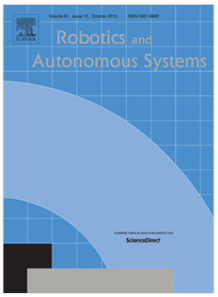
Accepted Manuscript



An improved A* algorithm for the industrial robot path planning with high success rate and short length

Bing Fu, Lin Chen, Yuntao Zhou, Dong Zheng, Zhiqi Wei, Jun Dai, Haihong Pan

PII: S0921-8890(17)30659-0
DOI: https://doi.org/10.1016/j.robot.2018.04.007
Reference: ROBOT 3019

To appear in: *Robotics and Autonomous Systems*

Received date : 3 October 2017
Revised date : 1 April 2018
Accepted date : 18 April 2018

Please cite this article as: B. Fu, L. Chen, Y. Zhou, D. Zheng, Z. Wei, J. Dai, H. Pan, An improved A* algorithm for the industrial robot path planning with high success rate and short length, *Robotics and Autonomous Systems* (2018), https://doi.org/10.1016/j.robot.2018.04.007

An improved A* algorithm for the industrial robot path planning with high success rate and short length

Bing Fu, Lin Chen,Yuntao Zhou, Dong Zheng, Zhiqi Wei, Jun Dai, Haihong Pan*

(Guangxi Colleges and Universities Key Laboratory of Modern Design and Advanced Manufacturing, GuangXi University, NanNing 530004, China)

Abstract Intelligent path planning is a significant tool for field of industrial robot. This field has attracted the attention of numerous researchers due to the great market demands, broad application prospects, and large potential development. Due to the limitation of neighborhood, the path search by the original A* algorithm is more likely to fail, and the solved path may contain too many local paths. In this study, an improved A* algorithm is proposed to solve the robot path planning problem. The first improvement of the advanced method is the local path between the current node and the goal node, which is planned before the next search in the neighborhood of the current node. And the local path will be adopted directly if it is safe and collisionless. The second advantage of this method is the utilization of post-processing stage to optimize the resulting path, by straightening the local path to reduce the number of local paths as well as the path length. In order to verify the theoretical advantages of the improved A* algorithm, a series of two-dimensional figures of the robot task was presented in this paper. In addition, some comparative experiments in the virtual and real robot manipulator platform are performed to examine the improved A* algorithm. Experimental results show that the search success rate of the improved A* algorithm is higher than the original A* algorithm, along with a shorter and smoother path could be obtained by the improved A* algorithm. Therefore, the success rate of robot path planning and the optimal extent of the robot path are effectively improved by the improved A* algorithm.

Keywords Robot manipulator; Path planning; Improved A* algorithm; Path optimization

1 Introduction

At present, the technology of industrial robot is ubiquitous in automatic field. Path planning is a fundamental issue in robotics research for the improvement of the autonomy of industrial robots in its static environments for offline programming. It is crucial to implement and can ensure industrial robots on-site operations safely, efficiently and orderly. However, planning for

manipulators poses additional challenges because each link of the robot is effectively constrained by the position and orientation of the other links[1].

Motion planning problem for the robot has been widely discussed in literature. And the sampling-based methods are the current state-of-the-art in motion planning[1,2], the main idea about sampling-based methods is computing multiple collision-free point in the robot task space and connect them to construct a tree or graph, after that a search method will be used to find a path. So there are two main classes: graph based methods (e.g., Probabilistic Roadmaps[3], Neural networks[4]) and tree-based methods (e.g., Rapidly-Exploring Random Trees [5], Guided Sampling Tree [6]). Sánchez[3] presented a graph-based method, a PRM planner samples the configuration space at random and retains the collision-free points as milestones, and it connects pairs of milestones by simple paths and retains the collision-free ones as local paths, The milestones and local paths form the probabilistic roadmap. Łe□Gowski[4] presented a method for planning trajectories for given set of points in task space with use of graphs and neural networks, the shortest path is determined to define required nodes in joint space and the trajectories were smooth and accuracy was good. These methods have been successfully applied to many different application domains. Qureshi [5] proposed the Potential Function Based-RRT* (P-RRT*) that incorporates the Artificial Potential Field Algorithm in RRT*. The proposed algorithm allows a considerable decrease in the number of iterations and thus leads to more efficient memory utilization and an accelerated convergence rate. Plaku [6] presented a motion planner, termed Guided Sampling Tree (GUST). GUST expands a tree of collision-free and dynamically feasible motions and uses a workspace decomposition to partition the motion tree into groups. GUST relies on shortest path distances in the workspace decomposition and penalty factors to identify candidate groups, which could result in rapid expansions of the motion tree toward the goal..

The search algorithm is very important in the execution of motion planning in robot task space and joint space. In general the working space of a robot converts to the corresponding graphical representation firstly[7], and then a suitable path search algorithm will be adopted to obtain the required path. So the kind of the path search algorithm will determine whether the shortest and optimal path will be obtained as well as the speed of the whole search process[8]. Therefore, the selection of the path search algorithm plays a key role in the planning results and the execution efficiency.

The problem of path search has seen numerous methods and means that solve the prevalent research issues to varying extents. The traditional algorithms are the breadth-first search algorithm [9,10], the depth-first search algorithm [11,12] and the Dijkstra algorithm [13]. They belonged to the blind search algorithm, and the characteristics of the problems are not considered. The path is searched by the strategy set beforehand for any problem and the search process will not be optimized according to the characteristics of the problem. So the time of the search will be longer and the efficiency will be lower when the problem is a complex one. However, the A* algorithm [14-18] belongs to the heuristic search algorithm [19]. The heuristic information obtained according to the characteristics of the problem, which will guide the search in the optimal direction. Such as speed up the search process and improve the efficiency. Therefore, the heuristic search algorithm has attracted other experts and scholars. Mo [19] proposed a particle swarm

optimization (PSO) that is a classical heuristic search method combined with a biogeography-based optimization (BBO) to optimize the paths in path network obtained by approximate Voronoi boundary network (AVBN) modeling in a static environment. But they provided the new approach that will take a lot of time to build models and just was verified it's feasibility in simulation situations.

Among them, A* algorithm is widely been concerned and is somewhat different. Abele[20] proposed an approach to find the shortest path by solving the traveling salesman problem of a graph automatically generated using the workpiece in the Surface Tessellation Language (STL) data file, the optimization is based on the A* algorithm and path smoothing by deleting unnecessary points. Persson [14] presents a generalization of the classic A* algorithm to the domain of sampling-based motion-planning. The root assumptions of the A* algorithm are examined and reformulated in a manner that enables a direct use of the search strategy as the driving force behind the generation of new samples in a motion-graph. Trovato [15] proposed an improved differential A* algorithm for the path planning of the manipulator. Its reconfigurability and real-time capability were superior to that of the A* algorithm. However, this method has not been applied in the robot. Zuo [16] proposed a hierarchical path planning approach for mobile robot navigation. It combined the A* algorithm and the least squares policy iteration (LSPI) algorithm together to get better path planning results with a two-level structure. It is used to generate smooth paths under kinematics constraints of the robot. The simulation results have demonstrated the effectiveness of the proposed approach, and in the future, they will need to further prove the practicability of this method on a real mobile robot. Kala [17] proposed a Multi-Neuron Heuristic Search (MNHS) algorithm for solving the problem of path planning in a static environment. This algorithm is a modified A* algorithm that performs better than normal A* when heuristics are prone to sharp changes, but the algorithm again needs to be validated by using a physical robot and the physical paths and obstacles in their future endeavors.

