applied sciences

MDPI

Article

An Efficient RRT Algorithm for Motion Planning of Live-Line Maintenance Robots

Jiabo Feng and Weijun Zhang *

School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; fjbfei@sjtu.edu.cn
* Correspondence: zhangweijun@sjtu.edu.cn

Abstract: The application of robots to replace manual work in live-line working scenes can effectively guarantee the safety of personnel. To improve the operation efficiency and reduce the difficulties in operating a live-line working robot, this paper proposes a multi-DOF robot motion planning method based on RRT and extended algorithms. The planning results of traditional RRT and extended algorithms are random, and obtaining sub-optimal results requires a lot of calculations. In this study, a sparse offline tree filling the planning space are generated offline through the growing–withering method. In the process of expanding the tree, by removing small branches, the tree can fully wiring in the planning space with a small number of nodes. Optimize wiring through a large number of offline calculations, which can improve the progressive optimality of the algorithm. Through dynamic sampling and pruning, the growth of trees in undesired areas is reduced and undesired planning results are avoided. Based on the offline tree, this article introduces the method of online motion planning. Experiments show that this method can quickly complete the robot motion planning and obtain efficient and low-uncertainty paths.

Keywords: motion planning; Multi-DOF Robot; live-line maintenance; RRT Algorithm; growing–withering

check for updates

Citation: Feng, J.; Zhang, W.; An Efficient RRT Algorithm for Motion Planning of Live-Line Maintenance Robots. *Appl. Sci.* **2021**, *11*, 10773. https://doi.org/10.3390/app112210773

Academic Editor: Wonhee Kim

Received: 16 September 2021
Accepted: 22 October 2021
Published: 15 November 2021

1. Introduction

Live-line maintenance includes replacing or testing of power system equipment without interrupting the current. It can effectively improve power supply reliability, reduce outage losses, ensure grid safety and reduce economic losses. However, the risk of electric shock is very high when the work is performed manually. With the development of robotics, robotics have been gradually applied to the electric power industry since the 1980s to replace human beings in performing high-risk operation tasks [1].

Early robotic systems for live-line maintenance include the remote-controlled robot system of Quebec Hydropower in Canada [2,3], the Spanish remote robot system ROBTET [4], and the Japanese robot phase series [5]. These robot systems are all operated remotely, and an operator manually controls a robot arm to perform complicated tasks. In live-line working scene, manually operating a robot for live-line maintenance is an inefficient, dangerous and difficult task. Automatic robot motion planning instead of manual operation can effectively improve operational efficiency and reduce risks.

Robot path planning algorithms include graph search [6,7], visibility graph [8,9], artificial potential field [10,11], probabilistic roadmap [12,13], and rapid exploring random tree (RRT) [14]. RRT and extended algorithm are widely used in the planning problem of multi degrees of freedom (multi-DOF) robot. Compared with other planning methods, this method is more suitable for dynamic obstacle avoidance path planning of a manipulator in high dimensional space under a complex environment. Moreover, RRT is applicable to the motion planning problem under kinematic constraints [15].

RRT was first proposed by LaValle in 1998 for solving high-dimensional space problems with algebraic constraints (imposed by obstacles) and differential constraints (imposed by nonintegrity and dynamic environments). RRT samples points in the planning space, and eligible points are added to a tree. As the number of samples increases, the tree is expanded to the entire space. The RRT method does not need to map obstacles to the joint space, and it is more suitable for dynamic obstacle avoidance path planning of robots in high-dimensional space in complex environments than other planning methods. It is a probabilistically complete path planning method with good scalability [14,16]. The paths planned by RRT have randomness and are not optimal solutions. The randomness of sampling also reduces the speed of convergence of the algorithm.

In response to the shortcomings of RRT, researchers have proposed many improvements. RRT* [17,18] adds a reconnection process to the original method to obtain a sub-optimal solution. However, the reconnection method has a slower convergence speed, especially in larger planning spaces. Informed RRT* [19,20] improves the convergence speed of RRT* by centralized sampling. Kourosh provides the RT-RRT* algorithm to further improve operations, making it possible to complete real-time path planning. The RRT-connect [21–23] algorithm uses two trees growing in opposite directions and speeds up the search with a greedy strategy. The method saves search time by reducing useless searches in blank areas. In addition, there are many other variants of RRT algorithms, such as the parallel-RRT [24] algorithm that uses GPU parallel computing to accelerate RRT solution speeds, real-time RRT [25] that improves the real-time performance, and dynamic domain RRT [26] for dynamic environments.

Most RRTs and their algorithms are designed for robot planar motion planning and are not suitable for robots with multiple DOFs. In the multi-DOF robot motion planning problem, the existing RRT algorithm has the following problems: (1) The expansion of RRT requires frequent collision detection calculations. To obtain a better path, it is usually necessary to increase the number of samples. This leads to too long planning times. (2) The uncertainty brought about by the results of motion planning increases the risks in operation processes.

In the previous research [27], the entire robot system was introduced, including path planning algorithms. The focus of this article is a new motion planning method suitable for task scenarios. In this research, a multi-DOF robot motion planning method for fixed scenes, dynamic targets and dynamic obstacles is proposed based RRT*. The growth-withering method is proposed to expand the tree offline, which can fill the high-dimensional space with fewer nodes and realize the optimal wiring of the tree. Aiming at the constraints of the planning problem, the wiring is optimized by dynamic sampling and pruning to make the planning results meet the task requirements and reduce the randomness. For live-line working scenarios, this paper introduces the method of path planning for dynamic targets and dynamic obstacles by using an offline tree. Simulations and experiments show that this method can realize automatic path planning of manipulators in live-line working scenarios and meet the requirements of live-line maintenance.

