



# RRT- Dijkstra: An Improved Path Planning Algorithm for Mobile Robots

<sup>1</sup>Mahmut Dirik , <sup>2</sup>A. Fatih Kocamaz 

<sup>1</sup>Department of Computer Engineering, Sırnak University, TURKEY

<sup>2</sup>Department of Computer Engineering, Inonu University, TURKEY

Corresponding author: Mahmut Dirik ([mhmd.dirik@gmail.com](mailto:mhmd.dirik@gmail.com))

**ABSTRACT** The Path planning problem is one of the most researched topics in autonomous vehicles. During the last decade, sampling-based algorithms for path planning have gained significant attention from the research community. Rapidly exploring Random Tree (RRT) is a sampling-based planning approach, which is a concern to researchers due to its asymptotic optimality. However, the use of samples close to obstacles in path planning and the path with sharp turns does not make it efficient for real-time path tracking applications. For overcoming these limitations, this paper proposes a combination of RRT and Dijkstra algorithms. The RRT-Dijkstra reliefs a shorter and collision-free path solution. It is measured by various factors such as path length, execution time, and the total number of turns. The aim here is review and performance comparison of these planners based on metrics, i.e., path length, execution time, and the total number of turning points. The algorithms are tested in complex environments structured with obstacles. The experimental performance shows that RRT-Dijkstra requires less turning point and execution time in 2D environments. These are advantages of the proposed method. The proposed method is suitable for off-line path planning and path-following.

**KEYWORDS:** Optimal criteria, Path planning, RRT, Dijkstra, Sampling-based algorithms.

## 1. INTRODUCTION

Mobile robots have been effectively adapted to perform a number of tasks in various fields over the past decade. Mobile robots gain more intelligence with machine learning technologies so they can increase productivity with their navigation features. Path planning is the basis of navigation that consist of a collision-free feasible path.

Several algorithms have been used for mobile robot path planning. These are A\* [1] which is an extension of the Dijkstra algorithm [2-4], Rapidly Exploring Random Tree (RRT) [5-7], Artificial Potential Fields (APF) [8, 9], visibility graph [10, 11], grid based methods [12, 13], Bidirectional-RRT (BRRT) [14, 15], Probabilistic Road Map (PRM) [16], Genetic algorithm (GA) [17-19], Fuzzy Type 1 and Fuzzy Type 2 [20, 21] planning algorithms. In global and local path planning, real-time applications, and solutions of high-dimensional problems, each of them has its own advantages and disadvantages. Several path planners can be used together to obtain a numerical value that represents the expected performance from path planning methods and techniques. The focus in this study is not only to provide a standard path planning test procedure but also to propose a solution to the path planning problem that an autonomous robot can follow in real-time in a low-cost environment. Here, RRT and Dijkstra, one of the best known/used algorithms, are examined, implemented, and tested.

The Dijkstra algorithm [2], which searches for the destination in all directions and searches for the most suitable route, is one of the most used routes planning algorithms. Using the grid map, the path

created with Dijkstra between the start and the target finds the optimum path where all the corners of the neighboring nodes are known as the edge of the straight connected link. The specific visibility points (collision-free points) are added to the map to construct the path. The vertices of the graph represent the set of configurations that are used by the planner as waypoints [22]. Dijkstra's algorithm is used in almost all path planning studies in recent years. This is because this algorithm is defined in the literature as a quick and simple route planning method [23].

RRT is a path planning techniques based on growing a random tree to search for the target position. This technique is suitable for identifying paths in uncomplicated environments without hitting obstacles to the desired target location regardless of path characteristics (ie shortest, softest etc.). RRT performs the starting path quickly and improves the quality of consecutive iterations. Thanks to its asymptotic optimal feature, it has become suitable for local path planning applications. Although the RRT algorithm is successful in navigation, convergence and real-time global path planning can also make it difficult for the mobile robot to track. This is because it has many turns and nodes close to obstacles. As a solution to this problem, the path coordinates obtained with RRT were chosen as the search area of the Dijkstra algorithm and a new path was created. Thus, the cost of the path is decreased and real-time path tracking is simplified by reducing the number of turning points.