In view of the shortcomings of the previous studies, this paper concentrate on researching the A* algorithm and a single industrial robot on-site operation in static environment as the study subject. The A* algorithm is an effective and direct method to search path in a static network. Nevertheless, those original A* algorithms are used to search in the neighborhood of the current node. Due to the limitation of the local path length, the one which is out of threshold length will be abandoned, and it will lead to the final path not be the optimal one. If the number of the sampling points is insufficient when the graph was built, the search success rate might be lower. In practice, increasing the number of sample points will make the cost more. Hence, an improved A* algorithm is introduced to enhance the optimal extent and the search success rate of the robot path. Since the path searched by A* algorithm is made up of multiple local paths, and the sampling

points are generated randomly, the path may not be the optimal one. Therefore, a post-processing stage is proposed to optimize the searched path. The proposed algorithms have a higher success rate of robot path planning and can get a more optimized robot path.

The rest of the paper is organized in five sections as follows: Section 2 presents a brief introduction of the original A*algorithm. An improved A* algorithm is proposed in section 3. Experimental results and detailed analysis are provided in Section 4. Finally, conclusions of the work are drawn in Section 5.

2 Brief review about original A* algorithm

The A* algorithm is an informed heuristic search algorithm, meaning that is solves problems by searching among all possible paths to the solution for the one that incurs the smallest cost. It visits nodes in a graph from the start node to the goal node. And the heuristic information related to the characteristics of the problem is utilized to guide its performance, so it is superior to other blind search algorithms. And as the heuristic algorithm, some key elements about the A* algorithm are OPEN list, CLOSED list and heuristic function $f(n)$ [14,17]. The OPEN list of the A* algorithm is established respectively to store the candidate nodes which will be detected then, and the CLOSED list is established to store the selected nodes of the final path. Accordance to the size of the evaluation value, the nodes in the neighborhood are arranged from lowest to highest into the OPEN list. The node with the minimum evaluation value is transferred to the CLOSED list. Then the transferred node is regarded as the current node and the above process is repeated until the target node is searched.

The evaluation function $f(n)$ is an estimate for the importance of the candidate node *n* in the path from the initial node *s* to the goal node *g* and it could offer a search order among candidate nodes for the next search. It could be calculated according to formula (1). It represents a cost of the path starting from the initial node *s*, going through the current node *n*, ending at the goal node *g*. It includes two parts: the actual cost $g(n)$ of the path from the initial node *s* to the current node *n* and the evaluation cost $h(n)$ of the optimal path from the current node *n* to the goal node *g*. The actual cost $g(n)$ is a known quantity, but the evaluation cost $h(n)$ is an unknown quantity, so the $h(n)$ plays a decisive role in the evaluation function $f(n)$.

The actual cost $g(n)$ could be calculated according to formula (2). It is the search depth of

the current node *n*. In other words, it is the number of nodes of the path from the initial node *s* to the current node *n*. The evaluation cost $h(n)$ is calculated according to formula (3). It is the Euclidean distance from the current node *n* to the goal node *g*. Then the practical needs could be met by the evaluation function $f(n)$.

$$f(n) = g(n) + h(n) \tag{1}$$

Where $f(n)$ is denoted the evaluation function of the current node *n*, $g(n)$ is denoted the actual cost of the path from the initial node *s* to the current node *n*, $h(n)$ is denoted the evaluation cost of the optimal path from the current node *n* to the goal node *g* (mm),

$$g(n) = d \tag{2}$$

Where *d* is denoted the search depth.

$$h(n) = \| p(g) - p(n) \| = \sqrt{(x_g - x_n)^2 + (y_g - y_n)^2 + (z_g - z_n)^2} \tag{3}$$

Where $p(g), p(n)$ are denoted respectively the position of the robot at the node *g* and the node *n*, x_g, y_g, z_g are denoted the coordinates of the goal node *g* (mm), x_n, y_n, z_n are denoted the coordinates of the current node *n* (mm).

In order to more clearly describe the search principle of the A* algorithm, a schematic diagram is given as shown in Fig. 1, the big red arrow indicate the search sequence. Interested readers may refer to references [14-16] for a more detailed description of this algorithm.

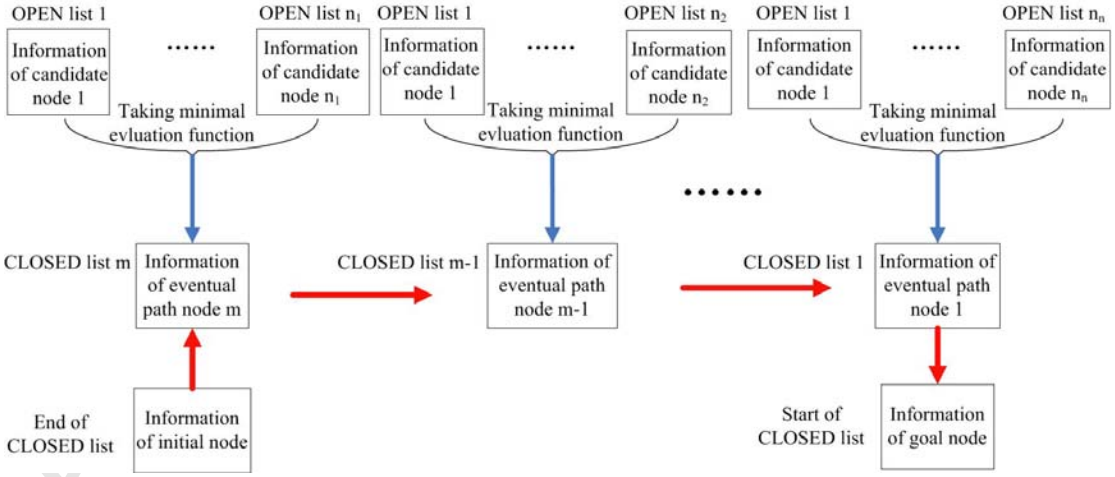


Fig. 1 The schematic diagram of the A* algorithm path search

However, for the robot path planning, there are two major drawbacks about the original A* algorithm. Firstly, if the number of candidate nodes is insufficient, it will lead path searching to

fail. Secondly, due to the candidate nodes are randomly distributed in robotic configuration space, the searched path between start point and end point is not a straight line, and actually, it is a tortuous path and it has room for improvement. Therefore, an improved A* algorithm is proposed to solve the inherent defects of the original A* algorithm, which describe in section 3.

3 The improved A* algorithm

Path planning based on the improved A* algorithm belongs to the query phase of the probabilistic roadmap planner[3,22]. The probabilistic roadmap planner consists of two phases: a preprocessing phase and a query phase. In the preprocessing phase, collisionless sampling points are generated randomly in robotic configuration space and detected by Hybrid Bounding Volume Hierarchy Tree algorithm [23] and also be called nodes during the query phase of the probabilistic roadmap planner, and then a local safety and collisionless path was built between the sampling points by local path planner. The local path planner will map the plans to robot joint space to check collision and determine whether the path is valid and can be execute by joint space constraint such as speed and acceleration、energy optimization constraints. So the probabilistic roadmap is constituted of collisionless sampling points and local safety paths. In the query phase, start point and end point of robotic movement are entered, the improved A* algorithm is adopted in the constituted probabilistic roadmap planner to search and get a safe path that connection the initial node s (start point) and the goal node g (end point). The proposed improved A* algorithm consists of two processing algorithms: the pre-processing stage and the post-processing stage.

3.1 The constraints in robot joint space

The joint space constraints refer to the angle range, joint velocity, acceleration constraint, velocity distortion constraint, and optimal inverse kinematics solution constraint. Those were used in building the probability map with the following steps:

Step1: A series of different robot six joint angle combinations are randomly generated within joint rotation angles, and collision detection is performed to eliminate the collision joint angle combinations.

Step2: Convert the generated joint angle values into the robot's task space by forward kinematics calculation. This method ensures that all generated points have inverse kinematics solutions.

