Robotica (2024), **42**, pp. 457–481
doi:10.1017/S0263574723001546

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Obstacle avoidance path planning of 6-DOF robotic arm based on improved A* algorithm and artificial potential field method

Xianxing Tang[1,2], Haibo Zhou[1,2] and Tianying Xu[1,2]

[1]School of Mechanical and Electrical Engineering, Central South University, Changsha, Hunan, China and [2]State Key Laboratory of High Performance Complex Manufacturing, Central South University, Changsha, Hunan, China
Corresponding author: Haibo Zhou; Email: zhouhaibo@csu.edu.cn

Received: 17 May 2023; **Revised:** 19 October 2023; **Accepted:** 22 October 2023; **First published online:** 29 November 2023

Keywords: 6-DOF robotic arm; obstacle avoidance; path optimization; artificial potential field method

Abstract
Most studies on path planning of robotic arm focus on obstacle avoidance at the end position of robotic arm, while ignoring the obstacle avoidance of robotic arm joint linkage, and the obstacle avoidance method has low flexibility and adaptability. This paper proposes a path obstacle avoidance algorithm for the overall 6-DOF robotic arm that is based on the improved A* algorithm and the artificial potential field method. In the first place, an improved A* algorithm is proposed to address the deficiencies of the conventional A* algorithm, such as a large number of search nodes and low computational efficiency, in robotic arm end path planning. The enhanced A* algorithm proposes a new node search strategy and local path optimization method, which significantly reduces the number of search nodes and enhances search efficiency. To achieve the manipulator joint rod avoiding obstacles, a method of robotic arm posture adjustment based on the artificial potential field method is proposed. The efficiency and environmental adaptability of the robotic arm path planning algorithm proposed in this paper are validated through three types of simulation analysis conducted in different environments. Finally, the AUBO-i10 robotic arm is used to conduct path avoidance tests. Experimental results demonstrate that the proposed method can make the manipulator move smoothly and effectively plan an obstacle-free path, proving the method's viability.

1. Introduction

Collision avoidance path planning is a fundamental technology in robotics and the foundation for robotic arm to complete complex work goals [1–3]. In recent years, a large number of heuristic algorithms [4–6] such as the genetic algorithm, neural network algorithm, particle swarm algorithm, and A* algorithm have been implemented. The genetic algorithm is an algorithm based on the evolution of biological populations that are widely used in path planning problems [7, 8] due to its excellent real-time performance and global search capability. However, the genetic algorithm suffers from a slow convergence rate and a propensity to settle on local optimal solutions [9, 10]. The particle swarm algorithm is an optimization algorithm that simulates the flight of birds, with the benefits of fast convergence speed and simple implementation [11, 12]. However, the particle swarm algorithm is prone to premature and inaccurate convergence [13, 14]. Neural network algorithm is an algorithm that mimics animal neural networks for distributed parallel information processing, which has the advantage of strong learning ability and robustness [15]. Nevertheless, the neural network algorithm has complex parameters, long running time, and slow convergence speed [16, 17]. The artificial potential field method proposed by Khatib is also utilized extensively in the field of obstacle avoidance in mobile robots and manipulators [18]. Ge [19] proposed a new potential field method to apply mobile robots to path planning in dynamic environments. However, the artificial potential field method is prone to local optimization, making it challenging to apply broadly to the joint obstacle avoidance of multi-degree-of-freedom robotic arms. Hart [20] proposed the A* algorithm in 1968 by designing a heuristic function, which has been widely utilized in path planning due to its low complexity, high search efficiency, and global optimality. Anshika [21] proposed a modified A* algorithm applied to path planning for multi-robot systems that achieves the shortest route with the least amount of energy and generates the smoothest paths. Ren [22] proposed an A* algorithm based on the combination of the static weight method and jump point search, which decreases the number of visited nodes and improves the search efficiency. Guruji [23] proposes an improved A* algorithm to determine the heuristic function before the collision phase, thereby reducing the search time enhancing the effectiveness of path planning. Li [24] incorporated the two-way alternating classification search strategy into the A* algorithm, which makes mobile robot path planning more efficient and smoother than the conventional A* algorithm. Zuo [25] proposed a hierarchical path planning method combining the A* algorithm and the least squares policy iteration algorithm for mobile robot navigation in complex environments. The algorithm suffers from computational complexity and low environmental adaptability. Wang et al. [26] proposed an A* algorithm with variable-step segment search, which can guarantee that the intermediate point is the optimal path. This algorithm is applied in obstacle avoidance path planning for a six-degree-of-freedom robotic arm, but it is less suitable for environments with complex obstacles. Bing et al. [27] proposed a local path planning method that applies the A* algorithm, which first reduces the local path length by straightening the local path to achieve collision-free path planning for industrial robotic arms, but the algorithm does not take the obstacle avoidance method of the robotic arm linkage into account.

In addition to the need for a comprehensive approach to obstacle avoidance that takes into account the robotic arms' end position and robotic arm linkage, the methods proposed in the literature [22–27] frequently only improve the path search efficiency and path smoothing of the A* algorithm. For the end position and joint overall obstacle avoidance problem of a robotic arm, the majority of studies employ path planning algorithms for obstacle avoidance [28–30]. However, these research methods are typically employed in simple obstacle avoidance environments, and only a few joint motions are considered to reduce the algorithmic complexity. In Section 4.2, a comparison between the approach proposed in this paper and the aforementioned concepts will be presented. The primary contribution of this paper is to propose a path planning method for overall 6-DOF robotic arms for obstacle avoidance in 3D environment based on an improved A* algorithm and artificial potential field method. First, an enhanced A* algorithm is proposed for robotic arm end path obstacle avoidance, followed by the development of a node collision detection method. The enhanced A* algorithm redefines the node search direction and proposes a local path optimization technique. Finally, the path nodes are smoothed by the cubic spline B-curve to enable the robotic arm to achieve continuous smooth path planning in the obstacle avoidance process. Notably, this paper is a continuation of previous research [1]. This method does not account for the obstacle avoidance of the robotic arm linkage. In this paper, the authors consider both end trajectory obstacle avoidance and robotic arm rod obstacle avoidance. Our main objective is to enable efficient path planning of a 6-DOF robotic arm in 3D environment to meet its obstacle avoidance requirements in certain motion environments.

The structure of the article is as follows: In Section 2, an enhanced A* algorithm for robotic arm end path planning is proposed. Based on this, a joint rod obstacle avoidance strategy for a 6-DOF robotic arm is designed in Section 3. In Section 4, simulations and experimentation are performed in detail. The final section provides a summary of the paper and its conclusions.

2. Improved A* algorithm for robotic arm end path planning

2.1. Traditional A algorithm*

The A* algorithm is a heuristic global optimal path planning algorithm that enables efficient path planning in an obstacle-aware environment [31]. The direction of the A* algorithm's path search is determined by the cost function. In each round of path search, the cost function value of each child node in the vicinity of the parent node is calculated, and the child node with the smallest cost function value

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

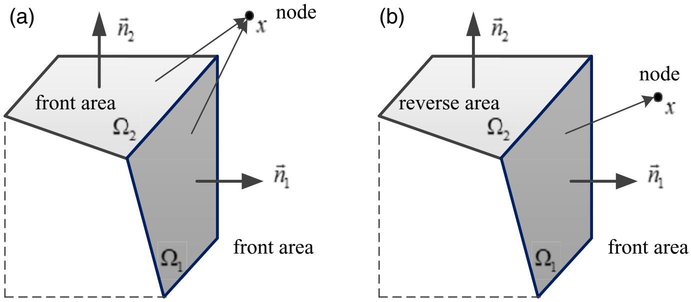

Figure 1. *Discrimination of obstacle planes.*

is chosen as the parent node in the next round. The final path result is then searched in this cycle. The cost function is typically expressed as follows:

$$F(n) = G(n) + H(n) \tag{1}$$

where $G(n)$ is the current path cost, which represents the cost of moving from the starting point to the child node, and $H(n)$ is the heuristic function known as the estimated cost, which represents the cost of moving from the child node to the target point.

However, if the traditional A* algorithm is directly applied to the path planning of the robotic arm in 3D environment, the following three issues will arise: (1) The A* algorithm's calculation speed in 3D environment will be drastically reduced; (2) when a node encounters obstacles, the A* algorithm's search efficiency is drastically reduced; (3) the traditional A* algorithm treats the moving subject as a point and only considers whether the moving point collides with environmental obstacles. However, the position relationship between the robotic arm's rod, the end position of the robotic arm, and the obstacles must be carefully considered when planning the robotic arm's path.

Due to the above-mentioned shortcomings of the traditional A* algorithm in practical applications, this paper proposes an enhanced A* algorithm to apply the motion planning of the robotic arm to the known obstacle environment model. The enhanced A* algorithm first proposes a node collision detection method and then improves the efficiency of obstacle avoidance by refining the node search direction and enhancing the local path optimization.

2.2. Path nodes collision detection

An important factor for ensuring that nodes can search the environment without colliding with obstacles is the collision detection [32, 33]. In this paper, all environmental obstacles are viewed as convex polyhedrons that undergo a particular expansion process, and the proposed algorithm can perceive the obstacles after the expansion. Since the search step size is relatively small compared to the polyhedral size of the obstacle, the essence of node and obstacle collision detection is to determine whether the node is located within this polyhedron. The plane Ω_j of the convex polyhedron Ω is arbitrarily extracted; the outward normal vector of the extracted plane Ω_j is the vector $\overrightarrow{n}$, and the point p belongs to the plane Ω_j. The subsequent definitions are provided first:

$$T\left(\Omega_j, x\right) = \overrightarrow{n} \cdot (x - p) \tag{2}$$

where x is the coordinate position of the current node.

If $T > 0$, it is defined that the node x is located in the front area of the plane Ω_j. If $T = 0$, it is defined that point x is located on the plane Ω_j. If $T < 0$, it is defined that point x is located in the reverse area of the plane Ω_j. For a clear illustration of the above definition, see Fig. 1. The node x is located in the front area of the plane Ω_1 and plane Ω_2 in Fig. 1(a), and the node x is located in the front area of the plane Ω_1 and the reverse area of plane Ω_2 in Fig. 1(b).

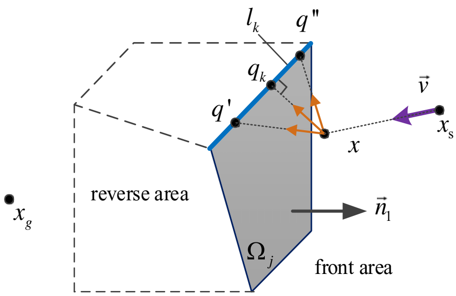

Figure 2. *Schematic diagram of node search.*

If for any plane Ω_i of the convex polyhedron Ω, the following equation exists:

$$T(\Omega_i, x) < 0 \tag{3}$$

Therefore, it means that the node x is in the obstacle.

2.3. Improved A algorithm*

When no obstacles are encountered, the search direction of the enhanced A* algorithm is the vector direction of the current node pointing to the goal point. The current node's search direction is $\bar{v}$, the current node is x, and the search step is d. Then, the search is performed on the child node x':

$$x' = x + \overrightarrow{v} \cdot d \tag{4}$$

If the child node x' interferes with the obstacle, the search cannot continue along the direction $\overrightarrow{v}$ at the node x. Therefore, the search direction of the child nodes needs to be redefined. The front area plane of the current node x is first selected from the obstacle O_i. Arbitrarily select a front area plane Ω_j of the obstacle O_i, take any edge l_k from the plane Ω_j, and take any point q on l_k, then the cost value of the point q can be expressed as follows:

$$F(q) = g(x) + h(q) + \|x - q\| \tag{5}$$

where $h(q)$ is the estimated cost value from the point q to the goal point, $g(x)$ is the current cost value of the node x, and $\|x - q\|$ is the path length value from node x to point q.

Then, the point on l_k where the smallest value of $F(q)$ exists is noted as q_k. q_k is denoted as a key node of the local path. Figure. 2 shows the search schematic.

