A Review on Optimal Trajectory Planning of Industrial Robot using Graph Search Algorithms

Yash Naik
Department of Mechanical Engineering, MIT Art, Design & Technology University, Pune, India.
rohansm2001@gmail.com

B. K. Patle
Department of Mechanical Engineering, MIT Art, Design & Technology University, Pune, India.
balu_patle@rediffmail.com

Praveen Kumar Bhojane
Department of Mechanical Engineering, MIT Art, Design & Technology University, Pune, India.
praveenbhojane@gmail.com

***Abstract*-- This comprehensive paper presents a comprehensive overview of graph search-based algorithms utilized for trajectory planning in the context of industrial robotic arms. The study focuses on four prominent algorithms which are A* Algorithm, Dijkstra's Algorithm, Rapidly-Exploring Random Tree (RRT), and Probabilistic Roadmap (PRM). This review synthesizes existing literature on these algorithms, highlighting their strengths, limitations, and applications in industrial robotic arm trajectory/path planning. Additionally, it identifies emerging trends, challenges, and opportunities for further research in this dynamic field. By offering a comprehensive overview, this paper serves as a valuable resource for researchers, practitioners, and enthusiasts seeking to understand and advance trajectory planning for industrial robotic arms.**

***Keywords*– Industrial robotic arm, trajectory planning, Graph-Search algorithms**

I. Introduction

Industrial robots have seen substantial growth in various industries, performing functions such as material handling, packaging, welding, labelling, painting, palletizing, assembly and disassembly, testing, transportation, inspection, space exploration, and more. The efficiency of industrial operations depends on the operational speed. Industrial robotic arms are important due to their ability to function constantly without tiredness, resulting in higher production rates and increased total productivity [1]. These robots can perform jobs with repetitive accuracy more quickly and consistently than human workers. Robotic arms are engineered to execute tasks with exceptional precision and accuracy. This is crucial in areas like manufacturing, where minor variances can greatly impact product quality. Robots can sustain steady performance for long durations, therefore keeping a continuous level of work quality. To enhance operational speed, it is essential to reduce the total travel time of the robot while considering the input limits related to torque. When working on a task, a robot's manipulator needs to carefully plan its trajectories, which can be difficult because of the numerous joints involved. The joints need to precisely track the desired trajectory to perform the given task efficiently, optimizing variables like time, energy, and jerk. Efficiently managing time, energy, and jerk in path planning is a critical topic in robotics that is receiving considerable interest from academics worldwide. Path planning is a crucial feature for enhancing the efficiency of industrial robots. Path planning involves finding the optimal sequence of motions for a robotic arm to achieve its goal, taking into account restrictions such obstacle avoidance, energy efficiency, and end-effector precision [2]. The objective is to reduce the time, energy, or resources needed to complete a task while avoiding obstacles. Path planning is also crucial in managing industrial robotic arms to execute jobs accurately and effectively. This introduction gives a summary of the path planning methods used to optimize the trajectory of industrial robotic arms. Path planning techniques are crucial for improving the efficiency of autonomous systems by assisting them in navigating complicated surroundings, avoid obstacles, and find the best routes. **Fig.1** shows a path planning environment with obstacles and industrial robotic arm.

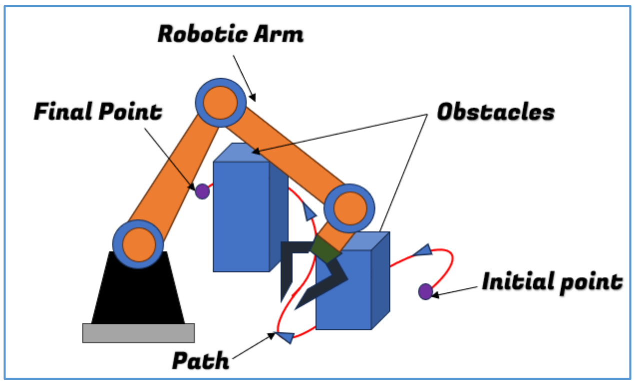


Fig. 1 Robotic arm Path planning avoiding obstacles

Multiple case studies illustrate the actual implementation of path planning methods for industrial robots. **Fig.2** represents the processes that are used for path planning of industrial robotic arm. Firstly, define initial and final point then according to the environment the maps are generated in through localization. Then algorithms are implemented and best path solution is formed out of generated paths. Lastly after best path solution the robotic arm manipulator moves from initial to final point [3].

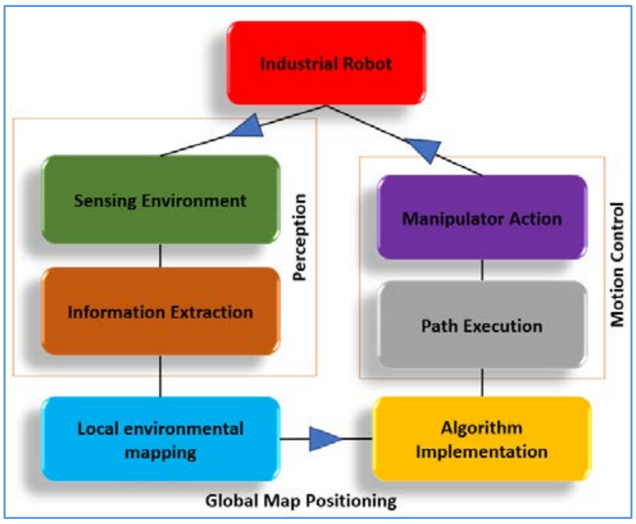


Fig. 2 Process of path planning of Industrial robotic arm

II. Trajectory Planning Technique of Industrial Robotic Manipulator

Graph search algorithms are used in trajectory planning for industrial robotic arms by representing the arm's configuration space and environment as a graph. Nodes represent possible states, and edges represent possible transitions. The goal is to find a path that leads the arm from its initial position to a desired goal position while avoiding obstacles and adhering to constraints like kinematics, workspace limitations and collision avoidance. This review work focuses on four prominent graph search algorithm which are A* Algorithm, Dijkstra Algorithm, RRT Algorithm, PRM Algorithm as shown in **Fig.3**.