This paper proposes an improvement over the algorithm (RRT, Dijkstra), for the optimal path planning for wheeled mobile robots. The proposed algorithm generates a better solution and shorter path with low cost. While reducing the cost of the path and reducing the turning points is the advantage of the algorithm, the increase in the computational complexity is the disadvantage. Comparative results are carried out in seven environments that confirm the effectiveness of the proposed methods. The purpose of this paper is to perform a comparative analysis of RRT and RRT+ Dijkstra based planning algorithms in a 2D environment as per stated metrics.

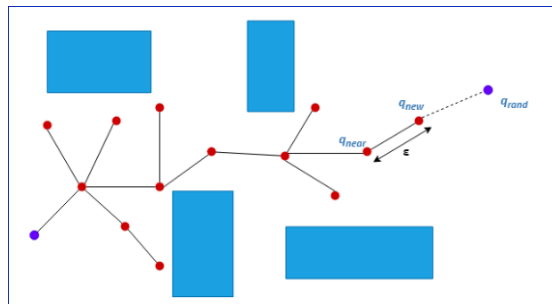
The rest of the paper is as follows: Section II addresses the problem methodology and explains the proposed algorithm. Section III presents the simulation results and analysis of performance. Section IV includes conclusion and future recommendations.

## **2. METHODOLOGY**

The proposed methods were conducted in a dispersed environment with different forms of obstacles. The experimental environments used here, and the image processing processes applied to achieve these environments, were detailed in our previous study [24]. These are an extended version of the work presented in [24] with a detailed evaluation of performance parameters for RRT and Dijkstra. MATLAB and LabVIEW software were used to perform numerical and graphical for comparison and analysis. Under the same conditions, the execution time, path length, and the number of turns performance parameters have compared and tested. As input to the respective planner, 2D environment maps of different cases are provided throughout tests. The proposed algorithms are described in the following subsections.

## A. RRT ALGORITHM

A rapidly exploring random tree (RRT) is a search algorithm with a single-query tree structure based on uniform random sampling, which is the start and goal point are certain known. The RRT algorithm grows based on the configurations of construction, a tree seeking the target point from a starting point, where each node of the tree is a point (state) in the workspace [25]. This algorithm was introduced by LaValle [26]. The RRT method first starts a roadmap configuration with a starting point ( $q_{start}$ ) as a tree root [27]. Next, the algorithm selects a random point ( $q_{rand}$ ) for each next iteration in the configuration space, and the nearest node ( $q_{near}$ ) from the existing graph is searched. A new sample ( $q_{new}$ ) is generated with a pre-defined distance ( $\epsilon$ ), namely a distance of step size, from  $q_{near}$  to  $q_{rand}$ . A collision is considered in the newly selected node ( $q_{new}$ ). If a collision occurs, a new step ( $q_{new}$ ) is discarded. Otherwise, it is added to the search tree so that for each new point, the distance between the newly generated node and the target point is considered [28]. The procedure is illustrated in Figure 1.



**Figure 1.** Extension of the RRT graph using the straight-line local planner with a resolution ( $\epsilon$ ).

Due to the random growth of the RRT, it is very time consuming when searching in the wider configuration area. The search for the growth of the tree will continue according to the predetermined threshold. The path is found if the distance is less than or equal to the set threshold. Otherwise, the distance is repeated until the distance is less than or equal to the specified threshold value, or the call times exceed the number of iterations. It can be considered using the forward tree expansion motion model, which allows the RRT to find viable trajectories even under differential constraints. That is one of the advantages and reasons for the extensive use of RRT in robotics for motion planning of systems such as mobile robots [29].

## B. DIJKSTRA ALGORITHM

Dijkstra's algorithm was published by Edsger W. Dijkstra in 1959 [2]. It is a graph search algorithm for finding the shortest path between nodes with a positive edge. It is separated from the A\* algorithm by not using the heuristic function [30]. Starting the calculation from the root node, the route with the lowest cost is selected, excluding routes with high-costs. The Dijkstra algorithm uses "infinite" for nodes that are marked as unvisited, and the starting point is marked as "0" ( $q_{init}$ ). From the current node, the temporary distance is calculated to all the unvisited neighbors. If the newly calculated distance is shorter than the previously saved temporary distance, this is selected as a new

node. By linking the shorter distances between the nodes, the most cost-effective path in the search area given between the start and the target is determined. The algorithm's pseudo-code is given below.