In this paper, the robot system and live-line maintenance are introduced in Section 2. In Section 3, we introduce the method for building rapid exploring random tree offline. Section 4 describes the method of path planning using offline random trees. Section 5 introduces the simulation and experiment. Finally, this article is summarized in Section 6.

2. Live-Line Maintenance and Robot System

Robotic live-line maintenance refers to the maintenance of overhead cables by robots using special tools without cutting off the power supply. Common maintenance for live lines includes the disconnection of drain wires, replacement of insulated terminals, and removal of foreign objects (tree branches, kites, etc.), as shown in Figure 1. Working at heights causes difficulties in terms of observations and operations. The threat of charged objects to the robot must be considered during operation tasks. Therefore, reliable and efficient robot operation methods are needed to help operators complete tasks. Therefore, reliable and efficient robot operation technology is needed to help operators complete tasks.

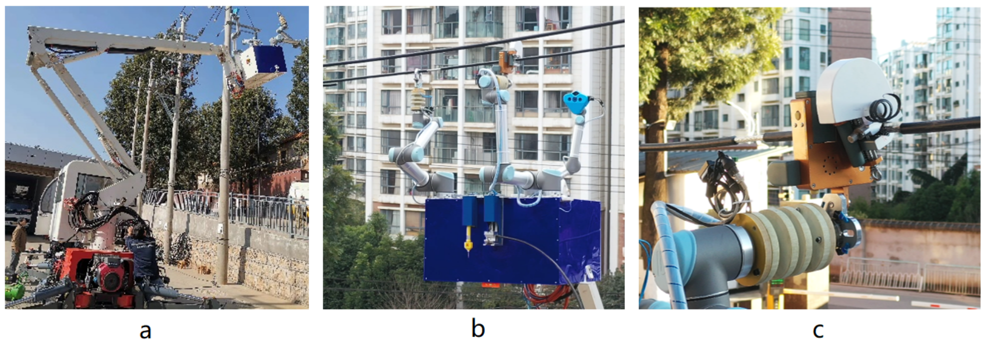


Figure 1. Schematic diagram of live-line working scene. ((**a**) Insulated bucket vehicle, (**b**) robot system, and (**c**) stripping operations).

2.1. Live-Line Maintenance Operations

The phased research goal of this paper is to achieve a drainage operation, and the main operation object is a swaying cable. The aim of the drainage operation of a 10 kV overhead line is to connect the distribution line and the drainage terminal through a drainage wire. The specific operation includes the following four parts:

(1) Wire Stripping: There is a 3–4 mm insulation layer on the surface of a 10 kV overhead cable, which needs to be removed, as shown in Figure 2a.
(2) Voltage detection: This is an essential part of connecting lead-flow wires. Electroscopy is used to detect whether the metal cable core is charged, as shown in Figure 2b.
(3) Wire clamp installation: The function of a wire clamp is to fix the lead-flow wire and the distribution network cable together. A wire clip needs to be installed on the cable, and the bolt needs to be tightened, as shown in Figure 2c.
(4) Lead-flow wire cut off: When the drainage power supply is not needed, the drainage line should be cut for safety reasons, as shown in Figure 2d.

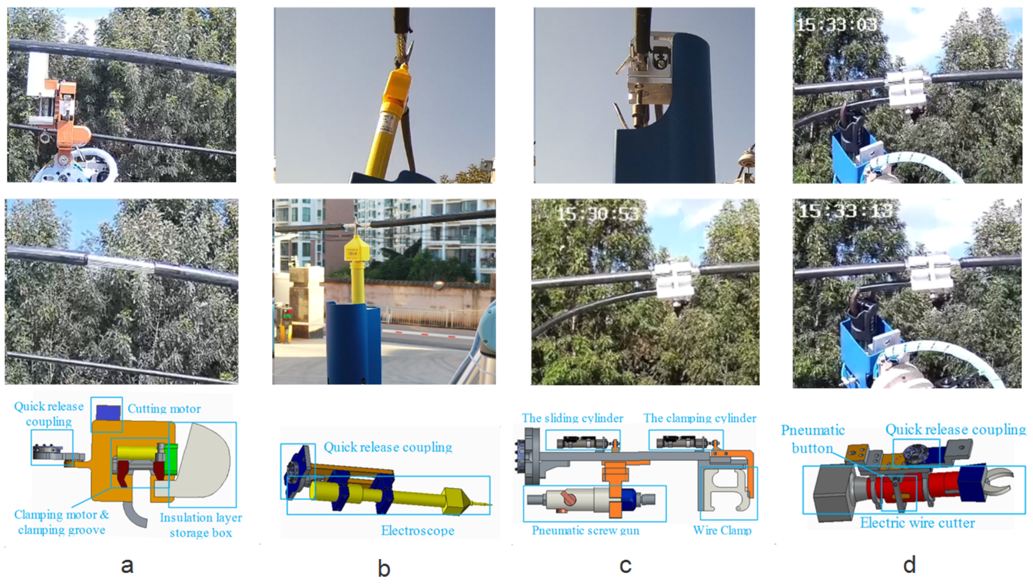


Figure 2. Operations of live-line maintenance [27]. ((**a**) Wire stripping, (**b**) voltage detection, (**c**) wire clamp installation, and (**d**) lead-flow wire cut off).

For the above tasks, special tools are designed, including strippers, electroscope, clamp installer, wire cutter, etc. During operation, the robot selects the appropriate tool, moves to the target position and operates the tool to perform the operation.

