Proceedings of CONF-APMM 2026 Symposium: Applied Mathematics and Quantitative Modeling
DOI: 10.54254/2753-8818/2026.PU35842

Robot Path Planning in Complex Environments: Methods, Challenges and Future Directions

Chanyu Wang

Chongqing No.1 Experimental Middle School, Chongqing, China
wangchanyu87@gmail.com

Abstract. Robot path planning is a fundamental problem in robotics enabling autonomous robots to navigate safely, efficiently and naturally from a start position to a target position. In real robotic systems, path planning is not only about finding a collision-free path, but also about generating motions that satisfy the robot's physical, sensory, and task constraints. In this essay, the main robot path planning methods, including graph search, artificial potential field methods, sampling-based planning, local obstacle avoidance, and trajectory optimization are reviewed. Their respective strengths, limitations, and applicable scenarios are examined, with particular attention to how these approaches address issues such as computational complexity, environmental structure, and real-time responsiveness. It also addresses present day issues such as uncertainty, dynamic environments, computational efficiency and physical feasibility, emphasizing that no single method is sufficient for all robotic applications. Practical systems increasingly rely on hierarchical integration to balance global navigation with local reactivity. Finally, the essay argues that the future of robot path planning will be dominated by hybrid systems that combine global planning, local replanning, optimization, and learning-based prediction, enabling robots to operate more safely, intelligently, and adaptively in complex real-world environments.

Keywords: Robot path planning, graph-search algorithms, obstacle avoidance, trajectory optimization

1. Introduction

Robot path planning is a fundamental problem in robotics, offering a method for autonomous robots to move safely and efficiently from a start point to a goal position. In a real robotic system, path planning means not only finding a route, but also producing motions that can be actually executed by the robot. A good path is collision-free, relatively short, smooth, consistent with dynamic constraints and adaptable to environmental changes. Latombe defines robot motion planning as the problem of finding the motions that a robot should perform so that it can reach a target state, which indicates that planning is closely related to robot action and control [1].

Graph search algorithms have been a strong influence on early path planning research. Dijkstra's algorithm is a classical algorithm for finding the shortest path in a weighted graph [2]. Later, the A* algorithm used the heuristic information to guide the search to the target, therewith the search efficiency was improved [3]. These algorithms are still very important for mobile robot navigation

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2753-8818/2026.PU35842

especially when the environment can be represented as a graph, a grid map or an occupancy grid. But real robots usually operate in continuous, uncertain, and unstable environments so classical search methods are not always enough alone.

Khatib has suggested the artificial potential field method for real-time robot motion. In this method, the target is attractive to the robot while the obstacles are repulsive [4]. This approach is suitable for rapid local avoidance of obstacles, but it can be affected by local minima and can produce unstable motion near obstacles. Sampling-based planning has emerged as a major direction for high-dimensional robotic systems, in particular, robotic arms. The Probabilistic Roadmap Method was first proposed by Kavraki et al., that samples collision-free configurations and connects them into a roadmap [5]. LaValle and Kuffner proposed Rapidly-exploring Random Trees that can successfully explore large continuous spaces and can be applied to systems with motion constraints [6]. Karaman and Frazzoli later proposed asymptotically optimal sampling-based algorithms, such as RRT*, that can improve the quality of the path as the number of samples increases [7].

In practical robotics, path planning should also consider the dynamic properties of the robot. A path that is geometrically safe may not be executable if it contains abrupt turns, jerk or unrealistic velocity change. Fox, Burgard and Thrun proposed a Dynamic Window Approach which searches in the robot's velocity space and directly takes into account dynamic constraints during local obstacle avoidance [8]. This idea is of utmost importance, since real robots need to convert planned paths into executable motion commands.

Robot path planning has developed from classical graph search to a wide research area, including heuristic search, local obstacle avoidance, sampling based planning and dynamic motion planning. Modern robotic systems generally utilize a combination of global planners and local planners. Global planners give the overall routes, while local planners use sensor information to adjust movements in real time. This paper reviews the main robot path-planning methods, their advantages and limitations and discusses current challenges including uncertainty, changing obstacles, computational effectiveness and real-life feasibility.

2. Current progresses in robot path planning

2.1. Overview of robot path planning methods

The methods for robot path planning can be mainly classified into several categories, such as graph-search methods, artificial potential field methods, sampling-based methods, local obstacle avoidance methods and trajectory optimization methods. Each method has its strengths and weaknesses and different applications of robotics often require different planning strategies. In practice, a robot path-planning system often employs more than one algorithm. Instead, many real robots generate safe and executable motions by combining global planning, local replanning and motion control. The details of these methods are as follows.

