Hindawi
Mathematical Problems in Engineering
Volume 2024, Article ID 3866834, 11 pages
https://doi.org/10.1155/2024/3866834

Research Article

Obstacle Avoidance Path Planning of a 4-DOF Weapon Arm Based on Improved RRT (RRT-H) Algorithm

Kaifan Zou, Xiaorong Guan, Zhong Li, Huibin Li, Changlong Jiang, and Zihui Zhu

School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Correspondence should be addressed to Xiaorong Guan; gxr@njust.edu.cn

Received 10 October 2023; Revised 2 December 2023; Accepted 9 February 2024; Published 22 February 2024

Academic Editor: Rohit Salgotra

To improve soldiers' combat capability, weapon arms have a good development prospect. However, due to special work scenarios and tasks, new requirements are exerted on. Based on the fast-expanding random tree algorithm (RRT), path algorithm optimization (RRT-H) is proposed for the path planning of weapon arms. Overall path optimization is achieved by reducing the local path length with a closer path point planning to the obstacle. In a complex environment, the RRT-H algorithm can avoid local traps by guiding the new path extension direction and exploring multiple different paths in the map. The superiority of this algorithm is verified with 2D plane obstacle avoidance and pathfinding simulation experiments. Compared to RRT*, RRT* smart, and information RRT*, the RRT-H can obtain high-quality calculation results in a shorter time. After setting degrees of freedom (DOF) as that of variables, the algorithm is applied to the 4-DOF weapon arm, which confirms an effective reduction to the 4-DOF weapon arm's motion costs.

1. Introduction

As a kind of individual intelligent equipment, wearable weapon arms gain much academic attention since they can provide wearers with accurate striking ability and a fire cover, while reducing soldiers' training periods and training costs. Compared with traditional manipulators (mechanical arms) mainly used in manufacturing processes, wearable weapon arms are equipped with lethal weapons and coordinate with human beings in battlefield environments. A good path-planning method can effectively improve man-manipulator cooperation and, therefore, soldiers' target tracking capacity and self-protection. For a weapon arm, it requires the good path planning to accurately avoid the human body, while efficiently aiming and tracking the target. Weapon arm researches are still at the initial stage, and there is no comprehensive theoretical support, so it is necessary to carry out relevant researches in this field. Current algorithms for path planning include artificial potential field method, ant colony optimization, genetic algorithm, A-Star (A*), probabilistic roadmaps, and fast expanding random tree algorithm (usually referred to as RRT) [1–8]. Not requiring a detailed map to find a path, the RRT algorithm has computing advantages over other algorithms in the case of a multidimensional path solution of a manipulator. Yet, due to its random space exploration, there may not be a unique final solution, and most probably, they are not the optimal solutions [9]. In order to overcome these path planning limitations, many researches have improved RRT by optimizing various aspects such as the path search processes, path results, and path smoothness. The RRT* further repairs the path obtained by RRT, making the final path closer to the optimal solution [10]. Informed RRT* limits the new expansion points in the space where new expansion points can optimize the existing path [11]. Islam et al. [12] presented RRT* Smart, which limits new expansion points near the obstacles around the path, resulting in a further optimized path than those by those obtained by the former two algorithms. Burget et al. [13] introduced a bidirectional extended random tree search algorithm, which can better deal with the complex environment and improve the search efficiency. Zhou et al. [14] used the Dubins curve strategy to process the informed RRT* path and improved the path generated by the RRT algorithm. All of these improved RRT algorithms have been applied to many fields, manipulator path planning being one prominent area. Cai et al. [15]

proposed an improved rapidly exploring RRT for manipulator path planning, taking the target point as the sampling point with a certain probability. The algorithm improved the directivity of path planning. To improve obstacle avoidance, the step size can be changed according to the number of obstacles in the surrounding environment [15]. Similarly, Ma et al. [16] improved the advantage of RRT* in time and path, optimizing RRT* for the manipulator. You et al. [17] preserved the initial tree, took it as the basis of pathfinding, and applied it to 9-degrees of freedom (DOF) manipulator pathfinding, with a much improved the obstacle avoidance efficiency. Yang et al. [18] combined RRT with an improved artificial potential field, provided avoid-rapidly exploring random tree. This algorithm was applied to the path planning of the forestry harvesting manipulator, which improved the smoothness of the path and the calculation time [14].

As a peculiar subcategory in manipulators, path planning of weapon arms requires a shorter calculation time and a shorter path to reduce energy loss. To better adapting to weapon arms, RRT algorithm efficiency needs tailor-made improvements, with a further reduced calculation time and optimized final path results. This paper provides an algorithm optimization called RRT-H, which can be close to the optimal solution in a shorter time. RRT*, RRT* smart, and informed RRT* are selected for comparison. The path planning in 2D space is verified to prove that the RRT-H can obtain a more efficient path in less time and less points. In this algorithm verification operation, a 4-DOF weapon arm ensures that it does not affect soldiers in space, so it can be used as the main weapon arm structure. The 4-DOF weapon arm's path planning can effectively avoid obstacles along its movement with a shorter Euclidean distance of the end of the 4-DOF weapon arm simulation model.

In this paper, Section 2 introduces RRT and its improved algorithms, with RRT-H is introduced in more detail. Section 3 illustrates the performance of RRT*, RRT* smart, in-formed RRT*, and RRT-H in three maps, and the respective advantages and disadvantages. Section 4 proposes a tailor-made escape from local traps method is proposed, making up for a weakness in RRT-H. Section 5 conducts a simulation model of a 4-DOF weapon arm. Based on the 4-DOF weapon arm's characteristics. Section 6 first designs a collision detection model, and applies this 4-DOF weapon arm for the path planning simulation. Concluding remarks are made in Section 7.

2. RRT-Based Obstacle Avoidance Algorithms

This section introduces and analyzes RRT algorithms, with RRT*, informed RRT*, and RRT* smart being follow-ups, and the latter of which will be compared with optimized RRT-H algorithm.

2.1. RRT Algorithm. RRT algorithm is a sampling process-based search algorithm. Its random expansion process is shown in Figure 1.