Step3: Connect adjacent points with a straight line in its neighborhood of the generated point in task space. After that acceleration and deceleration planning and interpolation by are performed on the straight line, and than convert the interpolation points into robot joint space by inverse kinematics. And the optimal inverse kinematics solution with the smallest angle difference is selected among adjacent interpolation point.

Step4: The local paths without inverse kinematics solutions and speed distortions are eliminated, and a probability map is established under the constraints in robot joint space.

3.2 The pre-processing stage

Compare to the original A* algorithm which is described in Section 2, the pre-processing stage of the improved A* algorithm is that the local path between the current node and the goal node is planned before the next search in the neighborhood of the current node. If the local path is the collisionless one, it will be adopted directly and the subsequent operations will be stopped. Otherwise, the next step is search in the neighborhood of the current node as same as the steps of the original A* algorithm search.

To compare the pre-processing stage of the improved A* algorithm with the original A* algorithm, a simple situation in the two-dimensional space was given randomly as shown in Fig. 2. Regardless of the robot's specific position, orientation and actual size, the robot will be regarded as a particle. The dark blue irregular rectangle is the obstacles in the environment, the green dot is the initial node *s* and the yellow dot is the goal node *g*. Then the path is searched from the initial node *s* to the goal node *g* by the pre-processing stage of improved A* algorithm and the original A* algorithm respectively. Their search process is illustrated in Figs. 3-6. Finally, the advantages of the pre-processing stage of improved A* algorithm could be shown from the two-dimensional figures.

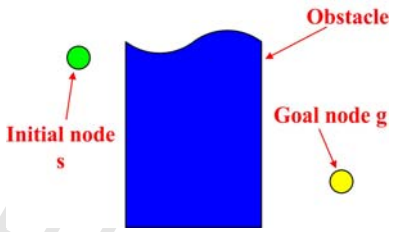


Fig. 2 A simple situation as the robot task

The paths searched by two algorithms for the first time are shown in Fig. 3. The path searched by the original A* algorithm is shown in Fig. 3(a). The path searched by the pre-processing stage of the improved A* algorithm is shown in Fig. 3(b). In Fig. 3, the neighborhood N_n is surrounded by the red dotted circle, the intermediate nodes n_i which has the minimum evaluation value $f(n_i)$ is searched by two algorithms are represented by the sky blue dots (they are marked as 1, 2, 3, 4 and 5 to distinguish each other), the feasible local paths searched by two algorithms are represented by black solid lines and the infeasible local paths searched by two algorithms are represented by black dotted lines.

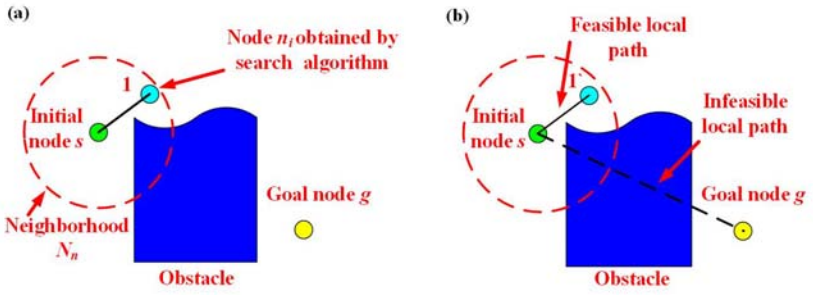


Fig. 3 Results of the first search (a) the original A* algorithm (b) the improved A* algorithm

As shown in Fig. 3(a), the next search will be executed directly in the neighborhood of the current node (here it was the initial node) when the original A* algorithm is adopted. The node with the minimum evaluation value will be the required one and it was represented by sky blue dot 1. But in Fig. 3(b), the local path between the current node and the goal node will be planned first when the improved A* algorithm is adopted. In fact, the local path represented by the black dotted line is collided with the obstacle, so it is the infeasible one, it will be abandoned and then the next search will be executed as same as the original A* search did. Like the first search,the local path between the current node and goal node is collided with the obstacle. So the results of the second and third search are the same by the two algorithms as shown in Fig. 4 and 5. Finally, the results of the next few steps searched by two algorithms are shown in Fig. 6. And Fig. 6(b) shows that in the fourth search, the local path between the current node (here it was the third searched node 3`) and the goal node is judged to be a feasible path. So that path will be adopted directly and then the search stopped, when the improved A* algorithm is adopted. But, it has shown in Fig. 6(a), the processes will not be stopped until the sixth search, when the original A* algorithm is adopted. As shown in Fig. 6, the final path obtained by the pre-processing stage of the improved A* algorithm is shorter than the one obtained by the original A* algorithm.

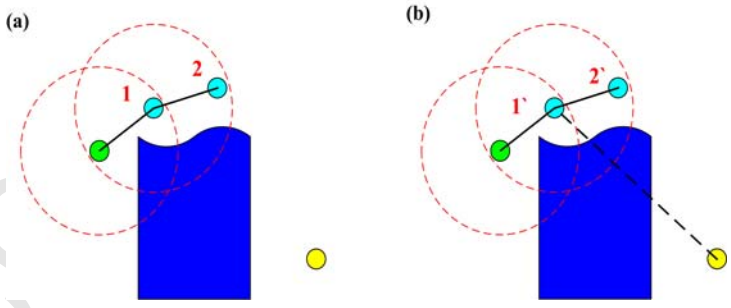


Fig. 4 Results of the second search (a) the original A* algorithm (b) the improved A* algorithm

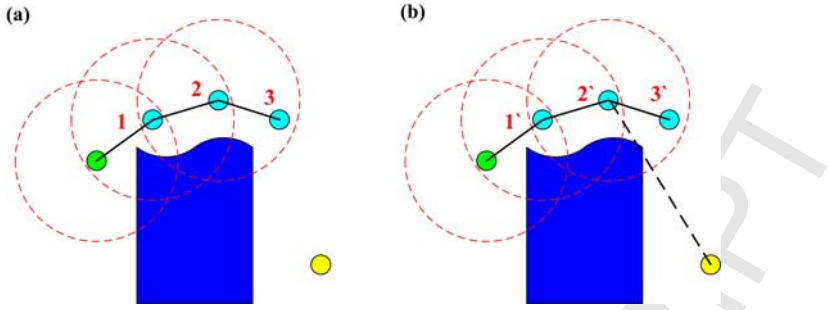


Fig. 5 Results of the third search (a) the original A* algorithm (b) the improved A* algorithm

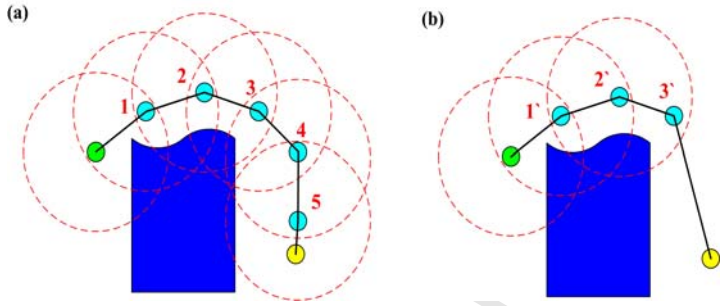


Fig. 6 Results of the last search (a) the original A* algorithm (b) the improved A* algorithm

The process of the pre-processing stage of the improved A* algorithm is shown in Fig. 7. The contents in the red dotted box in Fig. 7 are the improved part relative to the original A* algorithm. The detailed search processes of the pre-processing stage are as follows:

Step 1: Establish the OPEN list and the CLOSED list;

Step 2: Add the initial node *s* to the OPEN list;

Step 3: Find the node n_i which has the minimum evaluation value $f(n_i)$ in the OPEN list and then transfer it to the end of the CLOSED list. So it will be the footer node of the CLOSED list;

Step 4: Then the node n_i will be judged whether or not it is the goal node g; If the node n_i is the goal node g, the Step 14 will be executed; otherwise the Step 5 will be executed.