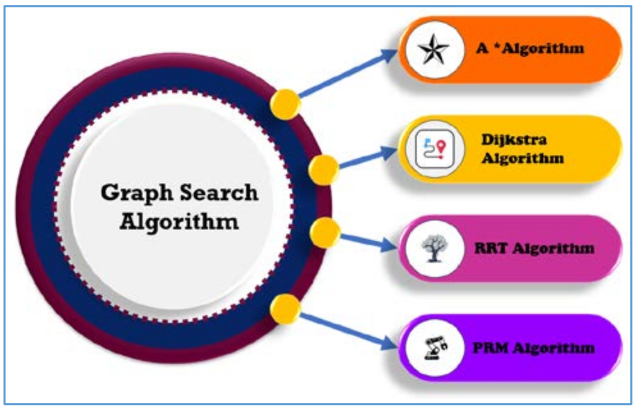


Fig. 3 Types of Graph Search Algorithms

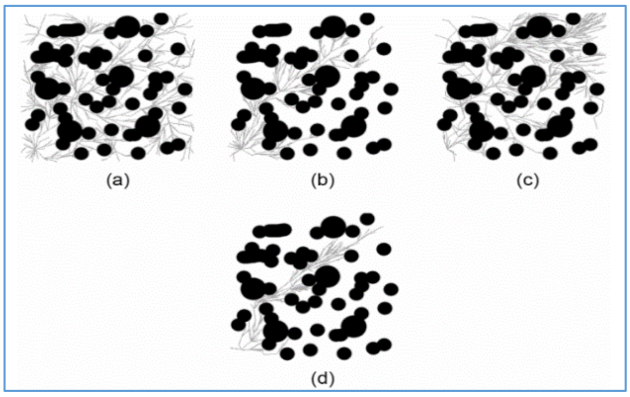
Graph search-based algorithms are commonly used in industrial robotic arms for path planning. These algorithms involve dividing the robot's configuration space into a graph, with each node representing a configuration and each edge representing a possible motion between two configurations [4]. The search starts with the current node and adds it to an open list. From this node, it explores its neighbours, calculates the cost of reaching them, and updates the cost if it results in a lower cost path. If the cost is lower than the path from the start configuration, the neighbour's information is updated in the open list. If the current node is the goal configuration, the search is successful, and the algorithm terminates. If not, the next node is moved until the goal is reached or the open list becomes empty. Once the goal is reached, the path is reconstructed by backtracking from the goal configuration to the start configuration [5].

2.1 A Algorithm*

The A* algorithm is renowned for its optimality and completeness, making it ideal for finding the optimal path in scenarios where finding the path is crucial. It involves initializing nodes, expanding them systematically, evaluating viable paths using a cost function, focusing on potential paths with lower projected costs, checking for environmental collisions, stopping at the goal configuration, and reconstructing the optimum path using search data [6]. It is commonly used in robotics, particularly for industrial robotic arms. The process of A * Algorithm involves initializing an open set with a start node and an empty closed set. The next step is to evaluate the node by selecting the one with the lowest f-score from the open set. The f-score is the sum of the cost to reach the node from the start and the heuristic estimate of the cost to reach the goal. The node is then moved from the open set to the closed set, considering its neighbouring nodes. The g-scores for these neighbours are calculated based on the cost of moving. If necessary, the neighbours' f-scores are added to the open set [7]. Finally, the path is reconstructed by backtracking through the parent nodes recorded during the search process.

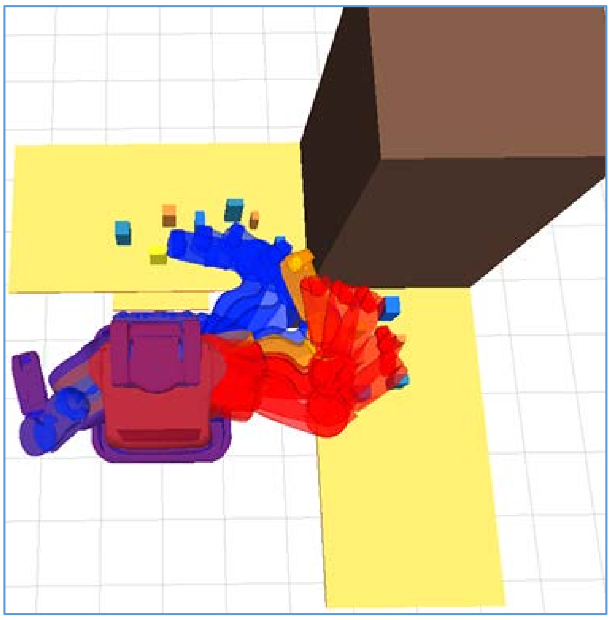
In this algorithm, the classical A* algorithm is less used compared to the improved A* algorithm. Also, this algorithm is mostly modified according to the requirements or used with its hybrids. A Path-planning algorithm which is generalization of the classic A* algorithm introduced by M.Perrson et.al [8], focusing on probabilistic expected values and density of nodes. Two combine algorithms, SBA* and RRT*, are used for path analysis and generation. SBA* improves success rates, solution quality, and improvement in later phases by exploring the optimal path region.

Fig.4 Trajectory graphs generated after 700 iterations on a highly clustered environment: (a) RRT*; (b) RRT* with BnB; (c) SA SBA*; (d) SA-SBA*with BnB [8]



Combining exploratory and exploitative rounds with a simulated annealing schedule enhances performance in most areas. **Fig.4** shows comparison of path generated. SBA* algorithms gracefully degrade in high-dimensional spaces, and the average number of nodes needed depends on dimensionality under obstacle-free conditions. Graph-based algorithms have two types of searches which are bidirectional and unidirectional searches. Bidirectional searches start in both directions, which is faster and more efficient in computing. A* connect algorithms, as discussed by V.Narayanan et.al [9], possess traits of bidirectional heuristic graph search based algorithms. He also introduced two arm robots for object placement. Although A* algorithms have limitations, a novel approach was developed that significantly outperforms conventional algorithms for path planning in industrial manipulators. Fig.5 gives clear understanding of trajectory generation from start point to end point. The Improved A* algorithm is a derivative of traditional algorithms. Zexin Huang et.al [10] introduced a method for path planning of industrial robotic manipulators, minimizing operational time, computational power usage, and producing shorter path dimensions. This technique optimizes the insertion of surgical needles into human blood vessels in complex procedures, outperforming RRT, APF, and Classic A* algorithms.

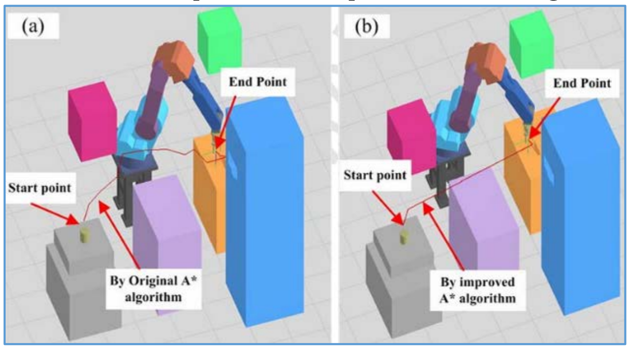
Fig. 5 A*connect algorithm is used to create optimal paths by displaying arm visualizations in red, blue, and orange, indicating the forward, backward, and intersection states of the path [9].