*Dijkstra (Graph G, Vertex s)*

1. *Initialize (G,S);*
2. *Priority\_Queue minQ={all vertices in V};*
3. **while** (*minQ*  $\neq \emptyset$ ) **do**
4.     *Vertex u=ExtractMin(minQ);//minimum est(u);*
5.     **for** (*each v*  $\in$  *minQ* *such that (u,v)  $\in$  E*
6.         *Relax (u,v);*
7.     **end for**
8. **end while**

The algorithm maintains a priority queue minQ order that used to store the shortest path estimates (v) as key values for unvisited nodes and repeatedly extracts the minimum est(u) from minQ and relaxes all edges incident from u to any vertex in minQ.

### 3. PERFORMANCE ANALYSIS

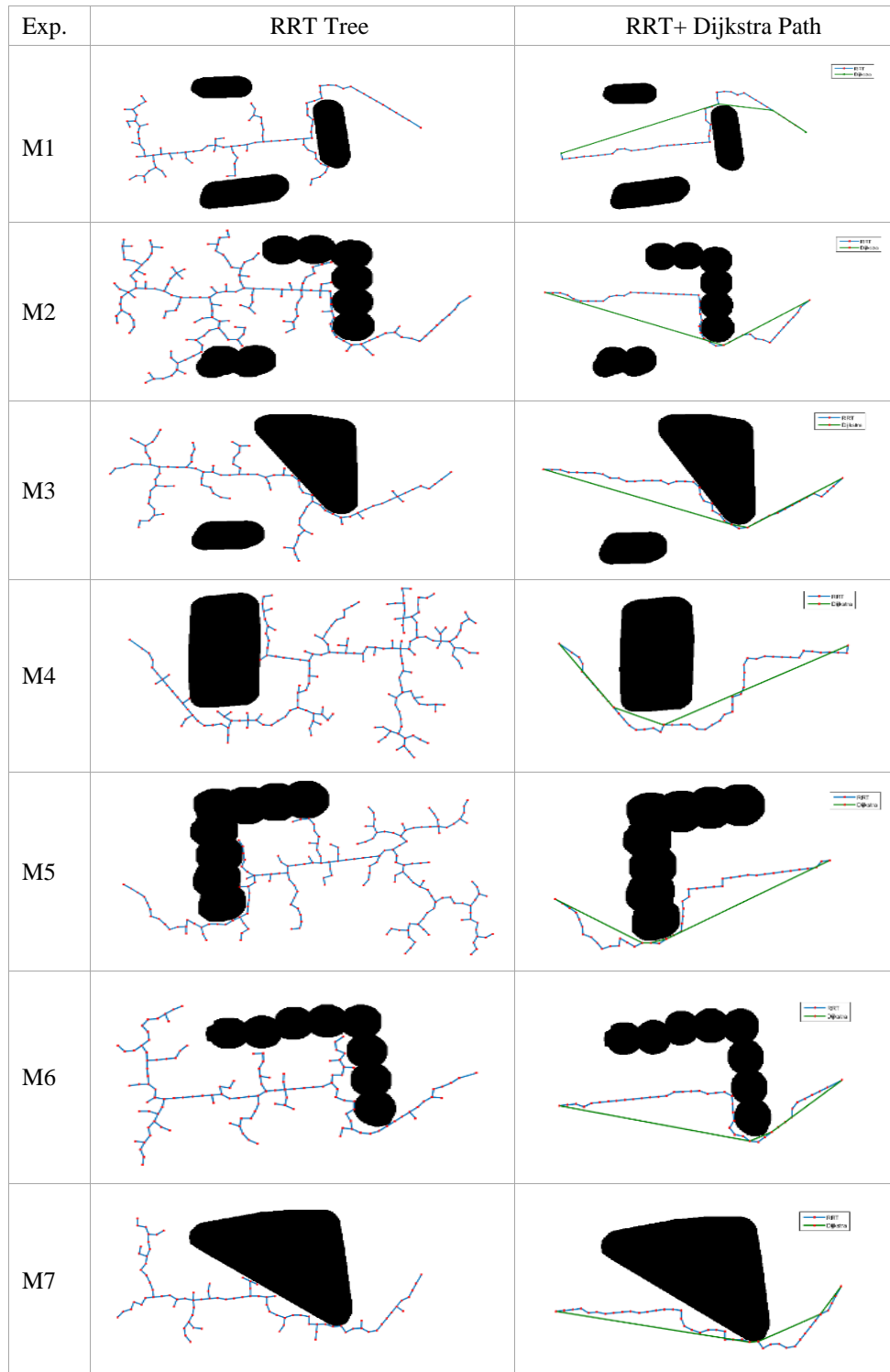
#### A. DATA SET

The experiments have been conducted in seven configuration spaces represented by M1 to M7 as shown in Figure 2. M1, M2, M4, M5 and M6 represents a basic environment with static obstacles. However, another scenario of the complex concave environments for testing the robustness of the proposed approach and safe passage is also represented by M3 and M7 maps, respectively. These are convex hull [31] technique applied environments. These environments are adopted from [24]. The proposed method is evaluated graphically and numerical comparative analysis are implemented and tested in LabVIEW+ MATLAB. The criteria to be considered when determining the optimal path for a mobile robot are measured by various factors such as