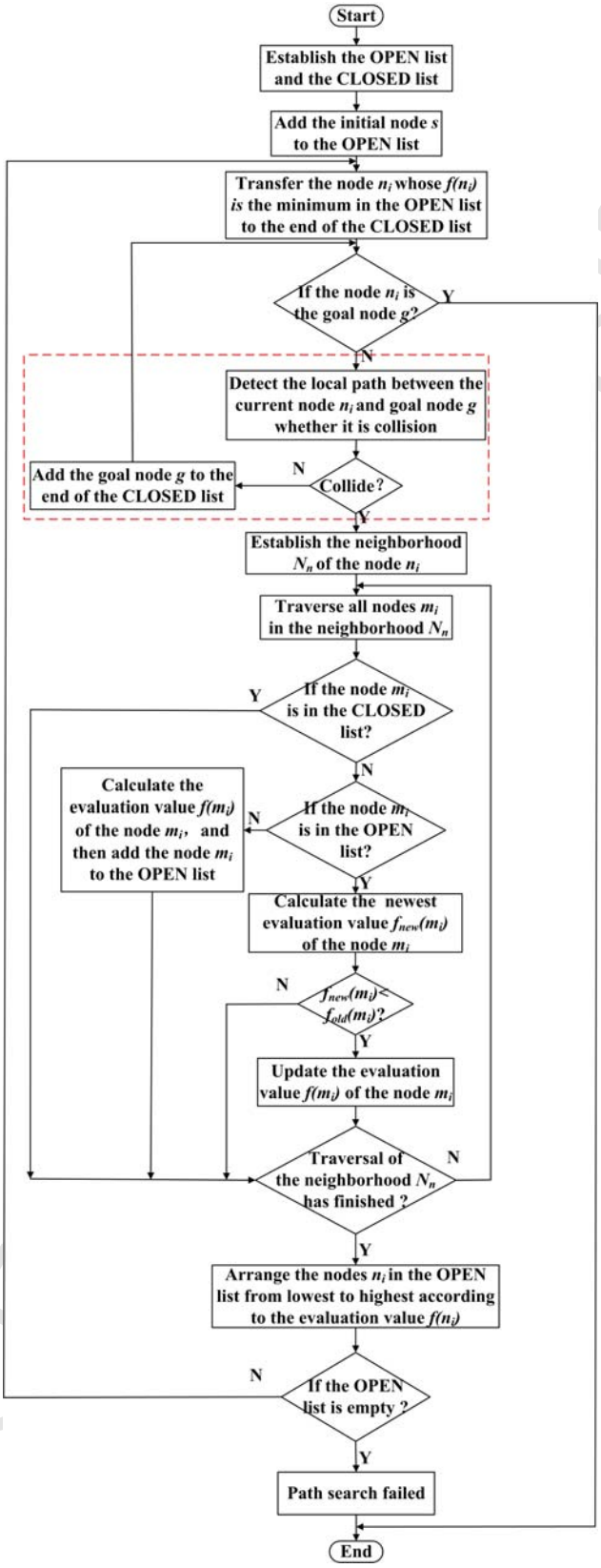


Fig. 7 Process of the pre-processing stage

Step 5: The local path between the node n_i and goal node g will be dispersed and detected whether it was collision by the Hybrid Bounding Volume Hierarchy Tree collision detection algorithm;

Step 6: The result of collision detection will decide the next step; if the collision occurred, the Step 7 will be executed; otherwise the goal node g will be added to the end of the CLOSED list and then the Step 14 will be executed.

Step 7: Establish the neighborhood N_n of node n_i ;

Step 8: Traverse all nodes in the neighborhood N_n ;

Step 9: If the node m_i in the neighborhood N_n is in the CLOSED list, execute Step 12; otherwise execute Step 10.

Step 10: Judge whether or not the node m_i was in the OPEN list; if the node m_i is not in the OPEN list, calculate the evaluation value $f(m_i)$. Then add the node m_i to the OPEN list and execute Step 12; otherwise calculate the latest evaluation value $f_{new}(m_i)$ and then execute Step 11.

Step 11: Judge whether or not the latest evaluation value $f_{new}(m_i)$ is less than the evaluation value $f_{old}(m_i)$ stored in the OPEN list previously; If the $f_{new}(m_i)$ is less than the $f_{old}(m_i)$, update the evaluation value $f(m_i)$ of the corresponding node and then execute the Step 12; otherwise execute Step 12 directly.

Step 12: If the traversal of the neighborhood N_n is finished, the nodes n_i in the OPEN list will be arranged from lowest to highest in accordance with the evaluation value $f(n_i)$ and then execute Step 13; otherwise back to execute Step 8.

Step 13: Judge whether or not the OPEN list is empty; if the OPEN list is empty, the result of the search will lost and it will be exported. Then all operations will be stopped; otherwise, back to execute Step 3.

Step 14: All nodes in the CLOSED list will be connected in a sequence.

3.3 The post-processing stage

Each search of the original A* algorithm is carried out in the neighborhood of the current node. Therefore, the neighbor threshold ρ is limited to the local path's length, the local paths out of range will be discarded, which leads to the path of the final search is not necessarily the optimal path. At the same time, when the initial node is far away from the goal node, the number of local paths in the final path will be larger due to the neighborhood threshold. And at the same time these nodes are randomly generated in robotic configuration space, so the robot will bend

several times along the final path, it will obviously reduce the efficiency of the robot movement, so it is necessary to use a post-processing stage to straighten the robot path to optimize the path.

In order to explain the implementation process of the post-processing stage more clearly, a schematic path in a two-dimensional space is used to describe the path obtained by the original A* algorithm (the specific position and actual size of robot are not consider and treat the robot as a particle as shown in Fig. 8, the dark blue graphics represent the obstacles in the environment, green dot represents the initial node p_1, yellow dot represents the goal node p_n, blue dot represents searched node p_j), and the post-processing stage is used optimize the schematic path . Figs. 9-11 show the main optimization process of the post-processing stage.

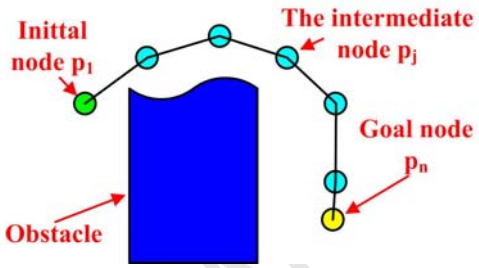


Fig. 8 The obtained path by the original A* algorithm

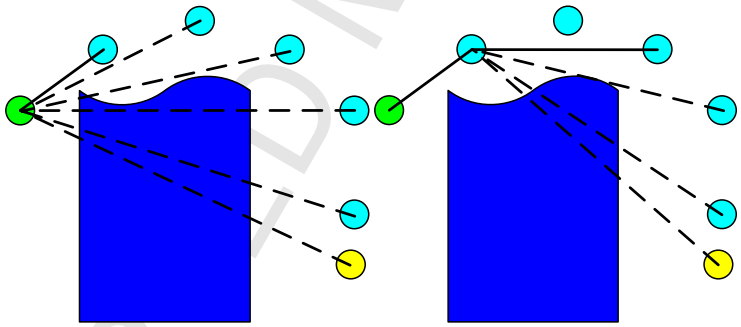


Fig. 9 Results of the optimization (a)first step,(b)second step

Fig. 9(a) and 9(b) shows the results of the first and second optimization steps. Firstly, the path is optimized by the local path planner between the initial node and the searched node p_i, and then determined whether these local paths will cause a collision to occur. If the local path is a feasible path and node's number is the maximum number, the local path is adopted; in addition to the initial node, the intermediate node before the current node is deleted. And use this current node as the new initial node. Otherwise, the local path does not change and the next node of initial node as the new initial node and search local path between last nodes. The above steps are then repeated in order until shortest path with least node is found.