Yang et.al [11] developed an improved A* algorithm using the 6-DOF Robotic arm and BM2DKPCANeT and SoftMax software for robot path planning. The modified algorithm creates zigzag paths and promotes search failure at random sample nodes. After processing, intervening nodes can be removed to shorten and straighten the local path. Experimental results show the modified algorithm outperforms the original. Zheng et.al [12] developed an improved A* algorithm for robot path planning, overcoming challenges caused by zigzag paths. The algorithm finds local paths between current and goal nodes before each search, eliminating intermediate nodes and straightening the path. Despite fewer sampling points, the algorithm outperforms the original in search success rates and path length, making it suitable for robot path planning in familiar environments. Fig.6(a and b) mentioned below is comparison of path generated by classical A* algorithm and Improved A* algorithm.

Fig. 6 The classical A* algorithm (a) compared to improved A* algorithm (b) [14].

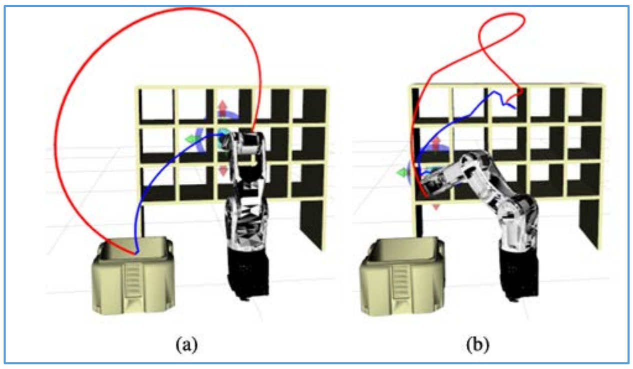
Kai Mi et.al [13] presents an improved SMHA* algorithm



for obstacle avoidance path planning in narrow spaces. It introduces multiple inadmissible heuristics to avoid search stagnation caused by inappropriate consistent heuristics. Additionally, it adds stagnation detection to each expansion node to address the issue of unnecessary heuristics increasing search burden when consistent heuristics are appropriate. Fig.7(a and b)mentioned below also shows the red and blue path generated using RRT connect and SD-SMHA* algorithmic respectively for two different operations.

Fig. 7 The end-effector paths, represented by the red curve and the blue curve, is generated by RRT Connect and SD-SMHA* respectively [13].

The improved algorithm uses stagnation detection as a



bridge, absorbing the advantages of the original Weighted A* and SMHA* algorithms. The proposed algorithm is more suitable for cluttered environments with narrow passages and offers more advantages in planning time and path length compared to sampling-based planners. The A* Algorithm is ideal for optimality-focused scenarios with static or known constraints. It ensures finding the shortest path using an admissible heuristic, making precision crucial. However, it may struggle in dynamic environments or with uncertain constraints, and can be computationally expensive in large graphs.

2.2 Dijkstra Algorithm

Dijkstra's Algorithm is a classical graph search algorithm used for finding the shortest path from a start node to all other nodes in a weighted graph [15]. In the context of trajectory/path planning for industrial robotic arms, Dijkstra's Algorithm is employed to determine the optimal path for the robotic arm to traverse from its initial configuration to a desired goal configuration while avoiding obstacles and adhering to constraints. The robotic arm's configuration space is represented as a weighted graph, with nodes representing different configurations and edges representing feasible transitions. The algorithm initializes all nodes with a tentative distance value, which represents the shortest known distance from the start node to each node in the graph. A priority queue is used to store nodes based on their tentative distance values. The algorithm iteratively selects the node with the smallest tentative distance from the priority queue and evaluates its neighbouring nodes. If the calculated tentative distance is smaller than the current distance, it is updated. Once all nodes have been evaluated, the algorithm reconstructs the shortest path from the start node to the goal node, representing the optimal trajectory for the robotic arm to navigate from its initial configuration to the desired goal configuration [**16**].

Muthuswamy et.al [**17**] introduces a computerized strategy to generate a desirable route for robot manipulators, taking into account hurdles and workspace singularities. The method utilizes robot design characteristics, the dimensions and positions of obstacles, as well as the initial and objective states to produce a path that is free from collisions. The robot workspace is created and divided into discrete units, while impediments are represented as areas that the robot is not allowed to enter. The process of finding the most efficient route starts with establishing a search space, listing all potential routes inside a network graph structure, and implementing Dijkstra's algorithm for determining the path with the lowest cost. The algorithm's computational complexity is inversely proportional to the quantity and dimensions of obstacles present in the workspace. A computer software with interactive capabilities has been developed to implement this concept for a manipulator with two links in a planar configuration.

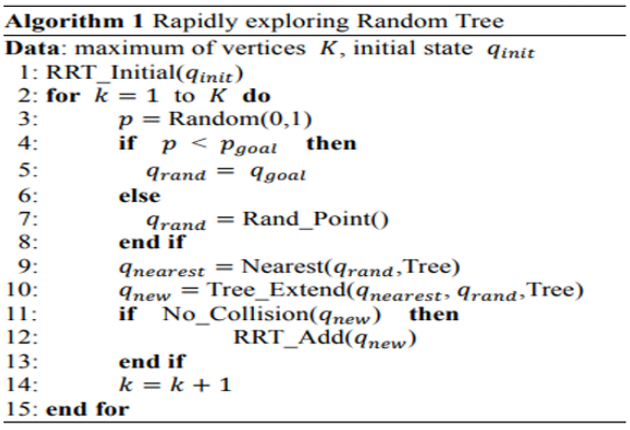
Dirik et.al [**18**] addressed a path planning problem in mobile robots has been a topic of significant research. Recently, sampling-based algorithms like RRT have gained attention due to their asymptotic optimization. However, RRT's use near obstacles and sharp turns makes it inefficient for real-time path tracking applications. This paper proposes a combination of RRT and Dijkstra algorithms, RRT-Dijkstra, to provide a shorter and collision-free path solution. The paper reviews and compares these planners based on metrics like path length, execution time, and total number of turns. The experimental performance shows that RRT-Dijkstra requires less turning point and execution time in 2D environments, making it suitable for off-line path planning and path following.