The node $\hat{x}$ will traverse all the edges on the frontal area plane Ω_j, and each edge will generate a key node, as shown in Fig. 3. If there are n boundaries on the front area plane Ω_j, n alternative directions will be generated, denoted by $\{\tau_{ij1}^{x_1}, \tau_{ij2}^{x_1} \cdots \tau_{ijn}^{x_1}\}$. The path cost values of n directions are put into the set $c_{ij1}^{x_1}$, and the minimum cost value in the set $c_{ij1}^{x_1}$ is selected as the movement direction.

After locating the key node, the enhanced A* algorithm optimizes the local path based on the path nodes and the key node. The local path before the optimization is denoted by $\{K_s, P_1, P_2, P_3 \cdots P_{n-1}, P_n, K_g\}$, K_s, K_g denote the start point and goal point of the local path, respectively, and $P_1, P_2, P_3 \cdots P_{n-1}, P_n$ denote the nodes of the path. The local path optimization process is as follows: Starting from the starting point K_s, connect K_s to P_1. If K_s and P_1 do not interfere with the obstacle, connect K_s and P_2 until K_s and $P_m (k = 3, 4 \cdots, m)$ interfere with the obstacle. Connect K_s to P_{m-1} and clear all path nodes between the starting point K_s and node P_{m-1} and update the path. Repeat this operation from the node P_m until the key node K_s is searched. Figure 4 compares the situation before and following path optimization. Local path optimization can effectively reduce the path's length and number of turns.

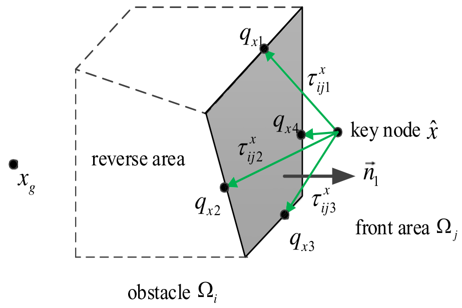

Figure 3. *Key nodes for the current node.*

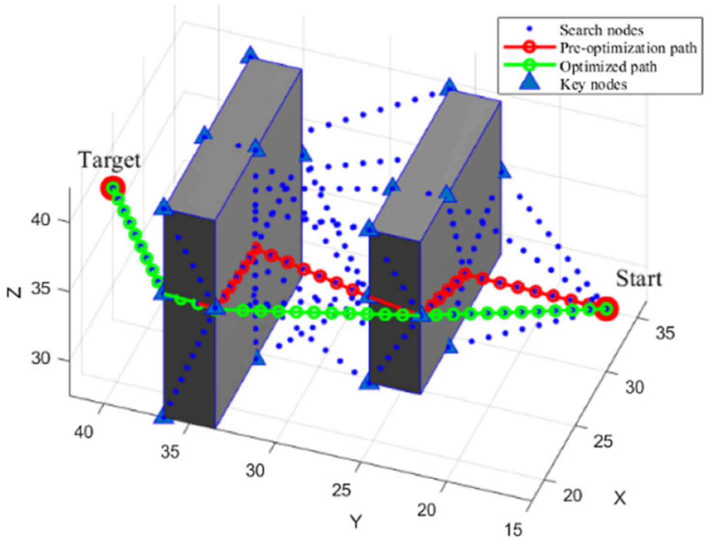

Figure 4. *Comparison of paths before and after optimization.*

3. Joint rod obstacle avoidance technique based on enhanced A* algorithm and artificial potential field method

During the process of path planning, the improved A* algorithm proposed above only modifies the position of the robotic arm's end, while the robot arm posture is ignored. In this paper, the local optimization property of the artificial potential field method is used to ensure that the robotic arm rods do not collide with environmental obstacles by adjusting the robotic arm's attitude.

3.1. Rod collision detection of the 6-DOF robotic arm

The robotic arm bars can be viewed as cylindrical features; therefore, it is necessary to determine if each cylindrical feature of the robotic arm bars collides with each environmental obstacle. The position of the robotic arm rod in space can be determined based on the robotic arm's current pose. The current pose of the robot arm is $X = (x, y, z, \alpha, \beta, \gamma)$, and the joint angle $Q_X = (q_1, q_2, q_3, q_4, q_5, q_6)$ can be obtained from the inverse kinematic model of the robot arm. As shown in Fig. 5(a), for the *ith* rod of the robot arm, the rod axis line segment is l_i, the rod radius is r_i, and the shortest distance from the line segment l_i to the obstacle O_j is denoted by $D(l_i, O_j)$. Then, the distance d_{ij} between the *ith* rod of the robot arm

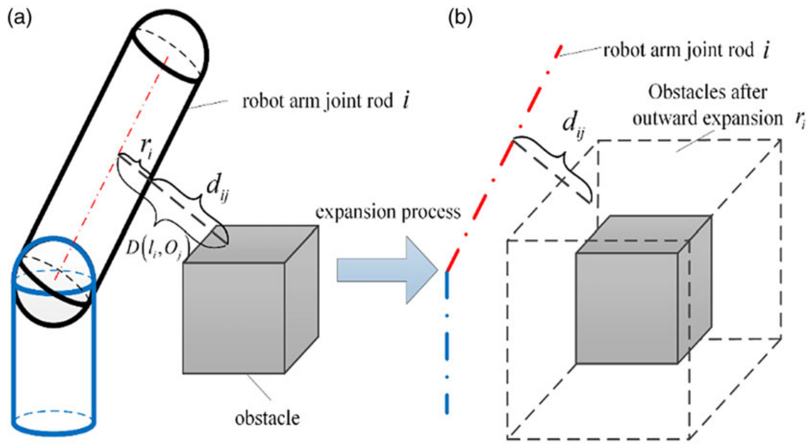

Figure 5. *Distance between the robotic arm joint rod and the obstacle.*

and the *jth* obstacle O_j is expressed as follows:

$$d_{ij} = \begin{cases} D(l_i, O_j) - r_i & D(l_i, O_j) > r_i \\ 0 & D(l_i, O_j) \leq r \end{cases} \tag{6}$$

In this study, the distance between the robotic arm and the obstacle is calculated as follows: first, the linkage is simplified into spatial line segments, and then, the obstacle is inflated, as depicted in Fig. 5(b). Determine if there is a point of intersection between each linkage segment and the obstacle plane region. If an intersection point exists, the robotic arm will collide with the obstruction. Obtain the distance between the line segments of each link and the line segments comprising the obstacle plane if there is no intersection point. An illustration is provided below. Suppose that the endpoints of the *ith* link of the robot arm are $q_1^i = (x_1^i, y_1^i, z_1^i)$ and $q_2^i = (x_2^i, y_2^i, z_2^i)$. Then, *ith* link can be regarded as a line segment $\overline{q_1^i q_2^i}$. Assume that the plane region is enclosed by points $\vec{p}_1 \vec{p}_2 \cdots \vec{p}_n \vec{p}_1$ connected in counterclockwise order. Suppose the normal vector of the plane is vector (A, B, C), the equation of the plane is given below:

$$Ax + By + Cz + D = 0 \tag{7}$$

Substitution of the coordinates $q_1^i = (x_1^i, y_1^i, z_1^i)$ and $q_2^i = (x_2^i, y_2^i, z_2^i)$ of the endpoints of the segment into the above equation yields:

$$d_1^i = Ax_1^i + By_1^i + Cz_1^i + D \tag{8}$$

$$d_2^i = Ax_2^i + By_2^i + Cz_2^i + D \tag{9}$$

If d_1^i and d_2^i have the same sign, then there is no intersection of line $\overline{q_1^i q_2^i}$ with the plane, and there is no collision between that line and the obstacle. If d_1^i and d_2^i have opposite signs, then there is an intersection of line $\overline{q_1^i q_2^i}$ with the plane, and it is easy to solve for the location of the intersection. Since the plane region $\vec{p}_1 \vec{p}_2 \cdots \vec{p}_n \vec{p}_1$ mentioned in this study is only a small part of the whole plane, when there is an intersection of the line segment with the whole plane, there may be a situation where the line segment $\overline{q_1^i q_2^i}$ does not intersect with the plane region $\vec{p}_1 \vec{p}_2 \cdots \vec{p}_n \vec{p}_1$. Therefore, it is necessary to determine whether the intersection point is in the plane area. To better illustrate the judgment of whether the intersection point is in the plane region $\vec{p}_1 \vec{p}_2 \cdots \vec{p}_n \vec{p}_1$, Fig. 6 is depicted below.

Assume that the intersection point is o. The points $p_1 p_2 \cdots p_n$ are forming the planar region form the vectors $\overrightarrow{p_1 p_2}$, $\overrightarrow{p_2 p_3}$, $\overrightarrow{p_3 p_4}$ and $\overrightarrow{p_4 p_5}$ in order, and the vertices form the vectors $\overrightarrow{p_1 o}$, $\overrightarrow{p_2 o}$, $\overrightarrow{p_3 o}$, $\overrightarrow{p_4 o}$ and $\overrightarrow{p_4 o}$

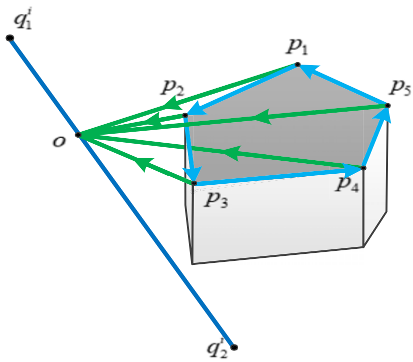

Figure 6. *Schematic diagram for judging the intersection point.*

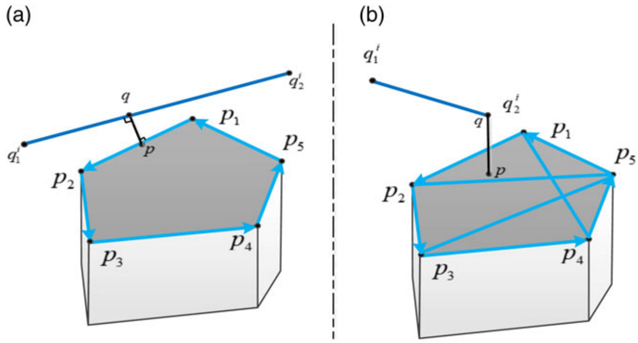

Figure 7. *The shortest distance between a line segment and a planar region.*

with the point o, respectively, and satisfy the following equation:

$$\overrightarrow{p_ip_{i+1}} \times \overrightarrow{p_io} = m_i \quad , \; i = 1 \sim n \tag{10}$$

According to the above equation, the vector $\overrightarrow{p_ip_{i+1}}$, formed between the vertices of the plane region and the vector $\overrightarrow{p_io}$ (formed by the vertices and the intersection point) is multiplied.

If the sign of the results of the calculation is the same, then the intersection point is in the plane region, and line segment $\overline{q_1^iq_2^i}$, must collide with the obstacle region. On the contrary, the intersection point is outside the plane region, then the line segment $\overline{q_1^iq_2^i}$ will not collide with the obstacle region. When no collision occurs, solve for the distance between line segment $\overline{q_1^iq_2^i}$ and line segment $\overline{p_1p_2}, \overline{p_2p_3}, \overline{p_3p_4}, \overline{p_4p_5}, \overline{p_1p_3}, \overline{p_1p_4}, \overline{p_2p_4}$ and $\overline{p_2p_5}$, respectively. The line segments of each link of the robot arm and the obstacle plane are evaluated sequentially to determine if an intersection point exists. If there is no intersection point, calculate the distance between the arm and the obstacle and take the smallest value as the shortest distance.