path length, collision-free area, the total number of turns, and execution time. The experiments have been performed in cluttered and complex unstructured environments with obstacles of different shapes. The results of all algorithms provided in same place for comparing performance parameters in path length, execution time, and the number of graphs.

#### B. EXPERIMENTAL RESULTS

In this section, we present the experimental results of the proposed algorithm and the comparison of their results. All approaches are shown for maps M1 to M7 in Figure 2. Path length, execution time, total number of nodes used in the path grid with comparison are also shown in Table 1 and Table 2. The visual comparison of RRT and RRT+ Dijkstra algorithms are shown in Fig. 3 (a) and (b). From these Figures it is clear that the proposed approaches has generated shorter paths and less turning points than RRT methods. It was seen that the execution time of the algorithm increased. This is because the two algorithms worked consecutively. The node-set (number of cells) generated by

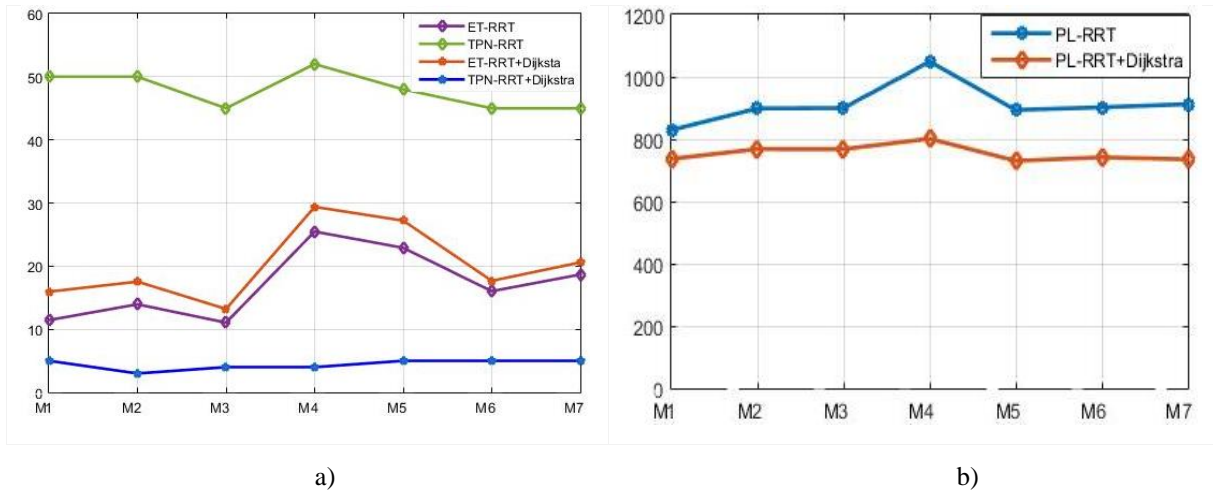
RRT algorithm created the search space of the Dijkstra algorithm. The proposed approach is flexible enough to generate safe and shorter path with less turning points. Reducing the number of nodes processed also reduced the path length by eliminating extra path points on the path.