The expansion principle is as follows: P_{start} and P_{goal} are the starting and ending points of the path. The point set T is used to store points of the tree, and T's first point is P_{start}. The

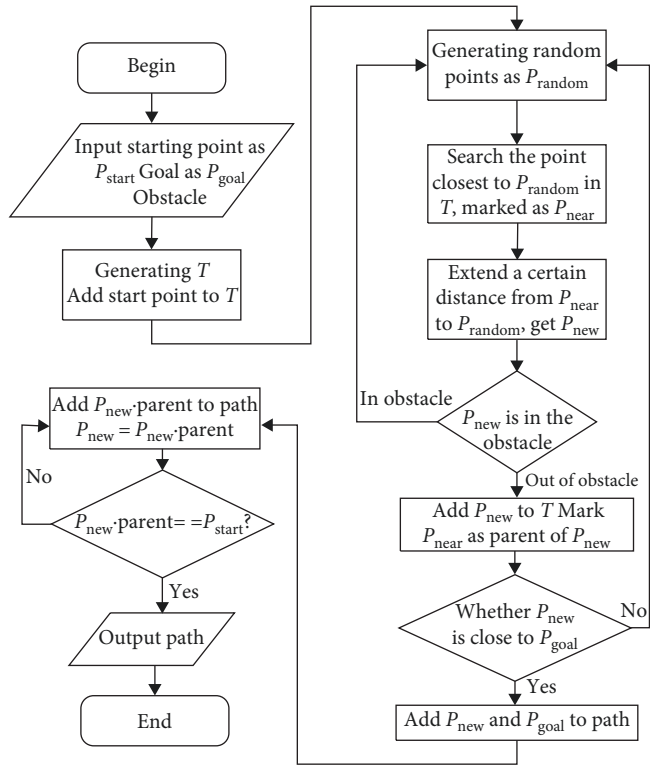

Figure 1: Random expansion process of RRT.

algorithm randomly generates P_{random} in the map. The point closest to P_{random} in T is marked as P_{near}, which is P_{new} is generated between P_{random} and P_{near}. If there is no collision between P_{new} and P_{near}, this P_{new} will be added to T, and this P_{near} will be marked as P_{parent} to P_{near}. When a generated P_{new} is close enough to P_{goal}, the new path will take the P_{new} and P_{goal} as the last two points, and P_{goal}'s P_{parent} is the P_{new}. Starting from the P_{goal}, the path is generated by finding each point's parent until the P_{start} is found.

Despite the great advantages of higher dimension obstacle avoidance, the path generated by the basic RRT algorithm is usually tortuous, with the possible failure of an optimal solution in the environment, a lengthened time, and unstable results.

2.2. Improved RRT Algorithms. To overcome the abovementioned weaknesses, many studies [9–12, 13] are being conducted to improve the RRT algorithm. RRT* is to add paths and points to optimize the original RRT-generated path (Figure 2).

After generating a new P_{new}, P_{parent}, and P_{next} will be selected to reduce the path cost. The points around P_{new} in T are marked as potential $P_{parents}$. The path length of each potential P_{parent} as parent to P_{new} are calculated and used to select the shortest P_{parent} as the best P_{parent}. Again, points around P_{new} in T are marked as potential P_{next}s. By calculating the path length of each potential P_{next} and comparing the path of the potential P_{next} with the original parent point, the shortest path will be selected, and this P_{new} will be updated as the parent point to P_{next}.

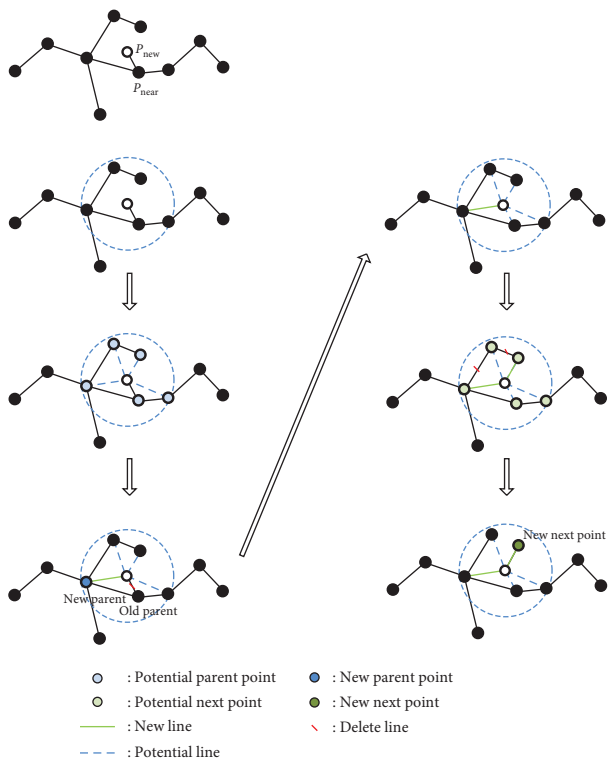

FIGURE 2: The optimization process of RRT*.

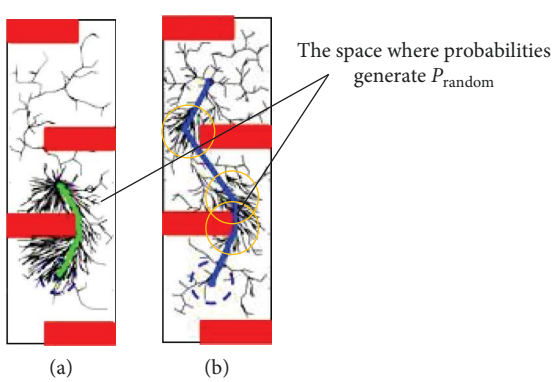

FIGURE 3: Simulation results of (a) informed RRT* and (b) RRT* smart.

Informed RRT* and RRT* smart further optimize the RRT* algorithm, both intelligently selecting the P_{random} (Figure 3). Simulational result from informed RRT* is shown in Figure 3(a). Informed RRT* algorithm immediately calculates the path cost as soon as it obtains the first path. Path cost sum from subsequent P_{random} –P_{start} and from the subsequent P_{random}–P_{goal} should not be higher than that of the existing path. In this way, a space is obtained, and the generated points in the space can optimize the path. For RRT* smart, after obtaining the first path RRT* smart algorithm simplifies the path. With reference to simplified results, it then analyzes the obstacle position according to the simplification results and generates a new P_{random} near the obstacle. RRT* smart simulation is shown in Figure 3(b). Both of them

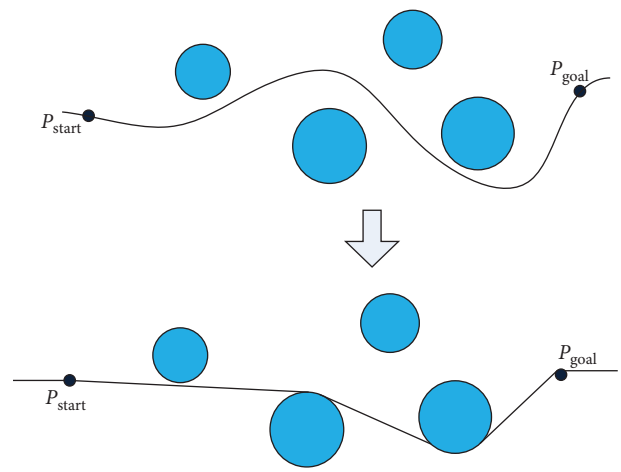

FIGURE 4: Schematic diagrams of RRT-H.

further optimize RRTRRT* algorithm but increase the calculation cost and slow down the pathfinding speed.

To maintain the previous advantages and overcome disadvantages, once the first is found, it will be directly and immediately optimized in RRT-H. It functions as if placing a curve between obstacles; stretching the two ends of a line will shorten the path within (Figure 4).

