

Several Global Path Planning Algorithms and Modification

Weimin Ouyang

Department of Glasgow College Hainan, University of Electronic Science and Technology of
China, Hainan, China

2022360903022@ std.uestc.edu.cn

Abstract. Mobile robots are being used in a wide range of scenarios with increasing frequency. But it is also an extremely difficult field with an interdisciplinary nature. Path planning (PP) is one of the most challenging problems in robotics, which can be further categorized into global navigation and reactive navigation. This article will focus on global navigation, aiming to provide a general review of several important global PP algorithms on their principles and significant improvement as well as propose some tenable modification suggestions. Dijkstra algorithm, as the most classic and mature algorithm, has been executed many modifications to address its inherent limitations. For instance, an optimizing data storage structure has been employed to obtain enhanced efficiency. The genetic algorithm is the most important evolution algorithm, giving a satisfactory result especially in large-scale searching. Modification of GA has also been conducted by introducing newly designed operators to facilitate the evolution and combining the APF method to guarantee a smoother path. A star is another popular heuristic algorithm, introducing a turning factor based on Dijkstra to reduce the searching area. It has also been improved in various aspects, particularly in refining its heuristic function. A neural network encoder has also been utilized to generate a satisfactory path after constructing a differentiable A star module. Three algorithms are then compared to further analyze their characteristics respectively, followed by proposed enhancements tailored to each algorithm. Through the content above, this paper gives a compact view of the main PP algorithms and suggests some potential directions for future modification.

Keywords: Path planning, Dijkstra algorithm, A star algorithm, Genetic algorithm.

1. Introduction

Mobile robots are currently applied in various fields, such as space exploration and emergency rescue [1, 2]. They can do many dangerous and arduous tasks for humans, which explains why they are so popular nowadays [1]. Among all the issues encountered in robotics, navigation is one of the most challenging ones because of the varying environment, especially for robots with a high degree of freedom operating on hostile roads [2]. The definition of navigation (or path planning) is quite straightforward: find a path (usually an optimal one) from the start point to the end point without collision in a certain environment [2]. It can be further categorized into global navigation and reactive navigation, depending on whether the precise map information is accessible for the robot [3]. If the robot has no knowledge of the environment, it must obtain the information using different sensors, mapping the environment while performing tasks [3]. On the contrary, if precise map information is provided, such as the position of stationary obstacles and the trajectory of dynamic obstacles, the robot can employ many algorithms to find the optimal path [3].

Up to now, many global path planning algorithms have been developed to fit different application situations, such as the Dijkstra algorithm, Genetic Algorithms (GA), and A star algorithm [3-5]. Dijkstra algorithm is one of the earliest methods that emerged in solving the shortest path (SP) problem [5]. The most important characteristic of the Dijkstra algorithm is that it can determine the shortest path from the starting node to any destination node along the route since every sub-path of the shortest path is also the shortest path from the initial node to the final node [6]. Among all the global path planning algorithms, A star is one of the most popular ones, widely used in robotic path planning [3, 7]. A star is developed depending on the Dijkstra algorithm by introducing a turning factor [6]. This modification avoids searching all the nodes in the map to reduce computational costs. Except for heuristic algorithms like the A star algorithm, evolutionary algorithms are also widely

used in path planning, especially for highly complicated environment [8]. Genetic algorithm is particularly distinctive among them because of its high robustness and strong global search capability [4].

This article will concentrate on global path planning, where the robot has no need to map the environment while performing tasks. Several key points of the main global PP algorithms mentioned above will be introduced and compared to give a concise view of this field and propose some viable suggestions for each algorithm based on these descriptions.

2. Algorithm and Modification Description

2.1. Dijkstra Algorithm

2.1.1 Basic Principle

The traditional Dijkstra finds the shortest path from the initial point to the end point by progressively exploring paths of increasing length [6]. Define a directed graph $G = (N, A)$, where N is the set of nodes and A represents the set of paths. $W(A, B)$ stores the weight of path $\{A, B\}$. The weight will be infinite if path $\{A, B\}$ does not exist. This is usually denoted by using the maximum integer number in the program, such as 4096 [6]. List U is then employed to store the nodes that have not been added to $closeList$, while list S stores the shortest distance for every node that has been searched [6, 7].

