3. Shift step propagation (Source Y.Dong et al.[23])

D. Rewiring,Node store and feasible path tracing

Implementation of RRT in a 3D environment with obstacles for a quadrotor is shown in Fig.4. This is similar to what the author has implemented in his own paper for finding a path between the start and goal. It can be inferred that, a feasible path is found between the start and the goal node.But, this path is not optimal. Therefore, to get a more optimised path, from the newly added node to the tree we consider K-Nearest neighbours approach and find k-number of nodes nearest nodes in the tree and try to see if we can reduce the overall distance from the root to newly added node by rewiring them(lines 25-27).Node number distance, Euclidean distance, and parent node number are stored in each newly added node. For each node on the tree, while multiple children nodes can exist, only one parent is mapped to each node.Given the final goal point, its parent node is the last node added onto the tree. Based on the parent node information stored in line 31, the feasible path can be generated by tracing back from goal point to its parent till the first parent of vehicle start position (tree root).

E. Path Optimization

From the path generated from the RRT*- an off-line optimization approach similar to [19] is used For practical implementation quadrotor dynamics and closed-loop controller will be considered,but for simulation purpose work quadrotor dynamics is considered as mass point and a straight line is used to connect the nodes of the tree [20][21].The optimization is done by randomly picking up two nodes from the path obtained from the RRT*,and trying to connect them directly as the length of this straight-line segment will be shorter than original (zigzagging) path If the connected line segment is collision free the original path between these two randomly-picked points will be replaced using the new connection; or if the connected line segment is not collision-free, an empty-trial will be returned.As we are trying to prune the original path using a new shorter path, overall length of the original path decreases, and when algorithm-stop conditions such as a set number of iterations are met the optimization process stops and the returned path is considered an optimized one.

III. Simulation Results

In this paper, the efficiency of both RRT and RRT* algorithms have been tested in ROS - Gazebo for a quadrotor in a 20 x 15 x 10 meter map with upright building like structures as shown in Fig 2. Various optimization techniques have been studied and it has been inferred that the rate of convergence and optimality of the path has a trade off. Depending upon the application either an algorithm which provides fast convergence or optimal path can be used. The obstacles are designed as cuboidal blocks and a quadrotor is used to perform the obstacle avoidance and landing task in the simulator environment. Extensive software simulations have been done in ROS (Robot Operating System) which is a general purpose robotics library that can be used with PX4 for offboard control. MAVROS node is used to communicate with PX4 running on the Gazebo Simulator.

*A. RRT vs RRT**

The path planning has been implemented using both the RRT and RRT* algorithm. The RRT algorithm is faster as it forms random sampling but does not guarantee of an optimized path. In the case of RRT* the path obtained is optimal, but it takes more time to find the goal. Fig. 4 represents the path planning done using RRT algorithm. Comparing this with that obtained using RRT*, as shown in Fig.5, it can be seen that the path planning done using RRT* is much more optimized, where as, the RRT algorithm it is very random.

B. RRT with and without pruning*

Three different optimization techniques were implemented with RRT* to get a better optimal path with faster convergence. Fig.5 shows the general implementation in which the final path formed is represented using the green lines. It can