Ramkumar et.al [**19**] study explores robot path planning and trajectory planning using Dijkstra algorithm parameters in various environments. The mobile robot model was developed using V-REP open-source simulation software and the Dijkstra algorithm for identifying sub-optimal and collision-free paths. The simulated results showed that the reduction method was effective in terms of time and velocity in the created environments for robot path planning. This approach simplifies the process of finding the most efficient path in a given environment. C Zhang et.al [**20**] proposes an improved Dijkstra algorithm for visual servo robot control systems to avoid fixed and non-fixed obstacles. Maps are created in the robot's actual movement environment, and a model is established to mark target points and obstacle information. The Dijkstra algorithm is used for optimal path planning, combined with hardware sensors. Dynamic obstacles are detected to improve the algorithm, as Dijkstra's algorithm cannot avoid them. The improved Dijkstra algorithm is applied to grip control of visual servo robots, and a practical running experiment verifies the method's robustness and effectiveness. Dijkstra's Algorithm is suitable for applications requiring optimality and static environments. It ensures finding the shortest path in weighted graphs, making it ideal for precise trajectory planning. However, it may become computationally expensive in large graphs or high-dimensional spaces and may not efficiently handle dynamic environments.

2.3 Rapidly Exploring Random Tree Algorithm

The RRT technique is an algorithm specifically developed to effectively explore nonconvex, high-dimensional areas by randomly constructing a tree that fills the space. By randomly selecting samples from the search space, the tree grows gradually into large, unexplored workspace regions [**21**]. The RRT technique involves identifying the initial and final positions of the robot, and then gradually building a tree by randomly selecting samples from the search area. The tree exhibits an innate tendency towards expanding its growth in the direction of sizable unexplored regions of the problem.

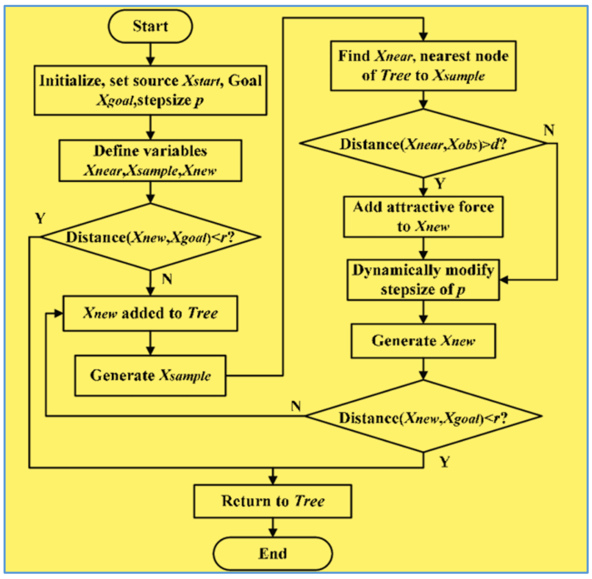
Fig. 8 Pseudo Code of RRT Algorithm distance from the obstacles nearest to the robot, providing the robot with a comparatively secure path [22].



```
Algorithm 1 Rapidly exploring Random Tree
Data: maximum of vertices $K$, initial state $q_{init}$
 1: RRT_Initial($q_{init}$)
 2: for $k = 1$ to $K$ do
 3:     $p$ = Random(0,1)
 4:     if $p < p_{goal}$ then
 5:         $q_{rand} = q_{goal}$
 6:     else
 7:         $q_{rand}$ = Rand_Point()
 8:     end if
 9:     $q_{nearest}$ = Nearest($q_{rand}$,Tree)
10:     $q_{new}$ = Tree_Extend($q_{nearest}$, $q_{rand}$,Tree)
11:     if No_Collision($q_{new}$) then
12:         RRT_Add($q_{new}$)
13:     end if
14:     $k = k + 1$
15: end for
```

In **Fig.8** above , the code utilized for the RRT algorithm is also mentioned.

Fig.9 Flow-diagram of RRT Algorithm [23]


Start
Initialize, set source Xstart, Goal Xgoal,stepsize p
Define variables Xnear,Xsample,Xnew
Distance(Xnew,Xgoal)<r?
Y
N
Xnew added to Tree
Generate Xsample
Find Xnear, nearest node of Tree to Xsample
Distance(Xnear,Xobs)>d?
N
Y
Add attractive force to Xnew
Dynamically modify stepsize of p
Generate Xnew
Distance(Xnew,Xgoal)<r?
N
Y
Return to Tree
End

\

The above mentioned flowchart **Fig.9** gives a clear understanding about the working of the RRT algorithm for generating highly efficient paths. The two mechanisms for various RRT algorithms of industrial robots, in the context of configuration space path planning is presented in the work of Zheng et.al [24]. The improved RRT algorithm proposed utilizes the boundary expansion mechanism and the regression mechanism to solve the problem of being often trapped in local minima during the planning process **Fig.10** shows that how the nodes are created from the initial position to the final position while creating a roots/branches like structured paths in basic RRT algorithm. In **Fig.10** , the **(a)** part represents all the groups of nodes that are formed through the algorithm . The other part is **(b),** which consists of selected nodes that will be added to the final path of tree which will give the optimal path for the industrial robotic arm [24]. It highlights the work done with the help of Nurbs safety formula and RRT algorithm collectively that showcases the obstacles in three dimensional workspace.

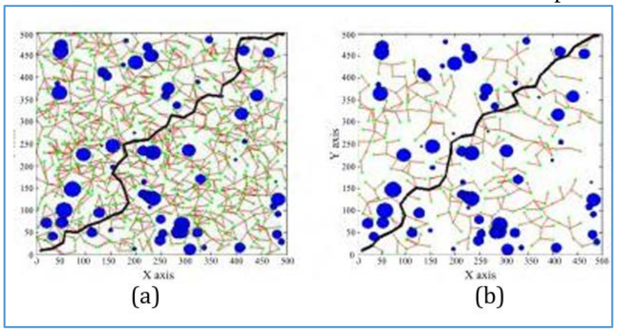


Fig. 10 Tree root/branch like structure formation[24]

The optimal path as shown in RRT algorithm is a work done by Ru-Xiao et.al [25], you can see that in **Fig.11** below. In **Fig.11(a)** explains 3D representation of the workspace whereas in **Fig.11(b)** represents 2D workspace. In both the figures we can clearly see the path that is formed after selection of various nodes. Analysis of significant advancements in optimal path planning with RRT* planning algorithm and its expanded variations within the past six years is presented by the Zulfiqar Habib et.al [26].Using the v-rep and MATLAB , occlusion free path planning is carried out using 7-DOF robotic arm with the help of vision sensor which is presented by Jae Kim et.al [27]. In this case arm is placed in the wall and objects are placed on the table and with the help of RRT path planner ,obstacle-avoidance paths are generated. Another approach is with the help of lab analysis, the manipulator's kinematics, D-H parameter model, and working space are determined using Monte Carlo based solution. A collision detection approach integrating the cylindrical bounding box and AABB method is developed and presented by the M Wang et.al [28].

