Received: 5 April 2021 | Revised: 7 May 2021 | Accepted: 8 May 2021
DOI: 10.379/ijeee.17.1.13 Vol.17 | Issue 1 | June 2021

Open Access

Iraqi Journal for Electrical and Electronic Engineering
Original Article

Hybrid RRT-A*: An Improved Path Planning Method for an Autonomous Mobile Robots

Suhaib Al-Ansarry*[1], Salah Al-Darraji[1]
[1] Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah, Iraq

Correspondence
*Suhaib Al-Ansarry
Department of Computer Science, College of Education for Pure Sciences,
University of Basrah, Basrah, Iraq
Email: pgs2186@uobasrah.edu.iq

Abstract
Although the Basic RRT algorithm is considered a traditional search method, it has been widely used in the field of robot path planning (manipulator and mobile robot), especially in the past decade. *This algorithm has many features that give it superiority over other methods. On the other hand, the Basic RRT suffers from a bad convergence rate (it takes a long time until finding the goal point), especially in environments with cluttered obstacles, or whose targets are located in narrow passages. Many studies have discussed this problem in recent years. This paper introduces an improved method called (Hybrid RRT-A*) to overcome the shortcomings of the original RRT, specifically slow convergence and cost rate. The heuristic function of A-star algorithm is combined with RRT to decrease tree expansion and guide it towards the goal with less nodes and time. Various experiments have been conducted with different environment scenarios to compare the proposed method with the Basic RRT and A-star under the same conditions, which have shown remarkable performance. The time consumed to find the path of the worst one of these scenarios is about 4.9 seconds, whereas it is 18.3 and 34 for A-star and RRT, respectively.*

KEYWORDS: Path planning, Autonomous mobile robot, Configuration space, Cost rate, Convergence rate, Grid-map.

I. Introduction

Since the emergence of humanity and the gradual development of societies until the establishment of the first civilization, all efforts directed to build that civilization were achieved by humans with the help of animals. When mechanical machines have been discovered, more tasks became easier due to the help of these machines. Robotics, in general, is the science that is concerned with trying to comprehend the environment in which the robot moves, through a set of measuring and sensing equipment that attempts to build an integrated vision of what that environment is, then guiding the mechanical parts based on that information [1]. In the same context, an autonomous mobile robot is defined as a machine that has the ability to perform a specific job without human intervention (i.e. make decisions independently) to decrease the effort and cost. Robotics has made big gains over the past years, especially in the field of medicine, industry, manufacturing, agriculture, space, and army [2,3]. Therefore, it can also be defined as the science which represents all the fields that have a direct relationship with the robot, such as designing, manufacturing, and implementing, as well as the fields of navigation, mapping, and others [4]. Navigation is one of the most important areas of research that has grown significantly in recent years due to the importance of an autonomous mobile robot field. This field includes four subfields [5] that still need a lot of research and development, such as perception, localization, motion control, and path planning. Path planning, as a general concept, varies based on the field that it represents and may reach far beyond that, such as the field of manufacturing and aerospace. In robotics, the problem of path planning, which is also called a "Piano Mover's Problem", aimed to develop methods that serve this purpose besides equipment capable of dealing with the challenges of this field. Rapidly exploring random tree (RRT) is one of the classical sampling-based algorithms, which is used widely in the field of path planning due to the ability of its fast growth within cluttered high-dimensional space as well as finding the route that reaching the goal if there an existing one. However, it has some problems in convergence rate besides the cost. A-star is a classical graph-search algorithm, which is used as another path planning method. It is able to find the optimal path depending on the use of distance-cost but within low dimensional space. On the other hand, the A-star algorithm has some disadvantages like the high-cost rate, big time complexity, and falling within local minima.

The contributions of this work can be summarized as the following:

1. A hybrid method called (Hybrid RRT-A*) is proposed by combining both basic RRT and A-star algorithms to overcome the rate of bad convergence for the basic RRT besides reducing the cost.
2. Evaluate the performance of the Hybrid RRT-A by comparing the results of the cost and convergence with Basic RRT and A-star under the same conditions (computer hardware and simulation environment).

The rest of the paper is organized as follows: Section II presents the related work of path planning algorithms. Section III describes Basic RRT and A-star, as well as the proposed methods Hybrid RRT-A* within the same environmental conditions. Section IV outlines the experiments and results. Section V discusses the final results. Section VI presents the conclusions and future works.

II. Related work

Rapidly Exploring Random Tree (RRT), which is developed by Steven M. LaValle [6] and James J. Kuffner [7], is a type of Roadmaps algorithms (RM). The basic idea of RRT is generating random samples from the configuration space based on the uniform distribution, then trying to links these samples. If the connection is feasible to the nearest point and passes entirely through free space without any collision of any obstacle, the result then is adding a new point to the tree. This algorithm is a single-query scheme (single tree), which takes a long time in search operation (convergence rate), while it has the ability to explore the whole search space with the lowest number of nodes (cost rate) compared to the bidirectional schemes. Basic RRT has some other advantages over other classical methods that make it a good choice for mobile robot path planning, which can be summarized as follows:

- Completeness algorithm, which can always terminate the search by finding the goal point and returning the path if there exists one.
- Used as a global planner (offline) and also as a real-time local planner (online) depending on the robot's sensor.
- Used for indoor, outdoor high complex, and large dimensional space environment.
- Dealing with the non-holonomic and Kinodynamic constraints efficiently (differential constraints).