The whole algorithm can be described as follows [6, 7]:

(1) Initialization: Import start node (s), target node (t), and weights for paths between each node. Define U and S sets following the above instructions, S contains only the start point while U contains all the other nodes.

(2) First Searching: all the adjective nodes of the start node s and compare their weights to find the shortest node, say i_1 , store it and the corresponding distance into the list S . Add i_1 to the optimal path list and set it as next searching center.

(3) Keep Searching: skipping all the nodes in list S , repeat the above operations to find i_2 while refreshing the distance for all the scanned nodes based on whether new path (via the new searching center) is shorter than the previous shortest distance.

(4) End of Searching: repeat step (3) until all the nodes are added to list S .

One advantage of the Dijkstra algorithm indicated in above description is that it can find the shortest path for every node in the graph [6].

2.1.2 Modification

In most Automated Storage and Retrieval System, the graph is usually modeled as a rectangle or square shape, where the equally short paths tend to be ignored [6]. A method was thus proposed to find all the potential shortest paths by selecting all the equally short nodes in one research as the next searching centers [6]. Eventually, all the paths will be compared to pick the shortest path.

Since distance together with consumed time are both criteria for an optimal path in that scenario, the turning node inside each path is counted and compared since the turning process costs considerable time for robots in the AS/RS system [6]. This can be easily done by performing subtraction between coordinates of the previous node ($m-1$) and current node (m), current node and latter node ($m+1$). Turning points should have unequal results [6].

Still, in the context of grid map, one modified eight-direction searching method has been utilized to flexibly detect surroundings, potentially providing a more optimal path [5].

Another method was suggested in one article to improve the Dijkstra algorithm by reducing its consumption of storage and computational resources in the situation of real traffic networks [7]. The article first explains the reason why storing the topological relationship of the classic Dijkstra algorithm through an adjacency matrix is extremely inefficient [7]. It then purposed to create a list for each node to store the adjective nodes and corresponding path distance using name-value pairs [7]. This was proved to be a more efficient method for eliminating storage redundancy and useless

computation through the respective of space complexity and time complexity [7]. The illustration for the above method is presented in the following two figures. The Figure 1 represents the stored name-value pairs for each node inside the second figure while the Figure 2 shows the connection between each node and weights for each path.

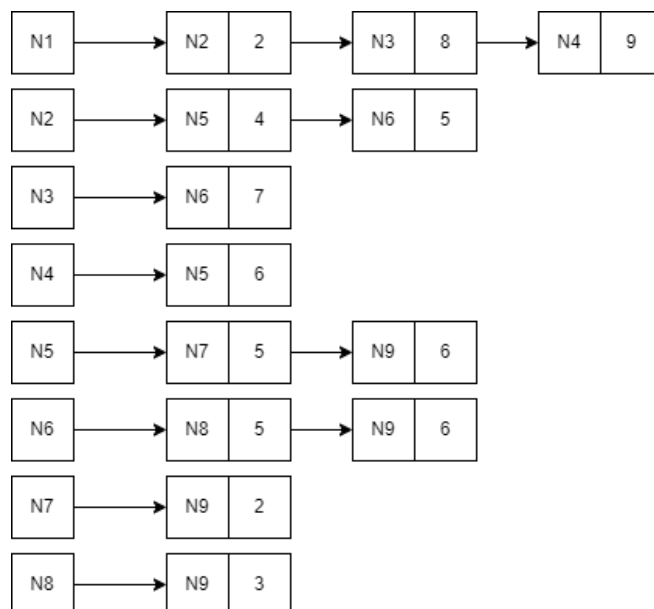


Fig 1. Storage Structure [7]