2.2. Robot System

The autonomous live-line maintenance robot (ALMR) consists of an insulated bucket vehicle and a working platform, as shown in Figure 3. The insulated bucket car has a crawler chassis and four auxiliary supports, which can adapt to complex terrain. The boom can lift up to 13 m, meeting the requirements of live-line maintenance of the distribution network. The insulated bucket was transformed into a live-line maintenance platform, as shown in Figure 3. Three 6-DOF robots [28] are installed on the platform and are used for operations, assistance and observations.

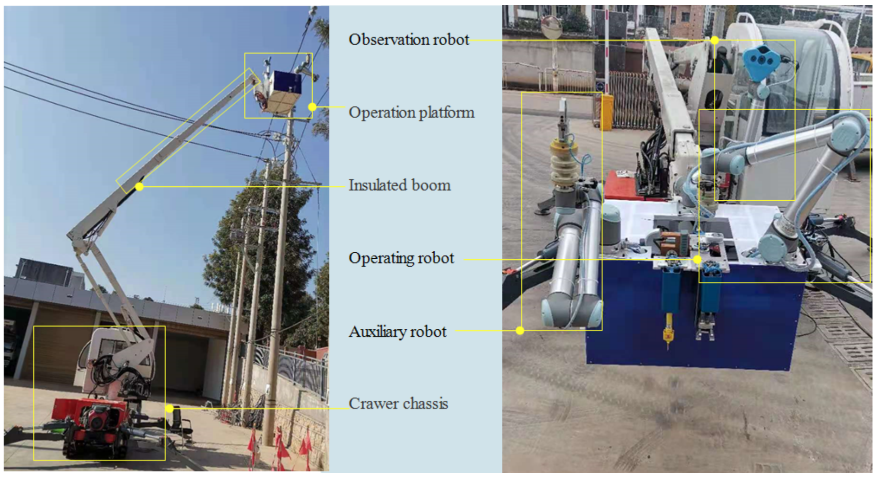


Figure 3. Autonomous live-line maintenance robot.

2.3. Particularity of Live-Line Working Scene

The Live-line Working scene has particularities, and the constraints caused by it need to be considered in the path planning. Affected by the cable layout, the position of the task target relative to the platform in different scenarios is quite different. The method of fixed path with end correction cannot meet the task requirements. Obstacles to robot motion planning include platforms, cables, and other robots. Affected by the size of the platform and the diversity of tools, the main obstacle in the movement of the manipulators is the platform itself, which can be regarded as a fixed obstacle, as shown in Figure 4. In different scenarios, the position of obstacles such as cables and crossarms relative to the platform is variable.

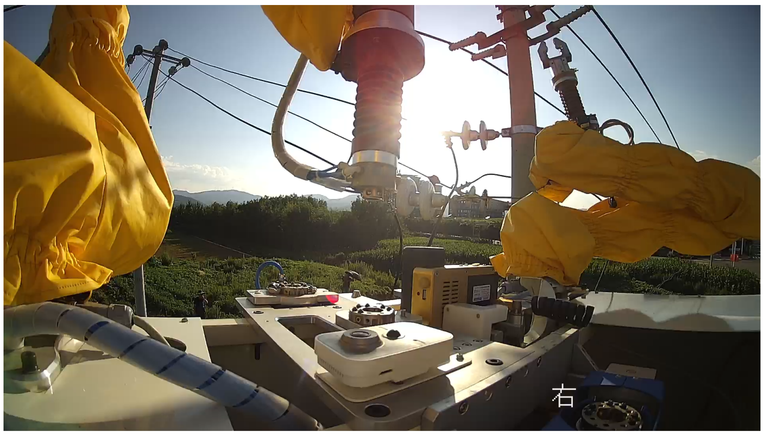


Figure 4. Dense tool placement and small space limit robot movement.

The live-line working operation is a highly risk task, and planning results with strong randomness are not acceptable. In addition, the stable planning results make it easy for the operator to monitor the movement process and prevent accidents.

The live-line working operation has special requirements for the robot's end posture. For example, it is necessary to prevent the unfixed parts from closing under gravity when moving the stripper. The drainage line will bend unpredictably during movement, which may lead to the cable winding to the robot body. Therefore, it is necessary to keep the robot body away from the cable when moving.

In conclusion, the path planning in the live-line working scene has the following particularities:

(1) There are a lot of obstacles in the environment, and some of the obstacles are mobile;
(2) The end posture of the robot is restricted during the movement of the robot;
(3) Based on safety factors, the robot stays away from charged objects as much as possible during the movement.

2.4. Planning Problem Definition

During the task process, it is necessary to plan a path from the tool position to the working position in the joint space. After the operation is completed, re-plan a path from the working position to the tool position in the changed environment. The motion planning problem can be described as:

$$\begin{cases} \Theta = f(\boldsymbol{\theta}_0, \boldsymbol{\theta}_n) \\ \Psi(\boldsymbol{\theta}_i) < 0, (\boldsymbol{\theta}_i \in \Theta) \end{cases} \tag{1}$$

where $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_n$ are the joint angles of the beginning and end positions, respectively. Θ is the planned path, which is composed of ordered path points $\boldsymbol{\theta}_i$. Ψ is the constraints. The constraints of this problem mainly include four aspects: obstacle constraint, preference constraint, attitude constraint and moment constraint. Obstacle constraints for robot movement include static obstacles (O_S) and dynamic obstacles O_d. The position of static obstacles relative to the platform for each task is constant, while dynamic obstacles need to be measured by sensors. Obstacle constraints can be described as:

$$f_{\text{coli}}\left(\boldsymbol{\theta}_i, O_S, O_d\right) < 0 \tag{2}$$

The preference constraint is to keep the robot away from charged objects and avoid cable entanglement. The preference constraint is achieved by limiting the range of motion of the joint, described as:

$$\boldsymbol{\theta}_{imin} < \boldsymbol{\theta}_i < \boldsymbol{\theta}_{imax}, (\boldsymbol{\theta}_i \in \Theta) \tag{3}$$

The joint limits θ_{imin} and θ_{imax} needs to be set based on experience.