2.2. Graph-search methods

Among the most classical methods for robot path planning are graph-search methods. These methods model the environment or state space of the robot as a graph of nodes connected by edges. The planner then searches for a path from the start node to the target node. Dijkstra's algorithm is one of the first and most important graph-search algorithms. It can find the shortest path in a weighted graph when all edges have non-negative costs [2]. Dijkstra's algorithm, however, can explore a very large number of nodes, especially if the search space is large. The A* algorithm

DOI: 10.54254/2753-8818/2026.PU35842

enhances this process by employing a heuristic function to estimate the cost remaining from the current node to the target [3]. If the heuristic function is well designed, this heuristic guidance makes A* generally more efficient in searching than Dijkstra's algorithm. Graph-search methods are especially useful for mobile robot navigation in environments that can be represented by a grid map, a topological map, or an occupancy grid. The advantage of this is mainly that they are reliable. Their performance depends a lot on the quality and resolution of the map. If the map is too coarse, it may produce unsafe or inaccurate paths. But if the map is too detailed, it may require an elevated computational cost.

2.3. Replanning methods for dynamic environments

Replanning is critical, especially for robots operating in uncertain or partially known environments. The environment can also change during operation, and the robot may not know all obstacles before it starts to move. The D* algorithm was introduced by Stentz to allow robots to efficiently update their paths when new information about the environment is discovered [9]. The idea was later simplified by D* Lite and became an incremental search method for robot navigation in unknown terrain [10]. These algorithms are very important, because real robots often have to make decisions with incomplete information. For example an indoor service robot can be confronted with a corridor that is temporarily blocked or with an object that is not part of the original map. Replanning methods allow the robot to replan only the affected part of the path due to the disturbance without replanning the whole path from scratch, thus saving computation time.

2.4. Artificial potential field method

The artificial potential field method is another option for robot path planning. In this approach the target is considered as attraction source and the obstacles as repulsion sources [4]. The robot moves to the combination of these virtual forces. This method is simple and computationally efficient, therefore, it is suitable for real-time local obstacle avoidance. Unexpected obstacles that suddenly appear nearby can help the robot to react quickly. But the artificial potential field method has obvious disadvantages. The robot might end up in a local minimum where the attractive and repulsive forces balance each other, and the robot is unable to continue moving towards the target. The path may also be unstable or oscillatory in the vicinity of obstacles. Thus, the potential field method is typically employed as a local planning tool rather than as a complete planning solution.

2.5. Sampling-based methods

Sampling-based methods are another important group of techniques for robot path planning. These methods are particularly suited for high-dimensional systems, e.g. robotic arms with many joints. Sampling-based planners do not discretize the whole state space. Instead, they randomly sample possible robot configurations and connect feasible configurations together. The Probabilistic Roadmap Method constructs a roadmap by sampling collision-free configurations and connecting them with local planners [5]. This approach is useful when robots have to plan several paths in the same environment. Rapidly-exploring Random Trees (RRT) construct a tree rooted at an initial state by extending the tree iteratively towards randomly sampled states [6]. The advantage of RRT is that it can explore large continuous spaces quickly and handle systems with motion constraints. But the original RRT does not guarantee an optimal trajectory. The RRT* gradually improves the tree structure and converges to the optimal solution as the number of samples increases [7]. Sampling-

DOI: 10.54254/2753-8818/2026.PU35842

based methods are popular in robotic manipulation, autonomous vehicles, and aerial robots because they can handle complex configuration spaces better than simple graph-search methods.

2.6. Local obstacle avoidance methods

When robots operate inside dynamic environments local obstacle avoidance methods are needed. The global path might be planned prior to the robot's movement. However, the presence of moving people, other robots or unexpected objects can make the original path unsafe. The Dynamic Window Approach (DWA) selects safe velocity commands based on the robot's current velocity, acceleration limits, and obstacles in the neighborhood [8]. This approach is very convenient since it operates directly in the velocity space and produces commands that can be effectively implemented by the robot. Velocity Obstacle methods are also defined in velocity space and can be used to avoid moving obstacles [11]. They predict speeds that could cause future collisions and then choose safer alternative speeds. In multi-robot systems, reciprocal collision avoidance methods distribute the responsibility for collision avoidance among the robots [12]. Such local methods are important in warehouses, hospitals, airports, and other public indoor spaces, where robots need to safely navigate around people and other machines.

