

Comparative Analysis of Path Planning Algorithms and Prospects for Practical Application

Yihan Ke*

College of Letters & Science, University of Wisconsin-Madison, 702 West Johnson Street, Suite 1101.

* Corresponding Author Email: yke23@wisc.edu

Abstract. Path planning, as an important applications of artificial intelligence technology in the field of intelligent transportation, has a wide range of application prospects in achieving efficient, intelligent, and safe travel. This paper first introduces the importance of path planning technology in the field of transportation, and then discusses the development of path planning from two aspects: traditional methods and new methods based on artificial intelligence. In terms of traditional methods, the principles and advantages and disadvantages of methods such as Dijkstra's algorithm, A* algorithm, and Floyd's algorithm are included. In terms of new methods based on artificial intelligence, the principles and application examples of methods such as genetic algorithms, simulated annealing algorithms, and deep reinforcement learning are included. Finally, the problems existing in path planning field are analyzed, and the future research directions are prospected in order to provide useful references for the development of intelligent transportation.

Keywords: Route planning, artificial intelligence, transportation, traditional methods, new methods.

1. Introduction

Traffic congestion has become an increasingly pressing issue around the world, leading to significant economic and environmental consequences, as well as negatively impacting the quality of life for urban dwellers. As urbanization continues to rise and cities become more densely populated, there is a growing demand for efficient and intelligent transportation systems. In this context, path planning has become a critical need for transportation systems, facilitating optimal route selection and reducing travel time for vehicles [1].

With the growing availability and advancements in artificial intelligence (AI) technology, deep learning, and reinforcement learning algorithms, its application in the field of path planning holds promising potential to improve traffic efficiency, reduce travel time, and optimize resource utilization. AI-based path planning can contribute to more effective decision-making, adaptability to dynamic environments, and ultimately enhance the safety and efficiency of transportation systems [2].

This paper aims to explore the development and application of AI in path planning, specifically in the context of autonomous vehicles, and provide an outlook on future directions. The integration of AI in path planning for autonomous vehicles is a crucial step towards realizing the full potential of self-driving cars in revolutionizing the transportation landscape.

The paper is divided into four parts: basic concepts of path planning, traditional path planning methods, AI applications in path planning with a focus on autonomous vehicles, and future development directions. By offering a comprehensive overview and analysis of AI-based path planning algorithms, this paper seeks to provide valuable insights for researchers and practitioners in the field, ultimately contributing to the advancement of intelligent transportation systems.

2. Basic Concepts of Path Planing

Path planning is a fundamental technology in robotics systems field, which involves finding an optimal or feasible path for a mobile agent (e.g., a robot or a vehicle) to navigate while avoiding obstacles and satisfying certain constraints [3]. The mainly goal of path planning is to determine the

safest, shortest, or most energy-efficient route, depending on the specific application and requirements.

There are several key concepts and terminologies associated with path planning, such as configuration space, search space, and cost functions. The configuration space is a representation of all possible positions and orientations that a mobile agent can occupy in the environment [4]. Search space, on the other hand, refers to the set of possible solutions to the path planning problem, which can be explored using various algorithms to find an optimal path. Cost functions are utilized to evaluate and compare the quality of different paths, often considering factors such as path length, travel time, energy consumption, and safety [5].

Path planning technology can be classified into two types. Path planning in global map is aiming to find an optimal path from the start to the destination, considering the entire environment and all known obstacles [6]. This is often achieved through algorithms such as A* search, Dijkstra's algorithm, or graph-based methods. Path planning in local map, on the other hand, focuses on real-time obstacle avoidance and navigation in a dynamic environment, where the mobile agent must react to changes and uncertainties. Common techniques for local path planning include potential field methods, velocity obstacles, and artificial potential fields [7].

In recent years, AI-based methods have emerged as powerful tools for path planning, incorporating machine learning, deep learning, and reinforcement learning techniques to enhance the efficiency, adaptability, and robustness of path planning algorithms. These approaches have shown great promise in handling complex and dynamic environments, particularly for applications such as autonomous vehicles, drones, and mobile robots.

3. Traditional path planing methods

3.1. Dijkstra's algorithm

This is a widely recognized and influential graph search algorithm. It is primarily used for finding the shortest path in map, which may represent various types of networks, such as road networks, communication networks, or social networks. The algorithm is applicable to both global and local scenarios, as it can be applied to any weighted graph with non-negative edge weights.

Edsger W. Dijkstra (1930-2002) was a pioneering computer scientist who made significant contributions to the fields of programming languages, algorithms, and distributed computing. His work on the shortest path problem, which led to the development of Dijkstra's algorithm, remains one of his most prominent contributions to the field of computer science [8].