It takes three steps of optimization processes, and step one is illustrated in Figure 5.

As is shown in Figure 5, the first step is to optimize the existing path. Step 1 takes three consecutive points: P_i, P_{i+1}, and P_{i+2} in the existing path. Multiple points distributed along the lines of P_{i+1} and P_{i+2}, respectively, are marked as P_{Ts}. P_{Ts} are connected with P_i to generate connection lines L_{Ts}. According to the lengths of L_{Ts}, step one evenly distributes point P_{Cs} on each L_T. If any of P_{Cs} has no collision with the obstacles, delete P_{i+1} and the path between P_i,P_{i+1}, and P_{i+2}, connect P_i and P_{i+2} as the new path, mark P_i as a new P_{i+1}, $i = i+1$. If there is a collision in $P_T(n)$, mark $P_T(n-1)$ as P_{i+1}, $i = i+1$. The optimization operation goes on. Reaching the P_{goal}, the first step of optimization is over.

Then, starting from the P_{goal}, the second round of path optimization is carried out in the same way. The first two steps lay a solid foundation for the third step, optimization. The third step is shown in Figure 6.

As is shown in Figure 6, step three selects three consecutive points P_i, P_{i+1}, and P_{i+2} in the path, similar to that of the first two optimization steps. The path of P_{i+1} and P_{i+2} generates multiple points P_T, which are, respectively, connected with P_i, and then the points P_{Cs} are evenly distributed on L_T. Collision detection is carried out for each P_C. Assuming that the first collision position is the $P_C(m)$ on $L_T(n)$, and $L_T(n)$ is the path between $P_T(n)$ and P_i, enter $P_C(m)$ of $L_T(n-2)$ into the path set, and mark $P_C(m$ of $L_T(n-2)$ as P_i, $P_T(n)$ as P_{i+1}, then optimize P_i, P_{i+1}, and P_{i+2} again. Distribution gaps between detection points on the path make it liable for sharp obstacles permeate in between, as is shown in Figure 7, which gives misjudgment of "no collision." In this case, the optimization will fall into an endless loop, and the distance between P_i and P_{i+1} will be reduced close to 0.

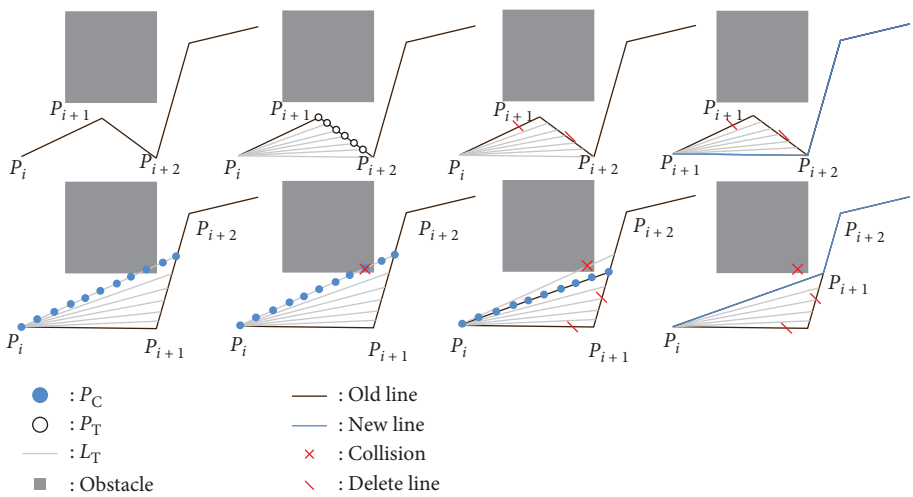

FIGURE 5: Schematic diagrams of the first step optimization.

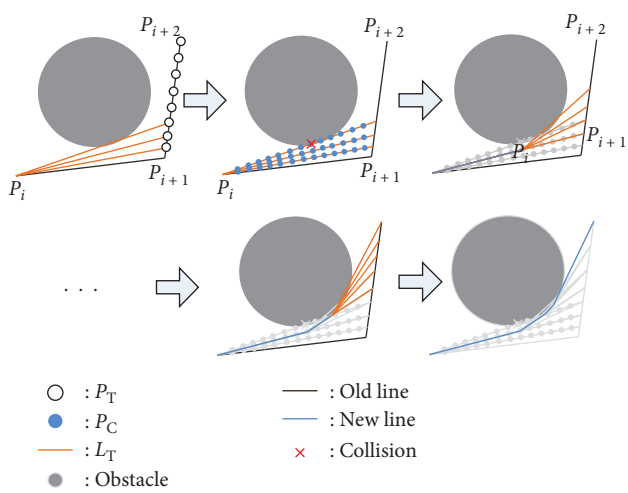

FIGURE 6: The schematic diagram of the third step optimization.

In order to avoid this problem, the judgment points distribution should be set as dense as possible. In order to avoid the occurrence of extreme conditions when the density of judgment points distribution is small enough, k is set as the minimum distance permitted between P_i and P_{i+1}. If the calculated distance between P_i and P_{i+1} is less than k, then $i = i + 1$. Meanwhile, the path passes through the sharp parts of the obstacle. An endless loop is thus avoided, and a new path optimization starts, until getting the final optimal solution. The calculation time of the algorithm will increase with the increase of P_C density and the decrease of the k value. A balanced distribution density of P_C and a value of k are helpful for the algorithm to achieve satisfactory results (Figure 7).

According to the obstacle avoidance strategy of weapon arms in this paper (Section 5), there will be a certain fault tolerance space between the edge of obstacles and the human body. The method for jumping out of the endless loop will not affect the obstacle avoidance effect if the distribution density of points used for collision detection is reasonably planned.

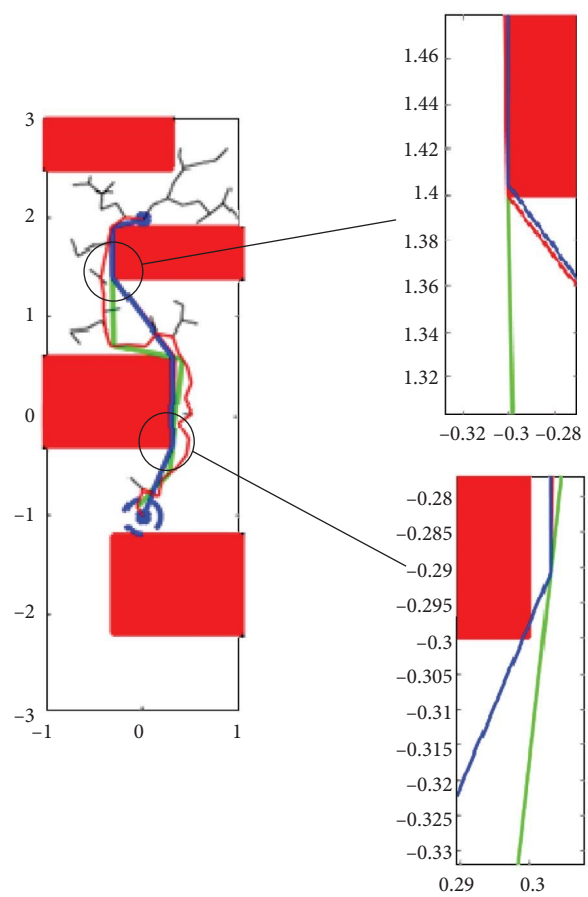

FIGURE 7: Sharp obstacles permeation between two detection points.