In order to meet the requirements of the posture of the tools for the task, it is necessary to limit the robot end posture during the movement. The posture constraint can be described as:

$$\begin{cases} R = f_{kin}(\boldsymbol{\theta}_i) \\ R_{min} < R < R_{max} \end{cases} \tag{4}$$

where f_{kin} is the positive kinematics formula of the robot, and R is the robot posture. R_{max} and R_{min} is the attitude limit, depending on the tool requirements.

Finally, due to the large quality of the tool ruler and the influence of the cable, the robot needs to avoid positions with poor load capacity during the movement, such as the robot being straight in horizontal. The dynamical constraints are described as follows:

$$T(\boldsymbol{\theta}_i) < T_{\text{limit}} \tag{5}$$

In the path planning, in order to simplify the model, we merge the attitude constraint and the moment constraint into the obstacle constraint.

3. Offline Rapid Exploring Random Tree

According to the characteristics of the live-line maintenance task, there is a need for a planning algorithm that can quickly plan paths for multi-free manipulators in an environment with dynamic obstacles and output optimized planning results. According to task requirements, the algorithm should be able to set constraints to avoid undesirable planning results.

According to the task requirements, the growth–withering extension method is designed. Theoretically, the random tree is sufficiently expanded by a large number of samples to obtain a better path. In a high-dimensional planning space, the method requires a large number of sample points. Taking a 6 DOF robot with a joint movement range of 0–360° as an example, when the sampling step is 5°, the number of samples is as high as 1.39×10^{11}. The robot controller is not up to such a large amount of computation. In nature, trees sprout new branches in the spring and shed their leaves in the fall, leaving only their branches. Inspired by this, in the tree exploring process, the leaf nodes are removed periodically and the branches are retained. In this way, the tree can fill the planning space with only a few nodes, and realize full wiring in high-dimensional space. In addition, a dynamic sampling method is designed for special scene constraints, which can affect the expansion area of the random tree according to the constraints, and then affect the planning results. Using the simulation environment to verify the planning results can eliminate branches that do not meet expectations and avoid bad planning results.

3.1. Dynamic Area Sampling

The RRT method expands the tree through random sampling. Therefore, by changing the uniformity and randomness of sampling, the tree can be controlled to grow according to the constraint conditions. Representative RRT improved algorithms based on this principle include RRT-informed and RRT-connect. In multi-DOF robot motion planning, it is not suitable to improve the algorithm by sampling in a small area because the obstacles are in a flow pattern in the planning space. The method of changing the sampling probability using the traditional greedy strategy may lead to oversearching and increase the computational effort of collision detection. As shown in Equation (6), x_{rand} is a randomly generated sample point. When a random number p is greater than $(1-\alpha)$, the random sample points are selected on the line between the nearest tree node and the target point p_{goal}.

$$\mathrm{x}_{\mathrm{rand}} = \begin{cases} \mathrm{LineTo}\left(p_{\mathrm{goal}}\right) & \text{if } p \geq \alpha \\ \mathrm{Uniform}\ (P) & \text{if } p < 1-\alpha \end{cases} \tag{6}$$

This strategy helps the tree to find the target location quickly when there are few obstacles. However, in the case of more obstacles, it affects the efficiency of the search. As shown in Figure 5, under the influence of the greedy strategy, the tree grows towards the target. Since 6D space cannot be displayed intuitively, the algorithm is explained using a tree in a 2D space. When there is an obstacle on the line to the target, a large number of samples will be generated that do not satisfy the collision constraint. The nodes in the red area are denser, which means that the algorithm conducts many searches near the obstacle. The problem is particularly acute in robot joint spaces where flow pattern obstacles exist.

To solve this problem, we proposed a dynamic sampling method using different sampling probabilities at different tree growth stages. In Equation (8), Function Rect(A,B) represents a rectangular region with vertices A p_{goal} and B $p_{\mathrm{goal}} + (1-k)p_{\mathrm{start}}$. The variable $k \in [0,1]$ is the growth index of the tree and is expressed by the ratio of the number of nodes to the constant N in Equation (8). The greedy sampling area is controlled according to the change in k. When the value of k is small, the sampling value is close to the average sampling value, the value of k increases, and the sampling area decreases.

$$\mathrm{x}_{\mathrm{rand}} = \begin{cases} \mathrm{Uniform(dynarea)} & \text{if } p \geq \alpha \\ \mathrm{Uniform}\ (\Phi_J) & \text{if} p < \alpha \end{cases} \tag{7}$$

$$\text{dynarea} = \text{Rect}\left(p_{\text{goal}},\quad k \cdot p_{\text{goal}} + (1-k)p_{\text{start}}\right) \tag{8}$$

$$k = \text{size(tree)} / \text{N} \tag{9}$$

Figure 6 shows the planning results of dynamic area sampling. Compared with the traditional greedy algorithm, the sampling density near the obstacle is lower, and there is no oversampling phenomenon. Close to the target area, the branches and leaves of the tree become denser. This can help trees grow quickly towards the target location.