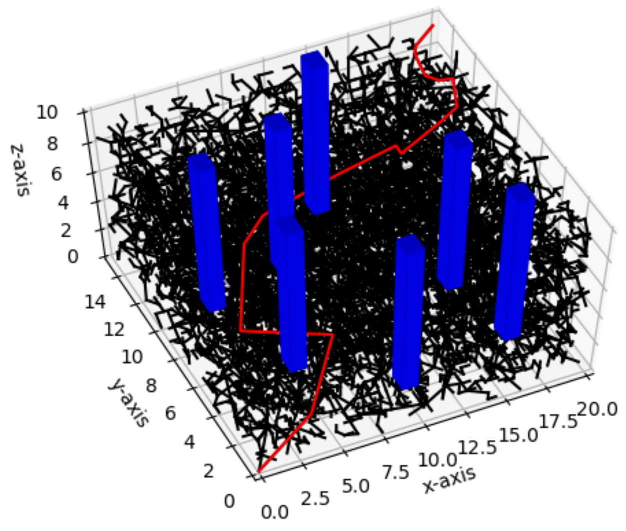

Fig. 4. RRT Path Planning

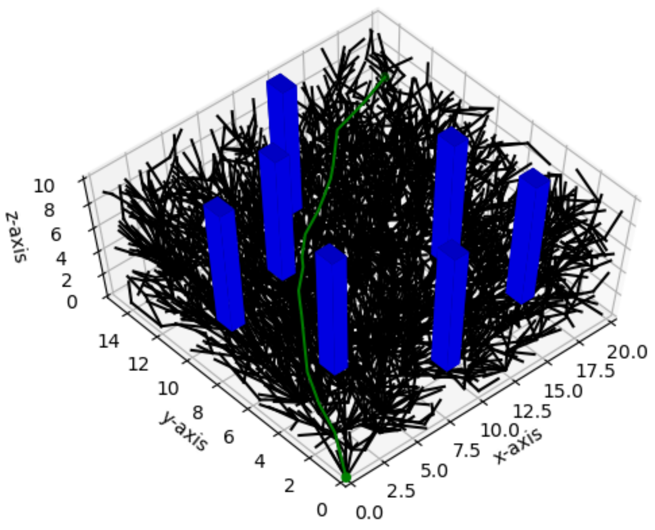

Fig. 5. RRT* without Pruning

be inferred that, this path is much more optimal than the path given by the RRT algorithm in Fig 4. The final path has many intermediate waypoints to reach the goal, making the path smooth. This proves that, there is still a room for optimization of the path with lesser number of waypoints from the start to goal.

This time, we can perform an offline optimization from the final path obtained from RRT* instead of making changes to the algorithm to get a much better optimized path with lesser number of waypoints. "Early stopping" algorithm is used to optimize the final path. Two random waypoints are picked from the final path and they are tested if there is any obstacle between them, if an obstacle is encountered, next set of points are chosen for a give number of iterations. When there is no obstacles between the chosen two random points, they are connected to each other and the intermediate waypoints between them are discarded. This will ensure we get a much more optimized path compared to Fig 5. The final optimized path will lesser number of waypoints is shown in Fig.6.

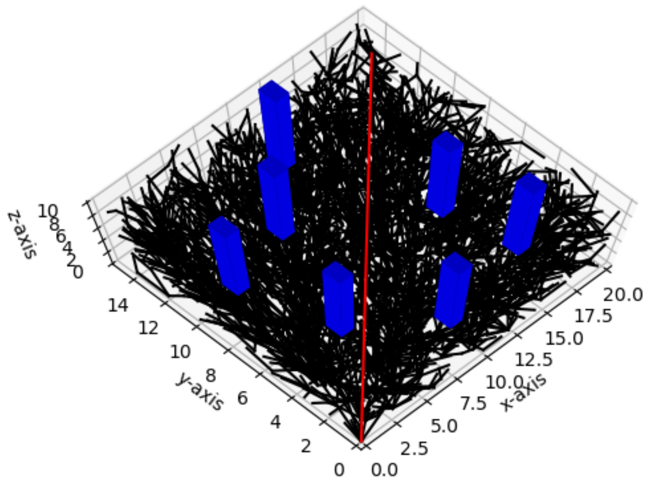

Fig. 6. RRT* with Pruning

The next optimization technique for faster convergence is the early stopping technique. Once the seed is randomly sampled and connected to its parent its also checked, if the goal node is visible from the seed, then the program terminates by connecting the seed directly to the goal.This optimization gives the fastest possible path planning for the quadrotor.The number of seeds produced by the search algorithm will also be very less as once the goal node is seen without any obstacles the seed is directly connected. Fig 7 shows the results obtained with type of optimization technique for faster convergence. It can be inferred that, compared to any other techniques discussed earlier, this approach does not have much exploration and hence it gives us a solution faster. Though, this algorithm gives us a faster solution, the path is not optimal. But it can be inferred that, its a better alternative to RRT as the goal convergence rate is really high compared to other path planning algorithms discussed in this paper.

IV. CONCLUSION

This paper has discussed the implementation and study of different sampling based path planning algorithms. The path planning algorithms are evaluated on the basis of faster goal convergence and optimality of the path. Sampling based algorithms such as RRT, RRT*, RRT* with offline path optimization and Early Stopping RRT* were studied. The programming was done in Python and the graphs showing the exploration of the tree and the final path were plotted using Matplotlib. The simulation was carried out in Gazebo environment using ROS using a position controller to move the drone to the waypoints outputted by the path planning

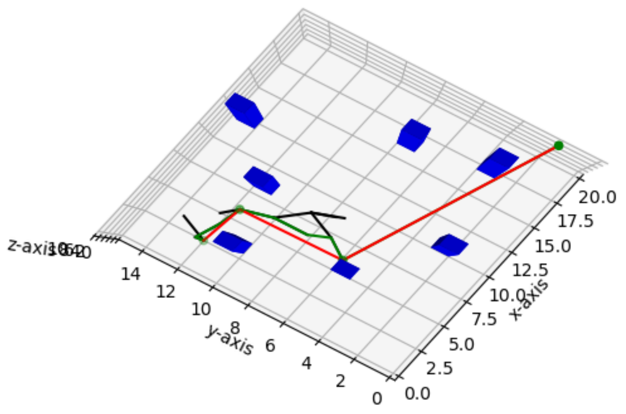

Fig. 7. Early Stopping

algorithm.