3. Comparison of Path Planning Effects

With referring to the maps used in other papers [19–22], three kinds of maps are designed in this paper, which are C-shaped, spiral-shaped, and fence-shaped, respectively. These maps in this paper are only used to prove the algorithm's superiority, ignoring map complexity influences (Figure 8).

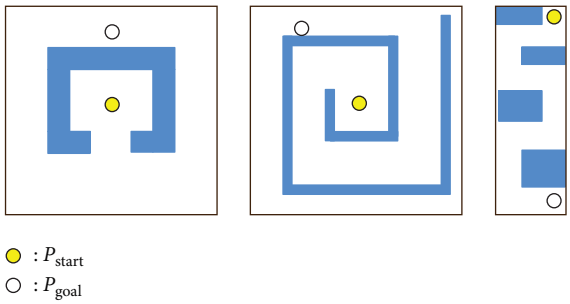

Figure 8: Shapes of the pathfinding maps.

In this paper, the algorithms are compared in terms of calculation time, number of P_{random}s, and the generated path length. Both RRT and RRT-H algorithm in this paper stop the calculation after finding the path, while RRT*, RRT* smart, and informed RRT* stop the calculation after reaching 2,000 steps. Figure 9 displays the results of the five algorithms in the above three different environments.

As is shown in Figure 9, all algorithms can obtain a basic path in a short time; the paths of RRT* smart and RRT-H are shorter, that of RRT* and informed RRT* algorithms being approximately longer. In the above three types of maps, the first path obtained by informed RRT* is large. According to the principle of informed RRT* algorithm, the optimization space is planned according to the first path's length. In the above three maps, the optimization space covers almost the entire map. Therefore, RRT* and informed RRT* would show similar performances within 2,000 steps. Thus, the informed RRT* algorithm does not have any obvious advantage in the tortuous path planning.

The path cost of RRT*, RRT-H, RRT* smart, and informed RRT* vary with the number of P_{random}s (Figure 10(a)). In order to highlight the advantages of the informed RRT*, new P_{start} and P_{goal} are defined, and the path planning effect is shown in Figure 10(b).

RRT-H algorithm does not need a new P_{random} once it finds a path. In the above simulation experiment, the RRT-H algorithm can reach the last P_{random} within 200 points without any new P_{random}. Also, the RRT-H algorithm has the smallest P_{random} number in the four algorithms. RRT-H has a longer path than the other three before optimization, while a shorter one after optimization.

Because the asymptotic optimality of the RRT*, RRT* smart, and informed RRT*, the longer the time consumed, the better the result they have. The time consumption of RRT* smart, informed RRT*, and RRT-H here is compared with that of informed RRT* algorithm, to show the relationship between their respective path cost and time. Based on RRT algo, the RRT-H algorithm has a high cost at the beginning and gradually decreases with the three-step optimization. After the optimization, it can quickly exceed the other three ones and reaches a smaller path cost (Figure 11).

Adopting the common practice, each algorithm takes ten groups of data for analysis. Of all four algorithms, the average path cost and time consumption are obtained from 10 experimental data groups and compared within the same map. Then, RRT-H result histograms are compared with the 2,000 steps results of the other three algorithms results. RRT-H algorithm can optimize the path to a better result in a shorter time compared with other algorithms (Figure 12).

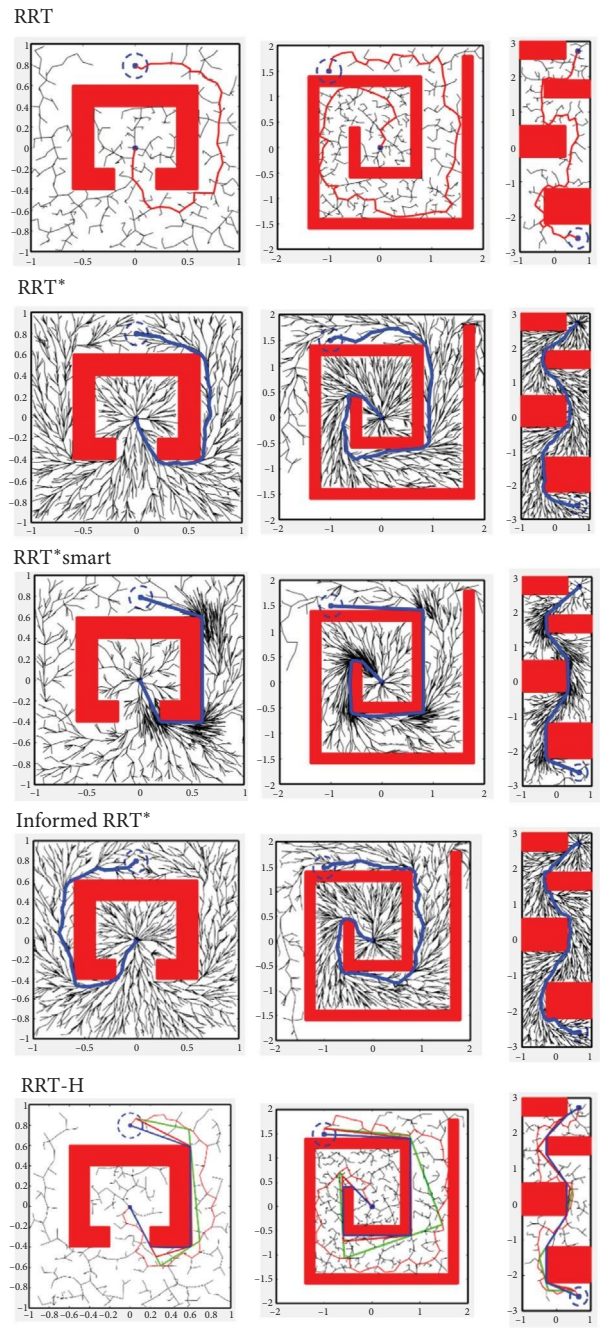

Figure 9: Results from different algorithms in three maps.