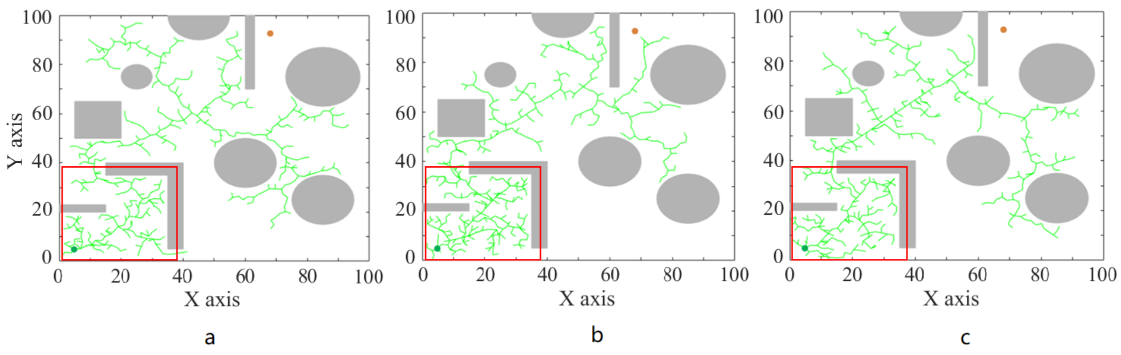


Figure 5. Expanded tree with greedy strategy ((**a**) α = 0.9, step = 2, sampling times = 500, collision detection times = 1536; (**b**) α =0.7, step = 2, sampling times = 500, collision detection times = 2394, (**c**) α = 0.5, step = 2, sampling times = 500, collision detection times = 3956).

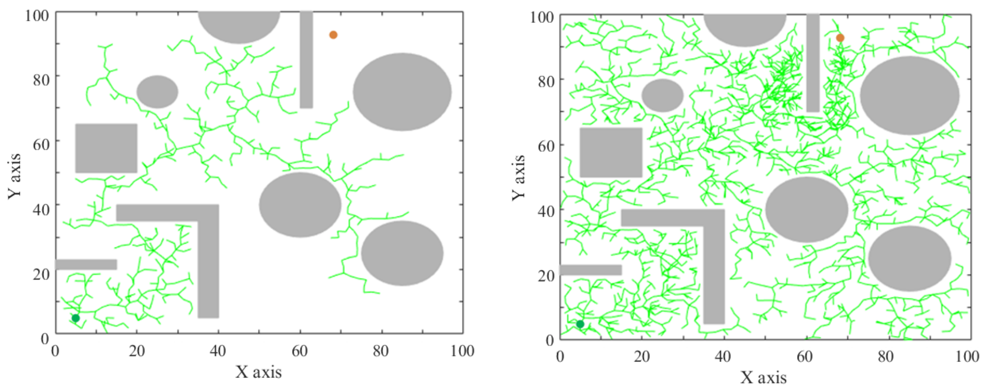


Figure 6. Tree expansion using dynamic area sampling ((**Left**) sampling time = 500, α = 0.5, collision detection times = 1647; (**Right**) sampling time = 2000, α = 0.8, and collision detection times = 15,357).

In motion planning, some paths are not expected. Reducing the sampling frequency for a specific area can reduce the growth of trees in that area. As a result, the probability of generating undesired paths is greatly reduced. In Equation (10), $\beta \in (0, \alpha)$ is a small constant, and the region is sampled only when $p < \beta$.

$$x_{\text{rand}} = \begin{cases} \text{Uniform}(rect) & p \geq \alpha \\ \text{Uniform}(\text{C}\ _{P}P_{\text{bor}}) & \beta \leq p < \alpha \\ \text{Uniform}\ (P_{\text{bor}}) & p < \beta \end{cases} \tag{10}$$

Path planning must not only consider the impact of obstacles on the movement, but also meet the requirements of special scenes, such as the end posture constraints in Figure 7. In addition, during the installation of the drainage line, it is necessary to avoid the twisting drainage line from hitting or even entangled with the robot body, as shown in Figure 8. During the path planning, the robot body is kept away from the lead line by limiting the range of joints motion.

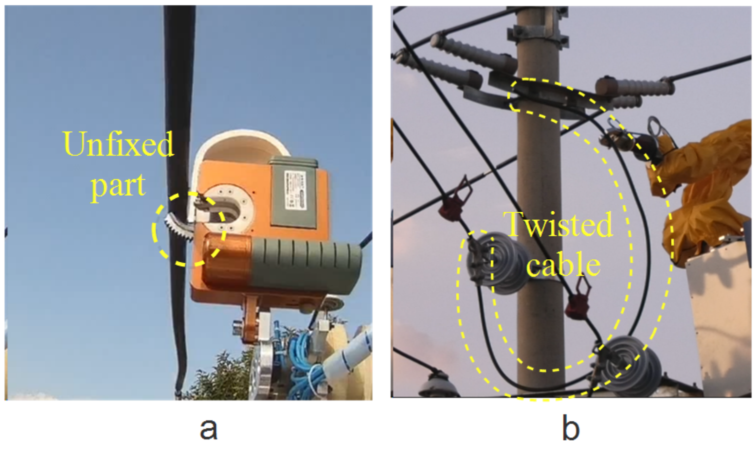


Figure 7. Tool restrictions on robot movement. ((**a**) The unfixed parts need to be kept open for cable entry. (**b**) Cable with uncertain twist).

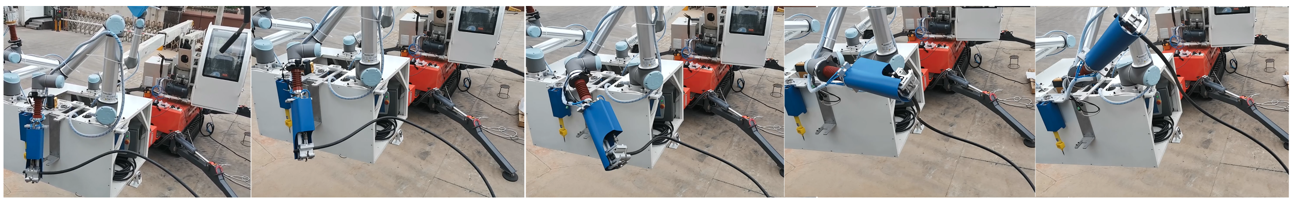


Figure 8. The twisting of a drainage line during installation.