2.7. Trajectory optimization methods

Trajectory optimization methods are not only about finding a feasible path but also about improving the quality of robot movement. Even if a path is free of obstacles, it may still not be good for execution if it has sharp turns, sudden acceleration, or if it goes beyond the physical limits of the robot. Covariant Hamiltonian Optimization for Motion Planning (CHOMP) optimizes trajectories in terms of smoothness and obstacle costs [13]. Stochastic Trajectory Optimization for Motion Planning (STOMP) uses stochastic optimization to improve trajectories, even if the cost function is hard to differentiate [14]. TrajOpt generates collision-free and smooth trajectories using sequential convex optimization [15]. These methods are very useful for robotic arms, drones and legged robots, because these systems require high quality of motion and dynamic feasibility. Many practical systems use graph search or sampling based planning to generate an initial path and then optimization to create a smoother and more executable trajectory.

2.8. Hierarchical integration and method comparison

In general, different robot path-planning methods solve different parts of the problem. Graph-search methods can be utilized on structured maps, replanning methods allow robots to incorporate new information, potential field methods allow fast local reaction , sampling based methods deal with high dimensional space, local obstacle avoidance methods deal with dynamic obstacles, and trajectory optimization improves the smoothness and feasibility of movement. Powerful robot path-planning systems usually combine various methods. This hierarchical approach enables the robot to perform global planning and local response but also realistic movements that fit the physical structure of the robot.

DOI: 10.54254/2753-8818/2026.PU35842

Table 1. Method comparison

Method category	Example	Main advantages	Main limitations	Suitable conditions
Graph-search methods	Dijkstra , A*	Reliable, easy to understand, and capable of finding an optimal or near-optimal path	Highly dependent on map resolution and may require substantial computation in large environments	Robots operating in known and relatively structured environments
Replanning methods	D* , D*Lite	Can efficiently update an existing path when new environmental information is discovered	More complex to implement and may still require considerable computation in highly dynamic environments	Robots operating in partially known or changing environments
Artificial potential methods	APF	Fast and suitable for real-time local reactions	May become trapped in local minima and produce unstable movements near obstacles	Robots requiring rapid local obstacle avoidance
Sampling-based methods	PRM ,RRT,RRT*	Effective in high-dimensional and complex configuration spaces	Results may be inconsistent because of random sampling, and the generated path may be irregular	High-dimensional robotic systems, particularly robotic arms
Local obstacle avoidance methods	DWA,VFH	Can respond quickly to moving or newly detected obstacles	Usually focus on short-term decisions and may not guarantee a globally efficient path	Robots operating in dynamic environments with moving obstacles

It can be seen from Table 1 that the choice of method depends on the robot type, the complexity of the environment, the real-time response requirements and the path quality requirements. In real robotic systems it is more effective to combine multiple methods rather than relying on a single planning algorithm.

3. Current challenges and future directions

3.1. Perception and location uncertainty

Robot path planning has made great strides, but real robotic systems face many challenges. The big problem is the uncertainty. Robots rely on cameras, LiDAR, radar, ultrasonic sensors and wheel encoders to sense their environment. However, the sensor data may be noisy, incomplete, or delayed. Robots may find it hard to detect transparent objects, reflective surfaces or small obstacles. Their positioning systems can also be wrong, so the robot may not know its own location.

So a road that looks safe on a map might not be perfectly safe in reality. To address this issue, future path-planning systems need to combine perception, localization and planning more closely. Robots should not rely on fixed maps only, but should keep updating their paths based on new sensor information.

3.2. Path planning in dynamic environments

The constantly changing character of the real world environments poses another major challenge. Many robots will be expected to operate in environments where people, vehicles or other robots are constantly on the move. In these settings, robots have to anticipate the likely movement of objects around them soon enough.

DOI: 10.54254/2753-8818/2026.PU35842

This makes planning a path more difficult, because robots have to avoid not only existing obstacles, but also possible future collisions. For example, hospital delivery robots must safely navigate around patients, nurses and medical equipment. Warehouse robots need to collaborate with other robots and human workers. In these cases motion planning, prediction and decision-making are combined to create a path plan.

3.3. Balance between optimality and real-time performance

Another challenge is to balance the optimality of the path and real-time performance. Some algorithms can yield high quality or near optimal paths, but involve a considerable amount of computation time. Other algorithms are fast, but the paths they produce may be less efficient or less smooth.

In practice, the shortest mathematically path is not always the best for a robot. It should be also safe, smooth, comfortable and generated quickly. That's why hierarchical planning systems are becoming more important. A global planner can provide the entire path. A local planner reacts fast to the local changes of the environment. Trajectory optimization can then further improve the final path for safety and execution.