4. A Tailor-Made Escape from Local Traps by RRT-H

RRT*, RRT* smart, and informed RRT* have the ability to avoid local traps. In particular, the informed RRT* algorithm can effectively jump out of local traps and obtain globally optimal solutions. RRT* algorithm is not limited by local traps due to

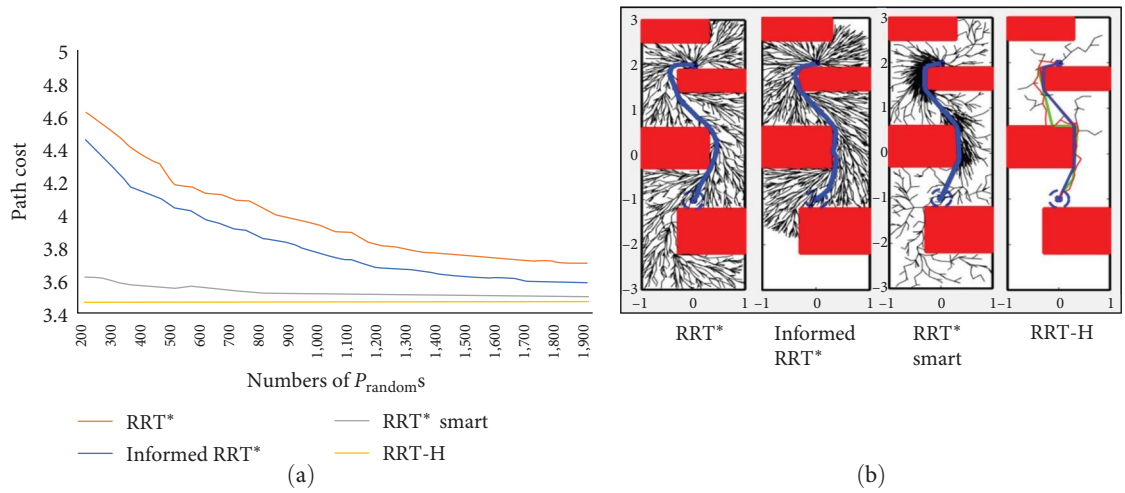

FIGURE 10: (a) Path costs against the number of P_{random}s and (b) path planning effect with new P_{start} and P_{goal}.

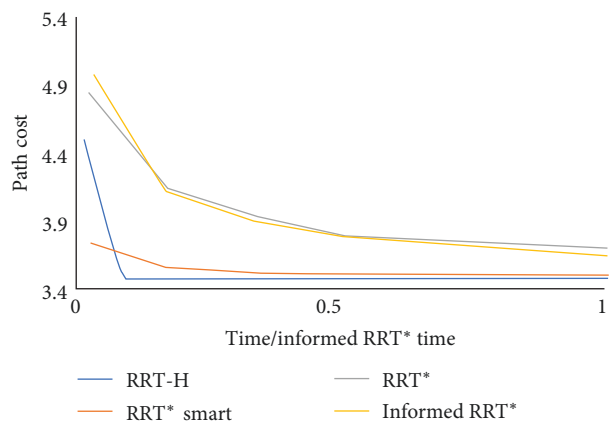

FIGURE 11: Relationships between path cost and time consumption.

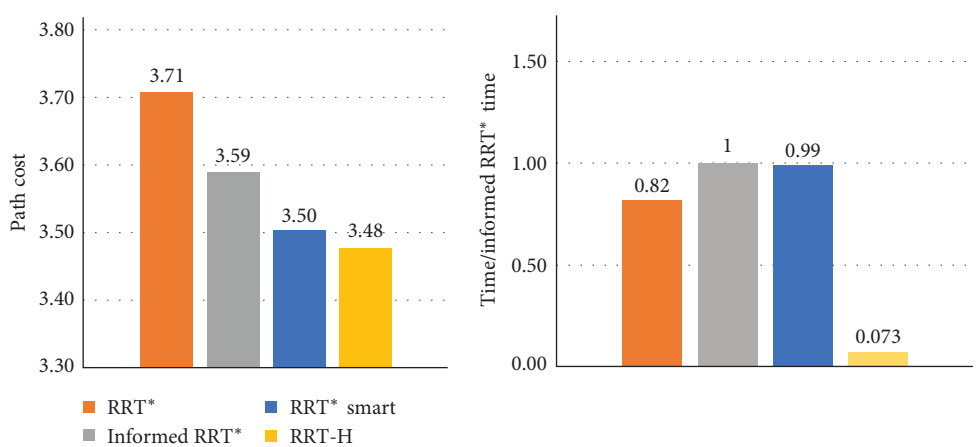

FIGURE 12: Path cost and time cost histograms of different algorithms.

P_{random}s, but it needs a large number of P_{random} to complete path comparison. RRT* smart is based on the paths found by RRT*. It is easier to fall into local traps, but there is still a certain probability to jump out of local traps due to the probability distribution of P_{random}s out of optimization space. Because the RRT-H algorithm directly optimizes the existing path, it is easy to fall into the local traps, so it optimizes the P_{random}s generating method here (Figure 13).

As is shown in Figure 13, the points on the existing path (before optimization) are entered into the path club. When a new P_{random} is generated, this P_{random} will be filtered according to the probability. The method searches a range near the

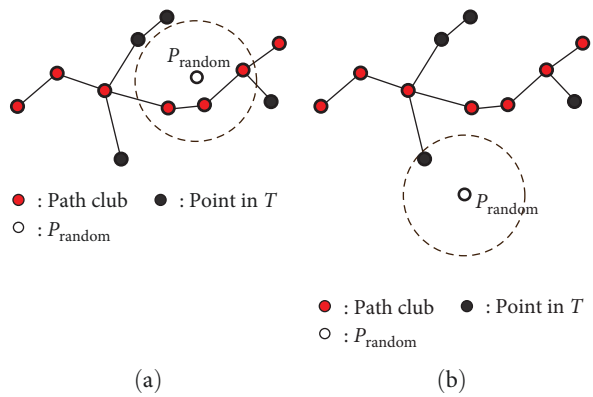

FIGURE 13: The escaping method from local traps. *The P_{random} in (a) is invalid, and the P_{random} in (b) is added to T.

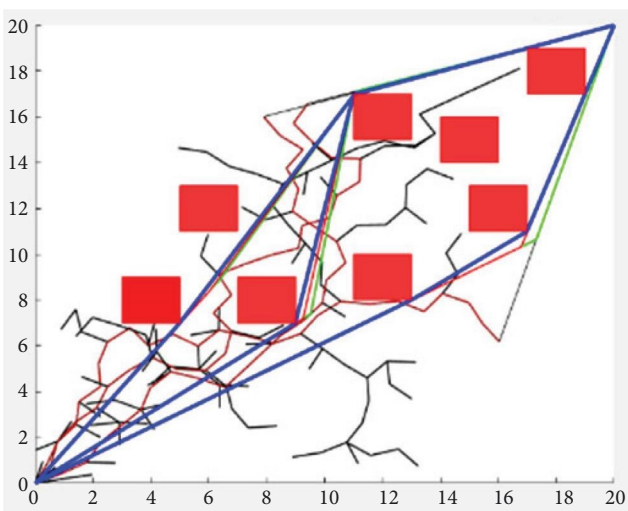

FIGURE 14: The optimization effect for escaping from local traps.