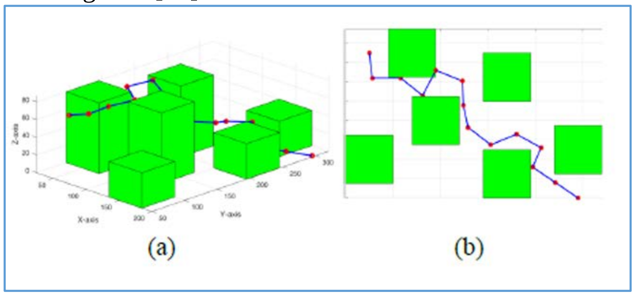


Fig. 11 3D and 2D representation of path through RRT[25]

The PR-RRT algorithm is developed using the RRT motion planning algorithm and RRT* rewiring approach. ROS based optimal path planning also done on 7 DOF Franka Emika Panda robotic arm where with the help of Motion Planner named MoveIt which was the research work of the Friedmann et.al [29] ,in which every RRT algorithms and its variants were tested. For pick and place operation RRT Connect algorithms is the optimal option as it reduces the computational time and path cost. RRT is a probabilistic algorithm that finds solutions in high-dimensional spaces by iteratively sampling and expanding the search tree. It is suitable for complex robotic arm environments and has fast convergence due to its iterative sampling and expansion. However, RRT lacks determinism, which can lead to variability in generated paths, making it less suitable for deterministic applications. It may also struggle to find optimal solutions, especially in narrow passages or complex obstacles [29]. Additionally, RRT's performance can be influenced by the sampling strategy used, with suboptimal sampling potentially resulting in inefficient exploration.

2.4 Probabilistic Roadmap Method Algorithm

The Probabilistic Roadmap Method (PRM) is a sampling-based technique for robotic motion planning, specifically for industrial robot arm optimal path planning. It solves high-dimensional configuration space and complex obstacle problems by creating a roadmap in configuration space by sampling random configurations and connecting them to generate a graph. This graph is used to plan robotic arm trajectories between initial and goal configurations [30]. The PRM works by sampling random configurations from the robotic arm's configuration space, checking for collisions with obstacles, discarding collision-prone configurations, and adding valid configurations to a graph representation known as the roadmap. Nodes in the graph correspond to valid configurations, and edges represent feasible paths between configurations. Connections between nodes are established by connecting nearby configurations in the roadmap, determined by a distance metric in the configuration space. Invalid edges are removed. Some variations of PRM include an optimization step to improve the quality of the roadmap or generated paths, such as adjusting node positions or refining paths. The roadmap is then used to perform a search using algorithms like Dijkstra's algorithm or the A* algorithm. After obtaining a path, a post-processing step may be applied to smooth or improve its quality. The final path obtained from PRM can be executed by the robotic arm in the real-world environment [31].The method involves sampling configurations and finding feasible paths for robotic arms, making it ideal for industrial applications where robots need to navigate cluttered environments and avoid obstacles. The above mentioned flowchart Fig.12 gives understanding of functioning of PRM algorithm on the robotic manipulators.

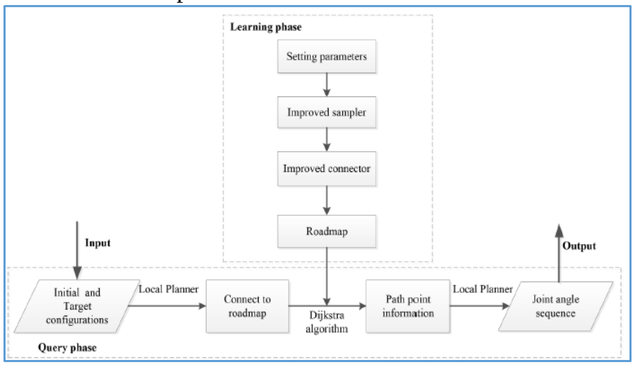


Fig.12 Flowchart of path planning of Robotic manipulator based on the PRM motion Planner [32].

An improved PRM method for efficiently planning a collision-free path for a high-DOF manipulator is introduced in the research work of D Liu et.al [32] which is used to improve sampling, a virtual force field is used to boost density in small passages and reduce redundancy in wide-open regions of free space. Secondly, a three-stage connection technique is created to link samples from the enhanced sampling strategy. Thirdly, the PRM algorithm model is enhanced to plan a collision-free path for manipulators, comprising the framework, local planner, and query phase. A semi-lazy probabilistic roadmap (SLPRM) method was introduced by the E Masehian et.al [33] for motion planning of industrial manipulators is a method that combines the features of the basic probabilistic roadmap (PRM) and lazy-PRM (LPRM) methods. The SLPRM algorithm differs from PRM and LPRM in that it does not do thorough collision-checking. Instead, it selectively analyses random configurations for collisions, focusing only on a specified number of terminal connections from the end-effector backward throughout the generation of the roadmap. Consequently, the building time for the roadmap is reduced compared to PRM since there are less collision checks.

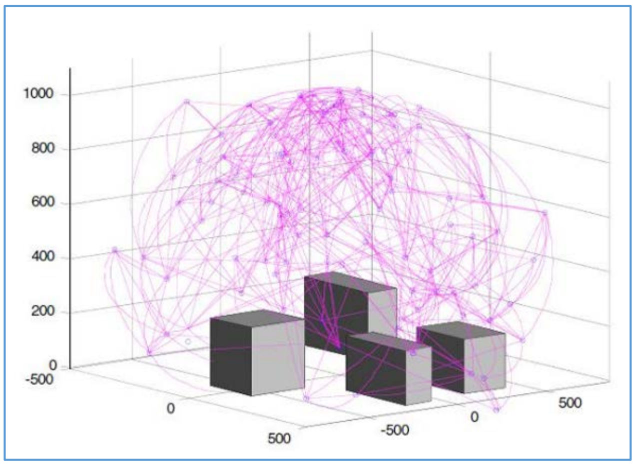


Fig.13 (a)Roadmap generation in SLPRM of 5 DOF robotic arm [33].

Fig.13 (a) shows us that all the possible paths that are generated in three dimensional space. Fig.13 (b) shows that final path generated by the algorithm. The two previous researches of PRM path planner were based on the static obstacle conditions ,although there is some work on dynamic obstacle conditions introduced by G Conclaves et.al [34] here the Dijkstra algorithms was introduced for shortest path along with the PRM. The approach comprises two phases: learning and query. In the learning phase, researchers sample the configuration space instead of computing it explicitly. In this phase, the roadmap is visualized as a graph with sampling configurations as vertex and connections between them as edges. In the query phase, they request a path between two free configurations. Collision detection can be a standalone module used in various steps of trajectory generation.