The algorithm operates by maintaining a set of unvisited nodes and their tentative distances from the starting node. First, the tentative distance between the starting node and all other nodes was set to zero and infinity. The algorithm then iteratively selects the unvisited node. The principal of Dijkstra's algorithm can be explained as follows [9]:

- 1) The initial distance value of each node is set, where the starting node and other nodes are zero and infinite, respectively.
- 2) The current node has been replaced with the starting node.
- 3) Iterative calculation of the distance from the starting node to adjacent nodes that have not been accessed. If the current tentative distance of the neighbor is greater than this distance, update the tentative distance.
- 4) The current node is marked as already visited.
- 5) The unreachable node with the minimum tentative distance is set as the current node, and steps 3-5 are repeated to guide all nodes to have been accessed or the target node to have been accessed.

The Dijkstra algorithm can be used to complete path planning tasks in a wide range of real-world scenarios. For example, in transportation networks, it can be employed to determine the shortest route between two locations, taking into account factors such as distance and traffic conditions [10]. In communication networks, the algorithm is used to optimize routing paths and minimize latency

between nodes. Furthermore, Dijkstra's algorithm is relevant in social network analysis, as it can identify the shortest path between individuals in a social graph [11].

3.2. A* algorithm

A-star (A*) algorithm is a graph traversal and path finding algorithm widely used in various fields. The principle of the A* algorithm can be explained as follows [12]:

- 1) Assign a cost value, $f(n)$, to each node n , where $f(n) = g(n) + h(n)$, where $g(n)$ is the actual cost between starting node and current node, and $h(n)$ is the estimated cost between current node and goal node.

- 2) Create an open set containing the starting node and a closed set that is initially empty.

- 3) $F(n)$ is selected as the smallest node when the open set is not empty and it is removed from the open set. If this node is the goal node, the algorithm terminates successfully.

- 4) Otherwise, add the chosen node to the closed set.

- 5) For each neighbor of the chosen node, calculate the tentative $g(n)$ value, assuming the path goes through the chosen node.

- 6) If there are no neighbors in the closed set and the tentative $g(n)$ value is less than the neighbor's current $g(n)$ value, update the neighbor's $g(n)$ value and set its parent to the chosen node. Determine whether neighbors are in an open set to operate on them.

The A* algorithm is applicable to both global and local scenarios and is widely used in real-world applications. [13]. In computer games, A* is employed to navigate virtual characters through complex environments.

Yang proposed a modified version of the traditional A* algorithm for more efficient path planning in a mobile robot. This modification involves the use of a new heuristic function that takes into account both the distance to the goal and the orientation of the robot. The modified A* algorithm also incorporates a local path planning method to avoid local minima and oscillations, which can occur in situations with closely spaced obstacles [13].

By combining global and local path planning, the modified A* algorithm effectively addresses the limitations of the traditional A* algorithm in dealing with complex environments, particularly when dealing with dynamic obstacles or changing environments.

3.3. Rapidly-exploring Random Tree algorithm

Rapidly-exploring Random Tree (RRT) algorithm is a widely used in many path planning scenes. It was specifically designed to handle high-dimensional configuration spaces and complex environments, making it suitable for various applications in robotics, computer graphics, and artificial intelligence [14]. The principle of this algorithm can be explained as follows [15]:

- 1) Initialize the tree with the starting configuration as its root node.

- 2) Repeat the following steps iteratively to find a satisfactory solution: Firstly, a random sample is generated in the configuration space. Subsequently, locate the node closest to the random sample in the tree. Finally, create a new node by predefined distances in the direction of the random sample and extending the tree to the random sample. If the path between the nearest node and the new node does not conflict, the new node is added to the tree.

- 3) If the tree has reached the goal configuration or a node sufficiently close to the goal, the path from the starting configuration to the goal configuration is extracted by traversing the tree.

RRT is designed to explore the entire configuration space quickly and effectively. However, it can be combined with local path planning methods to refine the solution or handle dynamic environments.

The RRT algorithm has found applications in various real-world scenarios. In robotics, RRT is used for motion planning in robotic manipulators, mobile robots, and autonomous vehicles [16]. In computer graphics, the algorithm is employed for generating realistic animations of characters navigating complex environments.

A recent study by Kim et al. (2018) presents an RRT-based approach for autonomous drone navigation in complex environments [17]. The authors propose a modified RRT algorithm that

incorporates a local path planning method to improve the efficiency and safety of drone navigation in real-time.

3.4. Dynamic A* algorithm

Dynamic A* (D*) algorithm, developed by Anthony Stentz in 1994, is a path planning algorithm designed for dynamic environments where edge costs can change during the search. It is well-suited for robotics and autonomous vehicle applications that require real-time adjustments due to changing conditions or new environment information [18].

The principle of D* is based on updating an initially established path when new information about the environment is obtained, making local adjustments as necessary [19]. D* is a global path planning algorithm but can make local adjustments in response to dynamic changes.

Real-world applications of the D* algorithm is mostly in robotics and autonomous vehicles for path planning and navigation in dynamic environments. A recent study by Manandhar et al. (2017) presents a modified D* Lite algorithm for unmanned ground vehicles (UGVs) in urban environments, incorporating geometric and semantic information from the environment to improve path planning efficiency and safety [20].

In summary, the D* algorithm by Anthony Stentz is a powerful path planning algorithm designed for dynamic environments, with applications in robotics and autonomous vehicles where real-time path adjustments are crucial for efficient and safe navigation.