**Figure 2.** Experimental results of environment maps M1 to M7 using RRT and RRT+ Dijkstra

The path created with RRT is not possible to follow because of the dynamic and kinematic constraints of mobile robots. This requires the robot to perform complex movements to track sharp

turns, resulting in high energy consumption, controller use, and premature aging of the robotic parts. The proposed method is presented as a solution to such situations in global path planning



**Figure 3.** Plot of path length (b), execution time and total path nodes (a) of all planners for all environment maps (PL: Path Length, ET: Execution Time, TPN: Total Path Nodes).

Plots of experimental results of the all planners are shown in Fig. 3. It is obvious from the results that the proposed methods generate shorter and relatively better solution paths for the densely cluttered environments. The proposed methods require fewer nodes, which makes the algorithm more efficient. From comparisons with RRT and RRT+ Dijkstra, we draw a conclusion that the performance of proposed methods in PL and TPN is best overall. However, the proposed algorithm had significantly lower performance, mainly due to the high computational time. For smoother routes and greater security levels, the proposed methods can be selected. It is suitable for wheeled mobile robot applications in offline path tracking. The numerical equivalent of the values of the above graph is given in Table 1.

**Table 1.** Experiments result for all planners (PL: Path Length, ET: Execution Time, TPN: Total Path Nodes)

Algorithm	M1	M2	M3	M4	M5	M6	M7	
RRT	PL-(px)	829,24	898,68	899,95	1048,7	893,76	902,20	912,45
	ET-(sc)	11,45	13,95	11,05	25,47	22,90	16,03	18,67
	TPN-(nodes)	50	50	45	52	48	45	45
	Z-Score	0,554	-0,155	0,257	0,454	0,168	0,327	-0,119
RRT+Dijkstra	PL-(px)	736,78	768,07	767,99	801,29	730,30	741,68	735,05
	ET-(sc)	15,95	17,54	13,21	29,38	27,21	17,67	20,59
	TPN-(nodes)	55	53	49	56	53	49	50
	Z-Score	0,127	-0,818	-0,063	0,302	-0,015	0,179	-0,263

Table 2 presents statistical analysis of results. PL, ET and TPN in all environments maps for all planner are listed in this Table. RRT+ Dijkstra algorithms generate better path with less PL and TPN.

<b>PL-(px)</b>	11,15 % less	14,53 % less	14,66 % less	23,59 % less	18,29 % less	17,79 % less	19,44 % less
<b>ET-(sc)</b>	39,30 % more	25,73 % more	19,55 % more	15,35 % more	18,82 % more	10,23 % more	10,28 % more
<b>TPN-(nodes)</b>	90,00 % less	94,00 % less	91,11 % less	92,31 % less	89,58 % less	91,11 % less	88,89% less

**Table 2.** Statistically, comparison of all approaches for ET, PL, and TPN (PL: Path Length, ET: Execution Time, TPN: Total Path Nodes)

Both algorithms (RRT and RRT+ Dijkstra) are efficient at finding an obstacle-free path in different experimental environments. However, it has been observed that the proposed planner reduces PL and TPN compared to traditional RRT and also increases ET. It is made possible by the proposed method to obtain values that can further reduce the path length and the number of path nodes and result in a flattened path. It is relatively better solution.

#### 4. CONCLUSIONS

In this paper, image-based RRT and Dijkstra algorithms are presented to find collision-free path planning in a cluttered environment. The performance of RRT was compared with the algorithms of RRT + Dijkstra for different scenarios.

The experimental results show that the path obtained only using RRT is not suitable for the mobile robot to follow because it has sharp transitions and a curved structure. Considering the dynamic and kinematic constraints of mobile robots, it is desirable to develop an appropriate algorithm for global path planning and path tracking. Because performing complex movements to follow sharp turns can result in the robot's high energy consumption, controller use, and premature aging of the robotic parts.