The detailed calculation of the shortest distance between the rod of the robotic arm and the obstacle is provided below. The shortest distance between the line segment $\overline{q_1^iq_2^i}$ and each planar region of the obstacle is solved separately. An example of the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and the plane region $\overrightarrow{p}_1\overrightarrow{p}_2\overrightarrow{p}_3\overrightarrow{p}_4\overrightarrow{p}_5\overrightarrow{p}_1$ is shown in Fig. 7. In essence, the goal is to find the minimum value of the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and each of line segments $\overline{p_1p_2}, \overline{p_2p_3}, \overline{p_3p_4}, \overline{p_4p_5}, \overline{p_5p_1}, \overline{p_1p_3}, \overline{p_1p_4}, \overline{p_2p_4}$ and $\overline{p_2p_5}$ on the plane region $\overrightarrow{p}_1\overrightarrow{p}_2\overrightarrow{p}_3\overrightarrow{p}_4\overrightarrow{p}_5\overrightarrow{p}_1$ as the shortest distance between the line segment and the plane region. It is noteworthy that the shortest distance between line segment $\overline{q_1^iq_2^i}$ and line segments $\overline{p_1p_3}, \overline{p_1p_4}, \overline{p_2p_4}$ and $\overline{p_2p_5}$ is necessary for this calculation. In general, the obstacle is relatively small compared to the rod of the robotic arm. Consequently, the

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

shortest distance between the line segment of the rod of the robotic arm and the plane region of the obstacle often falls on the line segment that encloses the flat plane region, such as the shortest distance $|pq|$ in Fig. 7(a). However, under certain conditions, the shortest distance between the line segment of the rod and the plane region of the obstacle can lie within the plane region of the obstacle, such as the shortest distance $|pq|$ in Fig. 7(b). Therefore, solving for the shortest distance between rod segment $\overline{q_1^i q_2^i}$ and line segments $\overline{p_1p_3}$, $\overline{p_1p_4}$ and $\overline{p_2p_4}$ can help reduce the calculation errors in a few cases.

Here is an example of solving for the shortest distance between line segment $\overline{q_1^i q_2^i}$ and line segment $\overline{p_1p_2}$. The coordinates of points q_1^i, q_2^i, p_1 and p_2 are $(x_{q_1}, y_{q_1}, z_{q_1})$, $(x_{q_2}, y_{q_2}, z_{q_2})$, $(x_{p_1}, y_{p_1}, z_{p_1})$ and $(x_{p_2}, y_{p_2}, z_{p_2})$, respectively. Assuming q is a point on the line $q_1^i q_2^i$; then, the coordinates of point q can be described as follows:

$$\begin{cases} x_q = x_{q_1} + s\left(x_{q_2} - x_{q_1}\right) \\ y_q = y_{q_1} + s\left(y_{q_2} - y_{q_1}\right) \\ z_q = z_{q_1} + s\left(z_{q_2} - z_{q_1}\right) \end{cases} \tag{11}$$

When there exists $0 \leq s \leq 1$, q is a point on line segment $\overline{q_1^i q_2^i}$. Conversely, q is a point on the extension of the line segment $\overline{q_1^i q_2^i}$.

Similarly, letting p be a point on the line p_1p_2, the coordinates of the point p can be described as follows:

$$\begin{cases} x_p = x_{p_1} + t\left(x_{p_2} - x_{p_1}\right) \\ y_p = y_{p_1} + t\left(y_{p_2} - y_{p_1}\right) \\ z_p = z_{p_1} + t\left(z_{p_2} - z_{p_1}\right) \end{cases} \tag{12}$$

When there exists $0 \leq t \leq 1$, p is a point on line segment $\overline{p_1p_2}$. Conversely, p is a point on the extension of line segment $\overline{p_1p_2}$.

Thus, the distance between points p and q can be expressed as:

$$|pq| = \sqrt{\left(x_q - x_p\right)^2 + \left(y_q - y_p\right)^2 + \left(z_q - z_p\right)^2} \tag{13}$$

Solving for the shortest distance between line segment $\overline{q_1^i q_2^i}$ and line segment $\overline{p_1p_2}$ is equivalent to solving for the shortest distance between the points p and q. The function can be set up as follows:

$$f(s,t) = |pq|^2 = \left(x_q - x_p\right)^2 + \left(y_q - y_p\right)^2 + \left(z_q - z_p\right)^2 \tag{14}$$

Calculate the partial derivative of the function $f(s, t)$ and compute the following equation:

$$\begin{cases} \dfrac{\partial f(s,t)}{\partial s} = 0 \\ \dfrac{\partial f(s,t)}{\partial t} = 0 \end{cases} \tag{15}$$

If the solutions s and t of the above equation satisfy the following equation:

$$\begin{cases} 0 \leq s \leq 1 \\ 0 \leq t \leq 1 \end{cases} \tag{16}$$

It can be determined that p is on line segment $\overline{p_1p_2}$ and q is on line segment q. The distance $|pq|$ can be obtained by solving the equations mentioned above.

If Eq. (16) is not satisfied, then it is calculated as follows: If $s > 1$, then the value of s is 1, and the value of point p is p_2. If $s < 0$, then the value of s is 0, and the value of point p is p_1. The problem mentioned above can be transformed into the shortest distance from point p to line segment $\overline{q_1^i q_2^i}$. Consequently, Eq. (4) is transformed into the following function:

$$f(t) = |pq|^2 = \left(x_q - x_p\right)^2 + \left(y_q - y_p\right)^2 + \left(z_q - z_p\right)^2 \tag{17}$$

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

Solve for the derivative of the function $f(t)$ and compute the following equation:

$$\frac{df(t)}{dt} = 0 \tag{18}$$

If $t > 1$, then the value of t is 1, and the value of point q is q_1^i. If $t < 0$, then the value of t is 0 and the value of point q is q_2^i. The distance between points p and q can be calculated by Eq. (13), which is also the shortest distance between line segments $\overline{p_1p_2}$ and $\overline{q_1^iq_2^i}$.

Finally, the method outlined above is applied to calculate the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and each line segment of the plane region $\overrightarrow{p}_1\overrightarrow{p}_2\overrightarrow{p}_3\overrightarrow{p}_4\overrightarrow{p}_5\overrightarrow{p}_1$. The smallest of these distances is then considered the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and the plane region $\overrightarrow{p}_1\overrightarrow{p}_2\overrightarrow{p}_3\overrightarrow{p}_4\overrightarrow{p}_5\overrightarrow{p}_1$. Following the same procedure, the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and each plane of the obstacle is calculated, and the smallest value among these distances represents the shortest distance between the line segment $\overline{q_1^iq_2^i}$ and the obstacle.

Therefore, if the robot arm pose is X, the distance between the robot arm and the obstacle O_j can be described as follows:

$$d_j^X = \min\left(d_{ij} | i \in (1, 2, \ldots, n)\right) \tag{19}$$

If the robotic arm rod does not collide with the obstacle, then the following equation exists.

$$d_j^X \geq 0 \tag{20}$$

3.2. Principle of robotic arm posture adjustment based on artificial potential field method

The artificial potential field method with local optimization properties is employed to adjust the robot arm's posture and find the local optimal posture at the current position. As posture adjustment is a locally optimal solution procedure, only the repulsive potential energy of the robotic arm rods is taken into account. It is expressed as:

$$U_j^X = \begin{cases} \dfrac{1}{2}k_r\left(\dfrac{1}{d_j^X} - \dfrac{1}{d_0}\right)^2, & d_j^X \leq d_0 \\ 0, & d_j^X > d_0 \end{cases} \tag{21}$$

where d_j^X is the distance between the robot arm bar and the obstacle O_j, d_0 is the repulsive range of the obstacle O_j, and k_r is the gain factor.

$$U^X = \sum_{j=1}^{n} U_j^X \tag{22}$$

Adjusting the posture of the robotic arm yields the minimum value of U^X to find the optimal solution. The robotic arm's repulsive force is calculated as follows:

$$f^X = \frac{\partial U^X}{\partial X} = \sum_{j=1}^{n} \frac{\partial U_j^X}{\partial X} \tag{23}$$

where

$$\frac{\partial U_j^X}{\partial X} = \begin{cases} k_r\left(\dfrac{1}{d_0} - \dfrac{1}{d_j^X}\right)\dfrac{1}{\left(d_j^X\right)^2}\dfrac{\partial d_j^X}{\partial X}, & d_j^X \leq d_0 \\ 0, & d_j^X > d_0 \end{cases} \tag{24}$$

where the nearest point of the robot arm to the obstacle O_j is x_j and the nearest point on the obstacle is x_j^o.

The following equation can be obtained:

$$d_j^X = \sqrt{\left(x_j - x_j^o\right)^T \cdot \left(x_j - x_j^o\right)} \tag{25}$$

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

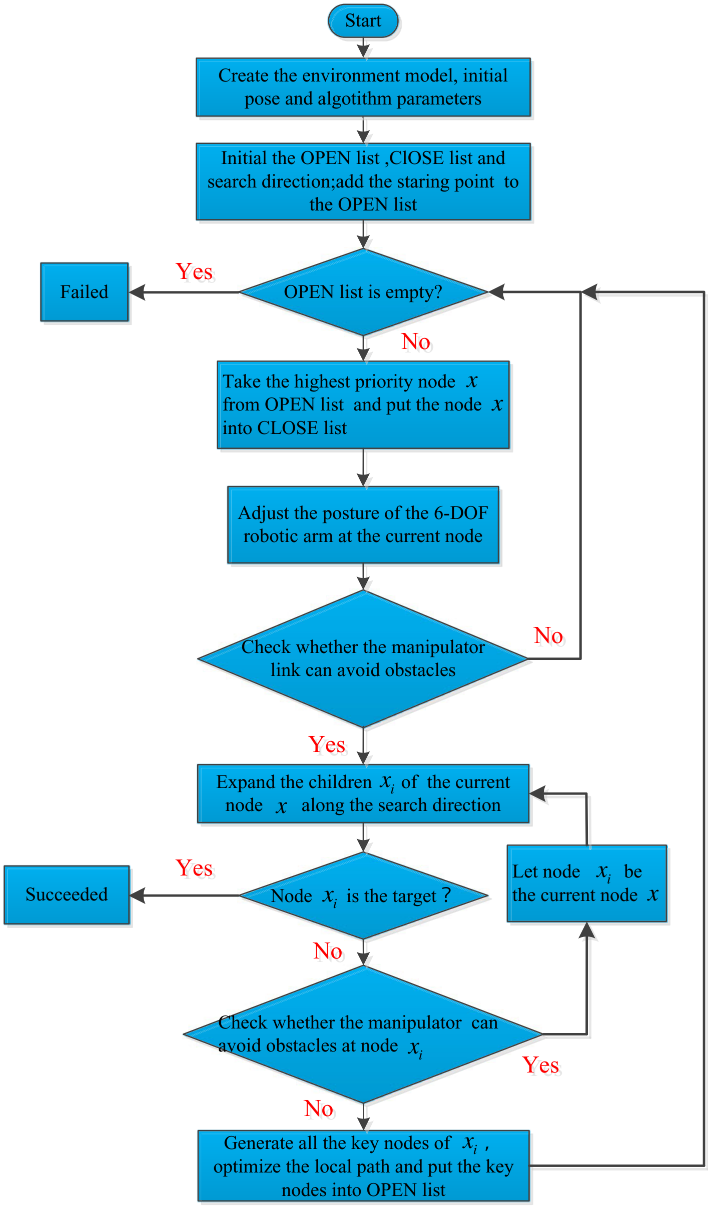

Figure 8. *Overall obstacle avoidance strategy of the robotic arm.*

Thus, the following equation can be derived:

$$\begin{aligned} \frac{\partial d_j^X}{\partial X} &= \frac{\left(x_j - x_j^o\right)}{\sqrt{\left(x_j - x_j^o\right)^T \cdot \left(x_j - x_j^o\right)}} \frac{\partial x_j}{\partial X} \\ &= \frac{\left(x_j - x_j^o\right)}{\left\|\left(x_j - x_j^o\right)\right\|} \frac{J_j^* \partial q}{\partial X} = \frac{\left(x_j - x_j^o\right)}{\left\|\left(x_j - x_j^o\right)\right\|} J_j^* (J_X)^{-1} \end{aligned} \tag{26}$$

where J_j^* denotes the Jacobi matrix of the point x_j on the robotic arm considering only small displacements and not the pose. J_X stands for the Jacobi matrix at the end of the robotic arm.