By changing the sampling frequency in the joint space, the range of motion of the joint can be limited. As shown in Figure 9, the sampling frequency of the red area is extremely low ($\beta = 0.05$). The tree has fewer branches in this area. Thus, it reduces the probability of the planned path passing through the red area. Through the setting of the preference area, we can influence the randomness of the trajectory to a certain extent, and the planning trajectory is biased towards our preferences.

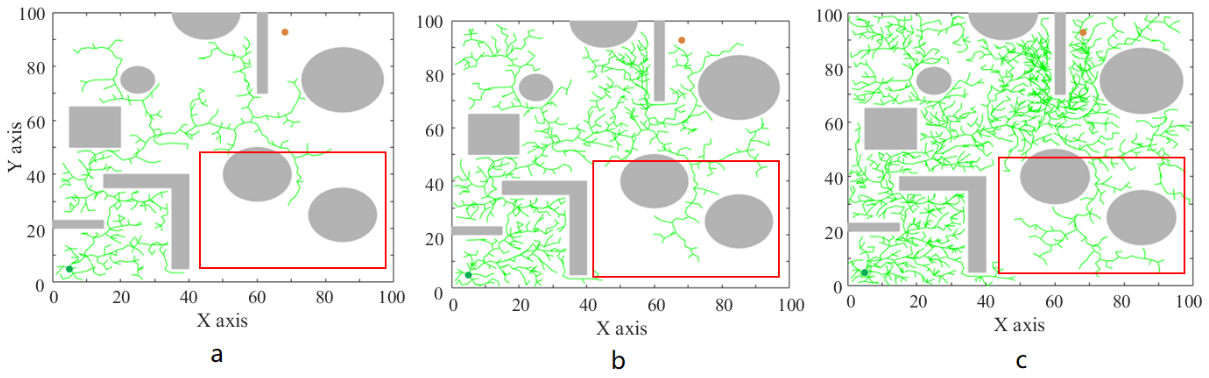


Figure 9. Tree expansion influenced by the change in regional sampling frequency (sampling times = 500/1000/2000, α = 0.5, and β = 0.05).

3.2. Expansion

Algorithm 1 introduces the expansion method of the random tree. Unlike the traditional expansion method, we add the withering operation to the expansion process. The expansion process stops the growth whenever n new nodes are generated. Then, Algorithm 2 is used to wither the tree, removing small branches on the random tree and keeping the main trunk. Through the cycle of growth and withering, tree $T_{offline}$ will eventually fill the planning space. During the expansion process, only the fixed barrier constraint $P_{StaticObs}$ is calculated.

Algorithm 1 Expand the tree by growing and withering.

```
Require: $\Phi_J$, $P_{StaticObs}$, $p_{start}$, $p_{goal}$,$n$,$N$
Ensure: $T_{offline}$
 1: Create $T_{offline}$ with root node $p_{start}$;
 2: loop
 3:    // Make the tree grow;
 4:    $cnt = n$;
 5:    while $cnt$ >0 do
 6:       Expand $T_{offline}$ using RRT*;
 7:       $cnt = cnt$ -1;
 8:    end while
 9:    // Wither the tree
10:    Remove the small branches of $T_{offline}$ using Algorithm 3;
11:    $num$ = sizeof($T_{offline}$);
12:    if $num \geq N$ then
13:       Break;
14:    end if
15: end loop
```

Algorithm 2 Wither the tree.

```
Require: $T$, $len$
Ensure: $T_{withered}$
 1: $cnt$ = size($T_{offline}$);
 2: while $cnt$ >0 do
 3:    $parent\_index$ = GetParentIndex($T$(cnt));
 4:    while $parent\_index$ >0 do
 5:       if $T(parent\_index)$ is not retrieved then
 6:          $branch$ = [$T(parent\_index)$;branch];
 7:          Mark $T(parent\_index)$ as retrieved
 8:          $parent\_index$= GetParentIndex($T(parent\_index)$);
 9:       else
10:          $parent\_index$= 0;
11:       end if
12:    end while
13:    if size(branch) >$len$ then
14:       Merge the branch into $T_{withered}$;
15:    end if
16:    $cnt = cnt$-1;
17: end while
```

The purpose of withering is to remove the short branches on the tree and reduce the number of nodes. Starting from the point with the largest index, the branches are constructed by retrieving the parent node. The retrieved nodes are marked to prevent repeated retrieval. A long branch (>len) is added to tree $(T_{withered})$. Since nodes with larger indices are newly added, starting from the largest index point can ensure the integrity of the branch. In addition, this also prevents new nodes at the ends of the branches from being pruned, ensuring that the branches are continuously expanded. The withering process is shown in Figure 10.