This scheme suffered from several problems, like other planners [8,9]. The bad rate of convergence is one of these problems, especially with a cluttered environment that has dynamic and static obstacles. This environment type leads usually to reduce the convergence and performance of the Basic RRT algorithm due to the slow tree expansion, besides the behavior of high randomness. So, as a result, it is not preferred to use Basic RRT in such an environment or that which has constraints. To overcome these shortcomings, Basic RRT has been combined with several other methods.

RRT-connect [10] is another type of Basic RRT, which operates to increase the Basic RRT convergence rate. This algorithm is based on the use of two functions: First, the greedy function that is used for rapid expansion to increase the convergence. Second, the probabilistic navigation function of the potential field to avoid falling in the local-minimum region. The main disadvantage of RRT-connect is the complexity increased in searching time for the nearest neighbor.

On the one hand, unlike Basic RRT, A-star is an intelligent graph-search algorithm using the heuristic information and can be applied to a configuration space topologically [11]. This algorithm is proposed by Peter Hart, et al. [12]. It is used as a part of the Shakey project [13] for mobile robot path planning, and a special case of the Dijkstra algorithm [14].

Karaman and Frazzoli [15,16] introduced an extended version of the Basic RRT algorithm called RRT*, which optimized the path non-optimality problem of the Basic RRT. The idea is similar to the RRT except two key differences make the influence of RRT* big. First, at each iteration, RRT* looks for the node with the lower distance-cost among the neighbors until one is found, then this node is examined again along with all adjacent nodes within a fixed radius to find the lowest-cost node. If a new one is found, then replace the old node (cheap) with the new one (cheapest). Second, RRT* reconstructs the tree frequently until the cheapest node has been reached, and so on. This method introduces a smooth path, and closest to an optimal solution compared to the Basic RRT, while its convergence rate is still weak. In some cases, such as cluttered environments within high-dimensional space, narrow passages, and if there are local minimum regions that cause to trap the mobile robot and lead the navigation to fail, it is preferred to use bidirectional approaches such as the Bidirectional-RRT* which proposed by M. Jordan and A. Perez [17].

Another form of Basic RRT called RRT*-smart algorithm is proposed [18,19]. This algorithm has two phases: the first for path optimization and the second for intelligent sampling. In the first stage, the RRT*-smart acts as RRT* and the redundant nodes will be removed from the initial path, and then defining beacon nodes. In the second phase, the algorithm starts an intelligent sampling rather than a random one. These nodes tend to expand towards the surroundings of the beacon nodes based on a predefined constant that is used as a biasing parameter that can be determined statically or adaptively. Once again, the resulting path is optimized in the same way that in the first stage, then iterations continue with more biasing towards beacons until the path is improved.

An improved variant of Basic RRT called RRT-Dijkstra is presented by Mahmut Dirik and A. Fatih Kocamaz [20]. This method is used as a global planner for mobile robot path tracking within a cluttered environment. It depends on taking the advantages of both Basic RRT and Dijkstra to obtain a new method that can overcome the problems which the previous two methods suffered from. The RRT-Dijkstra tries to produce an optimal path (safe, short, and smooth). This method bypassed the costly path problem due to the benefit of using Dijkstra's advantages of finding the shortest distance to the goal. On the other hand, it is computationally expensive.

Ayawli, et al. [21] proposed another version of Basic RRT called Optimized RRT-A*. In this method, four techniques are used. First, the step size-based RRT, which biases the search toward the target faster. Second, morphological dilation is another technique that helps to obtain a collision-free path by extending the external obstacle's boundaries. Third, the heuristic information obtained through the A-star algorithm is used to calculate the lowest distance-cost for neighbor nodes at each iteration to shorten the path. Finally, applying the interpolation of cubic spline to make the final path smoother. This method contributed to avoiding the region of local-minima besides providing an optimized path and reducing the computational load while navigating in a partially known environment.

The Adaptive Hybrid Dynamic Step-size and Target Attractive Force-RRT (AHDSTAF-RRT) is an improved method introduced by Zhang, et al. [22] which consists of two parts. The first part is represented by Basic RRT which is responsible for the tree growth, while the second is represented by two other techniques (Dynamic step-size & Target Attractive Forces). Dynamic step-size (DS-RRT) uses a step-size technique within the Basic RRT structure to deal with different situations. For example, in the case of passing through narrow regions, small steps are used whereas big steps are used in the case of open areas to speed up a tree growing. Target Attractive Force (TAF-RRT) is used to deal with the problem of growing trees in multiple directions. Finally, the DS-RRT and TAF-RRT are combined to address the most Basic RRT issues while preserving the complexity of space and time.

Zhang, et al. [23] proposed an improved form of the Basic RRT that takes the advantages of RRT while trying to overcome its drawbacks based on the use of two functions. The first is the Target Bias Search strategy (TBS). In this strategy, the Improved RRT (I-RRT) attempts to increase the convergence speed of the Basic RRT through biasing the search towards the goal point instead of searching the entire space, by predefining biasing threshold which guides the mechanism of RRT random sampling towards the goal, then reduces the sampling spread depending on the potential resulting value. This process makes the planning more accurate besides the ability to deal with planning situations in real-time. The problem of RRT path length and sharp turn is fixed by using the second technique that is called a New Metric Function (NMF).