P_{new}. When the number of points in the path club is large, the new P_{random} is invalid, and it generates a new P_{random} again. The optimization effect of this method is shown in Figure 14.

The algorithm will plan scattered multiple paths, which improves the map exploration efficiency and avoids falling into local traps to a certain extent.

Because RRT-H is easy to fall into local traps, the problem can be effectively avoided by this escape method in a complex environment, which can provide a better path planning effect when applied to the weapon arm.

5. Path Planning Modeling of a 4-DOF Weapon Arm

This 4-DOF weapon arm model is based on preliminary researches in the Lab for Individual Equipment Technology, Nanjing University of Science and Technology. According to weapon arm requirements, 4-DOF is defined to provide the weapon arm with an omnidirectional strike capability (Figure 15).

As long as a target point is on the outside ballistic track, the 4-DOF weapon arm can hit the target, regardless of the

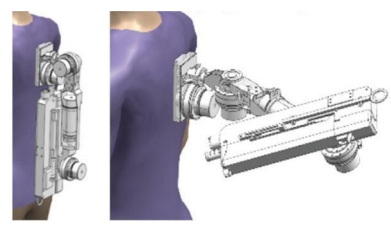

FIGURE 15: A 4-DOF weapon arm model.

position and posture of the weapon itself to which a weapon arm is attached. Therefore, the weapon system at the end of the 4-DOF weapon arm is regarded as a translational DOF, and the 4-DOF weapon arm's joints are regarded as rotational DOF. When the end of a translational DOF coincides with the target position, the position and posture of the weapon arm are regarded as the achievable strike posture. The arm can thus be regarded as a 5-DOF manipulator for kinematics modeling. The 5-DOF manipulator model is only used for the path planning algorithm here, and for other situations, such as actual prototypes, it is a 4-DOF model.

In this paper, the weapon arm space posture is described with the method of modified Denavit–Hartenberg (D–H) parameters. The D–H coordinate modeling is shown in Figure 16, and the D–H parameters are shown in Table 1. Then, the kinematics simulation model of the 4-DOF weapon arm is constructed and adopted for obstacle avoidance path planning, together with the obstacle position information.

6. RRT-H-based 4-DOF Weapon Arm Path Planning Algorithm

In order to apply the RRT-H algorithm to the 4-DOF weapon arm, algorithm variables are increased from 2 in the 2D space to 5 in the weapon arm simulation model. A collision detection model is set, aiming at the working conditions of the 4-DOF weapon arm. Simulation results confirm that the algorithm can be applied to an efficient path planning of the 4-DOF weapon arm.

6.1. Collision Detection of the 4-DOF Weapon Arm. With a relative position stability of both the weapon arm and the

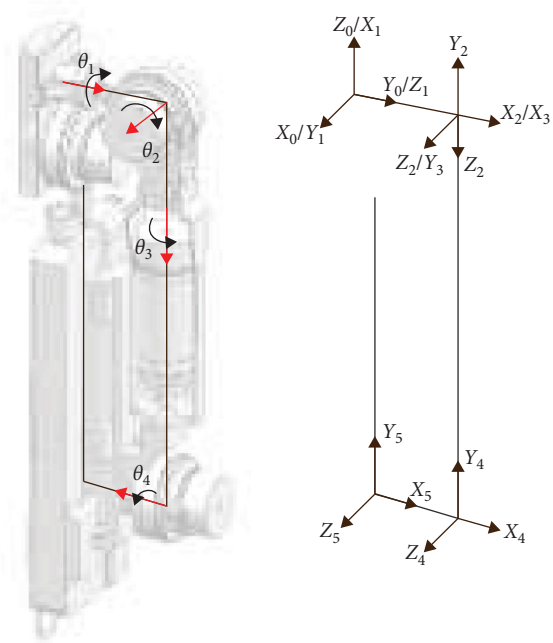

Figure 16: D–H coordinate modeling of the 4-DOF weapon arm.

Table 1: D–H parameters of 4-DOF weapon arm.

Link	Theta	d	a	Alpha (pi)	Offset (pi)	Joint type	Ranges
L1	$\theta 1$	0.126	0	−0.5*	−0.5*	Rotary joint	(−pi, pi)
L2	$\theta 2$	0	0	0.5*	−0.5*	Rotary joint	(−pi, 0])
L3	$\theta 3$	0.386	0	0.5*	0.5*	Rotary joint	(−pi, pi)
L4	$\theta 4$	0.038	0	0.5*	0	Rotary joint	(−pi, pi)
L5	0	1	0	−0.5*	0	Rectilinear motion joint	(0–∞)

human body, it is unnecessary to build a detailed human body model as avoidance obstacles. The weapon arm is simplified as a lever structure without volume, and the human body shape is simplified as a bounding tree composed of multiple regular cuboids. The whole obstacle avoidance issue thus is simplified as a lever–cuboid collision detection. Considering the safety and its volume ignorance in the weapon arm simplification, the cuboid representing the human body is moderately extended a certain distance larger than the actual human body. The collision simplification method is shown in Figure 17.

Figure 18 is the schematic diagram of the collision detection principle. The midpoint P_O of each cuboid face was taken as the starting point. Each face generates a vector perpendicular to the face and pointing to the outside of the cuboid. P_{Cs} on the manipulator are used as collision detection points and are taken as the vectors $D_{Ss'}$ starting points, with the direction P_O–P_C. The dot products P_S of S_S and D_S are used to judge whether there is a collision as follows:

$$P = \sum_{i=1}^{6} \frac{\overrightarrow{S_i} \cdot \overrightarrow{D_i}}{\left|\overrightarrow{S_i} \cdot \overrightarrow{D_i}\right|}, \tag{1}$$

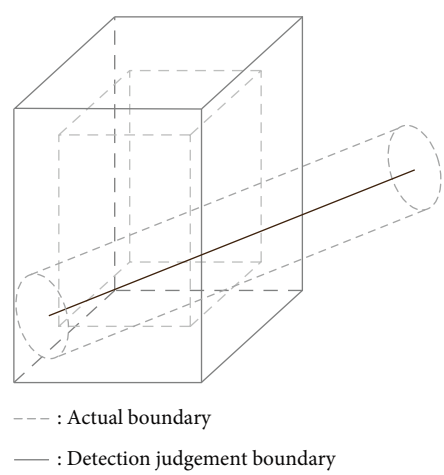

Figure 17: Collision model simplification.

$$\text{col} = \sum_{j=1}^{m} P_j, \tag{2}$$

where m is the number of points in P_C. Only when col $= 6m$, there is no collision between the weapon arm and the human body in this posture.