The future improvements involves accounting for the dynamic obstacles taking into account the perception of the quadrotor of the surroundings and its pose in real time which is an upgrade to the existing sampling based algorithms as discussed in the paper by Dr. Michael Otte at University of Maryland in his paper RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning [21].

REFERENCES

[1] D. Yiqun, F. Jun, Y. Bin, Z. Youmin, and A. Jianliang, "Position and heading angle control of an unmanned quadrotor helicopter using lqr method," in Control Conference (CCC), 2015 34th Chinese, 2015, pp. 5566–5571.

[2] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," IEEE Transactions on Control Systems Technology, vol. 17, no. 5, pp. 1105–1118, 2009.

[3] Nasir, J., Islam, F., Malik, U., Ayaz, Y., Hasan, O., Khan, M. and Muhammad, M., 2013. RRT*-SMART: A Rapid Convergence Implementation of RRT*. International Journal of Advanced Robotic Systems, 10(7), p.299.

[4] Kanehara, M., Kagami, S., Kuffner, J.J., Thompson, S., Mizoguhi, H. (2007) "Path shortening and smoothing of grid-based path planning with consideration of obstacles", IEEE International Conference on Systems, Man and Cybernetics, (ISIC), pp. 991–996.

[5] Sleumer, N.H., Tschichold-Grman, N. (1999) "Exact cell decomposition of arrangements used for path planning in robotics", Technical report. Switzerland: Institute of Theoretical Computer Science Swiss Federal Institute of Technology Zurich.

[6] Hwang, Y. K., Ahuja, N. (1992) "A potential field approach to path planning", IEEE Transactions on Robotics and Automation, vol. 8, no. 1, pp. 23–32.

[7] Yang, S.X., Luo, C. (2004) "A neural network approach to complete coverage path planning", IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 34, no. 1, pp. 718–724.

[8] En.wikipedia.org. 2020. Motion Planning. [online] Available at: <https://en.wikipedia.org/wiki/Motion_planningSampling-based_algorithms> [Accessed 11 April 2020].

[9] Geraerts, Roland (2006) "Sampling-based Motion Planning: Analysis and Path Quality", Ph.D. thesis. Utrecht University.

[10] Geraerts, Roland (2006) "On Experimental Research in Sampling-based Motion Planning", In (IROS) Workshop on Benchmarks in Robotics Research, pp. 31–34.

[11] Lavalle, S.M. (1998). "Rapidly-exploring random trees: A new tool for path planning", Computer Science Dept, Iowa State University, Tech. Rep. TR: 98–11. Retrieved 2008-06-30.S

[12] Karaman, S. and Frazzoli, E. (2011) "Sampling-based Algorithms for Optimal Motion Planning", International Journal of Robotics Research, vol. 30, no. 7, pp. 846–894.

[13] Garcia, I., How, J. P. (2005) "Improving the efficiency of Rapidly-exploring Random Trees Using a Potential Function Planner", in the proceedings of 44th IEEE Conference on Decision and Control, and the European Control Conference, pp. 7965–7970.

[14] Khanmohammadi, S., Mahdizadeh, A. (2008) "Density Avoided Sampling: An Intelligent Sampling Technique for Rapidly-Exploring Random Trees", Eighth International Conference on Hybrid Intelligent Systems, HIS, pp.672–677.

[15] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 2997–3004.

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

[17] C. Fu, A. Carrio, and P. Campoy, "Efficient visual odometry and mapping for Unmanned Aerial Vehicle using ARM-based stereo vision pre-processing system," in Unmanned Aircraft Systems (ICUAS), 2015 International Conference on, 2015, pp. 957–962.

[18] C. Fu, M. A. Olivares-Mendez, R. Suarez-Fernandez, and P. Campoy, "Monocular visual-inertial slam-based collision avoidance strategy for fail-safe uav using fuzzy logic controllers," Journal of Intelligent Robotic Systems, vol. 73, no. 1, pp. 513–533, 2014.

[19] Y. Dong and Y. Zhang, "Application of rrt algorithm to unmanned ground vehicle motion planning and obstacle avoidance," in 11th International Conference on Intelligent Unmanned Systems (ICIUS 2015), 2015

[20] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 2997–3004.

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

[22] Otte, M. and Frazzoli, E., 2015. RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning. The International Journal of Robotics Research, 35(7), pp.797-822.

[23] Y. Dong, C. Fu and E. Kayacan, "RRT-based 3D path planning for formation landing of quadrotor UAVs," 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), Phuket, 2016, pp. 1-6.