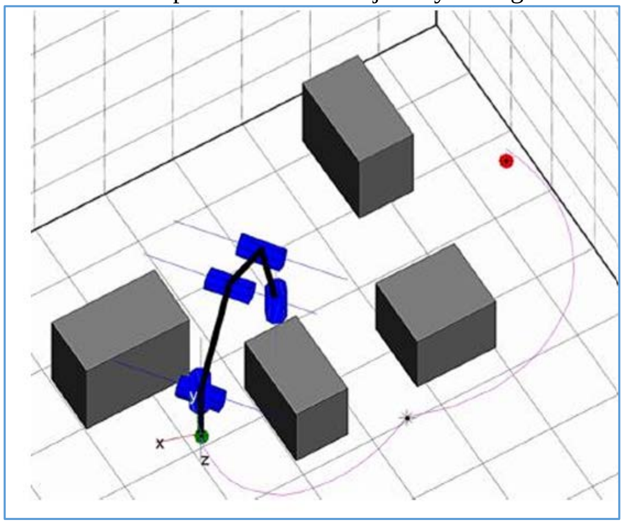


Fig.13 (b) collision-free path generated using SLPRM [33].

The Probabilistic Roadmap (PRM) is a tool that allows for global path planning by precomputing feasible paths between sampled configurations. It can handle uncertainty and variability by sampling multiple configurations and connecting them with collision-free paths. PRM also facilitates path smoothing techniques, enhancing the quality of generated paths and reducing jerky motions. However, it can be computationally complex, require significant memory resources, and be sensitive to sampling density, which can result in sparse roadmaps and suboptimal paths.

III. Result and Discussion

The proposed review paper provides an in-depth analysis of the contribution of graph search algorithm in the field for path planning techniques of industrial robotic manipulators. Path planning algorithm are most commonly used for time, energy and avoidance of obstacles. The shorter the path length more efficient is the algorithm. In the paper we have discussed about the graph search algorithm in which, while formation of path , they possess the traits of graph. The RRT and PRM algorithm are standalone in nature, yet they have some modified hybrids which stand them out in certain condition of trajectory planning of industrial robotic manipulators compared to that of other algorithms. In this paper for graph search algorithm, we have discussed four prominent algorithms which are RRT, PRM, A* algorithm, Dijkstra Algorithm. So, starting with the A* algorithm which is well-known and efficient path-finding method for finding the optimal path in scenarios where path discovery is crucial. It involves several steps, including initialization, cost evaluation, goal configuration, process steps, and reconstruction. The classical A* algorithm is less used compared to the improved A* algorithm, which is often modified or used with hybrids tailored to specific requirements. Dijkstra's Algorithm is a well-known graph search algorithm utilized to determine the shortest path from a given starting node to all other nodes in a graph with weighted edges. It is suitable for applications requiring optimality and static environments, ensuring the shortest path in weighted graphs. The Rapid Exploring Random Trees (RRT) technique is developed to explore nonconvex, high-dimensional areas by randomly constructing a tree that fills the space. It involves tree construction, node selection, and optimal path generation. RRT is suitable for complex robotic arm environments and has fast convergence due to iterative sampling and expansion.

However, it lacks determinism and may struggle to find optimal solutions, especially in narrow passages or complex obstacles. The Probabilistic Roadmap Method (PRM) is a sampling-based technique for robotic motion planning, specifically for industrial robot arm optimal path planning. It involves roadmap construction, path planning, and post-processing. PRM is suitable for handling uncertainty and variability by sampling multiple configurations and connecting them with collision-free paths. However, it can be computationally complex and sensitive to sampling density.

As these algorithm operates on basis of graph generation for path planning of industrial robotic manipulators hence as shown in **Fig.14** more research work is done in Static Environment compared to dynamic environment. Out of total, 90% of Research work is done for path planning in Static Environment and only 10% work is done in case of dynamic environment. After that Analysis is done on research work where 80% of work is only done for simulation analysis whereas 20% of work is done on real-time/experimental analysis as shown in **Fig.15**. Also, some of the researcher have used standalone approach for this path planning of robotic arm but some of them have used hybrid approach so as to form more perfect trajectories. So, in 80% of research work only single approach used whereas 20% work is done on Hybrid approach where more than two algorithm are used for optimizing the trajectories, this is shown in **Fig.16**. This review paper consists of 34 research work of the graph search algorithm , its represented by **Fig.17**.

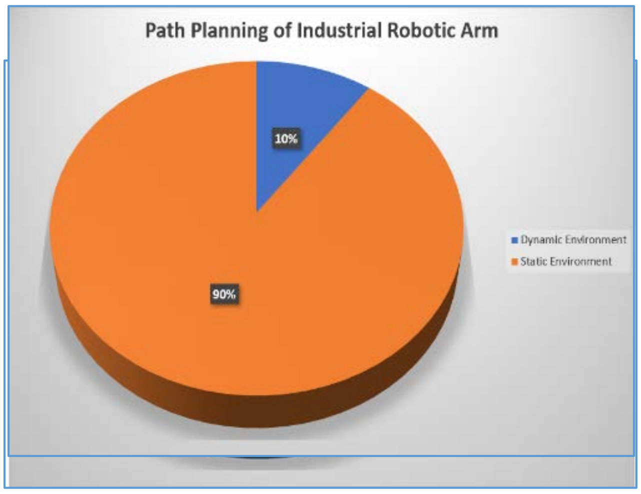


Fig.14 Number of Papers for Static and Dynamic Environment

Fig.15 Number of papers for Realtime Analysis and Simulation Analysis

Table 1 Analysis of Graph Search Algorithm for path planning
Here Y stands for "Yes" and N stands for "No"

