Motion Planning Algorithms in Robotics

Author Name (Martin Kobík)

Faculty of Mechanical Engineering, Brno University of Technology
Institute of Automation and Computer Science
Technicka 2896/2, Brno 616 69, Czech Republic
228709@vutbr.cz

Abstract: *This paper is concerned with the analysis of the most widely used motion planning algorithms in robotics. Each algorithm is described with its advantages and drawbacks. A practical example for robotic motion planning is provided as well, usually incorporating a variation of the basic algorithm.*

Keywords: *motion planning robotics RRT RRT* A* PRM D**

1 Introduction

Motion planning plays a crucial part in autonomous robot navigation. It is the process of finding an effective path from the start to the goal, avoiding obstacles, and considering constraints. Robots can map their surroundings, construct a map and use various algorithms to find an optimal path.

2 Rapidly exploring random tree (RRT)

RRT is a sampling-based algorithm used in motion planning. It searches space by randomly“ building up a tree structure. Though this algorithm can be used to search multidimensional state spaces, it will be explained in an example with a 2D space and a goal of finding an effective path while avoiding obstacles. Obstacles in state space would be considered undesirable states which the robot is permitted to enter. RRT is explained in the following example: Find a path from start to goal, while avoiding obstacles in a 2D environment.

Firstly, generate a random point in space. If the point is close to the start, the point becomes a node and is connected to the start node. Otherwise, create a node at some predefined max distance from the start in the direction of the random point and connect them. If the line between points intersects an obstacle, discard the

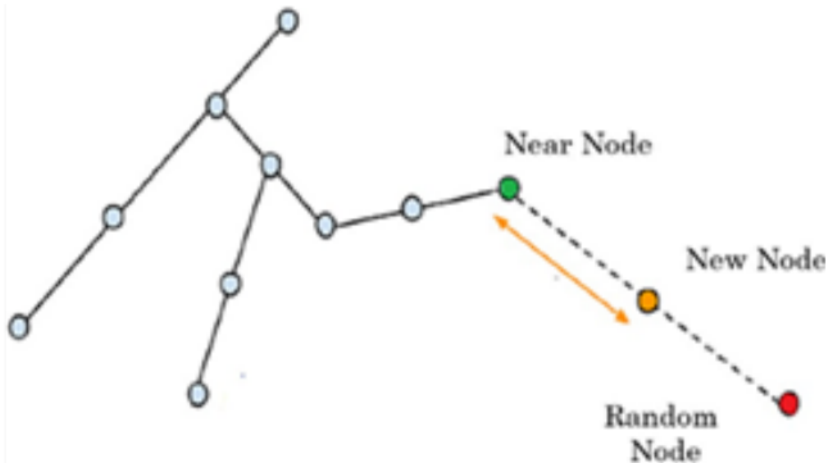

Figure 1: RRT algorithm [8].

node, create another random point and check for intersection again. Next, create another random point and its corresponding node by connecting it to the closest node according to the previously stated rules. Iterate this process until some node is close to the goal, or the predefined max node count is reached. Then just backtrack to the start to get the final path. [6]

Based on the algorithm, the randomly generated points will be likely generated far from the start at the beginning, so the nodes will be placed at the max distance. The tree will therefore expand quicky at the start, exploring large empty parts of the space. Major drawback of this algorithm is that when a path is found, adding more nodes hardly improves it.

2.1 Urban driving of Talos using RRT

RRT has been used in MIT's robot Talos for autonomous path planning in a dynamic environment, avoiding obstacles and abiding traffic rules. The robot calculated the path and speed while satisfying the requirements. The calculations had to be fast to account for the dynamic environment. While the algorithm was modified to reach better results, the main structure was RRT. [5]

3 Rapidly exploring random tree* (RRT*)

RRT* is very similar to the RRT with the difference of node connection. When a new node is generated, it is not connected to the nearest node. Instead, the algorithm checks within some predefined distance around the new node and connect is to a node that minimizes the path length back to the start. Then other nodes in that area are reconnected to the new node if it minimizes the distance. [8]

RRT* is superior to the RRT in that the path converges to the optimal, shortest path as the number of nodes gets close to infinity. This is of course at the cost of higher computational complexity. [8]