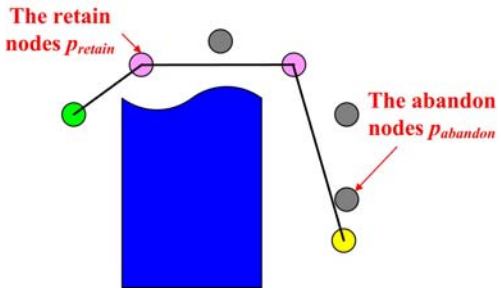


Fig. 10 Results of the last step optimization

Fig. 10 shows the result of the final optimization of the path, where the pink dots represent the nodes p_{retain} after optimization, and the dark gray dots represent the $p_{abandon}$ nodes that have been discarded after optimization. In this case, the optimized path contains only three local paths, whereas the original path contains six local paths as shown in Fig. 8, and the optimized path is significantly shorter than the original path, so the post-processing stage can effectively improve the final degree of optimization of the path.

The specific process of path optimization algorithm is shown in Fig. 11. The post-processing stage process can be divided into the following steps:

Step 1: Assign 1 to *i*, and assign *n-1* to *j*, where *i* and *j* are the subscript value of the node. Path optimization is from the head node p_1 and the termination node p_n of the CLOSED list;

Step 2: Establish all the local paths between the node p_i and p_j, and value *j* is larger than *i+1* and smaller than *n-1*;

Step 3: The local optimal paths are discretized by the local path planner and detected them by the collision detection algorithm. If all of these local paths can cause a collision occur to perform step 4. Otherwise go to step 5.

Step 4: Save the local path between node p_i to p_{i+1}, and then *i* plus 1;

Step 5: Save the local path which one contains node p_j and *j* is largest, and abandon the node *i+1* to *j-1*, and then assign *j* to *i*;

Step 6: Judge whether or not *i* and *n-1* are equal;

If the answer is yes, then execute step 7; otherwise, return to execute step 2.

Step 7: The previously saved locally optimization paths are connected sequentially as the global optimal path.

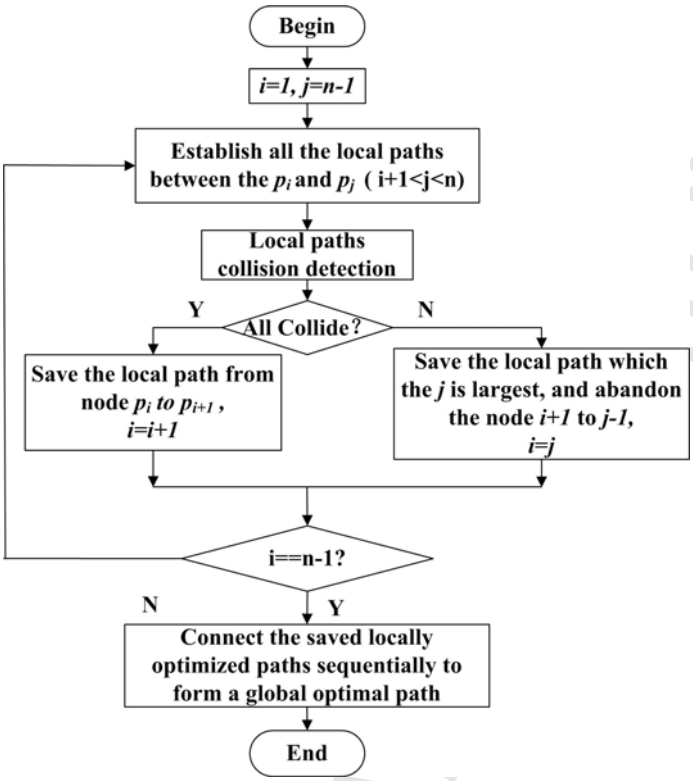


Fig. 11 Process of the path optimization algorithm

4 Experimental results and discussion

To validate the performances of the proposed approach, a six degrees of freedom revolute type robot manipulator (Motoman UP6) hardware platform and ambience is built independently as shown in Fig. 12. Fig. 13 shows the virtual simulation environment platform is modeled at the proportion of 1:1 relative to the actual environment in Fig.12. Its experimental subject was the industrial transport operation and the experiment will be performed both in the actual and virtual environment. One obstacle and three storages are placed on the surrounding ground of the robot respectively. Two obstacles are hanged above the storage 1 and storage 2 respectively. There is a workpiece put on the storage 1. There is a groove in the vertical direction of the storage 2, and a groove in the horizontal direction of the storage 3 too.

4.1 Evaluation about search success rate

In scene of Fig. 12, the preprocessing phase of probabilistic roadmap planner was finished and that the sampling points are randomly distributed in the robotic configuration space. The success rates of search paths are calculated, when the sampling points from less to more, to prove the advantages of the improved A* algorithm than the original A* algorithm. Ten groups consist of

the start joint angles and the end joint angles of the robot are provided and without repetition each other. The ten groups' detailed value of the start joint angles and the end joint angles are shown in Table 1.

Fig. 12 The actual experimental environment of robot

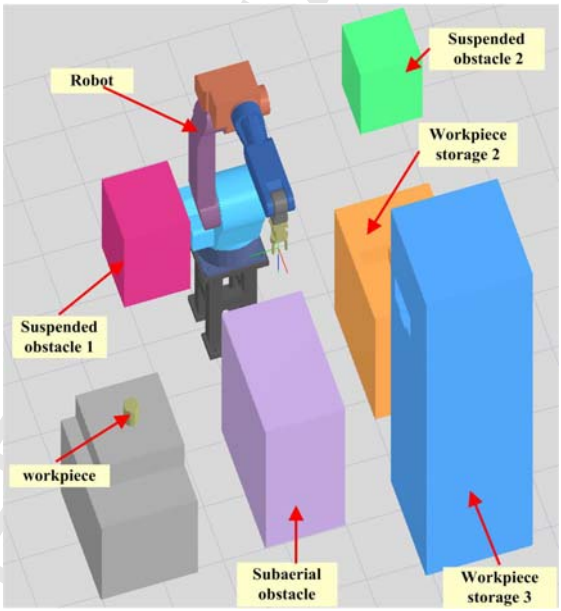


Fig. 13 The simulation experimental environment of robot

The paths of the above ten groups will be searched respectively when the number of the sampling points of the probabilistic roadmap planner is from 100 to 1500. Result of search path by the two algorithms is that some of them are successful search and some others are not when the sampling points less than a certain value. And success rate of search path by improved A* algorithm is always higher than the original A* algorithm. Their search success rates are shown in Table 2.