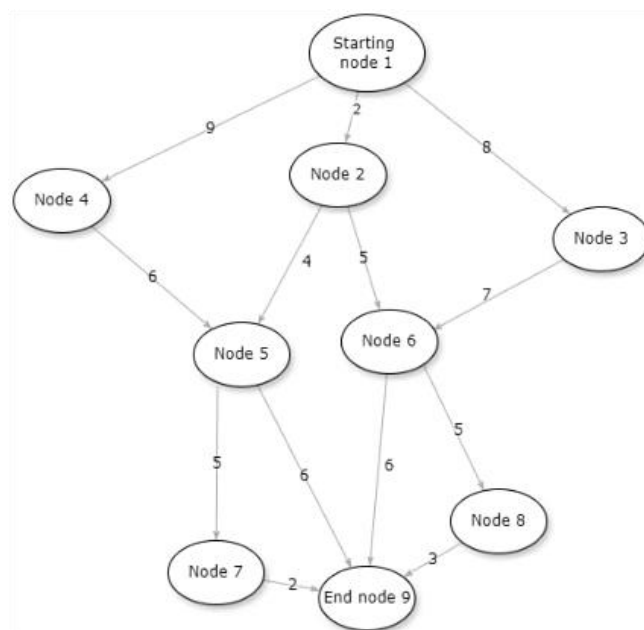


Fig. 2 Corresponding Node Map [7]

2.2. Genetic Algorithm

2.2.1 Basic Principle

The first step of the genetic algorithm is to define necessary parameters like population size, mutation probability, and crossover probability, which are typically decided according to human experience [4, 8]. After all the initial population are randomly generated, all the candidate solutions are evaluated depending on the fitness function, determined by the problem setting to be solved. Only suitable individuals can survive to reproduce offsprings [4, 8]. Additionally, to simulate the evolution process, the genetic algorithm introduces some operations like crossover, mutation, and inversion [4, 8].

The crossover is what would happen between two parent chromosomes to partially exchange the genes between parents, aiming to alike the chromosome within different individuals to help the algorithm converge [4]. This process in the algorithm is determined by the crossover probability and split points [4].

Mutation, on the contrary, will increase the genetic diversity within the population to avoid falling into local optimal value [4]. This is through interchanging the values between randomly selected points with a probability (P_M) [4].

Inversion has many types. Taking single-domain inversion for example, this will inverse certain domain of one chromosome to reproduce children [4].

These processes above will be continuously repeated until the convergence condition is satisfied.

2.2.2 Modification

Potential inefficiency of the traditional GA can be caused by the crossover operation since this process extremely destroys the chromosome and hinders the inheritance of advantage genes within parents [4]. Additionally, the mutation and single-domain inversion are not capable of diversing the parents' genes. Given the two reasons above, a new strategy named double-domain inversion is proposed [4]. As indicated in its name, this operation doubles the domain of inversion by randomly defining four points and selecting two pairs out of them, comparing the fitness of children and parent to determine which one can survive [4]. Here since four points can determine six domains in maximum, these children will all be compared with parent to increase the total number of offspring [4]. This operation is supposed to retain more competitive genes from the parent as well as improve the local searching ability simultaneously [4].

One hybrid algorithm was proposed to combine the advantage of the artificial potential field (APF) method and GA [8]. All the feasible piecewise linear paths from initial and destination points within a grid map are determined by the APF method to produce parent with higher quality [8]. At least one feasible path can be obtained if it exists, which are named Path Bases (PBs) [8]. It should be noted here that the PBs are a series of points to be connected using the Enhanced Genetic Algorithm (EGA), accounting for what "piecewise" means [8].

PBs are then passed to EGA, a customized version of GA with newly proposed operators [8]. This algorithm involves two modified crossover operators, one calculating the mean of two parents directly and the other conducting the linear combination of two parents [8].

$$x_f = \frac{x_{p1} + x_{p2}}{2}, y_f = \frac{y_{p1} + y_{p2}}{2} \quad (1)$$

$$x_f = \alpha x_{p1} + (1 - \alpha)x_{p2}, y_f = \alpha y_{p1} + (1 - \alpha)y_{p2} \quad (2)$$

Where α is a line vector of random numbers within [-1, 1].

The former operator aims to produce a shorter and smoother path, while the latter is employed to randomly search the surroundings and prevent convergence during unseasonable period [8].