III. The Proposed Method

This section presents the proposed hybrid method called (Hybrid RRT-A*). It involves a kind of combination of Basic RRT and A-star planners. While the RRT advantages are preserved, the heuristic function feature of the A-star planner is employed to solve the problem of bad RRT's convergence and decrease the overall cost of the resulting method.

A. *Hybrid RRT-A**

The proposed Hybrid RRT-A* method consists of two phases: The first can be represented simply as shown in the construct of Algorithm 1, by forming a tree within the configuration space, starting from an initial point - representing the root of the tree - and then gradually expanding into several branches to explore the entire environment and reaching the goal point. Figure 1 clearly shows this mechanism.

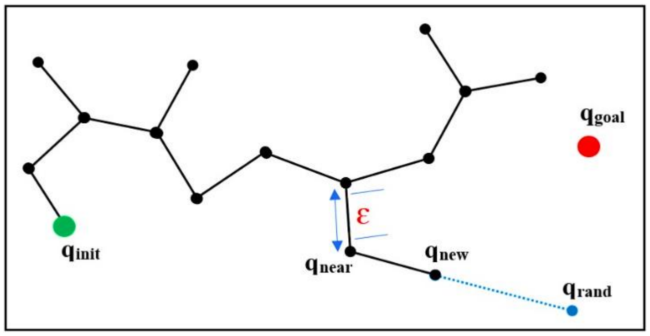


Fig. 1: The construction process of RRT.

Algorithm 1.
```
Input: Initial configuration qinit,
       Number of vertices in RRT V,
       Incremental distance ε,
Output: RRT graph G
G.init (qinit)
for v= 1 to V do
    qrand ← RAND_CONF()
    qnear ← NEAREST_VERTEX(qrand, G)
    qnew ← NEW_CONF(qnear, qrand, ε)
  G.add_vertex(qnew)
  G.add_edge (qnear, qnew)
return G
```

In the beginning, the dimensions of the main configuration C (workspace) are defined besides the obstacles regions (C_{obs}). Then, the locations for both start and goal points (q_{init}, q_{goal}) within the obstacle-free region (C_{free}) are located by placing them in the configuration space. In addition, the maximum number of nodes (V) that can be examined in all iterations should be predefined. The radius of the goal area is another condition that must be specified, which determines whether the A-star (sub-search operation) can be called at this range or not, as well as the distance between every two A-star algorithms. This condition (distance) is used as the basis for determining the number of the next sub-search functions that should be called within the RRT structure each time. After finishing these settings, the random tree starts its growth by frequently choosing a random sample (q_{rand}). The node of (q_{rand}) is used as a steer function to determine the direction of the new sample point (q_{new}) which will be added at each iteration. The new point will be used for the random tree expansion, while it does not fall in the region of (C_{obs}) and the distance between (q_{near}) and (q_{new}) achieves the condition of predefined step-size (ε), the nodes will be joined together by an edge. Basic RRT tends to grow rapidly in large unexplored regions besides biasing towards the configuration of the goal region. This algorithm is very sensitive against the obstacles, which means that the values of the two main parameters (step-size and bias probability) have to be carefully chosen. Several studies attempt to find some ways to help accurately select the most influential values [33]. In the Hybrid RRT-A*, the proposed parameters to be used are (1) for step-size and (0.25) for the probability of biasing to reduce the dependence of these parameters and showing the efficiency of the Hybrid RRT-A*.

At this stage of algorithm improvement, the first phase continues sequentially as long as the previous conditions are met until either the goal is found or all the nodes (V) are examined. In each iteration, the method has to verify two additional new conditions. First, if the node (q_{new}) which was generated and added to the configuration, does not fall within a radius of the goal area which is predefined depending on a specific threshold, then the main planner (RRT) will continues its search. Otherwise, the second phase (sub-search operation) is called to operate in this range. The second condition needs to check the distance (D) between the current node (q_{new}) and the last one that has been added to the tree whether satisfying a predefined metric (D) or not. If the condition is achieved, then the sub-search operation will be recalled again to perform the same task frequently until reaching the goal. Otherwise, this operation (sub-search) is aborted and the main RRT planner continues.

The second phase represents the use of the A-star functions. This method is used as a local planner that interpreting its environment basing on the robot's sensor. In the proposed method, A-star is used as a global planner supposing that the robot environment is known. A-star is sometimes called the Best-First planner because the configuration cells are evaluated based on the minimum value of f(n):

$$f(n)=g(n)+h(n) \qquad (1)$$

Where *f(n)* represents the cost value of the final path by passing through all points between the start and goal points. The *g(n)* represents the function that computes the path-cost from the initial node to the current one *(n)*, while *h(n)* represents the heuristic function, which estimates the distance between the current node and goal, by extending a straight line between the two nodes using (Euclidean, Manhattan, ..., etc.). A-star builds a graph structure saving all paths starting from the initial point until the goal. The procedure is continuously repeated by extending the node that has the lowest distance-cost at each iteration until hitting the goal, which enables us to decide which path should be expanded based on minimizing the *f(n)* value. Finally, when the goal point is found, the planner returns the path gradually in an opposite direction starting from the goal node, where each node keeps track of its previous path until reaching the start node.