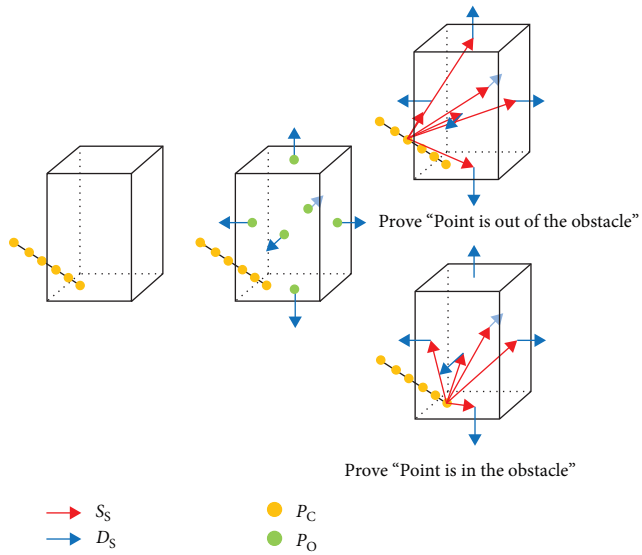

Figure 18: Schematic diagram of the collision detection principle.

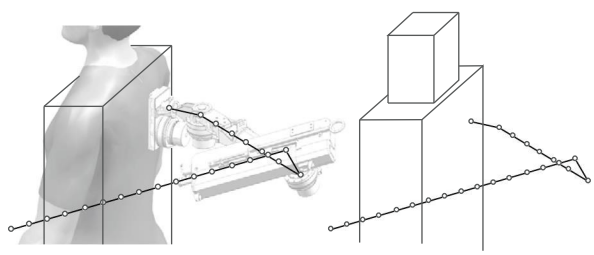

Figure 19: Extended lever and cuboid.

6.2. Obstacle Avoidance Strategy of the 4-DOF Weapon Arm. Given that the weapon arm is equipped with lethal weapons at its end and that it coordinates with the human wearer, the human body must be avoided, and its gun muzzle should never point at the human body in its operation process. Therefore, it is necessary to consider extending a certain distance forward at the gun muzzle. Since the length of the weapon arm in the model is short, 1 m is taken in front of the muzzle to add 10 points as P_C to participate in the collision detection in the path planning process (Figure 19).

The algorithm transforms the path from the folded state to the target-directed state by setting the same target position. The algorithm is applied to the path planning of a wearable weapon arm system, and eight path plannings are performed. The paths are then subjected to interpolation and smoothing. The resulting path is shown in Figure 20.

As shown in Figure 20, the RRT algorithm has some randomness in its computation results, while RRT-H is optimized and has relatively more stable results. The average cumulative angle changes of the 4-DOF in the path for RRT are 181.28, 172.57, 212.57, and 140.38, respectively, while for RRT-H, they are 88.38, 69.16, 92.00, and 93.89, respectively. RRT-H exhibits significantly higher path quality compared to the traditional RRT algorithm. The average optimization degree for the 4-DOF is 51.24%, 59.92%, 56.72%, and 33.11%, respectively, resulting in an overall optimization effect of 51.4%. The optimization effect is evident. In the workspace, the planning effects of the two path planning algorithms for multi-DOF manipulators are shown in Figure 21. It can be concluded that the RRT-H algorithm is meaningful for obstacle avoidance planning in multi-DOF robotic arms.

7. Conclusion

In this paper, a new RRT optimization algorithm (RRT-H) is proposed. It has obvious advantages for the environment with obstacles. RRT-H can effectively improve the pathfinding efficiency and obstacle avoidance in the given three maps. It can find the shortest obstacle avoidance path in a short time, and can avoid the local traps by tailor-made method in a complex environment. The algorithm is applied to a 4-DOF weapon arm system, which can effectively reduce the path search time and path length. The RRT-H algorithm in this paper caters to the rapid reaction needs of weapon arms with their better integration. Yet, due attention must be paid to the simplicity of the maps in this paper. RRT-H algorithm may process meaningless computations within complex maps, owing to numerous judgment points generated on the path. Increased time cost becomes inevitable. In future studies, more complex environmental information will be considered when generating collision detection points. Improvements should be directed at

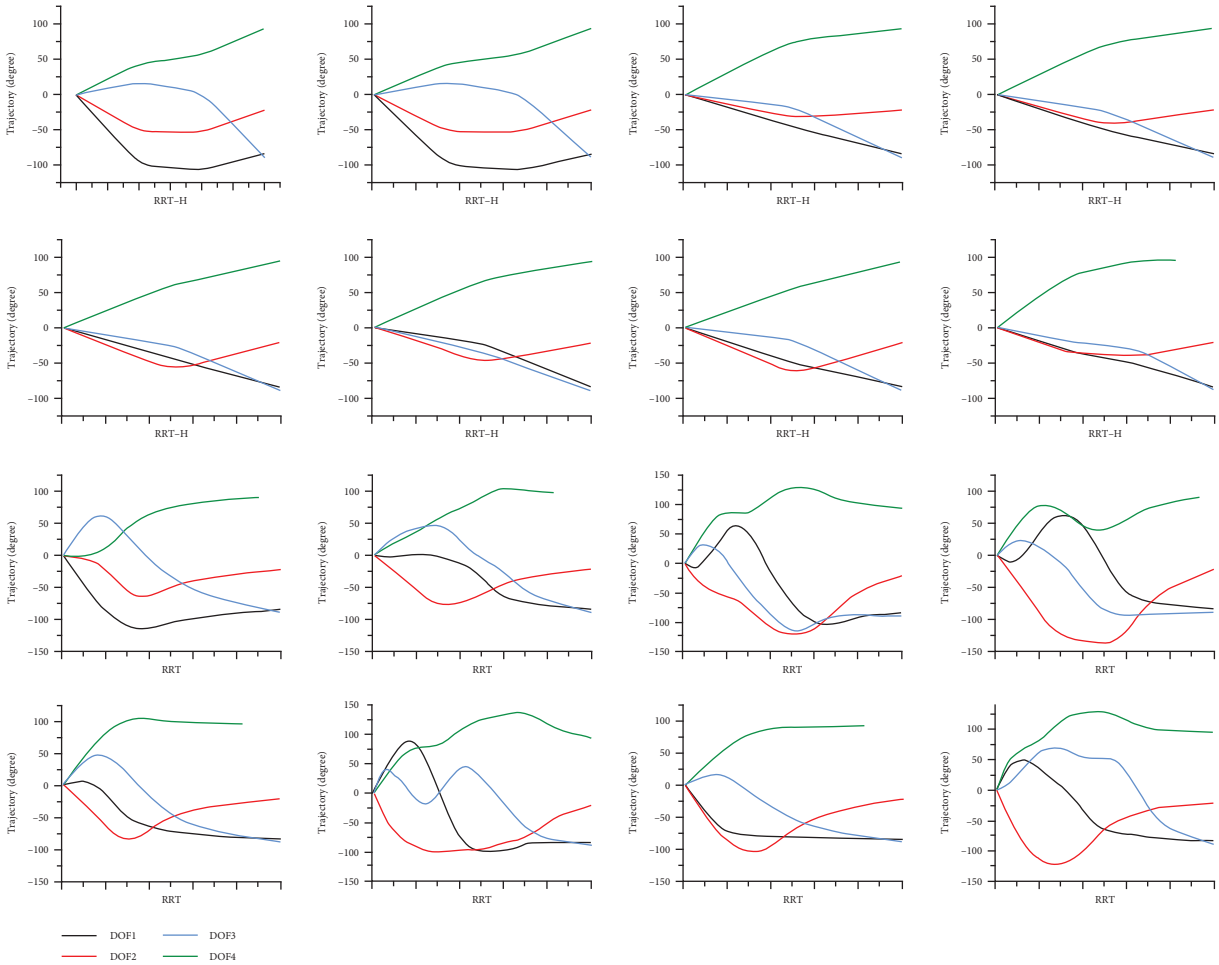

Figure 20: Path results of RRT-H and RRT.