Algorithm 3 Motion planning for the live-line maintenance robot.

```
Require: $\Phi_J$, $\mathbf{P}_{StaticObs}$, $\mathbf{P}_{DynObs}$, $\mathbf{p}_{start}$, $\mathbf{p}_{end}$
Ensure: $\Theta_{Path}$
 1: Create $T_{offline}$ with $\mathbf{p}_{start}$ in $\Phi_J$;
 2: Expand $T_{offline}$ using Algorithm 1 offline under the constraints of $\mathbf{P}_{StaticObs}$;
 3: Create $T_{goal}$ with $\mathbf{p}_{start}$;
 4: while new node $P_{new}$ cannot be connected to $T_{offline}$ do
 5:     Expand $T_{goal}$ using Algorithm 4;
 6: end while
 7: Plan $Path(\theta_0, \theta_1, ...\theta_n)$ from $\mathbf{p}_{start}$ to $\mathbf{p}_{end}$ through point $p_{new}$;
 8: if Path collides with $\mathbf{P}_{DynObs}$ then
 9:     Replan using Algorithm 5;
10: end if
```

Algorithm 4 Rewiring for the dynamic obstacle.

```
Require: $p_{dyn\_obs}$, $T_{offline}$, $Path$
Ensure: $Path_{dyn}$
 1: [$p_s$, $p_e$]=FindCollisionofPath($Path$,$p_{dyn\_obs}$);
 2: $T_{cut}$ = CutTreeCrownFromNode($p_s$);
 3: Inital $T_{dyn}$ with Path($p_e$ : end);
 4: loop
 5:     Expand $T_{dyn}$ using Algorithm 4 (line 4–11);
 6:     Find the $Path_{dyn}$ through the joint node
 7:     if collision($Path_{dyn}$, $p_{dyn\_obs}$) == false then
 8:         break;
 9:     end if
10: end loop
```

Algorithm 5 Expand the tree from the goal.

```
Require: $p_{goal}$, $T_{offline}$, $n$
Ensure: $Path$
 1: Inital $T_{goal}$ with $p_{goal}$;
 2: $cnt$ = 0;
 3: loop
 4:     Expand $T_{goal}$ using RRT*;
 5:     $p_{new} = T_{goal}(end)$
 6:     for $p \in T_{offline}$ do
 7:         if distance($p$, $p_{new}$) $\leq$ segmentlen & collision($p$, $p_{new}$) == false then
 8:             Mark $p$ and $p_{new}$ as joint nodes.
 9:             $cnt$ = $cnt$ + 1;
10:         end if
11:     end for
12:     if $cnt > n$ then
13:         break;//Stop extending if there are enough joint nodes.
14:     end if
15: end loop
16: Find a $Path(\theta_0, \theta_1, ...\theta_n)$ through joint nodes;
```

Through the growth–withering expansion method, the trunk of the tree continues to grow. In areas with dense branches, since the newly added branches are shorter, they are removed by the withering algorithm. As a result, the tree becomes leafless with long sparse branches, as shown in Figure 11. This process consumes many computing resources, so it needs to be performed offline. As the number of samples increases, the growth of the tree slows down or decreases when it grows to a certain size, as shown in Figure 12.

Therefore, it can be judged that the tree fills the space based on the growth status of the random number.

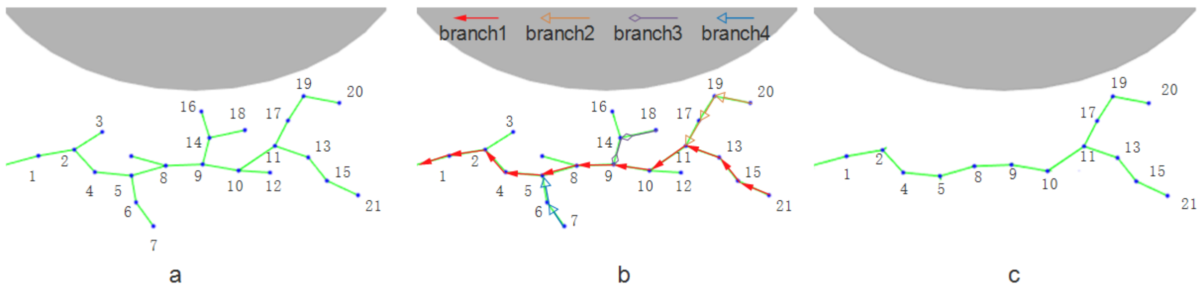


Figure 10. The withering of the exploring tree. ((**a**) The tree before withering and the numbers are the order in which the nodes are generated, (**b**) branches of the tree, (**c**) remove branches with length less than *len* = 3).

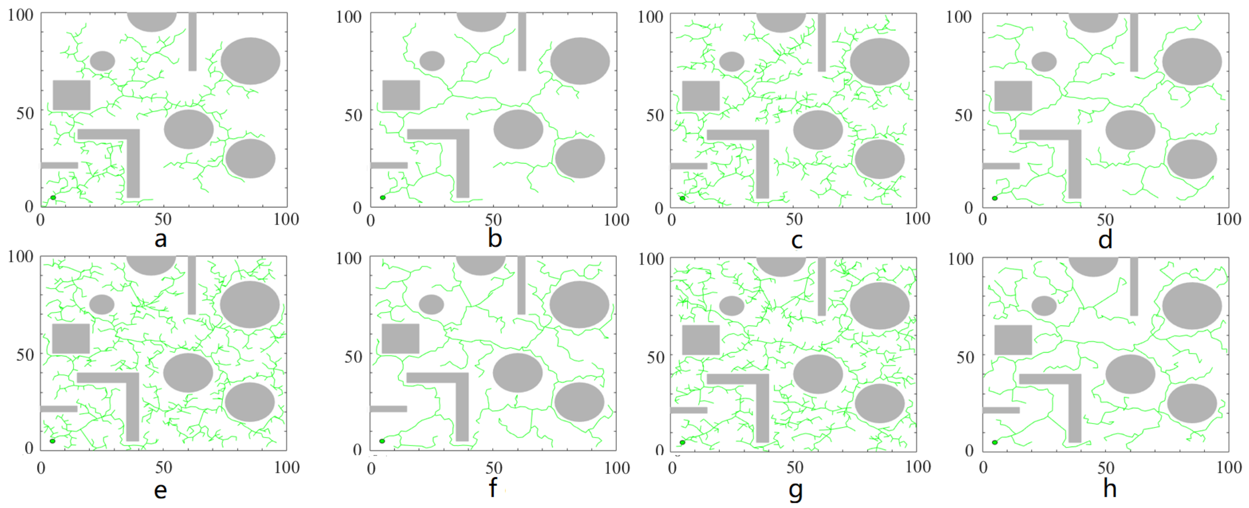


Figure 11. Expand tree using Algorithm 2 (**a**\c\e\f, expansion of tree after 500\\1000\\2000\\4000 sampling, **c\e\f**, the expansion is based previous withered tree, **b\d\f\h**, wither the tree in previous expansion with *len* = 4).