The A-star planner has many advantages, like the simple implementation, short running time, and high efficiency within a specific two-dimensional environment. On the other hand, space and time complexity are the main disadvantages, due to the need of checking and saving of all adjacent nodes in every round before deciding which node will take to be expanded later. That is considered a big issue facing the A-star with a large space environment.

Practically, the construct of A-star, as shown in Algorithm 2, uses two queues called (close-set & open-set) which are initially empty. In the beginning, the configuration nodes should be classified into two groups (non-expandable and expandable nodes). Firstly, the non-expandable nodes are all placed into the close-set queue representing nodes fallen on the map border or that fall within an obstacle region. Secondly, generating the start and goal nodes, then placing the start one into the second queue (open-set). This queue is used to perform the frequent choice for nodes with the lowest value which can be used for the expansion if it satisfies the criteria. This means that the algorithm has to verify all nodes adjacent to the current one and push them (add) into the queue before deciding which node has to visit next.

Algorithm 2.

```
A* (Start, Goal)
Closed-set = the empty set
Open-set = includes start node
G[Start] = 0, H[Start] = H_calc [Start, Goal],
F [Start] = H[Start]
While Open-set ≠ 0 Do
    CurNode ←EXTRACT-MIN- F (Open-set)
        If (CurNode == Goal), then return Best Path
          For each Neighbor Node N of CurNode
              If (N is in Closed-set), Then Nothing
              Else If (N is in Open-set),
                    calculate N's G, H, F
            If (G [N on the Open-set] > calculated G[N])
                RELAX (N, Neighbor in Open-set, w)
                N's parent=CurNode & add N to Open-set
            Else Then calculate N's G, H, F
    N's parent = CurNode & add N to Open-set
```

In this phase, the advantages of A-star are exploited to overcome the problem of Basic RRT's bad convergence rate as well as decreasing the cost rate. Basic RRT is a blind algorithm due to its randomness (uniform distribution), where the probability of selecting any point from the sample space is (1/n). This means that all points have the same selection probability whereas the cost-dependent behavior of A-star -as a feature- makes the RRT find the goal point in a relatively short time. Mostly, a Basic RRT is only needed when wanting to explore an open area without obstacles, but practically there is no obstacle-free or optimal environment in reality. When the RRT facing a large number of obstacles, or when the goal point falls within a local minimum region or falls behind or maybe between narrow passages, the RRT takes a long searching time until finding the goal. Therefore, A-star functions are used in the proposed scheme to overcome the low search-speed problem of the Basic RRT as well as the large cost. Finally, this leads to a fast and more efficient search. It should be noted that the values of both threshold (T) and distance (D) have to be proportional to the environment size. Figure 2 demonstrates Hybrid RRT-A* phases in more detail.