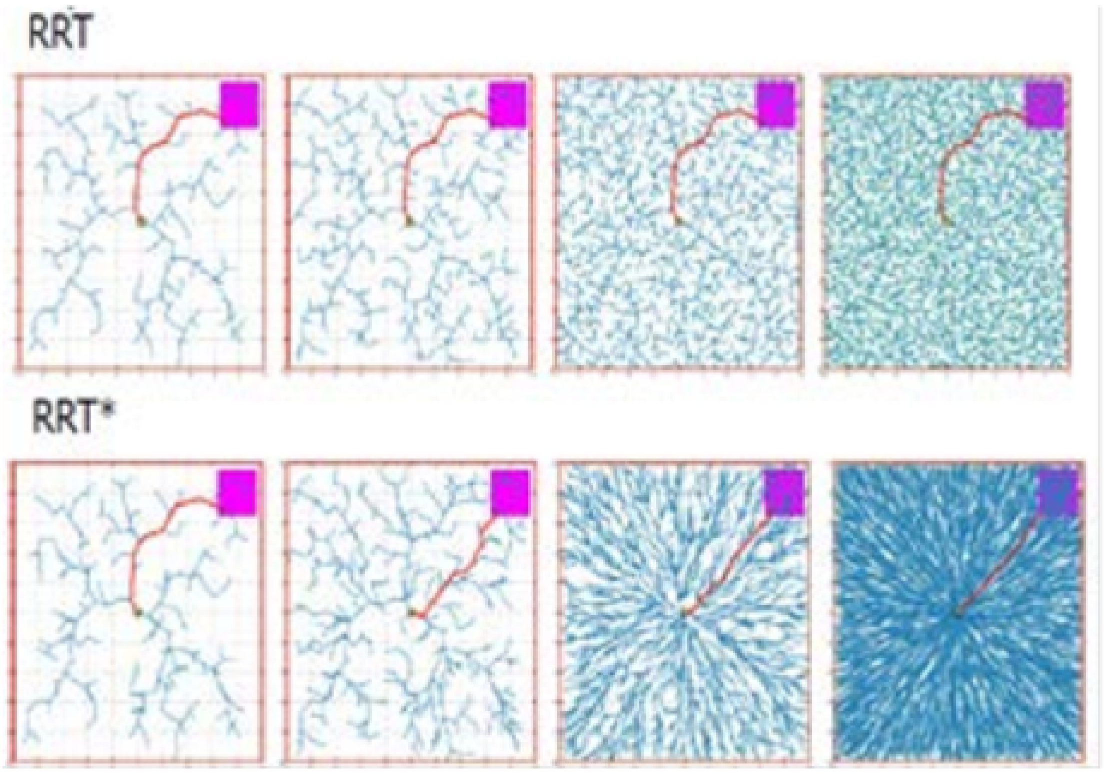

Figure 2: RRT vs RRT* [8]

3.1 Segway robot RMP 220 motion planning using risk RRT*

The RMP 220 robot was tasked with navigating through static and dynamic environment. In the dynamic environment, there were 2 pedestrians walking back and forth, which the robot had to avoid. It used risk RRT* algorithm, a variation of the RRT*, which considers a risk estimation of colliding with dynamic obstacles. The risk RRT* was compared to the risk RRT algorithm in both environments. Conclusion being that the latter had improved motion planning. [1]

4 A*

A* algorithm is a search-based algorithm used for finding the optimal path from start to goal. The space is divided into a grid of nodes. Initially, nodes are generated around the starting node, interconnected to it and have values assigned based on their total distance to the start node. Then another node is chosen based on a cost function, which considers the distance from the start node and the estimated distance to the goal node. This also takes into account obstacles. The chosen node has other nodes created around it. The process is iterated until a path to the goal is found. [10]