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

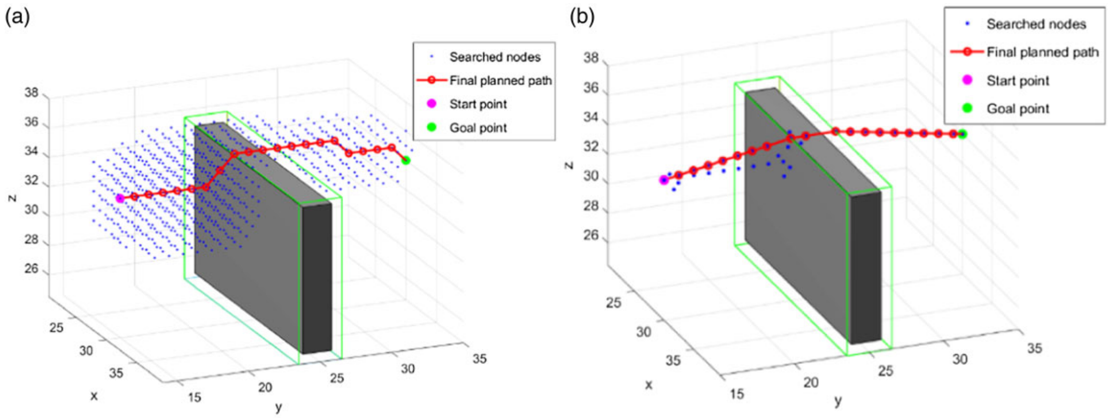

Figure 9. *Path planning in Environment 1. (a) Traditional A*, (b) improved A**

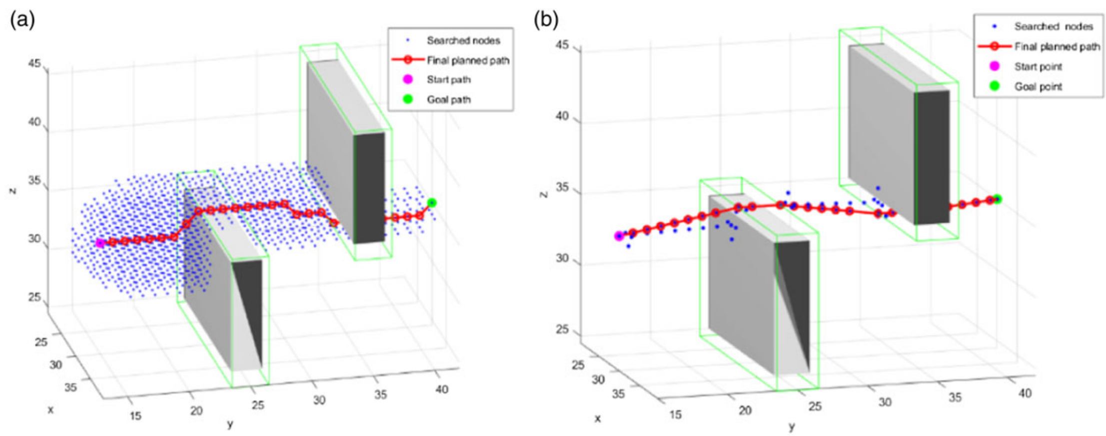

Figure 10. *Path planning in Environment 2. (a) Traditional A*, (b) improved A**

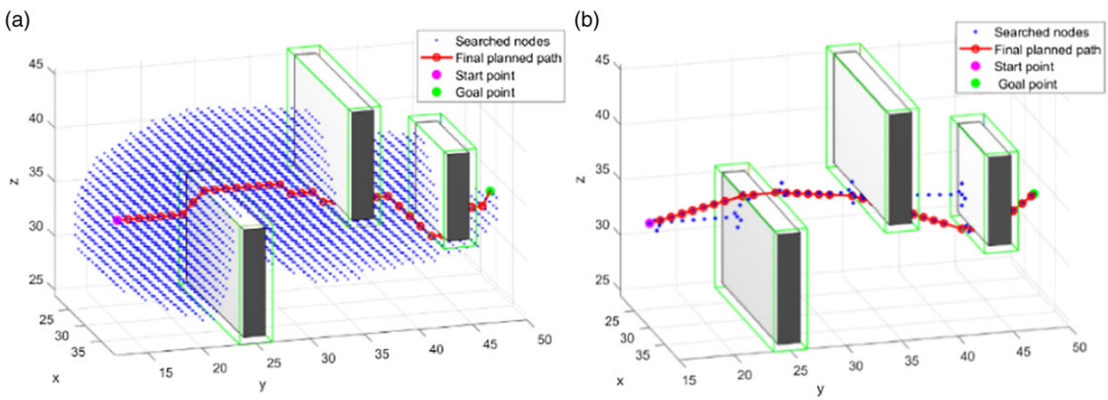

Figure 11. *Path planning in Environment 3. (a) Traditional A*, (b) improved A**

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

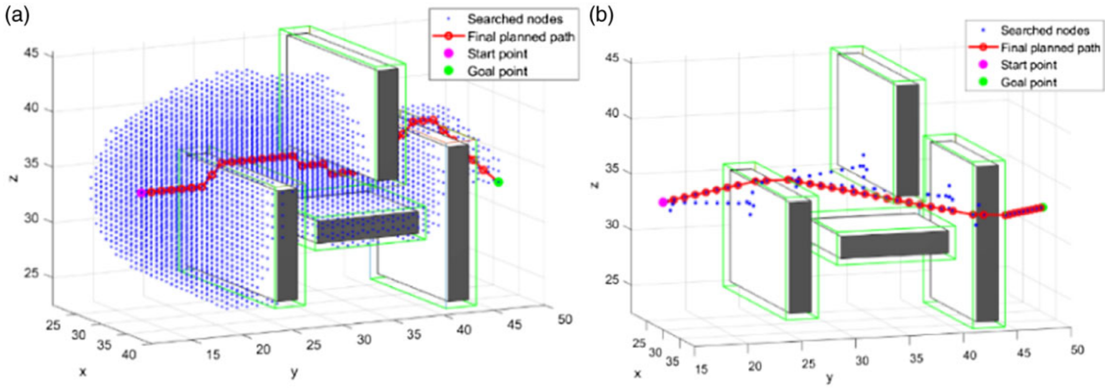

Figure 12. *Path planning in Environment 4. (a) Traditional A*, (b) improved A**

Therefore, it can be easily obtained that f^X is a 6×1 vector. The repulsive potential energy of the robot arm will decrease when its attitude changes in the direction of f^X, and only the attitude component of f^X needs to be considered to adjust the robotic arm in this paper.

3.3. Overall obstacle avoidance strategy for 6-DOF robotic arm

The robotic arm bars can be viewed as cylindrical features; therefore, it is necessary to determine if each cylindrical feature of the robotic arm bars collides with each environmental obstacle. The position of the robotic arm rod in space can be determined based on the robotic arm's current pose. To elucidate the overall obstacle avoidance strategy of the robotic arm, the working process of the improved A* algorithm and the artificial potential field method are analyzed in depth. Figure 8 depicts the overall obstacle avoidance strategy.

First, an optimal robotic arm end planning path is determined using the enhanced A* algorithm in the environment model, followed by the determination of the initial pose and search direction. The robot arm begins to move when the joint rod is about to collide with an impediment. If the pose of the robot arm at the node of collision is $X_0 = (x_0, y_0, z_0, \alpha_0, \beta_0, \gamma_0)$, only the posture of the robot arm is changed to adjust the spatial position of each bar to avoid the obstacle.

The direction of the robotic arm posture change is described in section 3.2, and its magnitude is as follows:

$$\delta X = (0, 0, 0, \delta\alpha, \delta\beta, \delta\gamma) \tag{27}$$

The position of the robot arm after the modification is as follows:

$$X_\delta = (x_0, y_0, z_0, \alpha_0 + \delta\alpha, \beta_0 + \delta\beta, \gamma_0 + \delta\gamma) \tag{28}$$

Using the robotic arm's Jacobi matrix, the relationship between the change in arm pose and the change in joint angle is determined and expressed as:

$$\delta X = J \cdot \delta Q \tag{29}$$

Consequently, the following equation can be derived:

$$\delta Q = J^{-1} \cdot \delta X \tag{30}$$

The shortest distance d_j^X between the robot arm and the obstacle can be solved using the changed joint angle Q. If $d_j^X > 0$, it is easy to obtain that the robotic arm can avoid the obstacle according to the changed pose. If $d_j^X \leq 0$, it indicates that the obstacle cannot be avoided regardless of the robot arm's orientation. Since excessive posture adjustment of the robotic arm can result in vibrations and abrupt changes, it is necessary to impose certain limits on the amount of attitude adjustment.

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

Table I. *Parameters set for obstacle avoidance environment*

Environment	**Parameters**	**Value (*Length* × *Width* × *Height*)**
Environment 1	Start point; Goal point	(30,15,34); (30,35,34)
	Size of obstacle 1	19 × 3 × 11
Environment 2	Start point; Goal point	(30,15,34); (30,42,35)
	Size of obstacles 1, 2	19 × 3 × 11, 19 × 3 × 11
Environment 3	Start point; Goal point	(30,15,34); (28,50,34)
	Size of obstacles 1, 2, 3	19 × 3 × 11, 19 × 3 × 11, 11 × 3 × 9
Environment 4	Start point; Goal point	(30,15,34); (31,50,32)
	Size of obstacles 1, 2, 3, 4	19 × 3 × 11, 19 × 3 × 11, 11 × 3 × 9, 15 × 3 × 15

Table II. *Comparison of search parameters before and after the development of the A* algorithm*

Environment	**Algorithm**	**Nodes searched**	**Path length**	**Time(s)**
Environment 1	Traditional A*	690	21	0.272
	Improved A*	40	20	0.175
Environment 2	Traditional A*	2484	28	1.162
	Improved A*	82	30	0.223
Environment 3	Traditional A*	5035	36	1.558
	Improved A*	89	34	0.241
Environment 4	Traditional A*	5357	36	2.006
	Improved A*	138	38	0.485

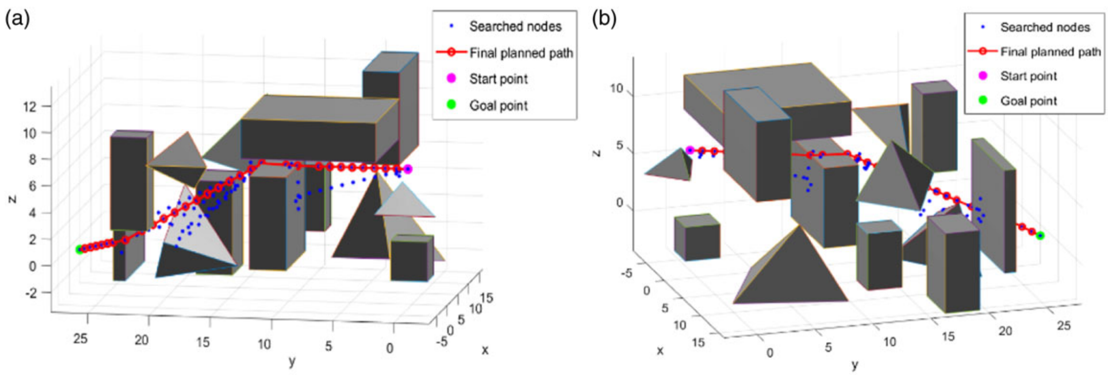

Figure 13. *Improved A* in case 1. (a) Front view, (b) side view.*

The constraints are as follows:

$$\begin{cases} |\delta\alpha| \le \varepsilon_\alpha \\ |\delta\beta| \le \varepsilon_\beta \\ |\delta\gamma| \le \varepsilon_\gamma \end{cases} \tag{31}$$

where ε_α, ε_β and ε_γ are the constraint values of posture change respectively. The values of ε_α, ε_β, and ε_γ are 0.2, 0.2, and 0.1, respectively in the obstacle avoidance algorithm of this study.