3.4. Hybrid planning frameworks

In the future, it is possible to see more hybrid planning frameworks. Alternatively, robots may not rely on a single algorithm but may combine graph search, sampling-based planning, local obstacle avoidance, and trajectory optimization. Such combinations can make planning systems more reliable and flexible.

Hybrid systems can also combine global route planning with reactive local responses. For instance, a graph-search algorithm can generate the initial route and a local obstacle avoidance strategy can adapt the robot's movement to unforeseen obstacles. Then, a trajectory optimization method can be used to improve the smoothness and physical feasibility of the final motion.

3.5. Integration of learning-based methods

Learning-based methods will also probably play a greater role in the path planning of robots. Machine learning can enable robots to anticipate human motion, evaluate risks and adapt their planning strategies to different environments.

But it is not required to replace classical planning algorithms by learning-based approaches. Classical methods offer structure, interpretability, and improved safety guarantees. Learning-based methods can enhance prediction and adaptation. A promising direction for future work is therefore the combination of reliability of traditional planning methods and flexibility of learning based models.

3.6. Multi-robot path planning

Another important direction for future research is multi-robot path planning. In warehouses, factories, delivery systems and smart cities many robots may need to share the same environment. These robots have to coordinate their paths so they don't collide or get congested.

In future planning approaches, the efficiency of individual robots should be considered, as well as the performance of the whole robotic system. This may require communication, task allocation, shared decision making, and coordinated motion planning among multiple robots.

DOI: 10.54254/2753-8818/2026.PU35842

3.7. Safety and explainability

And finally: safety and explainability will be more important than ever. As robots are more and more included in the daily life, the decisions taken during their path-planning must be understandable, predictable and safe.

A future path-planning system should be robust to uncertainty, efficient in real time and capable to adapt to complex human-centered environments to be successful. It also has to be able to explain why a certain path or action is chosen, especially for safety-critical applications such as hospitals, autonomous transport and public service robotics.

4. Conclusion

Robot path planning is an important ability of autonomous robots. Classical graph-search methods such as Dijkstra and A* provide a good foundation, while D* and D* Lite allow robots to replan in changing environments. The artificial potential field method provides fast local response and the sampling based techniques such as PRM, RRT and RRT* are suitable for high-dimensional robotic systems. Local obstacle avoidance techniques enable robots to operate in dynamic environments, while trajectory optimization enhances the smoothness and physical feasibility of their motions.

There is not one-size-fits-all approach for robotic applications. In practice a robot path-planning system will combine several methods depending on the robot's task, its environment and its physical constraints. Future robot path planning will probably rely on hybrid systems that combine global planning, local replanning, trajectory optimization and learning-based prediction. Such systems will allow robots to work more safely, efficiently and intelligently in complex real-world environments.

References

[1] Latombe, J.-C. (1991). *Robot motion planning*. Kluwer Academic Publishers.
[2] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.
[3] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics, 4*(2), 100–107.
[4] Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research, 5*(1), 90–98.
[5] Kavraki, L. E., Švestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation, 12*(4), 566–580.
[6] LaValle, S. M., & Kuffner, J. J. (2001). Rapidly-exploring random trees: Progress and prospects. In B. R. Donald, K. M. Lynch, & D. Rus (Eds.), *Algorithmic and computational robotics: New directions* (pp. 293–308). A K Peters.
[7] Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research, 30*(7), 846–894.
[8] Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine, 4*(1), 23–33.
[9] Stentz, A. (1994). Optimal and efficient path planning for partially known environments. In *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 3310–3317). IEEE.
[10] Koenig, S., & Likhachev, M. (2002). D* Lite. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence* (pp. 476–483). AAAI Press.
[11] Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research, 17*(7), 760–772.
[12] van den Berg, J., Guy, S. J., Lin, M., & Manocha, D. (2011). Reciprocal n-body collision avoidance. In C. Pradalier, R. Siegwart, & G. Hirzinger (Eds.), *Robotics research: The 14th International Symposium ISRR* (pp. 3–19). Springer.
[13] Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., & Srinivasa, S. S. (2013). CHOMP: Covariant Hamiltonian optimization for motion planning. *The International Journal of*

DOI: 10.54254/2753-8818/2026.PU35842

Robotics Research, 32(9–10), 1164–1193.

[14] Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., & Schaal, S. (2011). STOMP: Stochastic trajectory optimization for motion planning. In *2011 IEEE International Conference on Robotics and Automation* (pp. 4569–4574). IEEE.

[15] Schulman, J., Ho, J., Lee, A., Awwal, I., Bradlow, H., & Abbeel, P. (2014). Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research, 33*(9), 1251–1270.