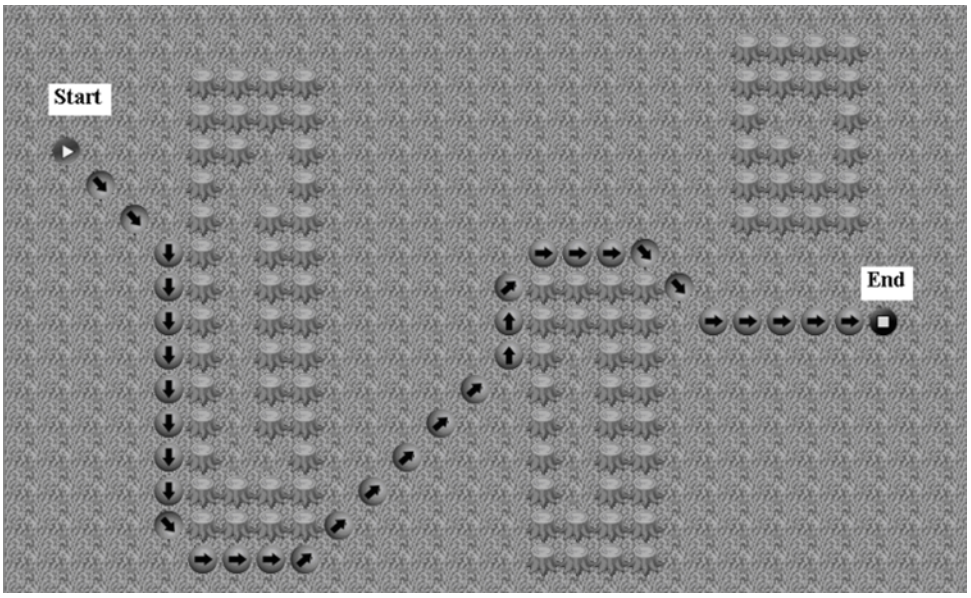

Figure 3: A* algorithm [10]

4.1 Robot manipulator Motoman UP6 with A*

This 6 DOF robot arm was tasked with moving a workpiece. The experimental setup involved 2 obstacles and 3 storage units. The manipulator used an A* algorithm and compared the results with an improved A*, which checks if the theoretical path from current node to the goal is obstacle free and prioritizes it. The improved A* algorithm yielded better results in terms of shorter path length, efficiency and higher success rate. [3]

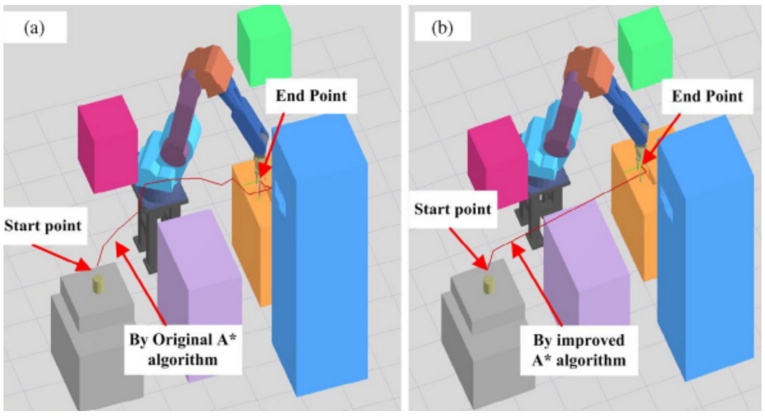

Figure 4: Motoman UP6 using A* [3]

5 Probabilistic road map (PRM)

PRM has 2 computational phases: preprocessing and query. In the preprocessing phase, a roadmap is constructed with points representing random configurations of the robot. These points are connected to each other by a planner and form a graph. In the query phase, we can choose 2 points in the graph and perform a graph search to find a path between them. [4]

5.1 Rear wheel drive robot motion planning using PRM and a fuzzy control system

This robot uses sensors to track obstacles. It generates a path to the goal with PRM and then smoothens the sharp corners with a fuzzy control system by adjusting the wheel velocities for smoother turning angles. This approach is shown to shorten the path and reduce time. [7]