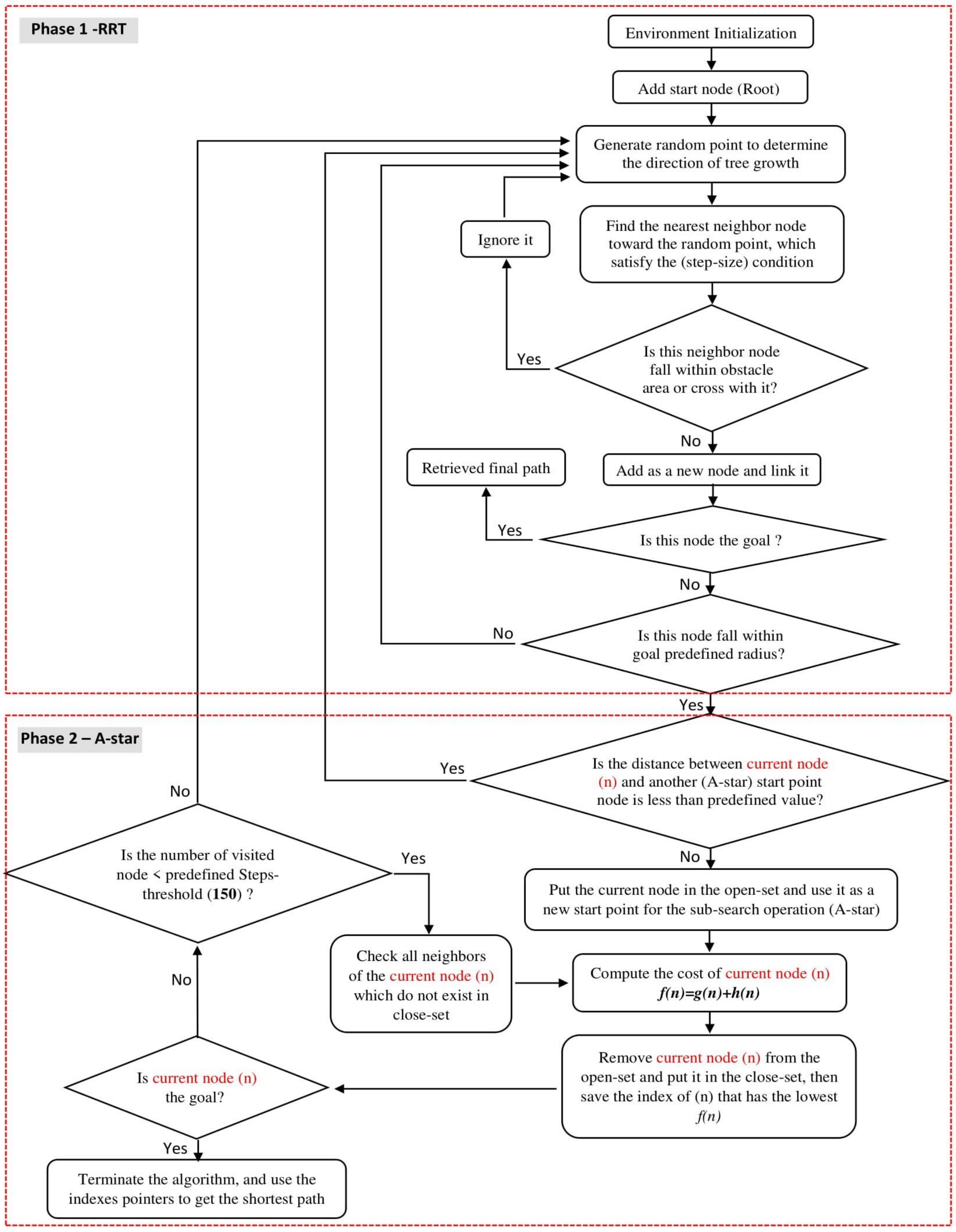

Phase 1 -RRT
Environment Initialization
Add start node (Root)
Generate random point to determine the direction of tree growth
Find the nearest neighbor node toward the random point, which satisfy the (step-size) condition
Ignore it
Yes
Is this neighbor node fall within obstacle area or cross with it?
No
Retrieved final path
Add as a new node and link it
Yes
Is this node the goal ?
No
No
Is this node fall within goal predefined radius?
Yes
Phase 2 – A-star
Yes
Is the distance between current node (n) and another (A-star) start point node is less than predefined value?
No
Is the number of visited node < predefined Steps-threshold (150) ?
Yes
No
Put the current node in the open-set and use it as a new start point for the sub-search operation (A-star)
Check all neighbors of the current node (n) which do not exist in close-set
Compute the cost of current node (n) f(n)=g(n)+h(n)
No
Remove current node (n) from the open-set and put it in the close-set, then save the index of (n) that has the lowest f(n)
Is current node (n) the goal?
Yes
Terminate the algorithm, and use the indexes pointers to get the shortest path

Fig. 2: Hybrid RRT-A* flowchart.

IV. Experiments and Results

In this section, various experiments have been conducted to evaluate the overall performance of the proposed method.

A. *Setup and Maps*

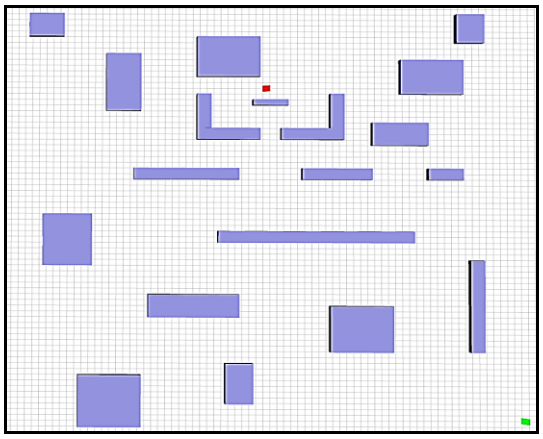
The experiments are conducted by using the simulation environment of the Robot Operating System (ROS Kinetic). This framework is operated under Linux (64-bit) operating system with (8GB) RAM and (intel Core-i7 8th Gen) processor. The visualize environment (Rviz) of ROS, C++ interface with the source file implementation, and the terminal windows are additional components that are utilized.

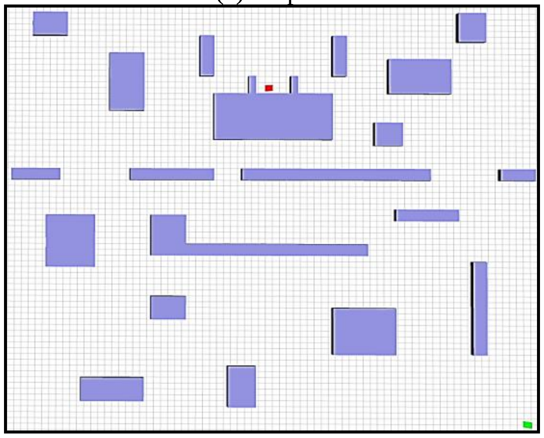
The simulation of planning the path is divided into three groups. Group-1 consists of three experiments: The First, for the (Basic RRT) method. The Second, for the (A-star) method. The Third, for the (Hybrid RRT-A*). All experiments are applied using a grid environment map of size 75*75 through the map (Map-1). Group-2 and 3 repeat the same methods listed above, experiments are applied again on the same grid dimensions through the maps (Map-2 & Map-3).
In Group-1, Group-2, and Group-3 experiments, the previous methods are tested by implementing them in different scenarios, then the results are compared and the performance is evaluated depending on several factors (cost, convergence). In these maps, shaded grid cells are referred to as an occupied region, while the green square shape refers to the point of start, and the red square to the goal. Finally, the rest grid-cells represent the free region.