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

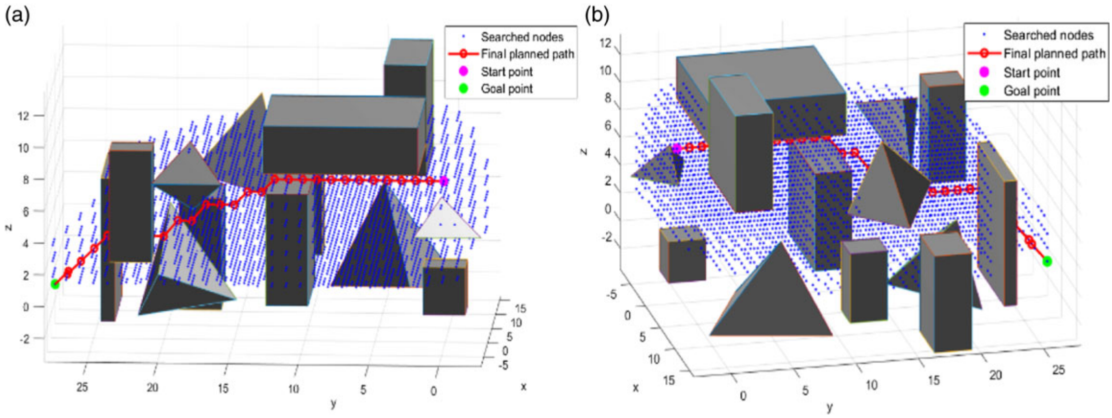

Figure 14. *Traditional A* in case 1. (a) Front view, (b) side view.*

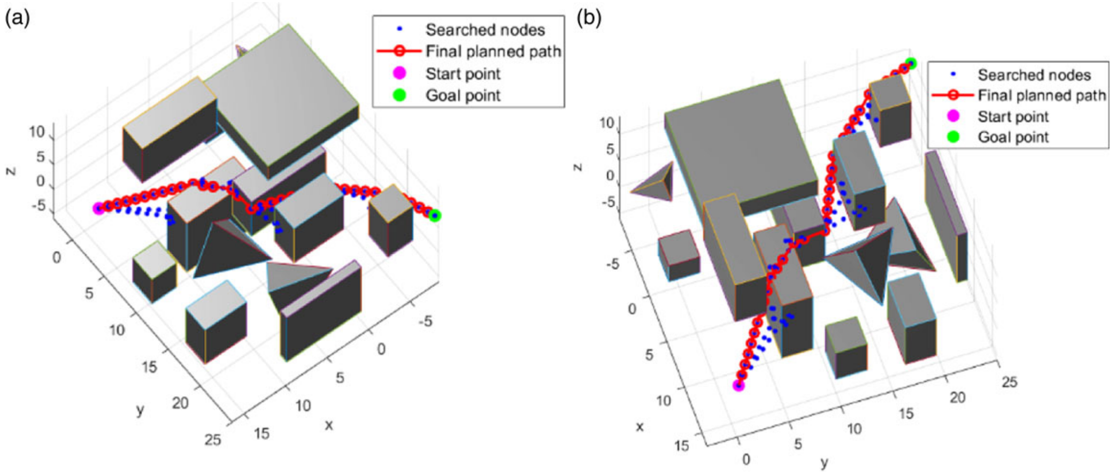

Figure 15. *Improved A* in case 2. (a) Front view, (b) side view.*

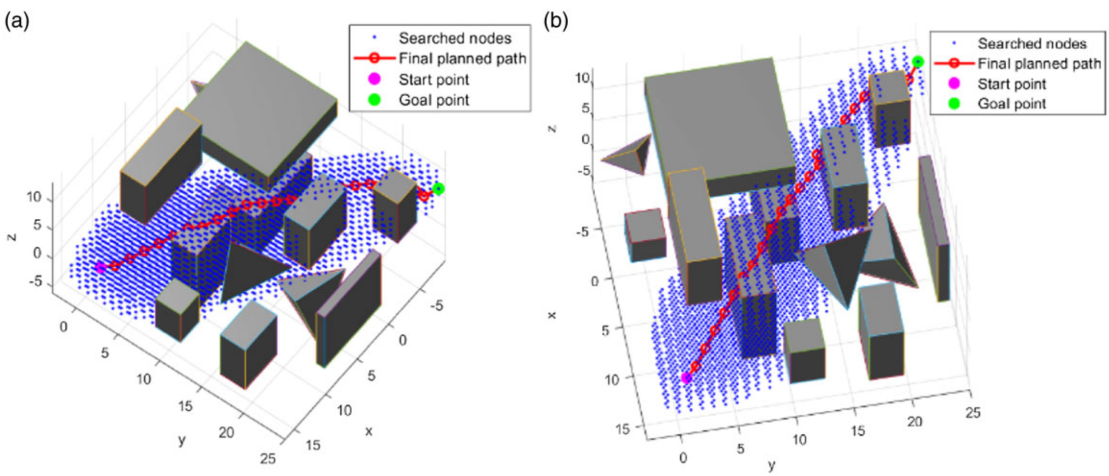

Figure 16. *Traditional A* in case 2. (a) Front view, (b) side view.*

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

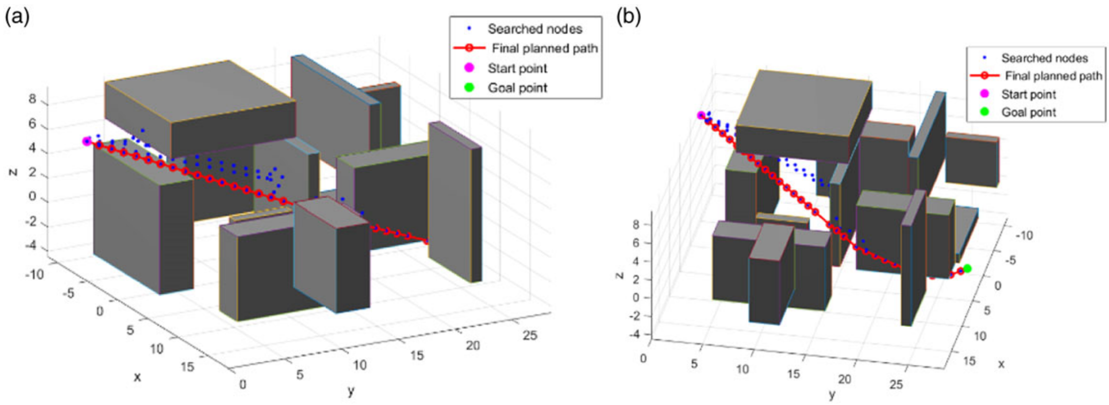

Figure 17. *Improved A* in case 3. (a) Front view, (b) side view.*

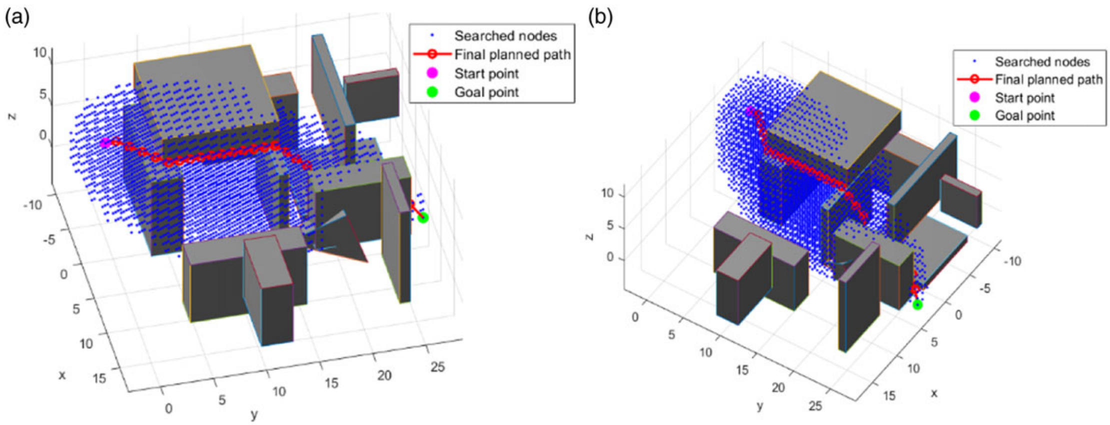

Figure 18. *Traditional A* in case 3. (a) Front view, (b) side view.*

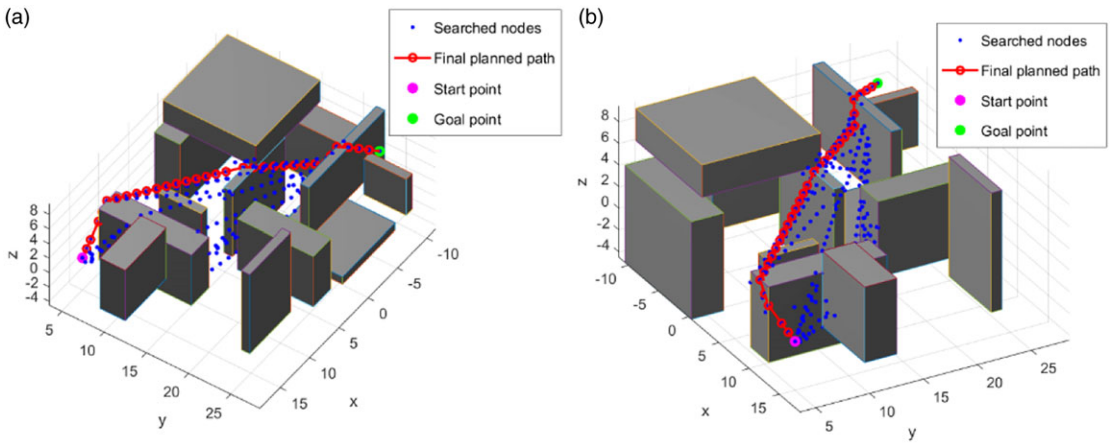

Figure 19. *Traditional A* in case 4. (a) Front view, (b) side view.*

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

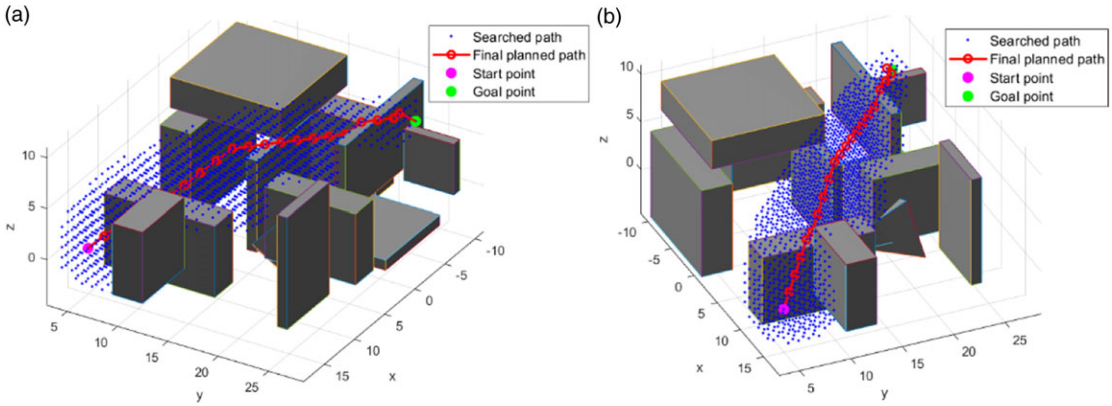

Figure 20. *Traditional A* in case 4. (a) Front view, (b) side view.*

4. Experiments and results

Simulation and experimental analysis are conducted in this section to verify the feasibility, effectiveness, and environmental adaptability of the robotic arm path planning algorithm proposed in this paper. The simulation and experimental analysis focus primarily on the search efficiency of the improved A* algorithm, the improved A* algorithm's adaptability in a multi-obstacle environment, and the practicability of the robotic arm pose adjustment strategy based on the artificial potential field method.