Ref No	Path Planning technique	Path planning in presence of: Static Obstacle	Path planning in presence of: Dynamic Obstacle	Realtime Result	Simulation Result	DOF	Standalone method	Hybrids methods	Robots Used	Path Planning	Inverse Kinematics	Forward Kinematics
[4]	A* Algorithm	Y	N	N	N	2	Y	N	Y	Y	Y	N
[5]		Y	N	N	N	3	Y	N	Y	Y	Y	N
[6]		Y	N	N	Y	2	Y	N	Y	Y	Y	N
[7]		Y	N	N	Y	3	Y	N	Y	Y	Y	N
[8]		N	Y	Y	Y	3	Y	Y	Y	Y	Y	N
[9]		Y	N	Y	Y	5	Y	N	Y	Y	Y	N
[10]		Y	N	Y	Y	6	Y	N	Y	Y	Y	N
[11]		Y	N	Y	N	7	Y	N	Y	Y	N	Y
[12]		Y	N	Y	N	5	Y	N	Y	Y	Y	N
[13]		Y	N	Y	Y	5	Y	N	Y	Y	Y	N
[14]		Y	N	Y	Y	3	Y	N	Y	Y	Y	N
[15]	Dijkstra Algorithm	Y	N	N	Y	6	Y	N	Y	Y	Y	N
[16]		Y	N	N	Y	4	Y	N	Y	Y	Y	N
[17]		Y	N	Y	N	2	Y	N	Y	Y	Y	N
[18]		Y	N	Y	N	2	Y	N	Y	Y	N	Y
[19]		Y	N	Y	Y	2	Y	N	Y	Y	N	Y
[20]		Y	N	Y	Y	2	Y	N	Y	Y	Y	N
[21]	RRT Algorithm	Y	N	N	Y	4	Y	N	Y	Y	Y	N
[22]		Y	N	N	Y	2	Y	N	Y	Y	N	Y
[23]		Y	N	N	Y	5	Y	N	Y	Y	Y	N
[24]		Y	Y	N	Y	6	Y	N	Y	Y	Y	N
[25]		Y	N	N	Y	5	Y	N	Y	Y	Y	N
[26]		Y	N	Y	Y	2	Y	Y	Y	Y	Y	N
[27]		Y	N	N	Y	7	Y	Y	Y	Y	N	Y
[28]		Y	Y	Y	Y	6	Y	N	Y	Y	Y	N
[29]		Y	N	Y	Y	6	Y	N	Y	Y	Y	N
[30]	PRM Algorithm	Y	N	N	Y	7	Y	N	Y	Y	Y	N
[31]		Y	N	N	Y	5	Y	N	Y	Y	Y	N
[32]		Y	N	Y	Y	8	Y	Y	Y	Y	Y	N
[33]		Y	N	Y	Y	5	Y	Y	Y	Y	Y	N
[34]		Y	N	Y	Y	2	Y	Y	Y	Y	Y	N

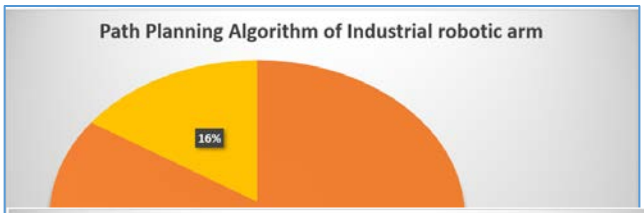
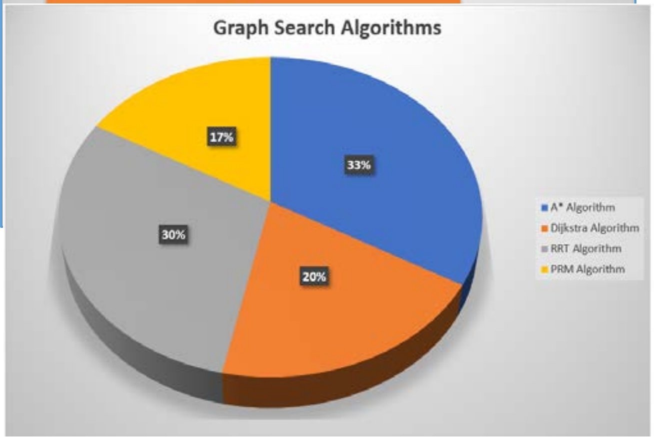


Fig. 16 Number of papers for Standalone Analysis and Hybrid Approach Analysis



The selection of the most suitable algorithm for the path planning of industrial robotic arms relies on several aspects, including the specific requirements of the application, the characteristics of the surroundings, and the compromises between different objectives such as optimality, efficiency, and robustness. If the environment remains constant throughout time and achieving the best possible outcome is of utmost importance, it may be preferable to use either the A* algorithm or Dijkstra's algorithm. Alternatively, if the capacity to plan in real-time and adjust to changing circumstances is crucial, methods such as RRT or PRM may be more appropriate. But also, for path planning in dynamic environment the RRT or PRM's standalone approach may not be suitable, so in that case it needs be integrated with other path planning algorithms so that any objectives of path planning for robotic arm can be achievable.

Fig. 17 Total Number of Paper of Graph Search Algorithm

IV. Conclusion

The prominent role of this review paper is investigation of optimal graph search algorithm for path planning of robotic manipulators. Graph search algorithms play prominent role in the path planning techniques of industrial robotic manipulators. These algorithms are primarily employed for optimizing time, energy, and obstacle avoidance in trajectory planning. A key metric for evaluating the efficiency of path planning algorithms is the length of the generated path, with shorter paths indicating greater efficiency. The review paper discusses four prominent graph search algorithms: A* algorithm, Dijkstra's Algorithm, Rapidly-Exploring Random Trees (RRT), and Probabilistic Roadmap Method (PRM).

- A* algorithm is highlighted for its efficiency in finding optimal paths, while Dijkstra's Algorithm is recognized for its optimality in weighted graphs. RRT is praised for its effectiveness in exploring non-convex, high-dimensional spaces, while PRM is acknowledged for its ability to handle uncertainty and variability in trajectory planning.
- The improved version of A* algorithm is preferred over the classical one due to its modifications and flexibility for specific requirements. Dijkstra's Algorithm is described as suitable for applications requiring optimality and static environments.
- RRT is commended for its suitability in complex robotic arm environments and its fast convergence due to iterative sampling and expansion. However, it may struggle to find optimal solutions in certain scenarios. PRM is recognized for its ability to handle uncertainty and variability through sampling and connection of collision-free paths. Nevertheless, it is noted for its computational complexity and sensitivity to sampling density.
- Research trends reveal a predominant focus on path planning in static environments compared to dynamic environments, with the majority of research conducted using simulation analysis. The review also highlights a preference for standalone approaches in path planning, although hybrid approaches integrating multiple algorithms are gaining attention for optimizing trajectories.
- The selection of the most suitable algorithm for path planning depends on various factors, including the specific requirements of the application, environmental characteristics, and trade-offs between objectives such as optimality, efficiency, and robustness.
- While A* algorithm or Dijkstra's Algorithm may be preferable for static environments and optimality, RRT or PRM algorithms are more suitable for real-time planning and adaptability to changing environment, even though requiring integration with other algorithms for dynamic environments.

V. Reference

[1] Patle, B.K., Chen, S.L., Singh, A. and Kashyap, S.K., 2023. Optimal trajectory planning of the industrial robot using hybrid S-curve-PSO approach. Robotic Intelligence and Automation, 43(2), pp.153-174..

[2] Lai, T.C., Xiao, S.R., Aoyama, H. and Wong, C.C., 2017, September. Path planning and obstacle avoidance approaches for robot arm. In 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE) (pp. 334-337). IEEE.

[3] Sharma, K. and Doriya, R., 2020. Path planning for robots: An elucidating draft. International journal of intelligent robotics and applications, 4, pp.294-307.