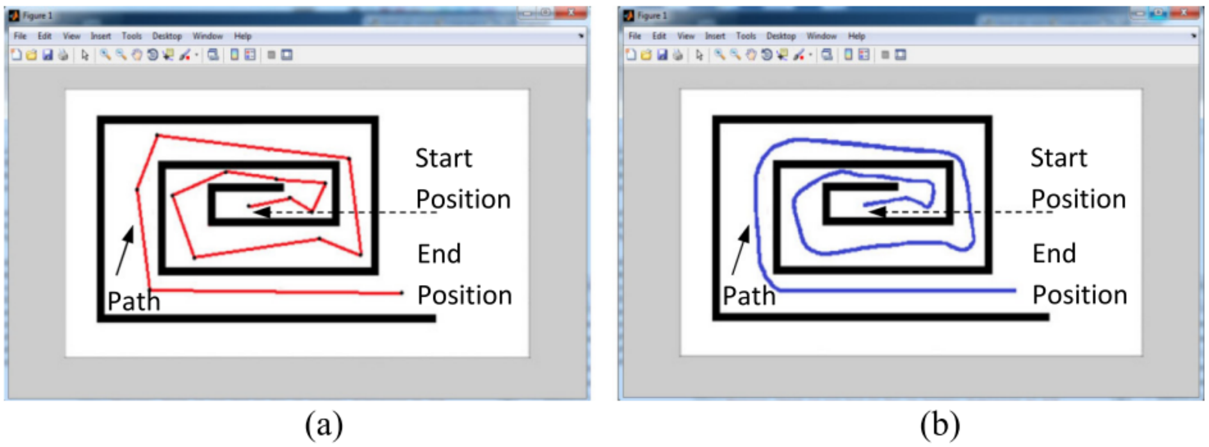

Figure 5: (a) Robot path using PRM, (b) Robot path using Fuzzy-RPM [7]

6 D*

D* algorithm focuses on motion planning in a dynamic environment. It is similar to A*, but it changes the cost as the environment changes and stores the data along with one step look ahead cost. As the cost of nodes changes, the algorithm replans the path based on the stored data, updating the newly changed parts of the space instead of recalculating the entire path. [2]

6.1 2 DOF robotic arm using D*

The arm integrates D* algorithm with ant colony algorithm which imitates the behavior of ants. Virtual ants are created and lead to explore the space, leaving pheromones on the edges, prioritizing edges with higher pheromones and lower cost. The optimal path is then evaluated based on the pheromones and the shortest distance to the goal provided by the D* algorithm. [9]

7 Conclusion

This paper provides an overview of some of the most widely used algorithms in motion planning including RRT, RRT*, A*, PRM, D*. Each algorithm is explained and showcased through a practical example in a robotics application. Variations of the algorithms are featured as well. Each algorithm has advantages and disadvantages over the others. The optimal choice of an algorithm depends on the application, environment and available computational power.

References

[1] Chi, W., and Meng, M. Q.-H. Risk-rrt*: A robot motion planning algorithm for the human robot coexisting environment. In *Proceedings of the 2017 18th International Conference on Advanced Robotics (ICAR)* (Hong Kong, China, July 2017).

[2] Ferguson, D., and Stentz, A. The delayed d* algorithm for efficient path replanning. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (Barcelona, Spain, April 2005). Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, {dif, tony}@cmu.edu.

[3] Fu, B., Chen, L., Zhou, Y., Zheng, D., Wei, Z., Dai, J., and Pan, H. An improved a* algorithm for industrial robot path planning with high success rate and short length. *Journal of Robotics* (2022). Guangxi Colleges and Universities Key Laboratory of Modern Design and Advanced Manufacturing, Guangxi University, Nanning 530004, China.

[4] Kavraki, L. E., and Latombe, J.-C. Probabilistic roadmaps for robot path planning. *International Journal of Robotics Research* (1996). Department of Computer Science, Rice University, Houston, TX 77005 and Department of Computer Science, Stanford University, Stanford, CA 94305.

[5] Kuwata, Y., Fiore, G. A., Teo, J., Frazzoli, E., and How, J. P. Motion planning for urban driving using rrt. In *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Nice, France, September 22-26 2008).

[6] Melchior, N. A., and Simmons, R. Particle rrt for path planning with uncertainty. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation* (Roma, Italy, April 10-14 2007), IEEE.

[7] Mohanta, J., and Keshari, A. A knowledge-based fuzzy-probabilistic roadmap method for mobile robot navigation. *Journal of Intelligent and Robotic Systems* (Year of Publication). Department of Mechanical Engineering, Motilal Nehru National Institute of Technology, Allahabad, India.

[8] Noreen, I., Khan, A., and Habib, Z. Optimal path planning using rrt* based approaches: A survey and future directions. *International Journal of Advanced Computer Science and Applications (IJACSA) 7*, 11 (2016), Not specified.

[9] Sadiq, A. T., Raheem, F. A., and Abbas, N. A. F. Ant colony algorithm improvement for robot arm path planning optimization based on d* strategy. *International Journal of Mechanical & Mechatronics Engineering (IJMME) 21*, 01 (2021).

[10] Yao, J., Lin, C., Xie, X., Wang, A. J., and Hung, C.-C. Path planning for virtual human motion using improved a* algorithm. In *Proceedings of the 2010 Seventh International Conference on Information Technology* (Xiamen, Fujian Province, China, 2010). School of Software, Xiamen University, Xiamen, Fujian Province, China, 361005 and School of Computing and Software Engineering, Southern Polytechnic State University, Marietta, GA, 30060, U.S.A.