4.1. Simulation and analysis

4.1.1. Comparison of the improved A* algorithm and the traditional A* algorithm

The traditional A* algorithm has many problems in the path planning of 3D environment, including a large number of search nodes, a lengthy search, and a decrease in computational efficiency as the number of obstacles in the environment increases. To verify the benefits of the improved A* algorithm proposed in this paper for path planning, the improved algorithm and the traditional A* algorithm are simulated and analyzed in four map environments, respectively. As illustrated in Figs. 9, 10, 11 and 12. The coordinates of the path's start and goal points and the parameters of the obstacles in the environment are displayed in Table I. The search step size of the improved A* algorithm is set to 1. The search terminates when the spatial distance between the search node and the goal point is less than 1.

From the simulation results of the four environments listed above, it can be concluded that the improved A* algorithm proposed in this paper can effectively reduce the number of search nodes in path planning when compared to the traditional A* algorithm. Table II displays the search results of the enhanced A* algorithm and the traditional A* algorithm in four environments. It can be observed that the number of search nodes utilized by the enhanced A* algorithm in various environments has decreased considerably. The proportion of search nodes decreased significantly as the number of obstacles in the environment increased, but the final search path length remained essentially the same. Therefore, the enhanced A* algorithm can effectively improve the search efficiency in 3D environment and significantly reduce the defects of the conventional A* algorithm.

4.1.2. Analysis of the environmental adaptability of the enhanced A* algorithm

The improved A* algorithm can remedy the traditional A* algorithm's low search efficiency, but the algorithm's adaptability to complex environments with multiple obstacles requires further investigation. Two distinct complex environment maps are constructed, and different starting and ending points are

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

Table III. *Environmental parameter*

Environment	Case	Start point	Goal point
Environment 1	Case 1	(−2, 0, 7)	(5, 28, −1)
	Case 2	(14, 1, −1)	(−8, 25, 8)
Environment 2	Case 3	(−3, 0, 7)	(5, 28, −1)
	Case 4	(15, 5, 0)	(−7, 25, 8)

Table IV. *Parameters set for obstacle avoidance environment*

Environment	Parameters	Value (*Length* × *Width* × *Height*)
Environment 1	Size of obstacles 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13	$3 \times 3 \times 3, 5 \times 3 \times 7, 11 \times 11 \times 3, 9 \times 3 \times 7, 9 \times 3 \times 7, 7 \times 7 \times 11, 9 \times 5 \times 7, 3 \times 5 \times 5, 9 \times 7 \times 13, 7 \times 7 \times 13, 3 \times 3 \times 5, 9 \times 1 \times 9, 3 \times 3 \times 7$
	Center points of obstacles 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13	(0,0,0), (13,18,0), (3,7,10), (0,11,3), (7,3,10), (10,15,5), (−3,18,6), (−5,0,4), (11,5,2), (5,20,2), (15,11,2), (5,24,2), (−5,22,6)
Environment 2	Size of obstacles 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	$11 \times 3 \times 9, 3 \times 11 \times 7, 11 \times 11 \times 3, 3 \times 9 \times 713 \times 1 \times 7, 7 \times 3 \times 7, 9 \times 1 \times 9, 3 \times 9 \times 7, 7 \times 1 \times 11, 7 \times 7 \times 13, 1 \times 5 \times 5, 9 \times 9 \times 1$
	Center points of 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	(−5,5,0), (10,10,0), (−5,10,8), (−10,15,1), (5,22,5), (15,11,2), (0,15,2), (5,22,2), (11,24,3), (5,20,−2), (−7,26,4), (−3,22,−2)

Table V. *Comparing the two algorithms in four distinct situations*

Environment	Algorithm	Nodes searched	Path length	Time (s)
Case 1	Traditional A*	3242	30	9.193
	Improved A*	137	32	1.402
Case 2	Traditional A*	2514	26	8.024
	Improved A*	107	32	1.263
Case 3	Traditional A*	3280	29	9.231
	Improved A*	314	36	2.130
Case 4	Traditional A*	1589	24	7.781
	Improved A*	330	30	2.202

chosen for simulation in each map. The outcomes of the simulation are depicted in Figs. 13, 14, 15, 16, 17, 18, 19, and 20. Table III compares the outcomes of the two search algorithms in four distinct instances. The planned routes are displayed from two distinct angles. The beginning and end parameters of the environment-selected path are displayed in Table III. The parameters of the environment obstacles are shown in Table IV, and the improved A* algorithm employs a search step size of 1. The search terminates when the spatial distance between the search node and the goal point is less than 1.

From the simulation results of two distinct complex environments, it is evident that the proposed improved A* algorithm can effectively perform obstacle avoidance planning and that the planned path in a complex environment with numerous obstacles is relatively short. Consequently, the enhanced A*

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

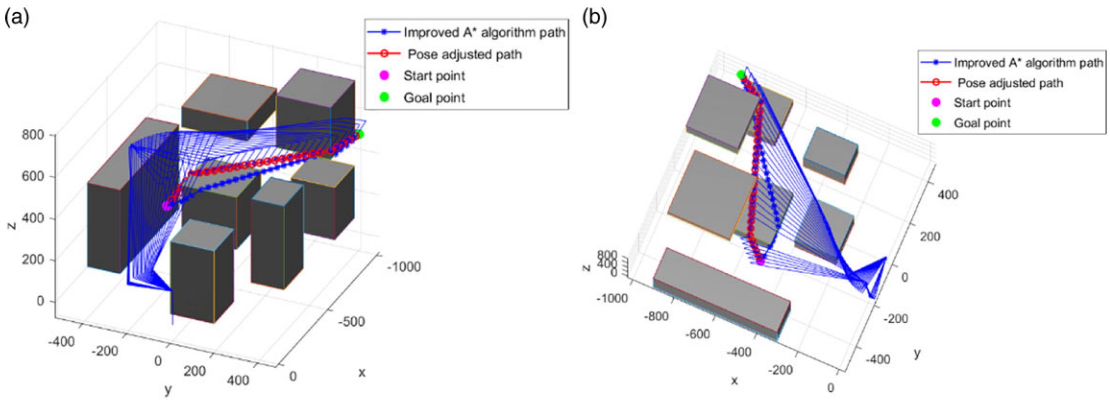

Figure 21. Path planning in Case 1. (a) Front view, (b) side view.

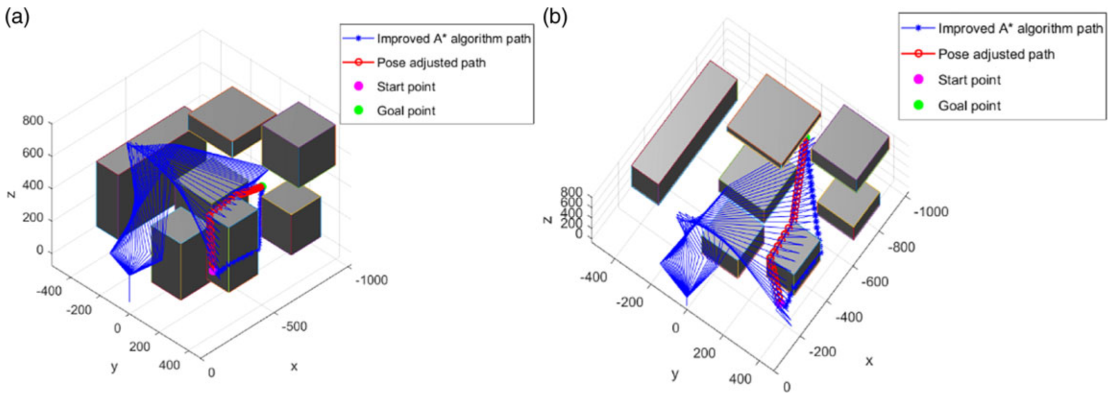

Figure 22. Path planning in Case 2. (a) Front view, (b) side view.

algorithm has enhanced adaptability to complex environments and a degree of generality in path planning. The search time experiences an increase in both the enhanced A* algorithm and the conventional A* algorithm when operating within a complex environment, as opposed to the simpler obstacle avoidance environment previously discussed. Although the simple obstacle avoidance environment exhibits a higher number of nodes and path length compared to the complex environment, the search time in the former remains lower than that in the latter. This phenomenon arises due to the algorithm's requirement for increased computational time to assess collisions with obstacles and identify viable pathways within intricate environments. Furthermore, the enhanced A* algorithm exhibits significantly reduced search time compared to the conventional A* algorithm, irrespective of the prevailing obstacle environment conditions (Table V).

4.1.3. Simulation of 6-DOF robot arm posture adjustment strategy

The preceding simulation is limited to the path planning of the robotic arm end by the enhanced A* algorithm and does not account for the possibility that the robotic arm's joint rod will collide with the obstacle. To improve the 6-DOF robotic arm's overall obstacle avoidance strategy, simulation analysis is performed in the obstacle environment. The simulation is depicted in Figs. 21 and 22, which depict the path planning results of the robot arm with different starting and endpoints. A blue line segment represents the joint rod of the 6-DOF robotic arm, and the environment's obstacles have been enlarged to be larger than the manipulator's radius. The parameters of the simulation are presented in Table VI. Following this

Table VI. *Two simulation conditions*

Case	Start point	Goal point
Case 1	(−500, −250, 250)	(−900, 450, 550)
Case 2	(−200, 370, 290)	(−900, 0, 200)

Table VII. *Comparison of the three algorithms in three different cases*

Environment	Algorithm	Success rates	Mean time(s)
Case 1	Traditional A* for end position obstacle avoidance	100%	10.856
	Improved A* for end position obstacle avoidance	100%	3.346
	Posture adjustment for overall obstacle avoidance	99.2%	8.142
Case 2	Traditional A* for end position obstacle avoidance	100%	11.321
	Improved A* for end position obstacle avoidance	100%	3.125
	Posture adjustment for overall obstacle avoidance	99.3%	7.781

premise, a total of 1000 starting points and goal points are randomly generated within a space characterized by a radius of 5 units. The coordinates of these points are determined such that the starting point corresponds to the center of the sphere mentioned in Table V, while the goal point corresponds to the center of the sphere as well. The path planning is executed following the proposed overarching obstacle avoidance strategy in two distinct scenarios, and the outcomes are presented in Table VII.

In the above simulation, both the traditional A* algorithm and the improved A* algorithm proposed in this paper are used for end position obstacle avoidance, achieving a 100% success rate of obstacle avoidance. However, when the posture adjustment method proposed in this paper is used for the overall obstacle avoidance of the robotic arm, there is a slight reduction in the success rate. This decrease primarily stems from the fact that the traditional and improved A* algorithms for end position obstacle avoidance do not consider collisions between the individual links of the robotic arm and the obstacles. However, when the posture adjustment method is employed for the overall obstacle avoidance of the robotic arm, the posture adjustment strategy, based on the artificial potential field, may lead to the local optimum in certain extreme cases. This can, to some extent, reduce the success rate of the search. The path results of the posture adjustment strategy for a 6-DOF robotic arm based on the improved A* algorithm and the artificial potential field method differ from the path results planned by the improved A* algorithm alone, and the path length and cost time have increased significantly. This is due to the posture adjustment strategy determining whether the obstacle collides with the joint rod and adjusting the original path result to accommodate the manipulator's movement. The attitude adjustment strategy does not guarantee complete obstacle avoidance as the A* and improved A* algorithms do, but its obstacle avoidance success rate is still quite high. It can be seen that the algorithmically planned path has poor smoothness, and jitter may occur in the manipulator's trajectory motion. Finally, cubic spline processing is applied to the planned path to increase the stability of the manipulator's motion.