As for the mutation operators, one brand new operator was also introduced to integrate its current point and adjacent two points [8]

$$x_{fi} = x_{pi} + \alpha(x_{p_{i-1}} - x_{pi}) + \beta(x_{p_{i+1}} - x_{pi}), y_{fi} = y_{pi} + \alpha(y_{p_{i-1}} - y_{pi}) + \beta(y_{p_{i+1}} - y_{pi}) \quad (3)$$

Where α and β are tiny positive factors. The principle of this operator can be shown through parallelogram law as indicated in Figure 3 below, giving a smoother and shorter distance [8].

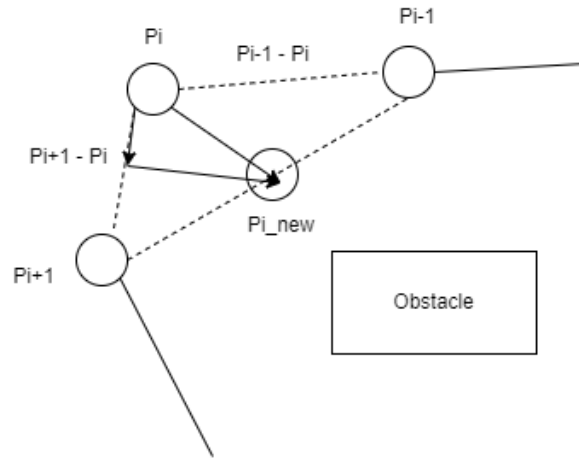


Fig. 3 Illustration for the Mutation Operator above [8]

Another deletion operator is also required to delete redundant PBs within each initial path since they will undermine the flexibility of the fitting process of PBs based on piecewise linear or spline paths [8]. This is done by evaluating each PB sequentially to determine whether to delete this point or not, depending on whether the deletion can make a positive contribution [8]. The positive here means something like reducing the path length or increasing smoothness [8].

To implement multi-objective path planning, the strategy included in this paper is to include three types of criteria in the fitness function, that is length, smoothness, and safety [8]. Three terms are combined using three factors [8]:

$$F(p) = W_L \times L(p) + W_s \times S(p) + \frac{W_R}{R(p)} \quad (4)$$

where W_L , W_s , and W_R are the weights of length, smoothness, and safety, respectively.

The issue of multi-robot PP falls in the category of a dynamic path planning problem involving moving obstacles [8]. Two mechanisms are introduced to solve this problem, first one measures the minimum distance between every two paths of robots and assigns a penalty value to the fitness function based on the distance [8]. Typically, a linear penalty function that is inversely proportional to the measured distance will be employed. But it can also be an exponential one [8]. Another mechanism is to modify the set of PBs by randomly selecting and moving one of the PBs within the path [8]. The improved path will be evaluated and modified again, if necessary, until a collision-free condition is satisfied [8].

2.3. A Star Algorithm

2.3.1 Basic Principle

A star algorithm was first proposed by Hart [9] and it has been extensively used to deal with optimal path problems. A typical A star involves the following steps [3, 9]:

- Import the map information in the form of grid map, indicating the obstacles, start, and target information in the grid.
- Define closeList which contains only the initial point and openlist which contains all adjacent cells.
- Traverse eight adjacent cells of the current cell to evaluate the possible distance away from the target point. The evaluated distance consisted of two parts: real distance covered and estimated distance given by the heuristic function. The formula is as follows.

$$f(v) = g(v) + h(v) \quad (5)$$

Where $f(v)$ is the total distance from the initial point to the target point $g(v)$ is the distance covered from the initial point to the evaluated point, $h(v)$ is the evaluated distance generated by

heuristic function [3, 9]. There are three main kinds of heuristic functions: Manhattan distance, Euclidean distance, and Chebyshev distance. They are suitable for different application situations [3].

- The cell with the lowest $f(v)$ will be added to the closeList and selected as the next node in the path.

- Repeat the above searching steps until the goal node is contained in the closeList.

The introduction of the heuristic function effectively increases the path-finding process. It does not need to traverse all the nodes in the map like the Dijkstra Algorithm [7]. However, it is obvious that the quality of the result given by the heuristic function is critical for the success of path searching. This is also where various modifications can be made by modifying or even involving another distance factor [3].