Many parameters should be initialized, which are the same for all experiments. These parameters are: Start point= (74,74); Goal point= (37,15); Max iteration no.=30000; Step size=1; Goal biasing=0.25; Goal threshold=13 (goal area radius); A* threshold=7 (distance between two A* start points); Obstacle padding=1; Robot frequency=0.05. The number of obstacles and their coordinates varies among the maps (Map-1, Map-2, and Map-3) of all groups. Each of these maps will be used with the previous methods for planning the robot path. The results of the search operations of all experiments will be discussed at the end of the next section.

B. *Experiments*

The experiments of all groups are applied on a 75*75 environment grid-map as shown in Figure 3 with a number of sparse obstacles to be used in the simulation process of finding the robot path.


(a) Map-1.


(b) Map-2.

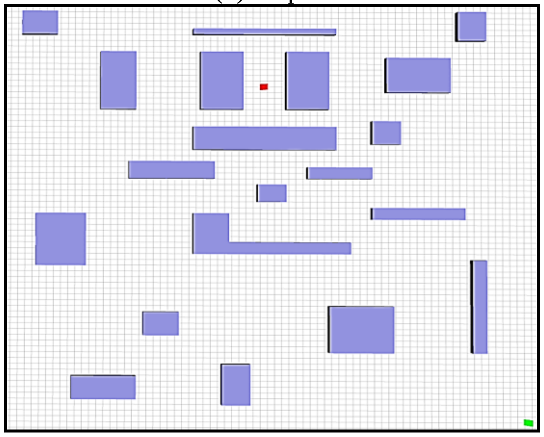
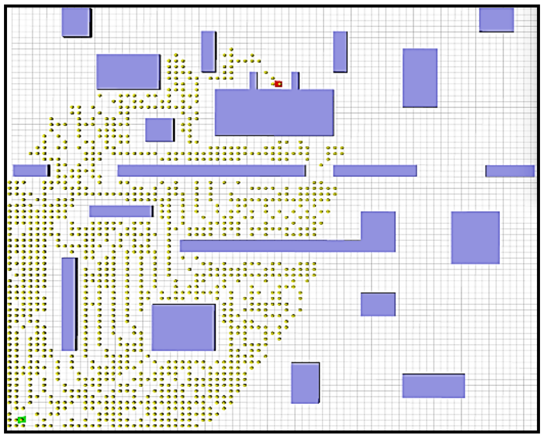
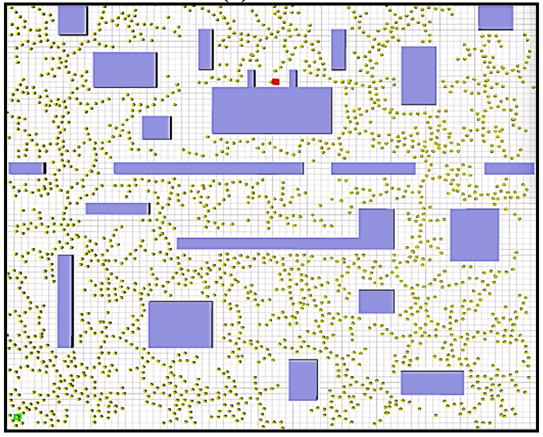

(c) Map-3.

Fig. 3: 75*75 Grid environment of (a) Map-1, (b) Map-2, and (c) Map-3.

In each group, three different methods (Basic RRT, A-star, Hybrid RRT-A*) are applied through three separate experiments to explore the entire environment map and locating the goal point as shown in Figure 4 with a low rate (cost and time), as well as avoiding the collisions, and finding the path between source and destination.


(a) A-star


(b) Basic RRT

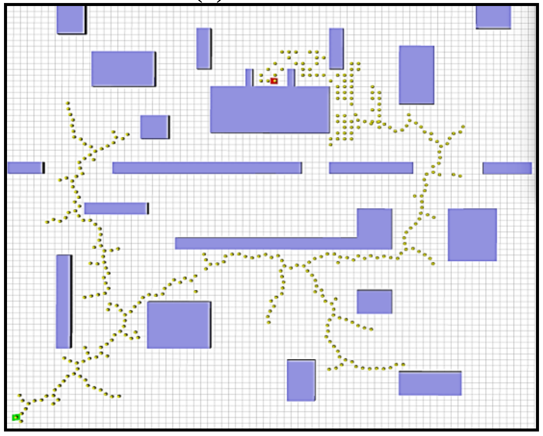
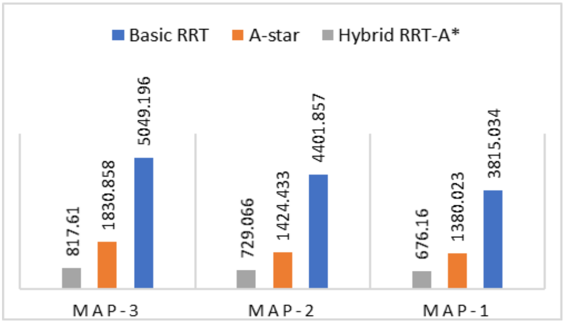

(c) Hybrid RRT-A*

Fig. 4: 75*75 Grid environment of Map-2 (a) A-star, (b) Basic RRT, and (c) Hybrid RRT-A*.