4. AI-based algorithms

AI-based path planning algorithms have emerged as powerful alternatives to traditional methods like Dijkstra's, A*, RRT, and D* algorithms[21]. These AI-based approaches can handle complex and dynamic environments, often providing more efficient and robust solutions. Some popular AI-based path planning methods include deep reinforcement learning, ant colony optimization, and differential evolution.

4.1. Deep reinforcement learning algorithm

Deep reinforcement learning (DRL) is an AI technique to enable agents to learn directly from raw input data and make decisions to achieve a specific goal [22]. Unlike traditional path planning algorithms, which are primarily based on graph search and heuristics, DRL algorithms can learn and adapt their behavior over time by interacting with the environment. This allows them to handle complex and dynamic environments more effectively. One popular DRL algorithm is Proximal Policy Optimization (PPO), which has demonstrated superior performance in various applications, including robotics and autonomous vehicles.

4.2. Ant Colony Optimization algorithm

Ant Colony Optimization (ACO) is a bio-inspired optimization technique to find optimal paths in a graph [23]. While traditional methods like Dijkstra's and A* algorithm rely on deterministic search strategies, ACO uses a probabilistic approach to explore the search space. This can result in more efficient solutions, especially in large-scale and complex environments. ACO algorithms have been successfully applied to various scenes, including vehicle routing and network optimization.

4.3. Differential Evolution

Differential Evolution (DE) is another AI-based optimization technique that has been employed for path planning. DE is a population-based, stochastic optimization method that iteratively refines candidate solutions through mutation, crossover, and selection operations. This enables DE to efficiently search the global solution space and avoid local minima, which can be an issue in traditional methods like A* and RRT. DE has been used in a range of applications, including robotics and sensor network deployment.

5. Conclusion

The transition to AI-based path planning methods has been driven by their ability to adapt to complex and dynamic environments more effectively than traditional methods. For instance, DRL algorithms can learn to navigate environments with changing obstacles, whereas traditional methods like Dijkstra's and A* would require continuous updates to their search graphs. Similarly, ACO and DE algorithms can efficiently explore large-scale search spaces and avoid local minima, which can be problematic for RRT and D* algorithms.

In conclusion, AI-based path planning algorithms offer promising alternatives to traditional methods by providing more adaptive and efficient solutions in complex environments. As research in AI continues to advance, we can expect further improvements in path planning algorithms, leading to increased practical applications in fields such as robotics, autonomous vehicles, and transportation systems.

References

- [1] Chen, T., & Shen, G. (2019). Review of intelligent transportation systems for urban network traffic signal control. *IET Intelligent Transport Systems*, 13(1), 1-11.
- [2] Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7), 760-772.
- [3] LaValle, S. M. (2006). *Planning algorithms*. Cambridge University Press.
- [4] Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. E., & Thrun, S. (2005). *Principles of robot motion: Theory, algorithms, and implementation*. MIT Press.
- [5] Ge, S. S., & Yang, Y. (2010). *Autonomous mobile robots: Sensing, control, decision-making, and applications*. CRC Press.
- [6] Kumar, S., & Cohen, I. (2000). Global path planning for autonomous robot navigation. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (Vol. 3, pp. 165-170)*. IEEE.
- [7] Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7), 760-772.
- [8] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269-271.
- [9] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.
- [10] Zhan, F. B., & Noon, C. E. (1998). Shortest path algorithms: An evaluation using real road networks. *Transportation Science*, 32(1), 65-73.
- [11] Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2), 163-177.
- [12] 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.
- [13] Yang, S. X., & Meng, M. (2000). Path Planning with Modified a Star Algorithm for a Mobile Robot. *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, 2000, 2, 1177-1182. doi: 10.1109/ROBOT.2000.844739
- [14] LaValle, S. M. (1998). *Rapidly-exploring random trees: A new tool for path planning*. Technical Report TR 98-11, Computer Science Department, Iowa State University.
- [15] LaValle, S. M., & Kuffner, J. J. (2001). Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, 20(5), 378-400.
- [16] Kuwata, Y., Fiore, G. A., Teo, J., Frazzoli, E., & How, J. P. (2008). Motion planning for urban driving using RRT. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1681-1686.

- [17] Kim, S., Lee, S., Lee, M., & Myung, H. (2018). Autonomous drone navigation system for indoor environments using RRT-based path planner and artificial potential field. 2018 15th International Conference on Ubiquitous Robots (UR), 887-891.
- [18] Stentz, A. (1994). Optimal and Efficient Path Planning for Partially-Known Environments. Proceedings of the IEEE International Conference on Robotics and Automation, 1994, 3310-3317.
- [19] Stentz, A. (1995). The Focussed D* Algorithm for Real-Time Replanning. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995, 1652-1659.
- [20] Manandhar, S., Song, J. B., & Yim, S. (2017). A modified D* lite algorithm for unmanned ground vehicle navigation in a dynamic urban environment. 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 519-524.
- [21] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533.
- [22] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.
- [23] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347.