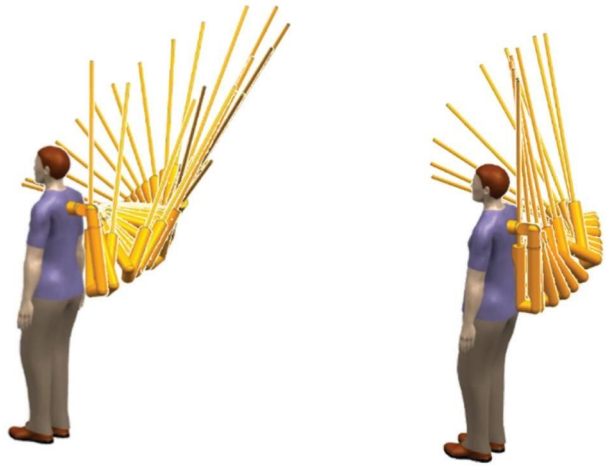

Figure 21: Simulation effects of RRT and RRT-H.

RRT-H's adaptability and computational efficiency for different maps. Besides, RRT-H only considers the muzzle safety, so further studies are called for to deal with issues like more complex weapon arm characteristics and the ejecting process. In further research, the algorithm will optimize the Prandoms generation position and collision detection point distribution, further improve the algorithm's speed and the complex environment adaptability.

Data Availability

The Python codes used and analyzed during the study are available from the first author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Kaifan Zou, Zhong Li, Huibin Li, Zihui Zhu, and Changlong Jiang contributed to the design and implementation of the research and the analysis of the results. Kaifan Zou contributed to the writing of the first draft of the manuscript. Xiaorong Guan commented on previous versions of the manuscript.

Acknowledgments

This research was funded by the National Defense Basic Scientific Research Program of China, grant number JCKY2019209B003. Thanks to the experiment participants for their help and the Nanjing University of Science and Technology for its support to this study.

References

[1] M. Brand, M. Masuda, N. Wehner, and X.-H. Yu, "Ant Colony Optimization algorithm for robot path planning," in *2010 International Conference on Computer Design and Applications*, pp. V3-436–V3-440, IEEE, Qinhuangdao, China, 2010.

[2] M. S. A. D. Ali, N. R. Babu, and K. Varghese, "Collision free path planning of cooperative crane manipulators using genetic algorithm," *Journal of Computing in Civil Engineering*, vol. 19, no. 2, pp. 182–193, 2005.

[3] Y. Peng and W. Wei, "A new trajectory planning method of redundant manipulator based on adaptive simulated annealing genetic algorithm (ASAGA)," in *2006 International Conference on Computational Intelligence and Security*, pp. 262–265, IEEE, Guangzhou, China, 2006.

[4] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[5] S. Q. Zheng, "Path planning for space manipulator to avoid obstacle based on A* algorithm," *Journal of Mechanical Engineering*, vol. 46, no. 13, Article ID 109, 2010.

[6] J. Fei, G. Chen, Q. Jia, and D. Liu, "Obstacle avoidance path planning for space manipulator based on improved probability roadmap method," in *14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 1963–1968, IEEE, Xi'an, China, 2019.

[7] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[8] S. M. Lavalle, "Rapidly-exploring random trees: a new tool for path planning," Research report, 1999.

[9] M. Du, J. Chen, P. Zhao, H. Liang, Y. Xin, and T. Mei, "An improved RRT-based motion planner for autonomous vehicle in cluttered environments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4674–4679, IEEE, Hong Kong, China, 2014.

[10] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2997–3004, IEEE, Chicago, IL, USA, 2014.

[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[12] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "RRT*-Smart: rapid convergence implementation of RRT* towards optimal solution," in *2012 IEEE International Conference on Mechatronics and Automation*, pp. 1651–1656, IEEE, Chengdu, China, 2012.

[13] F. Burget, M. Bennewitz, and W. Burgard, "BI2RRT*: an efficient sampling-based path planning framework for task-constrained mobile manipulation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3714–3721, IEEE, Daejeon, Korea (South), 2016.

[14] H. X. Zhou, Y. Cheng, and W. C. Liu, "Manipulator motion planning based on dubins-informedRRT* algorithm," *Techniques of Automation and Applications*, vol. 39, no. 10, pp. 67–74, 2020.

[15] W. T. Cai, Q. Deng, J. Zhang, Y. B. Zhang, S. Rao, and K. Yang, "Manipulator path planning based on improved RRT algorithm," *Transducer and Microsystem Technologies*, vol. 38, no. 5, pp. 121–124, 2019.

[16] H. L. Ma, Z. Q. Lu, and S. T. Wang, "Research on manipulator path planning based on improved RRT* algorithms," *Machine Design and Research*, vol. 36, no. 4, pp. 42–46, 2020.

[17] H. Y. You, S. He, H. W. Liu, and Z. B. Xu, "The mechanical arm of 9-DOF path planning based on bi-RRT algorithm," *Computer Simulation*, vol. 36, no. 7, pp. 308–313, 2019.

[18] Y. Yang, J. Liu, Y. Zheng, and Q. Huang, "Obstacle avoidance path planning of manipulator of forestry felling & cultivation machine," *Scientia Silvae Sinicae*, vol. 57, no. 2, pp. 179–192, 2021.

[19] R. Mashayekhi, M. Y. I. Idris, M. H. Anisi, and I. Ahmedy, "Hybrid RRT: a semi-dual-tree rrt-based motion planner," *IEEE Access*, vol. 8, pp. 18658–18668.

[20] J. Wang, M. Q.-H. Meng, and O. Khatib, "EB-RRT: optimal motion planning for mobile robots," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 2063–2073, 2020.

[21] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "PQ-RRT*: an improved path planning algorithm for mobile robots," *Expert Systems with Applications*, vol. 152, pp. 1–11, 2020.

[22] O. Adiyatov and H. A. Varol, "Rapidly-exploring random tree based memory efficient motion planning," in *2013 IEEE International Conference on Mechatronics and Automation*, pp. 354–359, IEEE, Takamatsu, Japan, 2013.