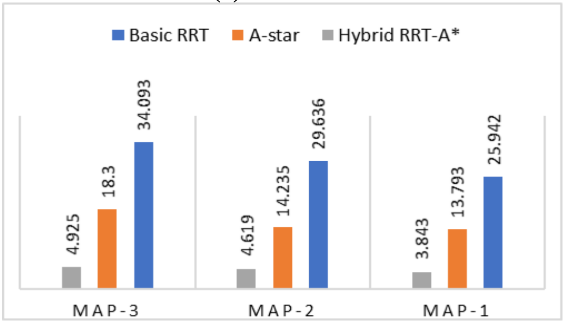
The experiment of each group introduces an analytical comparison among the results of the three methods depending on the cost and time rates. The results in Table I demonstrate that the highest cost rate is the Basic RRT, then A-star with a lower rate followed by Hybrid RRT-A* as the lowest one. While Hybrid RRT-A* is the lowest time rate followed by A-star then Basic RRT. Figure 5 shows these results through a plot graph.

TABLE I
EXPERIMENTS RESULTS FOR THE COST, CONVERGENCE, AND THE PERCENTAGE AMONG THE THREE METHODS IN EACH MAP.

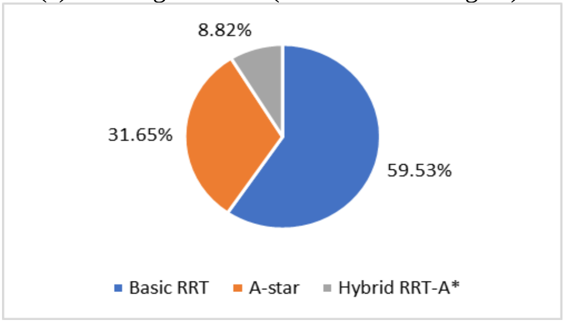
Methods	**Map-1**		
	Cost (node)	Time (sec.)	Total Percentage
Basic RRT	3815.034	25.942	59.53%
A-star	1380.023	13.793	31.65%
Hybrid RRT-A*	676.160	3.843	8.82%
Methods	**Map-2**		
	Cost (node)	Time (sec.)	Total Percentage
Basic RRT	4401.857	29.636	61.12%
A-star	1424.433	14.235	29.36%
Hybrid RRT-A*	729.066	4.619	9.53%
Methods	**Map-3**		
	Cost (node)	Time (sec.)	Total Percentage
Basic RRT	5049.196	34.093	59.48%
A-star	1830.858	18.300	31.93%
Hybrid RRT-A*	817.610	4.925	8.59%



(a) Cost-rate.



(a) Convergence-rate (time to reach the goal).



(c) Total percentage.

Fig. 5: (a) Cost, (b) Convergence, and (c) Total percentage graphs for the three methods in each map.

V. Discussion

In general, Hybrid RRT-A* initially applies the obstacle avoidance method and if the path intersecting some obstacles then re-planned the path around the obstacles at the available free spaces. In the case of facing narrow passages, or goal point is placed among many obstacles and the search tree is near to the goal region, then the heuristic function is used to improve the search cost and time. Many studies used maps of size (10*10 - 40*40) with about (5-10) obstacles distributed in simple scenarios. In this simulation, various scenarios are used with maps of size 75*75 and (18-30) obstacles, designed to simulate the worst cases. Moreover, in this simulation, the experiments are executed (1000 times) for each method individually. The results proved that the combination process used in the proposed method has contributed to decreasing the Basic RRT cost and time rate (convergence). Finally, in Table II, the proposed method (Hybrid RRT-A*) was compared with several earlier sampling-based algorithms depending on the characteristics of each one.

TABLE II
THE CHARACTERISTICS COMPARISON OF SAMPLING-BASED ALGORITHMS.

Algorithm	Completeness	Low Cost	Fast Convergence
RRT	Yes	No	No
RRT*	Yes	No	No
PRM	Yes	No	No
PRM*	Yes	No	No
RRG	Yes	No	No
Hybrid RRT-A*	Yes	Yes	Yes

VI. Conclusion

In this paper, a hybrid path planning method (Hybrid RRT-A*) is proposed and implemented depending on the rapidly exploring random trees (Basic RRT). The sampling technique of the RRT planner has been improved, so the present tree of the proposed method will guide the process of node sampling efficiently within the goal region. This is done by using a heuristic function of A-star in combination with the Basic RRT. The heuristic function is defined to speed up the convergence rate, besides reducing the cost. This is done by examining each RRT node to check if it falls within the goal region with a specific threshold. When reaching this region, the A-star algorithm is activated. The advantage of using A-star in this stage is to eliminate random search and find the shortest path in this region. The simulation results show that (Hybrid RRT-A*) achieves the best cost and time rate compared to both A-star and Basic RRT.
In future work, the aim is to optimize the shortest path and make it much smoother by using some kind of regression and curve methods like (multinomial-regression and b-spline). Moreover, as future work, the aim is to perform planning within an environment populated with moving (dynamic) obstacles.