In order to improve this situation or obtain more suitable path coordinates, RRT and Dijkstra algorithm have been applied together. It is clear that due to the use of Dijkstra and RRT together, the total length of the path decreases while the time spent in path planning is prolonged. The method we propose is mainly aimed at creating a costly and traceable route. It has been proven that RRT + Dijkstra effectively produces shorter paths. In addition, computing efficiency and less memory requirements are important criteria for small robots. Experiments are carried out in different 2D environments. The results obtained also show the effectiveness and applicability of the proposed method. In future research, we plan to carry out path tracking application of mobile robots on the planned route in real time.

## REFERENCES

- [1] F. Duchoň *et al.*, “Path Planning with Modified a Star Algorithm for a Mobile Robot,” *Procedia Eng.*, vol. 96, pp. 59–69, 2014, doi: 10.1016/j.proeng.2014.12.098.
- [2] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959, doi: 10.1007/BF01386390.
- [3] S. A. Fadzli, S. I. Abdulkadir, M. Makhtar, and A. A. Jamal, “Robotic Indoor Path Planning using Dijkstra ’ s Algorithm with Multi-Layer Dictionaries,” pp. 1–4, 2015.
- [4] P. Hart, N. Nilsson, and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968, doi: 10.1109/TSSC.1968.300136.
- [5] J. J. Kuffner and S. M. La Valle, “RRT-connect: an efficient approach to single-query path planning,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2000, doi: 10.1109/robot.2000.844730.
- [6] J. Bruce and M. Veloso, “Real-time randomized path planning for robot navigation,” in *IEEE/RSJ International Conference on Intelligent Robots and System*, 2002, vol. 3, pp. 2383–2388, doi: 10.1109/IRDS.2002.1041624.
- [7] A. Bry and N. Roy, “Rapidly-exploring random belief trees for motion planning under uncertainty,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, doi: 10.1109/ICRA.2011.5980508.
- [8] T. Weerakoon, K. Ishii, and A. A. F. Nassiraei, “An Artificial Potential Field Based Mobile Robot Navigation Method To Prevent From Deadlock,” *J. Artif. Intell. Soft Comput. Res.*, vol. 5, no. 3, pp. 189–203, Jul. 2015, doi: 10.1515/jaiscr-2015-0028.
- [9] E. Rimon and D. E. Koditschek, “Exact robot navigation using artificial potential functions,” *IEEE Trans. Robot. Autom.*, vol. 8, no. 5, pp. 501–518, Oct. 1992, doi: 10.1109/70.163777.
- [10] H. Kaluder, M. Brezak, and I. Petrović, “A visibility graph based method for path planning in dynamic environments,” in *MIPRO 2011 - 34th International Convention on Information and Communication Technology, Electronics and Microelectronics - Proceedings*, 2011.
- [11] R. Wein, J. P. Van Den Berg, and D. Halperin, “The visibility-Voronoi complex and its applications,” in *Computational Geometry: Theory and Applications*, 2007, doi: 10.1016/j.comgeo.2005.11.007.
- [12] P. Yap, “Grid-based path-finding,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2002, doi: 10.1007/3-540-47922-8\_4.
- [13] A. I. Panov, K. S. Yakovlev, and R. Suvorov, “Grid path planning with deep reinforcement learning: Preliminary results,” in *Procedia Computer Science*, 2018, doi: 10.1016/j.procs.2018.01.054.
- [14] E. Donmez, A. F. Kocamaz, and M. Dirik, “Bi-RRT path extraction and curve fitting smooth with visual based configuration space mapping,” in *International Artificial Intelligence and Data Processing Symposium (IDAP)*, Sep. 2017, pp. 1–5, doi: 10.1109/IDAP.2017.8090214.
- [15] R. Sadeghian, S. Shahin, and M. T. Masouleh, “An experimental study on vision based controlling of a spherical rolling robot,” in *Iranian Conference on Intelligent Systems and Signal Processing (ICSPIS)*, Dec. 2017, pp. 23–27, doi: 10.1109/ICSPIS.2017.8311583.
- [16] L. E. Kavradi, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, 1996, doi: 10.1109/70.508439.
- [17] C. Lamini, S. Benhlima, and A. Elbekri, “Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning,” *Procedia Comput. Sci.*, vol. 127, pp. 180–189, 2018, doi: 10.1016/j.procs.2018.01.113.
- [18] Jianping Tu and S. X. Yang, “Genetic algorithm based path planning for a mobile robot,” in *International Conference on Robotics and Automation (Cat. No.03CH37422)*, 2003, vol. 1, pp. 1221–1226, doi: 10.1109/ROBOT.2003.1241759.
- [19] L. Moreno, J. M. Armingol, S. Garrido, A. De La Escalera, and M. A. Salichs, “A genetic algorithm for