Table 1 The start point and the end point of ten groups

Test group	Start point (°)						End point (°)					
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	-41	-49	2	-7	47	-41	56	-78	45	-7	33	56
2	-41	-49	2	-7	47	-41	7	-43	-24	92	83	-22
3	56	-78	45	-7	33	56	7	-43	-24	92	83	-22
4	0	-90	0	0	90	0	-41	-49	2	-7	47	-41
5	0	-90	0	0	90	0	56	-78	45	-7	33	56
6	0	-90	0	0	90	0	7	-43	-24	92	83	-22
7	-16	-38	-34	-100	61	-23	-41	-49	2	-7	47	-41
8	-16	-38	-34	-100	61	-23	56	-78	45	-7	33	56
9	10	-49	-53	-7	102	10	56	-78	45	-7	33	56
10	10	-49	-53	-7	102	10	-41	-49	2	-7	47	-41

The results of the Table 2 showed that when the number of sampling points were 100, 200 and 300, the search success rate of the original A* algorithm were both 0%, but the search success rate of the improved A* algorithm are 80%, 90% and 90%, respectively. So the initial searches' success rate of the improved A* algorithm are higher than the original A* algorithm. When the number of sampling points are 400 to 700, all the searches' success rate of the improved A* algorithm are still higher than that of the original A* algorithm. When the number of sampling points is 800, the success rate of the improved A* algorithm has been 100%, but the success rate of the original A* algorithm is just 70%. Only when the number of sampling points was 1500, the success rate of the original A* algorithm is 100%. So it could be seen that the success rate of the improved A* algorithm will be higher than that of the original A* algorithm. In other words, the search success rate of the improved A* algorithm is much higher than the original A* algorithm with limited sampling points. To be sure, above of the corresponding relationship of the number of sampling points and search success rate are got in a given environment.

Table 2 The search success rate for two algorithms when the number of sampling points was from 100 to 1500

Number of sampling points	Search success rate of original A* algorithm	Search success rate of improved A* algorithm
100	0%	80%
200	0%	90%
300	0%	90%
400	10%	90%
500	10%	90%
600	20%	90%
700	20%	90%
800	70%	100%
900	70%	100%
1000	70%	100%
1100	70%	100%
1200	70%	100%

1300	70%	100%
1400	70%	100%
1500	100%	100%

4.2 Evaluation about length of obtained path

The robot path is planned under the following operating condition (Fig. 15). Table 2 shows that when the number of sampling points is 1500, the success rates of the two algorithms are both 100%. The ten groups in table 1 can all find a path by the two algorithms and the lengths of paths are show in Fig. 14.

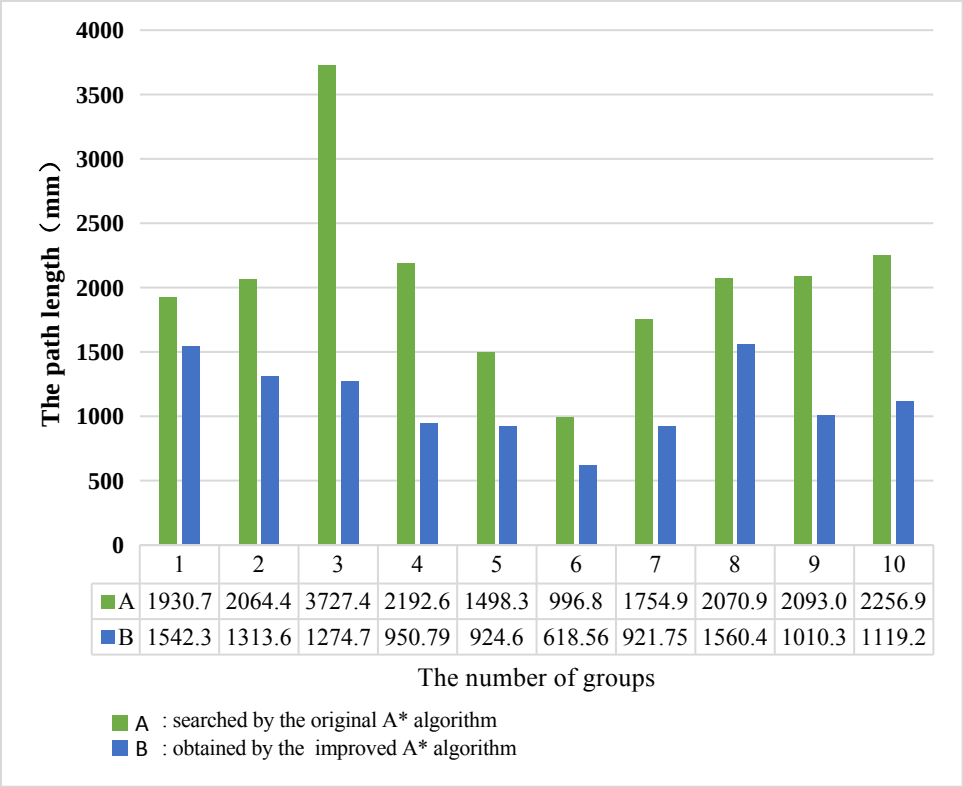


Fig. 14 The path length of the above ten groups obtained by the above algorithms (sampling points is 1500)

From Fig. 14, the all path lengths obtained by the improved A* algorithm is shorter than by the original A* algorithm. The experimental result indicated that the obtained paths by the improved A* algorithm are more optimized.

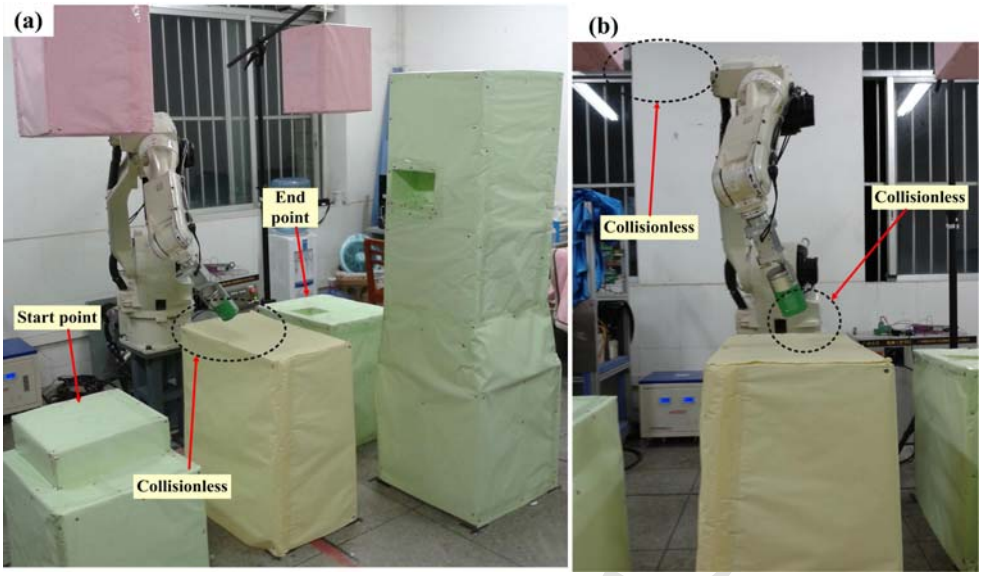


Fig. 15 The movement process with the improved A* algorithm in the actual experimental environment