References

[1] Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. "Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)." (2005).
[2] Alexis, Kostas, George Nikolakopoulos, Anthony Tzes, and Leonidas Dritsas. "Coordination of helicopter UAVs for aerial forest-fire surveillance." In *Applications of intelligent control to engineering systems*, pp. 169-193. Springer, Dordrecht, 2009.
[3] Doherty, Patrick, and Piotr Rudol. "A UAV search and rescue scenario with human body detection and geolocalization." In *Australasian Joint Conference on Artificial Intelligence*, pp. 1-13. Springer, Berlin, Heidelberg, 2007.
[4] Möls, Märt. *Linear mixed models with equivalent predictors*. Tartu University Press, 2004.
[5] Injarapu, Anantha Sai Hari Haran V., and Suresh Kumar Gawre. "A survey of autonomous mobile robot path planning approaches." In *2017 International conference on recent innovations in signal processing and embedded systems (RISE)*, pp. 624-628. IEEE, 2017.
[6] LaValle, Steven M. "Rapidly-exploring random trees: A new tool for path planning.", pp. 98-11 (1998).
[7] LaValle, Steven M., and James J. Kuffner Jr. "Randomized kinodynamic planning." *The international journal of robotics research* 20, no. 5, pp. 378-400 (2001).
[8] Karaman, Sertac, Matthew R. Walter, Alejandro Perez, Emilio Frazzoli, and Seth Teller. "Anytime motion planning using the RRT." In *2011 IEEE International Conference on Robotics and Automation*, pp. 1478-1483. IEEE, 2011.
[9] Atramentov, Anna, and Steven M. LaValle. "Efficient nearest neighbor searching for motion planning." In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 1, pp. 632-637. IEEE, 2002.
[10] Kuffner, James J., and Steven M. LaValle. "RRT-connect: An efficient approach to single-query path planning." In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2, pp. 995-1001. IEEE, 2000.
[11] Cui, Shi-Gang, Hui Wang, and Li Yang. "A simulation study of A-star algorithm for robot path planning." In *16th international conference on mechatronics technology*, pp. 506-510. 2012.
[12] Hart, Peter E., Nils J. Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths." *IEEE transactions on Systems Science and Cybernetics* 4, no. 2, pp. 100-107 (1968).
[13] Nilsson, Nils J. *The quest for artificial intelligence*. Cambridge University Press, 2009.
[14] Dijkstra, Edsger Wybe. "A note on two problems in connexion with graphs:(Numerische Mathematik, 1 (1959))", pp. 269-271 (1959).
[15] Karaman, Sertac, and Emilio Frazzoli. "Incremental sampling-based algorithms for optimal motion

planning." *Robotics Science and Systems VI* 104, no. 2 (2010).

[16] Karaman, Sertac, and Emilio Frazzoli. "Sampling-based algorithms for optimal motion planning." *The international journal of robotics research* 30, no. 7, pp. 846-894 (2011).

[17] Jordan, Matthew, and Alejandro Perez. "Optimal bidirectional rapidly-exploring random trees." (2013).

[18] Islam, Fahad, Jauwairia Nasir, Usman Malik, Yasar Ayaz, and Osman Hasan. "Rrt∗-smart: Rapid convergence implementation of rrt∗ towards optimal solution." In *2012 IEEE international conference on mechatronics and automation*, pp. 1651-1656. IEEE, 2012.

[19] Nasir, Jauwairia, Fahad Islam, Usman Malik, Yasar Ayaz, Osman Hasan, Mushtaq Khan, and Mannan Saeed Muhammad. "RRT*-SMART: A rapid convergence implementation of RRT." *International Journal of Advanced Robotic Systems* 10, no. 7, pp. 299 (2013).

[20] Dirik, Mahmut, and Fatih KOCAMAZ. "RRT-Dijkstra: An Improved Path Planning Algorithm for Mobile Robots." *Journal of Soft Computing and Artificial Intelligence* 1, no. 2, pp. 11-19 (2020).

[21] Ayawli, Ben Beklisi Kwame, Xue Mei, Moquan Shen, Albert Yaw Appiah, and Frimpong Kyeremeh. "Optimized RRT-A* Path Planning Method for Mobile Robots in Partially Known Environment." *Information Technology and Control* 48, no. 2, pp. 179-194 (2019).

[22] Zhang, Zhen, Defeng Wu, Jiadong Gu, and Fusheng Li. "A path-planning strategy for unmanned surface vehicles based on an adaptive hybrid dynamic stepsize and target attractive force-RRT algorithm." *Journal of Marine Science and Engineering* 7, no. 5, pp. 132 (2019).

[23] Zhang, Pengchao, Chao Xiong, Wenke Li, Xiaoxiong Du, and Chuan Zhao. "Path planning for mobile robot based on modified rapidly exploring random tree method and neural network." *International Journal of Advanced Robotic Systems* 15, no. 3 (2018).