4.2. Experiment in a real environment

To demonstrate the efficacy of the comprehensive obstacle avoidance strategy, an experimental evaluation is conducted to compare the performance of the six-degree-of-freedom joint obstacle avoidance

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

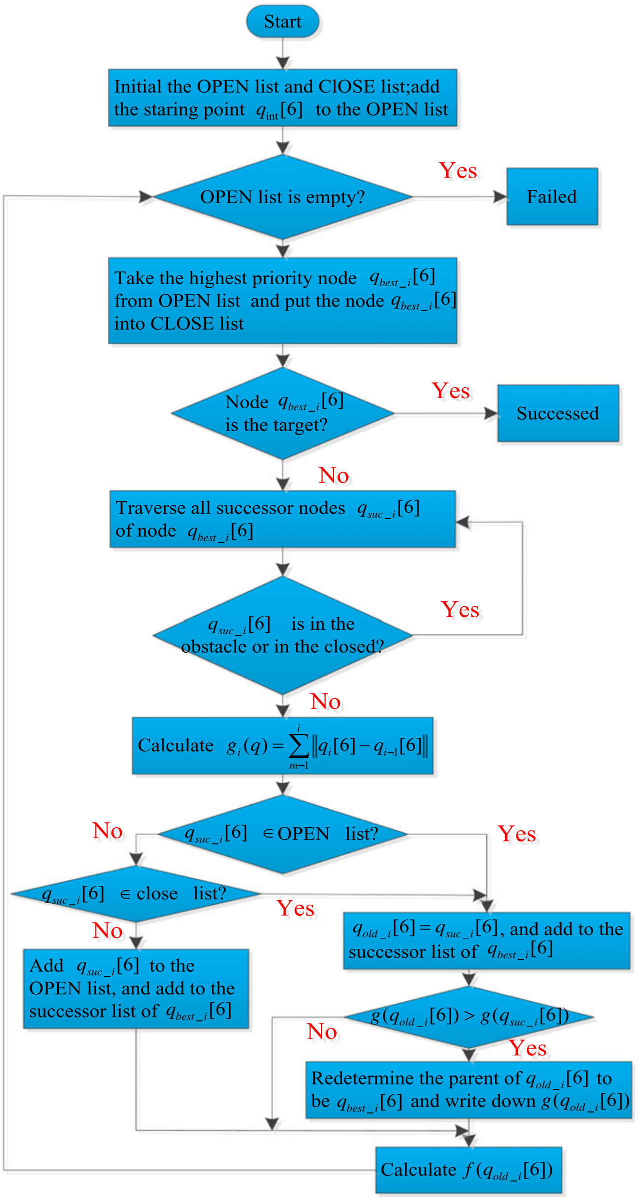

Figure 23. *Obstacle avoidance algorithm proposed by Jia.*

algorithm, which is based on the A* algorithm proposed by Jia [29], with the algorithm proposed in this study. The primary concept of the algorithm presented by Jia involves mapping the search for the position of the robotic arm in 3D space to the search for angles in joint space. The prescribed procedure is outlined as follows: the six joint angles of the robotic arm are designated to be documented as a six-dimensional array. Subsequently, the initial and target positions in three-dimensional space are determined through inverse kinematics, thereby facilitating the computation of the corresponding initial and target joint angles. In Eq. (1), define $G_i(q)=\sum_{m=1}^{i}\|q_i[6]-q_{i-1}[6]\|$, and $H_i(q)=\max_{m=1,2\cdots 6}|q_i[m]-q_{des}[m]|$. The flowchart of the six-degree-of-freedom obstacle avoidance algorithm proposed by Jia is shown in Fig. 23.

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

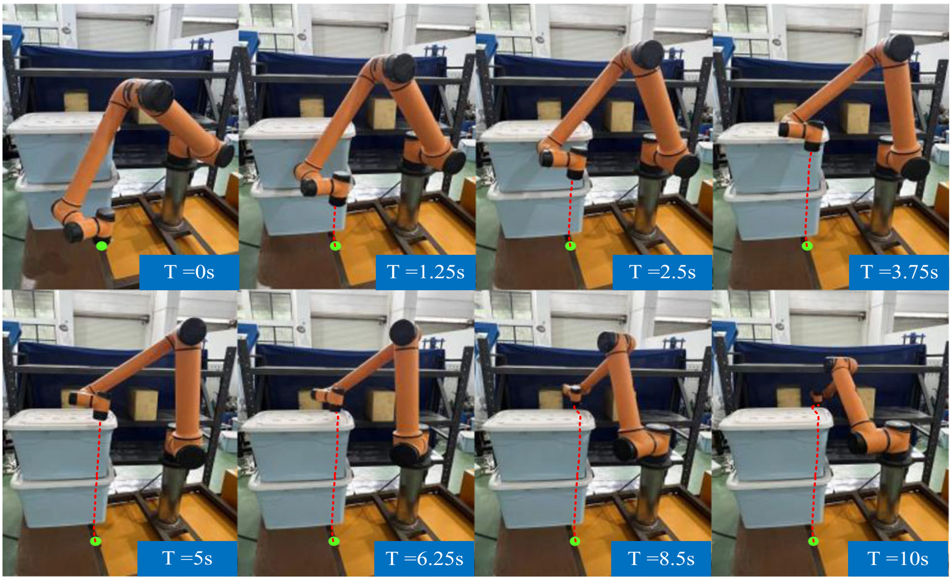

Figure 24. *Robotic arm obstacle avoidance experiment.*

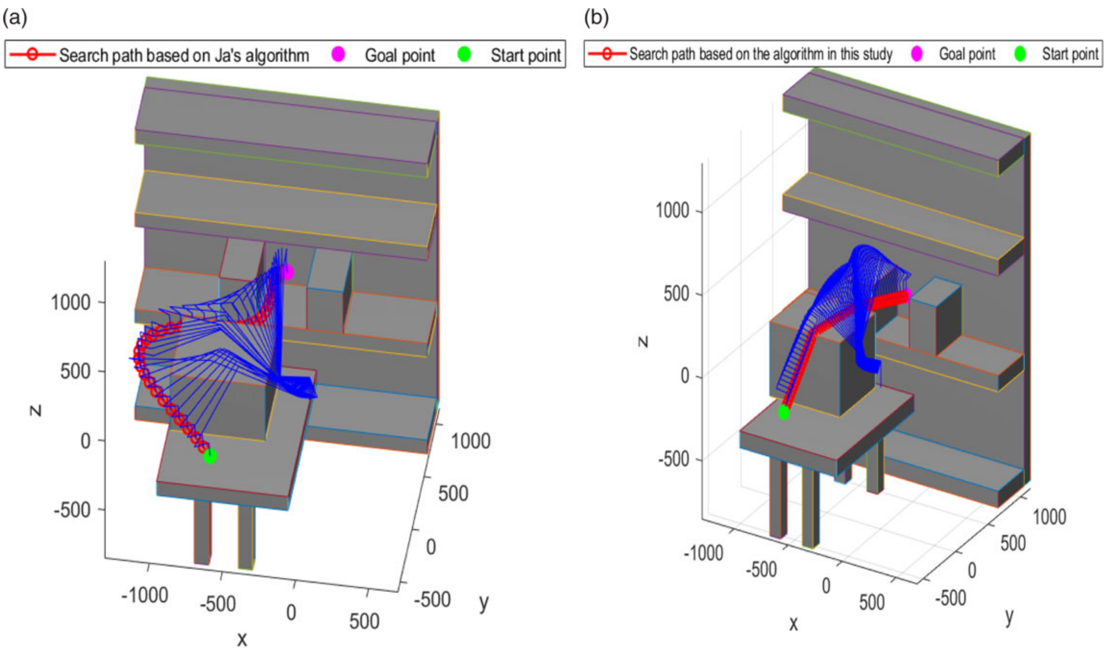

Figure 25. *Obstacle avoidance environment. (a) Path trajectory in Jia's method, (b) path trajectory in this study.*

The algorithm proposed in this study is used to conduct experiments on the AUBO-i10 robotic arm, and the arm's position at multiple points during its movement is recorded, as shown in Fig. 24. Figure 25(a) depicts the trajectory of the algorithm proposed in this study, while Fig. 25(b) depicts the trajectory of the algorithm proposed by Jia. Compared to the obstacle avoidance algorithm proposed by Jia, this study's algorithm has a significantly shorter path length in real 3D space.

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

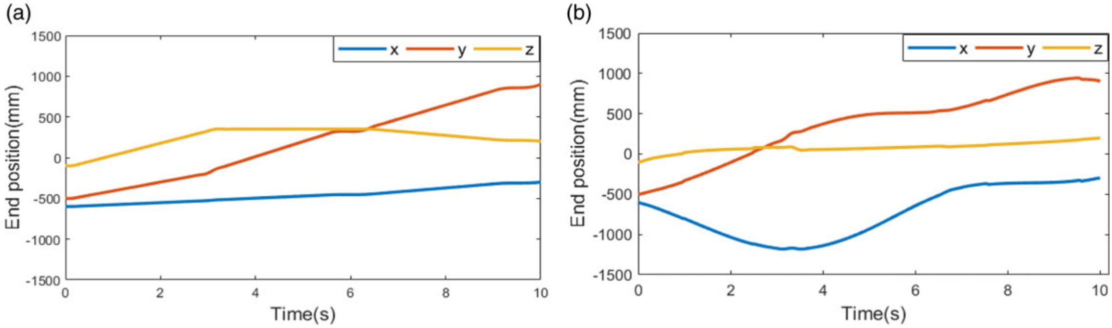

Figure 26. *Comparison of changes in end position. (a) The algorithm proposed in this paper, (b) the algorithm proposed by Jia.*

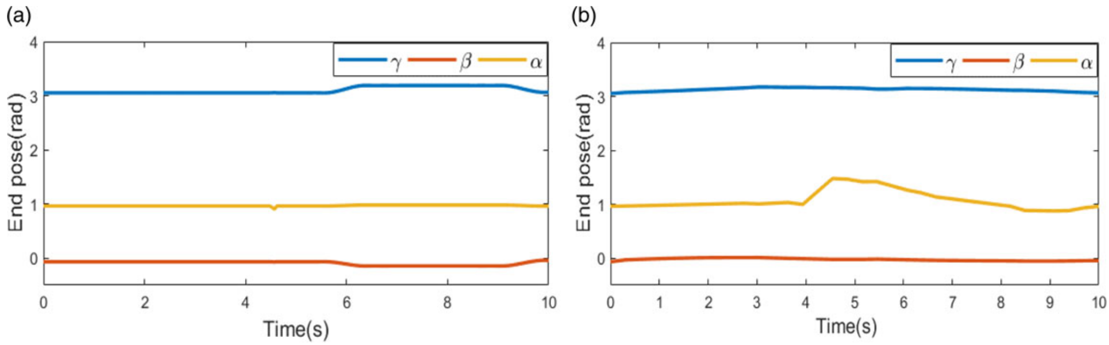

Figure 27. *Comparison of end-pose changes. (a) The algorithm proposed in this paper, (b) the algorithm proposed by Jia.*