2.3.2 Modification

Since the traditional A star algorithm can only search in 8 directions, potentially, it can be further optimized since there may exist much free space between two connected cells [3]. Theta* and Phi* algorithms are thus introduced to enable searching in every angle [3].

To continuously adapt the path, the Theta* algorithm inspects after each round of traversal and evaluation. Typically, the optimal cell s'' within all the 8 adjacent ones will be connected to parent node s' . But in Theta*, the parent node of s', s will be set as the parent node of s'' if two nodes are directly visible for each other [3]. The process is iterated until the initial point [3].

Phi* algorithm was developed based on the foundation of the basic Theta* algorithm by storing a local predecessor and two angles for every evaluated cell. The angle range determined by the recorded values specifies the area within which the predecessor can be found [3]. The characteristic of Phi* allows people to simulate the dynamics of a robotic system in the calculation of the algorithm by introducing extra restraints [3]. To reduce the number of turning points in the scenario of an autonomous land vehicle, a new heuristic function combined with the artificial potential field method was proposed [9]. The revised evaluation function is expressed as follows:

$$f(v) = g(v) + h_0(v) + p(v) \quad (6)$$

Where $f(v)$ and $g(v)$ represent the same meaning as the traditional function, $h_0(v)$ is the prediction given by traditional heuristic function, $p(v)$ is the cost given by potential field [9]. Improved heuristic function $h(v) = h_0(v) + p(v)$ thus contains both distance information and obstacle information [9]. The potential force consists of the repulsive force exerted by obstacles and the attraction force exerted by the goal [9]. The total force will be projected to eight searching directions to serve as $p(v)$. The characteristic of the potential field generates a smoother path [9].

Another improvement is to utilize a neural network to extract the characteristics in the map [10]. The strongest obstacle for this target is the discrete nature of path planning, impeding the predictability of this type of problem [10]. A differential A star algorithm is thus developed by transforming all the key variables like openList and closeList into matrix form, which means matrix operation can be directly applied [10]. A CNN-based encoder is employed to extract the characteristic of the map to generate a guidance map, which is then passed to differentiable A star module. The module can update the encoder by back-propagation [10]. An end-to-end model is thereby established and proved to have an excellent performance on various maps and situations [10].

3. Discussion

Three classic algorithms are described in this article. Dijkstra algorithm has an excellent performance for static, well-defined road networks without negative weights. However, the excessive focus on details consumes considerable computational resources, which is not necessary for a direct goal. A Star algorithm thus proposes a heuristic function to engage the goal point to minimize meaningless search. On the contrary, the Genetic algorithm was inspired by the evolution process, especially suitable for large-scale, intricate environment.

From the above description, all those algorithms can only fit finite situations. But depending on what application interested in, various modifications can be conducted. Among all the modifications introduced, one common method is to combine different algorithms to compensate intrinsic shortcomings of the original one. One modification of the A star algorithm considered the APF method to involve the information of obstacles in the decision of turning direction. Similarly, the ant community algorithm can be employed in the Dijkstra algorithm to provide information on the previous optimal path, assisting in reducing the redundant search in the grid map.

As for the A star algorithm, the most crucial improvement is located in the accuracy of the heuristic function. Many modifications have been conducted as described in previous sections. However, it should be promising to directly replace the heuristic function with a deep learning model, which can potentially extract the characteristics of obstacles map information and relative locations of start and target points to generate an accurate prediction of costs. The illustration of above discussion is in Figure 4.

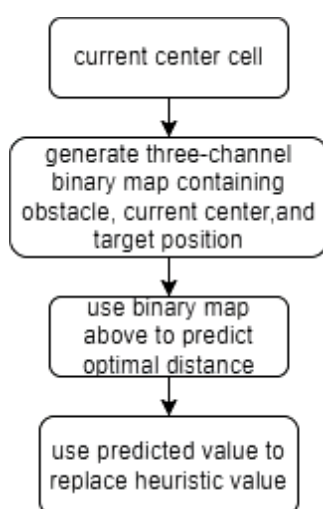


Fig. 4 Process of deep learning model prediction (Photo credited: original)

Genetic algorithm is essentially based on the real process of genetics, most operators are proposed to simulate the genetics and evolution of population. Therefore, it might be beneficial to simulate the generation and random recombination process of gametes, further improving the diversity of offspring. This can be done by using the weighted average of two gametes, the weight of which could be set according to the fitness value of parents. Figure 5 gives a flow chart of proposed improvement.