(a) scene of actual movement process (b) partial view

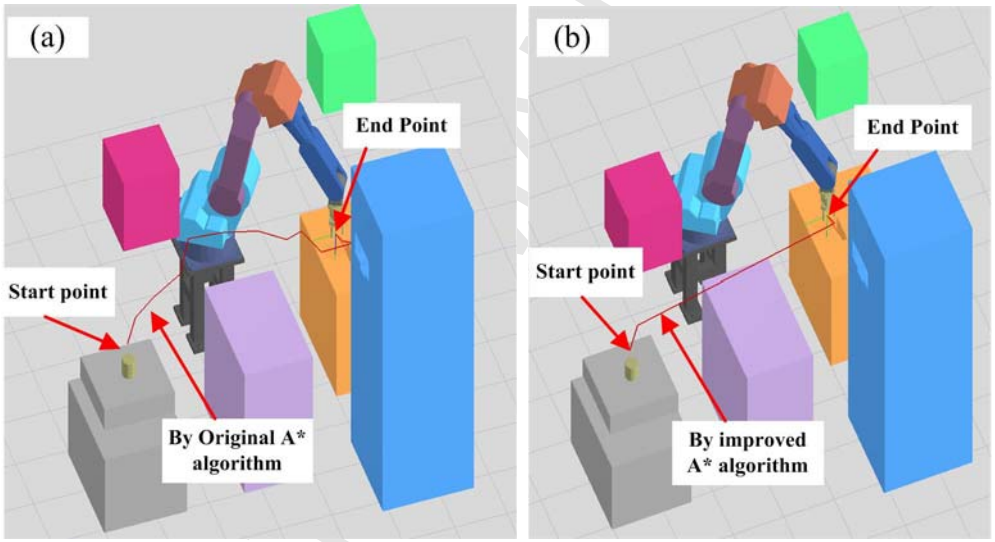
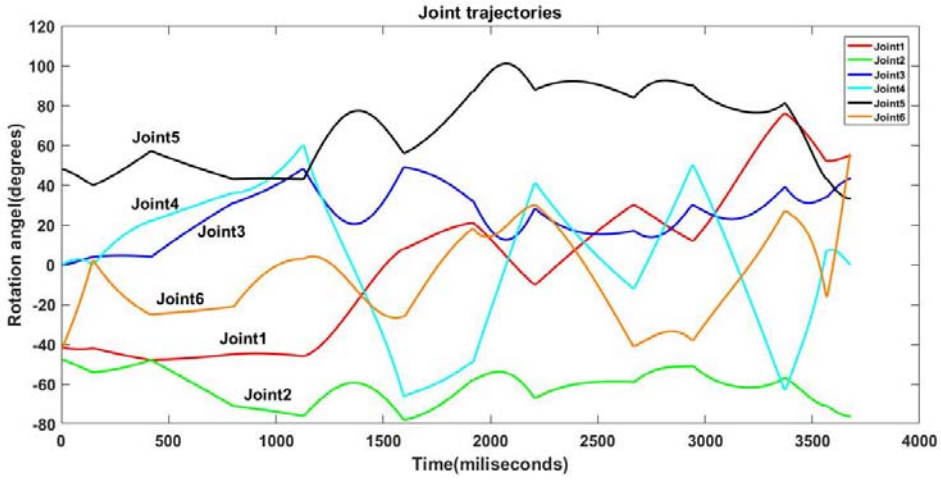
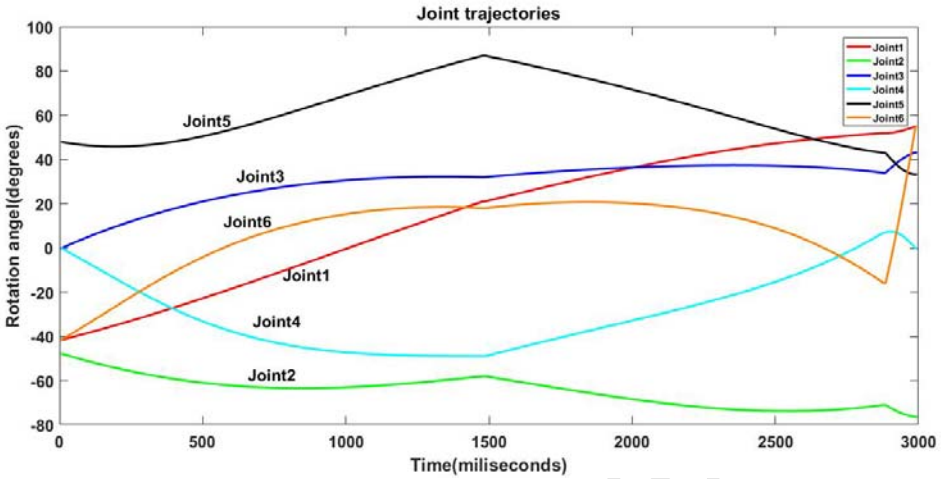


Fig. 16 The obtained path (Test group 1 and sampling points is 1500)

(a) Searched by original A* algorithm (b) searched by improved A* algorithm



(a)Joint trajectories obtained by Original A* algorithm



(b) Joint trajectories obtained by improved A* algorithm

Fig. 17 The obtained path converts to joint space (Test group 1 and sampling points is 1500)

Fig. 15 shows the scene of actual movement process by the improved A* algorithm on the independently building robot hardware platform. The obtained path of test group 1 are shown in the Fig. 16(a) and 16(b), the path searched by the improved A* algorithm is obviously more smooth and the number of local paths are less than the one searched by the original A* algorithm. At the same time, results in Fig.18 showed the path searched by the improved A* algorithm are 20% to 66% shorter than the one searched by the original A* algorithm, the reason is that the obtained path by the improved A* algorithm is much less twists and turns than that by the original A* algorithm. Fig. 17 showed the path get by the improved A* algorithm convert to robot joint space is more smooth than the original A* algorithm.

4.4 Discussions

In Section 4.1, the cause of experimental result was that when the number of sampling nodes is insufficient, the probabilistic roadmap planner will be not a connected graph. Many areas in the graph are disconnected with each other, so the search will be failure easily with the original A* algorithm. But when the improved A* algorithm is adopted to search the path, the local path between the current node and the goal node will be judged whether or not it is feasible. If the local path is feasible, it will be adopted directly. So even if the area including the current node is disconnected with the area that including the goal node, they can be connected with each other compulsorily by the above-mentioned way. Therefore, the search success rate of the improved A* algorithm could be higher than the original A* algorithm. Therefore, it can be seen that the search success rate of the robot path planning can be raised effectively by the improved A* algorithm.

In Section 4.2, the sampling nodes are generated randomly, and in the way of original A*

algorithm to search path, local paths are connected between the randomly sampling nodes, the obtained path consisted of many local paths. Therefore, the obtained path is a zigzag path. The improved A* algorithm is applied to search path, so some local paths are not connected any more, and in the most direct way to connect to the goal node. The local path between the current node and the goal node will be judged whether it is feasible. If the local path is feasible, it will be adopted directly. Otherwise, it will adopt original A* algorithm to search path. Finally, the post-processing stage is applied to remove some intermediate nodes to straighten the robot path. So the obtained path with improved A* algorithm is relatively smooth and it has a shorter length.

The proposed method can work well in robot path planning. It has been pointed out that the improved A* algorithm has been a strong performer in search success rate and the total length of the obtained path compared with the original A* algorithm. Therefore, the experimental results showed that the improved A* algorithm could effectively enhance the optimal extent of the robot path.

5 Conclusions

In this paper, an improved A* algorithm is proposed to overcome inherent drawbacks of the original A* algorithm for the robot path planning. Randomly sampling nodes lead to the eventual path is a zigzag one and the process of path search is easy to fail. The theoretical comparison preliminary verifies the feasibility of the proposed methods, and the experiment of real environment and simulation environment proves its availability and reliability further. By utilizing the improved A* algorithm, before each path search is executed in accordance with the process of the original A* algorithm, the local path between current node and goal node will be detected. If it is the collisionless path, it will be adopted directly. Otherwise the following processes are the same as the original A* algorithm. A post-processing stage is proposed to shorten the length of the path by removing some intermediate nodes to straighten the local path.

Experimental results illustrate that the search success rate of the improved A* algorithm is obviously higher than the original A* algorithm, according to the number of the sampling points is insufficient. Moreover, under the same experimental conditions, the length of searched paths with the improved A* algorithm are obviously shorter than those searched by the original A* algorithm. Overall, the improved A* algorithm is superior to the original A* algorithm on search success rate and the length of the obtained path. This method shows promising results and it’s more applicable for robot path planning in a known environment.

Future work will concern the path planning of multiple industry robots, the robots share the workspace, the robot is the move object to each other and change the situation to dynamic

environment. In the follow-up study is needed to handle those situations to get high search success rate, time-optimal smooth short path. After the path planning is completed, the trajectory optimization strategy needs to be further optimized for the minimization of the cycle time.