- mobile robot localization using ultrasonic sensors,” *J. Intell. Robot. Syst. Theory Appl.*, 2002, doi: 10.1023/A:1015664517164.
- [20] J.-Y. Jhang, C.-J. Lin, C.-T. Lin, and K.-Y. Young, “Navigation Control of Mobile Robots Using an Interval Type-2 Fuzzy Controller Based on Dynamic-group Particle Swarm Optimization,” *Int. J. Control. Autom. Syst.*, vol. 16, no. 5, pp. 2446–2457, Oct. 2018, doi: 10.1007/s12555-017-0156-5.
- [21] T. W. Liao, “A procedure for the generation of interval type-2 membership functions from data,” *Appl. Soft Comput.*, vol. 52, pp. 925–936, Mar. 2017, doi: 10.1016/j.asoc.2016.09.034.
- [22] M. Dirik, “Development of vision-based mobile robot control and path planning algorithms in obstructed environments,” Inonu University, 2020.
- [23] M. Fu, J. Li, and Z. Deng, “A practical route planning algorithm for vehicle navigation system,” in *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 2004, doi: 10.1109/wcica.2004.1343742.
- [24] M. Dirik and A. F. Kocamaz, “Global Vision Based Path Planning for AVGs Using A\* Algorithm.” *Journal of Soft Computing and Artificial Intelligence*, 1, pp. 18–28, 2020.
- [25] X. Cao, X. Zou, C. Jia, M. Chen, and Z. Zeng, “RRT-based path planning for an intelligent litchi-picking manipulator,” *Comput. Electron. Agric.*, vol. 156, pp. 105–118, Jan. 2019, doi: 10.1016/j.compag.2018.10.031.
- [26] S. M. LaValle, “Rapidly-Exploring Random Trees: A New Tool for Path Planning,” *Iowa State Univ. Ames, IA 50011 USA*, vol. 6, no. 2, p. 103, 1998.
- [27] N. Chao, Y. Liu, H. Xia, A. Ayodeji, and L. Bai, “Grid-based RRT\* for minimum dose walking path-planning in complex radioactive environments,” *Ann. Nucl. Energy*, vol. 115, pp. 73–82, May 2018, doi: 10.1016/j.anucene.2018.01.007.
- [28] L. Jaillet, A. Yershova, S. M. La Valle, and T. Simeon, “Adaptive tuning of the sampling domain for dynamic-domain RRTs,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 2851–2856, doi: 10.1109/IROS.2005.1545607.
- [29] A. Viseras, D. Shutin, and L. Merino, “Robotic Active Information Gathering for Spatial Field Reconstruction with Rapidly-Exploring Random Trees and Online Learning of Gaussian Processes,” *Jr. Sensors*, vol. 19, no. 5, pp. 10–16, Feb. 2019, doi: 10.3390/s19051016.
- [30] G. Klančar, A. Zdešar, S. Blažič, and I. Škrjanc, *Wheeled Mobile Robotics, From Fundamentals Towards Autonomous Systems*. Butterworth-Heinemann, © 2017 Elsevier Inc., 2017.
- [31] F. Yaacoub, Y. Hamam, A. Abche, and C. Fares, “Convex Hull in Medical Simulations: A New Hybrid Approach,” in *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, Nov. 2006, pp. 3308–3313, doi: 10.1109/IECON.2006.347668.