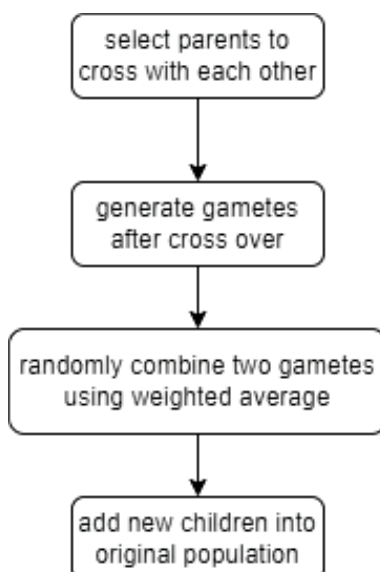


Fig. 5 Illustration of gamete simulation method (Photo credited: original)

The suggestions above could be feasible for future research, providing possible directions for algorithm development.

4. Conclusion

This article aims to provide a compact view of three popular global path planning algorithms, facilitating the researcher who knows little about this field to efficiently familiarize with the basic principles and recent modifications of these algorithms. In this article, Dijkstra, Genetic, and A star algorithms were described and analyzed in detail, providing a general review of mainstream path planning solutions and commonly used improving methods.

Based on this content, three algorithms were compared to further summarize their advantages and disadvantages respectively. Dijkstra is accurate but consumes too many unnecessary resources.

The efficiency of the Genetic algorithm primarily depends on the quality of parents and various operators, while A star's success is significantly influenced by heuristic function.

Possible improvement methods were then proposed for each algorithm to provide potential suggestions for future research. The pheromone factor is proposed to reduce the redundant searching given the global searching manner employed by Dijkstra. For GA, a simulated gamete method is suggested given almost all the operators used in GA come from real genetics. When it comes to A star, it might be beneficial to directly employ a deep learning model to serve as a heuristic function, owing to its excellent potential to learn characteristics from high dimensional data.

References

- [1] F. Rubio, F. Valer, C. Llopis-Albert. A review of mobile robots: Concepts, methods, theoretical framework, and application. *International Journal of Advanced Robotic Systems*, 2019, 16.
- [2] A. Gasparetto, P. Boscariol, A. Lanzutti, R. Vidoni. Path Planning and Trajectory Planning Algorithms: A General Overview. *Mechanisms and Machine Science*. 2015, 3-27.
- [3] F. Duchoň, et al. Path Planning with Modified a Star Algorithm for a Mobile Robot. *Procedia Engineering*, 2014, 96:59-69.
- [4] J. Zhong, F. Yang, Y. Cui, J. Sheng. An Improved Genetic Algorithm for Path-Planning of Unmanned Surface Vehicle. *Sensors*, 2019, 19:2640.
- [5] Y. Sun, M. Fang, Y. Su. AGV Path Planning based on Improved Dijkstra Algorithm. *J. Phys.: Conf. Ser.*, 2021, 1746:012052.
- [6] G. Qing, Z. Zheng, X. Yue. Path-planning of automated guided vehicle based on improved Dijkstra algorithm, 2017 29th Chinese Control And Decision Conference (CCDC), Chongqing, China: IEEE, 2017, 7138-7143.
- [7] D. Fan, P. Shi. Improvement of Dijkstra's algorithm and its application in route planning. 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery. 2010.
- [8] M. Nazarahari, E. Khanmirza, S. Doostie. Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. *Expert Systems with Applications*, 2019, 115:106-120.
- [9] J. Zhang, J. Wu, X. Shen, Y. Li. Autonomous land vehicle path planning algorithm based on improved heuristic function of A-Star. *International Journal of Advanced Robotic Systems*. 2021.
- [10] R. Yonetani, T. Tanai, M. Barekatin, M. Nishimura, A. Kanezaki. Path Planning using Neural A* Search. *arXiv*, 2021,