Acknowledgments We gratefully acknowledge the support of National Natural Science Foundation of China (Grant NO. 51465005) , Scientific and Technological Research Foundation of Guangxi (Grant NO. 1598008-21), and the Dean Project of Guangxi Colleges and Universities Key Laboratory of Modern Design and Advanced Manufacturing. We will also like to thank the reviewers for their constructive and inspiring comments and suggestions on our research.

References

[1] S. Pellegrinelli, N. Pedrocchi, L.M. Tosatti, et al., Multi-robot spot-welding cells for car-body assembly: Design and motion planning, Robotics and Computer-Integrated Manufacturing. 44(2017)97-116.

[2] L. Janson, B. Ichter, M. Pavone. Deterministic Sampling-Based Motion Planning: Optimality, Complexity, and Performance. Computer Science, 37(3)(2018).

[3] G. Sánchez, J.C. Latombe. On Delaying Collision Checking in PRM Planning: Application to Multi-Robot Coordination, International Journal of Robotics Research. (2002)21(1):5-26.

[4] A. Łe□gowski, M. Niezabitowski, An industrial robot singular trajectories planning based on graphs and neural networks, In: American Institute of Physics Conference Series. AIP Publishing LLC. 2016, pp.97-105.

[5] A.H. Qureshi, Y. Ayaz, Potential functions based sampling heuristic for optimal path planning, Autonomous Robots. 40(6)(2017)1079-1093.

[6] Plaku E. Region-Guided and Sampling-Based Tree Search for Motion Planning With Dynamics, IEEE Transactions on Robotics. 31(3)(2017)723-735.

[7] L. Johannsmeier, S. Haddadin, A. hierarchical human-robot interaction-planning framework for Task Allocation in Collaborative Industrial Assembly Processes, IEEE Robotics & Automation Letters. 2(1)(2017)41-48.

[8] L. Larsen , J. Kim , M. Kupke, et al., Automatic path planning of industrial robots comparing sampling-based and computational intelligence methods, Procedia Manufacturing. (11)(2017)241-248.

[9] J. Silvela, J. Portillo, Breadth-first search and its application to image processing problems, IEEE Trans. Image Processing. 10(8) (2001) 1194-1199.

[10] B. Awerbuch, R.G. Gallager, A new distributed algorithm to find breadth first search trees, IEEE Trans. Information Theory. 33(3) (1987) 315-322.

[11] R. Tarjan, Depth-first search and linear graph algorithms, SIAM J. Comput. 1(2) (1972) 146-160.

[12] Z. Chen, C. He, Z. He, et al. BD-ADOPT: a hybrid DCOP algorithm with best-first and depth-first search strategies, Artificial Intelligence Review. 5439(2017)1-39.

[13] J.L. Solka, J.C. Perry, B.R. Poellinger, G.W. Rogers, Fast computation of optimal paths using a parallel Dijkstra algorithm with embedded constraints, Neurocomputing. 8 (1995) 195-212.

[14] S.M. Persson, I. Sharf. Sampling-based A* algorithm for robot path-planning, International Journal of Robotics Research. 33(13)(2014)1683-1708.

[15] K.I. Trovato, L. Dorst, Differential A*, IEEE Trans. Knowledge and Data Engineering. 14(6) (2002) 1218-1229.

[16] L. Zuo, Q. Guo, X. Xu, H. Fu, A hierarchical path planning approach based on A* and least-squares policy iteration for mobile robots, Neurocomputing. 170 (2015) 257-266.

[17] R. Kala, A. Shukla, R. Tiwari, Robotic path planning in static environment using hierarchical multi-neuron heuristic search and probability based fitness, Neurocomputing. 74 (2011) 2314-2335.

[18] A.K. Guruji, H. Agarwal, D.K. Parsediya, Time-efficient A* algorithm for robot path planning, Procedia Technology. (23)(2016)144-149.

[19] H.W. Mo, L.F. Xu, Research of biogeography particle swarm optimization for robot path planning, Neurocomputing. 148 (2015) 91-99.

[20] E. Abele, F. Haehn, M. Pischan, et al., Time optimal path planning for industrial robots using STL data files, Procedia Cirp. (55)(2016)6-11.

[21] X. Yu, Y. Zhao, C. Wang, et al. Trajectory planning for robot manipulators considering kinematic constraints using probabilistic roadmap approach, Journal of Dynamic Systems Measurement & Control. 139(2)2017.

[22] C.M. Clark, Probabilistic Road Map sampling strategies for multi-robot motion planning, Robotics and Autonomous Systems. 53 (2005) 244-264.

[23] L. Chen, J. Dai, J.J. Feng, et al., Hybrid Bounding Volume Hierarchy Tree (HBVHT) Algorithm for Collision Detection of Articulated Model Robot, Information Technology Journal. 13 (10) (2014) 1723-1729.

Author Biographies

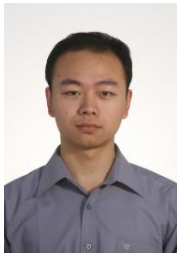
Bing Fu received the M.E. degree in mechatronic engineering from Guangxi University, Nanning, China, in 2013. He is currently an engineer in Guangxi University. His research interests in kinematics and dynamics modeling, motion plan and control of industrial robot manipulators.
Lin Chen received the Ph.D. degree from Huazhong University of Science and Technology, Wuhan, China, in 2008. She is currently a Professor of College of Mechanical Engineering, Guangxi University. Her research interests include numerical control, virtual reality, electromechanical detection and control technology use in rehabilitation robot and industrial robot manipulators, and the battery management system in electric vehicles.

Yuntao Zhou is a graduate student major in mechatronic engineering in Guangxi University, Nanning, China. His research mainly focuses on off-line programming, virtual simulation and cooperative control of multirobots.

Dong Zheng received the M.E. degree in computer technology from Guangxi University, Nanning, China, in 2016. He is currently an assistant engineer in Guangxi University. His research mainly focuses on off-line programming, virtual simulation of industrial robot manipulators.
Zhiqi Wei received the M.E. degree in mechatronic engineering from Guangxi University, Nanning, China, in 2017. His research mainly focuses on off-line programming, virtual simulation and cooperative control of multirobots.

Jun Dai received the M.E. degree in mechatronic engineering from Guangxi University, Nanning, China, in 2015. He is currently working toward the Ph.D. degree in Central South University. His research mainly focuses on motion control and trajectory planning of mobile robot.
Haihong Pan received the M.E. degree in mechatronic engineering from Guangxi University, Nanning, China, in 1991. And the Ph.D. degree in mechatronic engineering from Huazhong University of Science and Technology, Wuhan, China, in 2007. He is currently a Professor and the Vice President of College of Mechanical Engineering, Guangxi University. His research interests include numerical control, virtual reality, electromechanical detection and control technology use in rehabilitation robot and industrial robot manipulators, and intelligent manufacturing.

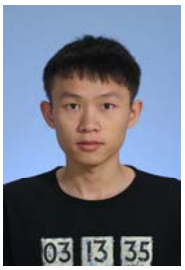
Author Photos



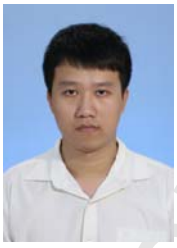
Bing Fu

Lin Chen

Yuntao Zhou



Dong Zheng



Zhiqi Wei

Jun Dai

Haihong Pan

1. Focus on path planning for industrial robots in complex environments.
2. A local safety and collisionless path was built between the sampling points by local path planner.
3. The local path to the goal node is planned before the next search in the neighborhood of the current node.
4. A post-processing stage used to optimize the resulting path.
5. Path planning problem is solved by the improved A* algorithm with high search success rate and short length.