Path Planning

Ahmed Gamal Mohamed

Alexandria university, Faculty of Engineering
Department of Communication and Electronics

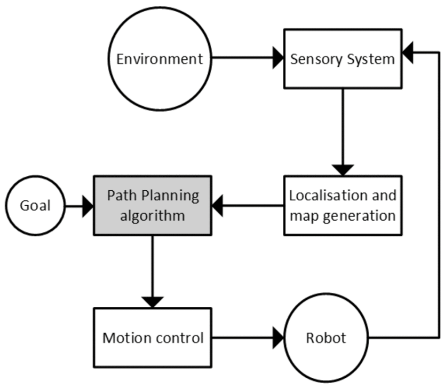
Abstract

Path planning, the basis of successful navigation and obstacle avoidance, is a fundamental and essential component of robotics and autonomous systems. In order to move from starting points to desired destinations while avoiding obstacles and obeying by constraints, path planning includes the strategic computation of possible routes for mobile robots or beings within a variety of environments. Environment representation, search algorithms, collision detection, and optimisation methods are essential elements of path planning. Real-time adaptation and dynamic environments further increase its complexity. Path planning has a wide range of uses, including in video games, industrial automation, aerial drones, and autonomous vehicles.

1 Introduction

The capacity to move through and interact with complicated settings is crucial in a world where automation and autonomy are becoming more and more prevalent. The idea of "path planning" is at the core of all of these efforts, whether it be an autonomous vehicle navigating city streets, a robotic arm in a manufacturing environment, or a drone navigating erratic airspace. The fundamental intelligence of path planning enables machines and robots to travel meaningfully, securely, and effectively from one location to another, frequently while navigating a maze of obstacles and limitations.Path planning, also known as motion planning, encompasses the art and science of finding a feasible and optimal route for a mobile agent – be it a robot, vehicle, or virtual entity – to reach its destination while avoiding collisions and adhering to various constraints. It's the digital equivalent of plotting a course on a map or a GPS system, but with the added complexity of adapting to dynamic, real-world scenarios.

2 Types of Path Planning Algorithm



2.1 Dijkstra Algorithm

The Dijkstra algorithm works by solving sub-problems to find the shortest path from the source to the nearest vertices. It finds the next closest vertex by keeping the new vertices in a priority-min queue and only storing one intermediate node, allowing for the discovery of only one shortest path. Additionally, it is a reliable path planning algorithm because it cannot tolerate negative edges and must calculate every potential outcome in order to identify the shortest path, it consumes a lot of memory. The Dijkstra algorithm is best suited for a static environment and/or global path planning since the majority of the data needed to determine the shortest path is pre-defined.

2.2 A* Algorithm

Similar to Dijkstra's method, the A* method is a popular network navigation path planning algorithm. But in order to potentially save time, it focuses its search on the most promising states. A* is the most popular technique for approaching a near-optimal solution with the provided data-set/node. A* is a heuristic-based search algorithm that combines the benefits of Dijkstra's algorithm with a heuristic function that guides the search toward the goal. It is widely used for finding optimal paths in various applications, including robotics and video games.The algorithm consists of three different terms, a cost function that takes into account the cost up to the current node$g(N_i)$, the cost from the current to the next node $c(N_i, N_{i+1})$, and the estimated cost from the next node to the destination $h(N_{i+1}, N_d)$. These costs are summed up to calculate the cost of the current node.