[4] Ma, H., 2022. Graph-based multi-robot path finding and planning. Current Robotics Reports, 3(3), pp.77-84.

[5] Yang, L., Li, P., Qian, S., Quan, H., Miao, J., Liu, M., Hu, Y. and Memetimin, E., 2023. Path Planning Technique for Mobile Robots: A Review. Machines, 11(10), p.980.

[6] Zhuang, M., Li, G. and Ding, K., 2023. Obstacle avoidance path planning for apple picking robotic arm incorporating artificial potential field and A* algorithm. IEEE Access.

[7] Xu, Z., Guo, S. and Zhang, L., 2022. A path planning method of 6-DOF robot for mirror therapy based on A* algorithm. Technology and Health Care, 30(1), pp.105-116.

[8] Persson, S.M. and Sharf, I., 2014. Sampling-based A* algorithm for robot path-planning. The International Journal of Robotics Research, 33(13), pp.1683-1708.

[9] Islam, F., Narayanan, V. and Likhachev, M., 2016, May. A-Connect: Bounded suboptimal bidirectional heuristic search. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2752-2758). IEEE.

[10] Li, F., Huang, Z. and Xu, L., 2019, December. Path planning of 6-DOF venipuncture robot arm based on improved a-star and collision detection algorithms. In 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 2971-2976). IEEE.

[11] Li, X., He, Q., Yang, Q., Wang, N., Wu, H. and Yang, X., 2022. Research on an Optimal Path Planning Method Based on A* Algorithm for Recognition. Algorithms, 15(5), p.171

[12] Fu, B., Chen, L., Zhou, Y., Zheng, D., Wei, Z., Dai, J. and Pan, H., 2018. An improved A* algorithm for the industrial robot path planning with high success rate and short length. Robotics and Autonomous Systems, 106, pp.26-37.

[13] Mi, K., Zheng, J., Wang, Y. and Hu, J., 2019. A multi-heuristic A* algorithm based on stagnation detection for path planning of manipulators in cluttered environments. IEEE Access, 7, pp.135870-135881.

[14] Fu, B., Chen, L., Zhou, Y., Zheng, D., Wei, Z., Dai, J. and Pan, H., 2018. An improved A* algorithm for the industrial robot path planning with high success rate and short length. Robotics and Autonomous Systems, 106, pp.26-37.

[15] Zhang, X., Yang, F., Jin, Q., Lou, P. and Hu, J., 2023. Path Planning Algorithm for Dual-Arm Robot Based on Depth Deterministic Gradient Strategy Algorithm. Mathematics, 11(20), p.4392.

[16] Zhou, X., Yan, J., Yan, M., Mao, K., Yang, R. and Liu, W., 2023. Path Planning of Rail-Mounted Logistics Robots Based on the Improved Dijkstra Algorithm. Applied Sciences, 13(17), p.9955.

[17] Muthuswamy, S. and Manoochehri, S., 1992. Optimal path planning for robot manipulators.

[18] Dirik, M. and KOCAMAZ, F., 2020. Rrt-dijkstra: An improved path planning algorithm for mobile robots. Journal of Soft Computing and Artificial Intelligence, 1(2), pp.69-77.

[19] Fusic, S.J., Ramkumar, P. and Hariharan, K., 2018, March. Path planning of robot using modified dijkstra Algorithm. In 2018 National Power Engineering Conference (NPEC) (pp. 1-5). IEEE.

[20] Zhang, C. and Wang, C., 2018, July. Service robot path planning based on improved Dijkstra algorithm. In Proceedings of the 12th International Convention on Rehabilitation Engineering and Assistive Technology (pp. 77-80).

[21] LaValle, S.M., Kuffner, J.J. and Donald, B.R., 2001. Rapidly exploring random trees: Progress and prospects. Algorithmic and computational robotics: new directions, 5, pp.293-308.

[22] Lai, T.C., Xiao, S.R., Aoyama, H. and Wong, C.C., 2017, September. Path planning and obstacle avoidance approaches for robot arm. In 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE) (pp. 334-337). IEEE.

[23] Zhang, Z., Wu, D., Gu, J. and Li, F., 2019. A path-planning strategy for unmanned surface vehicles based on an adaptive hybrid dynamic step size and target attractive force-RRT algorithm. Journal of Marine Science and Engineering, 7(5), p.132.

[24] Zhang, H., Wang, Y., Zheng, J. and Yu, J., 2018. Path planning of industrial robot based on improved RRT algorithm in complex environments. IEEE Access, 6, pp.53296-53306

[25] Lai, T.C., Xiao, S.R., Aoyama, H. and Wong, C.C., 2017, September. Path planning and obstacle avoidance approaches for robot arm. In 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE) (pp. 334-337). IEEE.

[26] Noreen, I., Khan, A. and Habib, Z., 2016. Optimal path planning using RRT* based approaches: a survey and future directions. International Journal of Advanced Computer Science and Applications, 7(11).

[27] Kim, Y.J., Wang, J.H., Park, S.Y., Lee, J.Y., Kim, J.J. and Lee, J.J., 2014, August. A RRT-based collision-free and occlusion free path planning method for a 7DOF manipulator. In 2014 IEEE International Conference on Mechatronics and Automation (pp. 1017-1021). IEEE.

[28] Liang, Y., Mu, H., Chen, D., Wei, X. and Wang, M., 2020, October. PR-RRT: Motion Planning of 6-DOF Robotic Arm Based on Improved RRT Algorithm. In 2020 10th Institute of Electrical and Electronics Engineers International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER) (pp. 417-422). IEEE.

[29] Friedmann, C.V., Collision-Free Path Planning Using Sampling-Based Algorithms for a 7DOF Robot Arm.

[30] Zafar, M.N. and Mohanta, J.C., 2018. Methodology for path planning and optimization of mobile robots: A review. Procedia computer science, 133, pp.141-152.

[31] Tamizi, M.G., Yaghoubi, M. and Najjaran, H., 2023. A review of recent trend in motion planning of industrial robots. International Journal of Intelligent Robotics and Applications, pp.1-22.

[32] Chen, G., Luo, N., Liu, D., Zhao, Z. and Liang, C., 2021. Path planning for manipulators based on an improved probabilistic roadmap method. Robotics and Computer Integrated Manufacturing, 72, p.102196.

[33] Akbaripour, H. and Masehian, E., 2017. Semi-lazy probabilistic roadmap: a parameter-tuned, resilient and robust path planning method for manipulator robots. The International Journal of Technology, 89, pp.1401-1430

[34] Iqbal, Z., Reis, J. and Gonçalves, G., 2019, June. Path Planning for an Industrial Robotic Arm. In Proceedings of the Eighth International Conference on Intelligent Systems and Applications (INTELLI), Rome, Italy (p. 39).