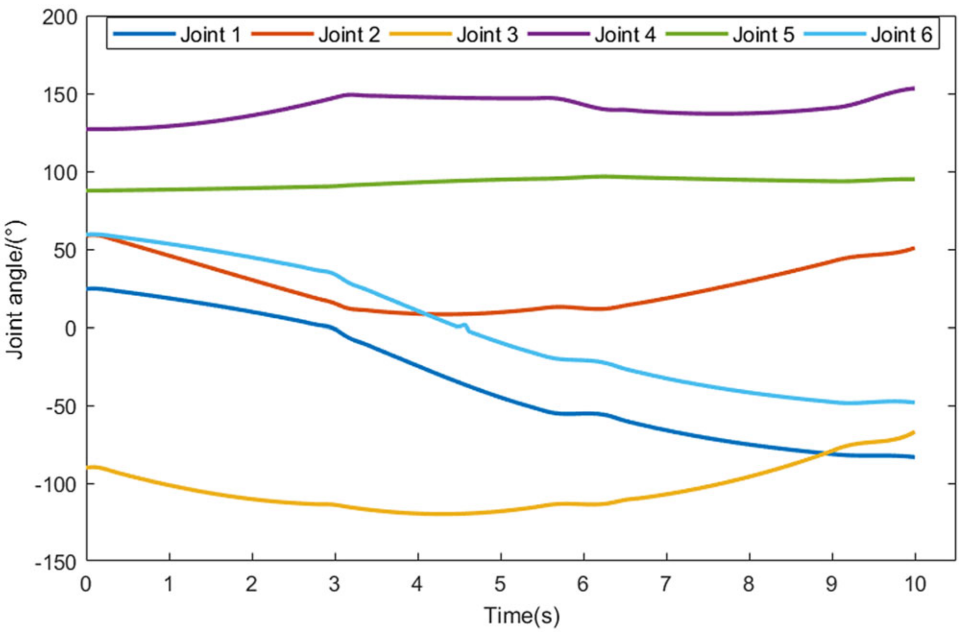

Figure 28. *Joint angle changes in the algorithm proposed in this study.*

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

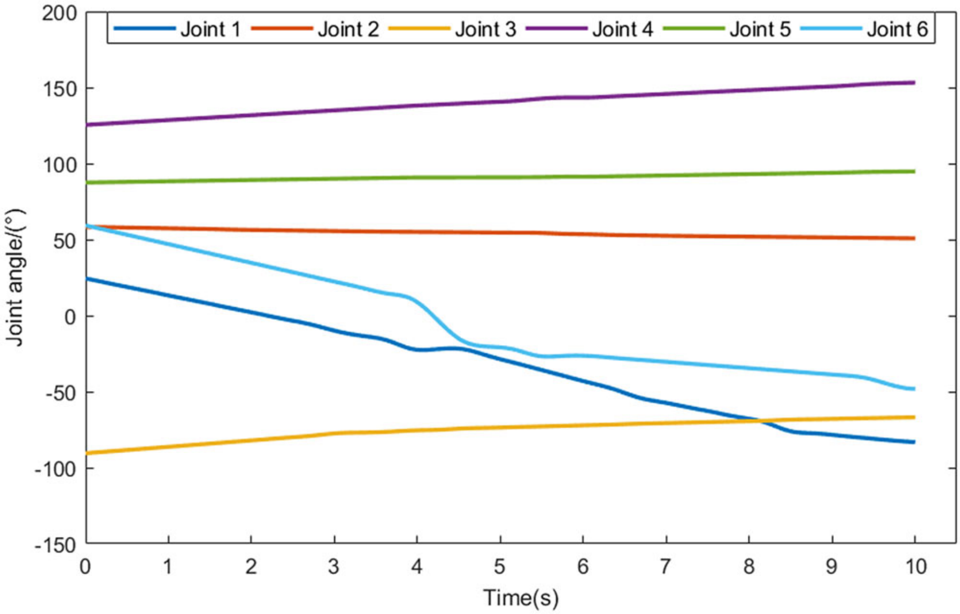

Figure 29. *Joint angle changes in the algorithm proposed by Jia.*

In this simulation, the search time for Jia's proposed algorithm is 35.386 s, while the search time for this study's algorithm is. 9.426 s. The primary reason for this is that the proposed method of this study is based on a three-dimensional positional space, and only three spatial positions must be altered during each search. However, Jia's method is based on six joint spaces, and each search requires changing six joint angles, which significantly increases the search algorithm's complexity. Under the same motion time, the search trajectories of the two algorithms are compared, and the changes in the end position, end pose, and joint angle are depicted in Figs. 26, 27, 28, and 29, respectively. According to Figs. 26 and 27, it can be determined that, compared to the algorithm proposed by Jia, the algorithm in this study has a smoother position change and posture change of the robotic arm in the three-dimensional space, resulting in less jitter at the end of the robotic arm in the actual motion space. According to Figs. 28 and 29, it can be determined that, compared to the algorithm proposed by Jia, the joint variation range of the algorithm proposed in this study is greater, and the joints in motion exhibit some jitter. This is because the algorithm in this paper searches in the 3D position space and solves the joint angles using inverse kinematics. In conclusion, the algorithm proposed in this study has superior performance in terms of search time, path length, and end position; however, the smoothing of joint angle changes should be enhanced.

5. Conclusion

When operating in three-dimensional environments, 6-DOF robotic arms commonly suffer from the time-consuming computation of obstacle avoidance algorithms, low flexibility of algorithms, and low adaptability to the environment. In this paper, a 6-DOF robotic arm obstacle avoidance path planning algorithm based on the improved A* algorithm and the artificial potential field method is proposed. The proposed improved A* algorithm is used for the path planning of the manipulator's end, which significantly improves the problems of numerous search nodes and low search efficiency that arise when the traditional A* algorithm is applied to 3D environment path planning. And the enhanced A* algorithm proposes a method for detecting node collisions and local path optimization. Then, based on the

https://doi.org/10.1017/S0263574723001546 Published online by Cambridge University Press

improved A* algorithm, a method for adjusting the manipulator's attitude using the artificial potential field method is proposed to prevent collisions between the robotic arm link and obstacles during movement. Simulation and experiments both validate the algorithm's practicability as described in the paper.

This paper proposes a 6-DOF robotic arm obstacle avoidance algorithm that is primarily used in static environments where obstacles are known and fixed. Nonetheless, the 6-DOF robotic arm must perform path planning in dynamic scenarios where the obstacles are not fully known. Future research will extend the obstacle avoidance method described in this paper to dynamic environments.

Author contributions. Xianxing Tang established the obstacle avoidance model and designed the path planning algorithm; he also drafted the manuscript. Tianying Xu carried out relevant experiments and data processing, and Haibo Zhou made suggestions and reviewed the manuscript.

Financial support. The authors would like to thank the National Natural Science Foundation of China for its financial support for research project No. 51975590.

Competing interests. All authors disclosed no relevant relationships.

Ethical approval. Not applicable.

References

[1] T. Xu, H. Zhou, S. Tan, Z. Li, X. Ju and Y. Peng, Mechanical arm obstacle avoidance path planning based on improved artificial potential field method," *Ind. Robot.* **2**, 49 (2022).
[2] C. kheireddine, A. Yassine, S. Fawzi and M. Khalil, "A robust synergetic controller for Quadrotor obstacle avoidance using Bézier curve versus B-spline trajectory generation," *Intel. Serv. Robot.* **15**(1), 143–15227 (2022).
[3] L. A. Trinh, M. Ekström and B. Cürüklü, "Dependable navigation for multiple autonomous robots with petri nets based congestion control and dynamic obstacle avoidance," *J. Intell. Robot. Syst.* **104**(4), 69 (2022).
[4] Z. He, C. Liu, X. Chu, R. R. Negenborn and Q. Wu, "Dynamic anti-collision A-star algorithm for multi-ship encounter situations," *Appl. Ocean. Res.* **118**, 102995 (2022).
[5] J. Ou, S. H. Hong, P. Ziehl and Y. Wang, "GPU-based global path planning using genetic algorithm with near corner initialization," *J. Intell. Robot. Syst.* **104**(2), 34 (2022).
[6] Z. Fang and X. Liang, "Intelligent obstacle avoidance path planning method for picking manipulator combined with artificial potential field method," *Ind. Robot.* **49**(5), 835–850 (2022).
[7] W. Lei, L. Ming, T. Dunbing and C. Jingcao, "Dynamic path planning for mobile robot based on improved genetic algorithm," *J. Nanjing Univ. Aeronaut. Astronaut.* **48**(06), 841–846 (2016).
[8] M. Elhoseny, A. Tharwat and A. E. Hassanien, "Bezier curve based path planning in a dynamic field using modified genetic algorithm," *J. Comput. Sci.* **25**, 339–350 (2018).
[9] A. Rs, B. Db and A. Nc, "Domain knowledge based genetic algorithms for mobile robot path planning having single and multiple targets," *J. King Saud Univ. Comput. Inf. Sci.* **34**(7), 4269–4283 (2022).
[10] K. S. Suresh, R. Venkatesan and S. Venugopal, "Mobile robot path planning using multi-objective genetic algorithm in industrial automation," *Soft Comput.* **26**(15), 7387–7400 (2022).
[11] J. Kennedy and R. Eberhart. Particle swarm optimization. **In:** *Proceedings of ICNN International Conference on Neural Networks*, **4**, (1995) pp. 1942–1948.
[12] B. Tang, Z. Zhu and J. Luo, "A convergence-guaranteed particle swarm optimization method for mobile robot global path planning," *Assembly Autom.* **37**(1), 114–129 (2017).
[13] H. Q. Jia, Z. Wei, X. He and L. Zhang, "Path planning based on improved particle swarm optimization algorithm," *Trans. Chin. Soc. Agric. Machin.* **49**(12), 371–377 (2018).
[14] B. Song, Z. Wang and L. Zou, "An improved pso algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve," *Appl. Soft Comput.* **100**(1), 106960 (2021).
[15] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(10), 993–1001 (2002).
[16] C. Miao, G. Chen, C. Yan and Y. Wu, "Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm," *Comput. Ind. Eng.* **156**, 107230 (2021).
[17] M. A. Pérez-Cutiño, F. Rodríguez, L. D. Pascual and J. M. Díaz-Báñez, "Ornithopter trajectory optimization with neural networks and random forest," *J. Intell. Robot. Syst.* **105**(1), 17 (2022).
[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *IEEE Int. Conf. Robot. Autom.* **2**, 500–505 (1985).

[19] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Auton. Robot.* **13**(3), 207–222 (2002).

[20] P. E. Hart, N. J. Nilsson and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cyb.* **4**(2), 100–107 (1968).

[21] A. Pal, R. Tiwari and A. Shukla. Multi Robot Exploration Using a Modified A* Algorithm. **In:** *International Conference on Intelligent Information & Database Systems*, Springer-Verlag (2011).

[22] Y. Y. Ren, X. R. Song and G. Song. Research on Path Planning of Mobile Robot Based on Improved A* in Special Environment. **In:** *IEEE International Symposium on Autonomous Systems*, Shanghai, China (2019) pp. 12–16.

[23] A. K. Guruji, H. Agarwal and D. K. Parsediya, "Time-efficient A* algorithm for robot path planning," *Proced. Technol.* **23**, 144–149 (2016).

[24] C. Li, X. Huang, J. Ding, K. Song and S. Lu, "Global path planning based on a bidirectional alternating search A* algorithm for mobile robots," *Comput. Ind. Eng.* (168-), 168 (2022).

[25] L. Zuo, Q. Guo, X. Xu and H. Fu, "A hierarchical path planning approach based on A* and least-squares policy iteration for mobile robots," *Neurocomputing* **170**(dec.25), 257–266 (2015).

[26] S. K. Wang and L. Zhu, "Motion planning method for obstacle avoidance of 6-DOF manipulator based on improved A* algorithm," *J. Donghua Univ. (Eng. Ed.)* **32**(1), 7 (2015).

[27] F. Bing, C. Lin, Y. Zhou, D. Zheng, Z. Wei, J. Dai and H. Pan, "An improved A* algorithm for the industrial robot path planning with high success rate and short length," *Robot. Auton. Syst.* **106**, 26–37 (2018).

[28] W. S. Newman and M. S. Branicky, "Real-time configurations for space transforms for obstacle avoidance," *Int. J. Robot. Res.* **10**(6), 650–667 (1991).

[29] Q. Jia, "Path planning for space manipulator to avoid obstacle based on A* algorithm," *J. Mech. Eng.* **46**(13), 109 (2010).

[30] N. Zhang, Y. Zhang, C. Ma and B. Wang. Path planning of six-DOF serial robots based on improved artificial potential field method. **In:** *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2017).

[31] L. Zhang, Y. Zhang, M. Zeng and Y. Li, "Robot navigation based on improved A* algorithm in dynamic environment," *Assembly Autom.* **41**(4), 419–430 (2021).

[32] X. Wang, M. Tang, M. Dinesh and R. Tong, "Efficient BVH-based collision detection scheme with ordering and restructuring," *Comput. Graph. Forum* **37**(2), 227–237 (2018).

[33] H. Liu, D. Qu, F. Xu, Z. Du, K. Jia, J. Song and M. Liu, "Real-time and efficient collision avoidance planning approach for safe human-robot interaction," *J. Intell. Robot. Syst.* **105**(4), 93 (2022).

Cite this article: X. Tang, H. Zhou and T. Xu (2024). "Obstacle avoidance path planning of 6-DOF robotic arm based on improved A* algorithm and artificial potential field method", Robotica **42**, 457–481. https://doi.org/10.1017/S0263574723001546