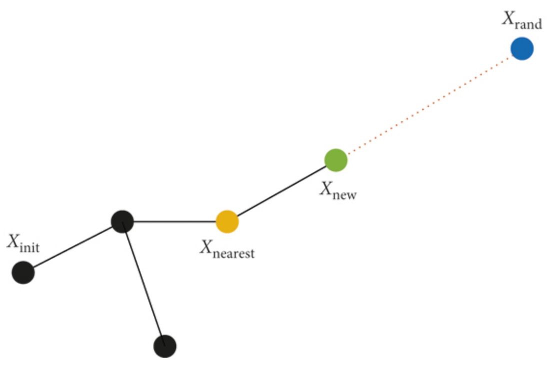
$$f(N_i) = g(N_i) + c(N_i, N_{i+1}) + h(N_{i+1}, N_d)$$

2.3 D* Algorithm

The importance of path planning in partially known and dynamic environments is rising, particularly for automated vehicles. To resolve this issue, the D* (or Dynamic A*) algorithm creates a collision-free path among moving obstacles. The cost map and the cost map that was previously calculated are both partially repaired by the D* cost map repair algorithm using informed gradual search.

2.4 Rapidly-Exploring Random Trees (RRT)

RRT is a probabilistic algorithm made for configuration spaces with many dimensions. It connects the random samples in a tree-like structure to form a path. RRT works especially well in complex, multidimensional environments.PRMs may require connections of thousands of configurations or states to find a solution, whereas RRTs does not require any connections between states to find a solution.RRT takes the initial point as the root node and generates a random extended tree by adding leaf nodes through random sampling (Figure 6). When the leaf nodes in the random tree include the target point, a path from the initial point to the target point can be found in the random tree. This helps in applying RRTs to non-holonomic and kinodynamic planning. RRTs expand by rapidly sampling the space, grow from the starting point, and expand until the tree is sufficiently close to the goal point.

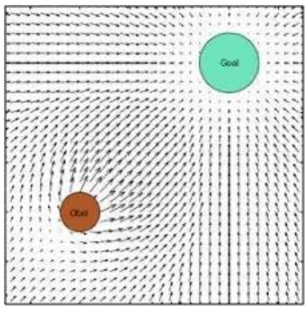


2.5 Visibility Graphs

Visibility graphs identify visible edges between obstacles in the environment. These edges are used to create a graph, and graph search algorithms like Dijkstra's or A* can find paths on this graph. Visibility graph methods work well for environments with polygonal obstacles.

2.6 Potential Field

Potential Field Algorithm is one of the most fundamental and traditional algorithms used for such a problem.A potential field is any physical field that obeys Laplace's equation. Some common examples of potential fields include electrical, magnetic, and gravitational fields. A potential field algorithm uses the artificial potential field to regulate a robot around in a certain space. For our ease, we consider a space to be divided into a grid of cells with obstacles and a goal node.



The algorithm assigns an artificial potential field to every point in the world using the potential field functions which will be described further in the explanation. The robot simulates from the highest potential to the lowest potential. Here, the goal node has the lowest potential while the starting node will have the maximum potential. Hence, we can say that the UAV moves from lowest to the highest potential.

3 Local planner and Global planner

The Local Planner and the Global Planner are two essential elements that stand out as the foundation of effective navigation in the context of path planning for mobile robots and autonomous systems. They coordinate the smooth movement of robots through complex environments, translating high-level objectives into actionable trajectories.

3.1 Global Planner

The process of path planning is imagined by the global planner. It performs high-level operations by outlining the robot's path from its starting point to its final destination. Its main responsibility is to establish the global path, a macro-level route that specifies the overall course and important checkpoints the robot should take. Global planners take into account the environment's overall map, taking into account terrain, obstacles, and any known limitations.
Key characteristics of the Global Planner include:
Environment Representation: Global planners work with a global map of the environment. This map can be represented in various ways, such as occupancy grids, graphs, or even high-level descriptions of the terrain.

Optimization: Global planners often strive to find an optimal path based on specific criteria, such as minimizing travel distance, time, or energy consumption. Popular global path planning algorithms like A* or Dijkstra's algorithm excel at finding these optimal routes.

Waypoint Generation: The global planner identifies a series of waypoints that guide the robot from start to finish. These waypoints help the robot make high-level decisions and provide context for the local planner.

Static Planning: Global planners usually assume a static environment, which means they don't account for dynamic obstacles or real-time changes. They calculate the path before the robot starts moving.

3.2 Local Planner

The Local Planner is the executor, guiding the robot through the environment's minute details while the Global Planner plots the overall course. It functions on a more fundamental level, concentrating on immediate obstacles, terrain variations, and real-time adjustments. The main goal of the Local Planner is to maintain the robot's compliance with the global path while avoiding collisions and making adjustments to dynamic changes in the environment.
Key characteristics of the Local Planner include:
Obstacle Avoidance: The Local Planner constantly assesses the immediate surroundings of the robot to detect obstacles and determine how to navigate around them. It uses sensor data, such as lidar or cameras, to make real-time decisions.

Dynamic Adaptation: Unlike the Global Planner, the Local Planner can adapt to dynamic obstacles, which are objects or agents that move within the robot's vicinity. It calculates avoidance maneuvers in real time to avoid collisions.

Low-Level Control: Local planners often interface directly with the robot's control system, sending commands for velocity, steering, or other control inputs. This enables precise execution of the planned path.

Real-Time Operation: Local planners operate in real time, continuously updating the robot's trajectory based on sensor data and current conditions. This responsiveness is critical for navigating through unpredictable environments.

4 Challenges of Path Planning

Collision Avoidance: The robotic arm must avoid running into any of the workspace's other items. To prevent collisions, the arm needs to be able to recognise obstacles and navigate around them.
Dynamic Obstacles: Dynamic obstacles are objects that move through the workspace, such as humans or other robots. The robotic arm must be able to detect and avoid these obstacles in real-time to prevent collisions.
Safety and Robustness: It is crucial to make sure autonomous systems are safe. To handle sensor noise, uncertainties, and unforeseen circumstances, path planners must be robust.
Energy Efficiency: Energy use is a major worry, particularly for mobile robots and drones. Planners of paths ought to prioritise routes that use little energy.

5 Future Trends

Machine Learning Integration: Path planning is increasingly integrating machine learning methods, such as deep reinforcement learning and neural networks, to improve adaptability and decision-making in dynamic environments.
Hybrid Methods: Robots will increasingly combine various path planning algorithms and optimisation methods to take advantage of the advantages of various strategies.
3D and Beyond: 3D path planning will be more and more crucial for space exploration and urban air mobility as drones and autonomous flying vehicles proliferate.
Semantic Mapping: Path planning and decision-making will be enhanced by improved mapping techniques that offer semantic information about the environment (such as identifying various objects or regions).

6 Applications

Autonomous Vehicles: For self-driving cars, trucks, and other autonomous vehicles, path planning is essential. It assists them in navigating complex road networks, dodging obstacles, and making safe driving judgements.
Aerial Drones: In order to fly autonomously and carry out tasks like aerial photography, surveillance, search and rescue, and package delivery, drones use path planning. They follow predetermined routes and avoid collisions thanks to path planning.
Warehouse Automation: Autonomous mobile robots (AMRs) use path planning to optimize the movement of goods within warehouses, reducing labor costs and increasing efficiency.
Space Exploration: Autonomous rovers and spacecraft rely on path planning to navigate celestial bodies, collect data, and perform scientific experiments in space missions.

7 Conclusion

Based on the previous evaluation, some planning algorithms still have drawbacks, despite significant advancements in the study of unmanned ship path planning algorithms. For instance, the Dijkstra and A* algorithms are slow and inefficient, and the particle swarm algorithm is less capable of local optimisation. There are issues with the artificial potential field method, such as local minimum. The commonly employed improved algorithms are primarily based on a combination of various path planning algorithms in light of the aforementioned issues. There are still many development bottlenecks, such as growing algorithm complexity, even though they make up for the shortcomings.

References

[1] https://medium.com/@rymshasiddiqui/path-planning-using-potential-field-algorithm-a30ad12bdb08

[2] https://roboticsbiz.com/path-planning-algorithms-for-robotic-systems/

[3] https://iopscience.iop.org/article/10.1088/1742-6596/1213/3/032006/pdf

[4] https://www.mdpi.com/2218-6581/12/4/113