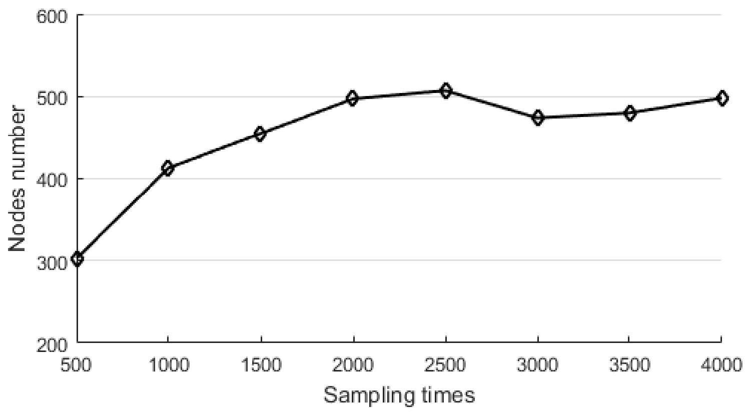


Figure 12. Changes in the number of nodes during expansion.

The sparseness of the random tree is determined by the parameter *len*. The larger *len* is, the sparser the tree, as shown in Figure 13. In a high-dimensional space, we use a larger *len* value to make the random tree fully grow while reducing the number of nodes.

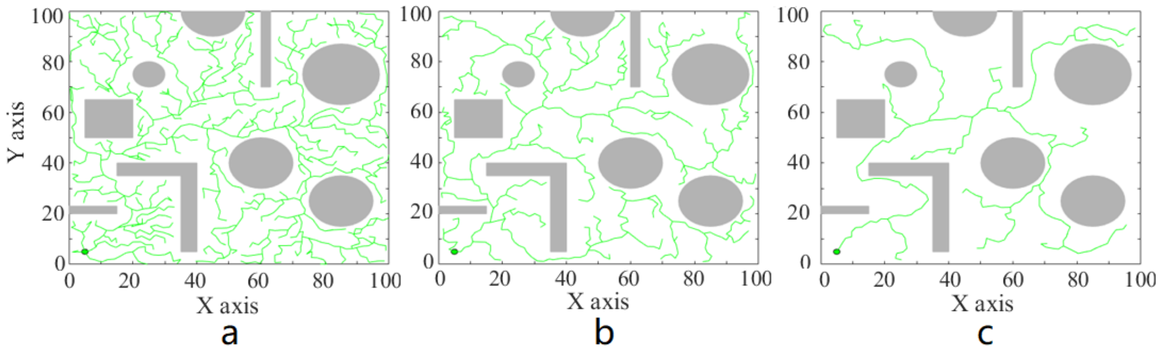
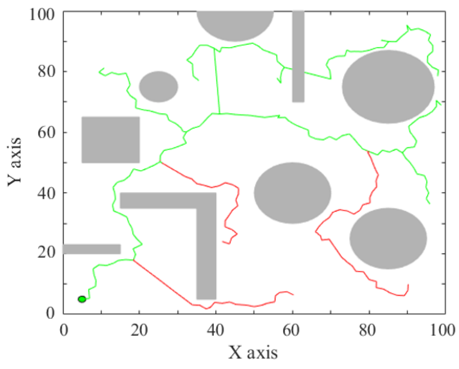


Figure 13. The effect of *len* on tree expansion (sample times = 4000, (**a**) len = 2, nodes num = 1086; (**b**) len = 4, nodes num = 498; (**c**) len = 6, nodes num = 301).

3.3. Pruning the Tree

For the problem of the randomness of paths, the undesired branches can be removed by pruning tree $T_{offline}$. Since a 6D joint space cannot be intuitively displayed with a graphical method, the pruning of branches needs to be done in a 3D simulation environment. A large *len* value is used to wither the tree once so that only a limited number of branches remain in the tree. In the simulation environment, the robot is run according to the path generated by each branch to check whether the path meets the expectation, as shown in Figure 14. Finally, the branches corresponding to undesired paths are removed from tree $T_{offline}$.



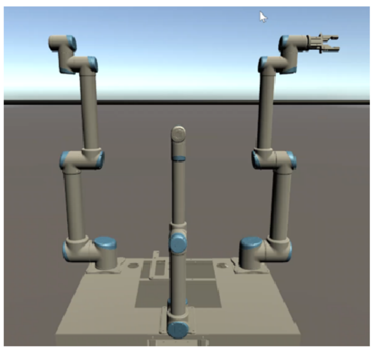


Figure 14. Trunks of the tree and the simulation environment. ((**left**) The trunk of a rapidly exploring random tree and the red branches will be removed, (**right**) simulation environment for robot motion).

4. Online Motion Planning

In this section, the method of using offline trees to plan paths for live-line maintenance is introduced, including dynamic obstacles and path smoothness. During the operation, the robot takes the tool from the platform and moves to the target position, as shown in Figure 15. The movement process will be affected by obstacles such as platforms and electrical facilities (cables, cross arms, etc.). The target position is indeterminate and is provided by the depth vision sensor. The steps of path planning are shown in Algorithm 3.

First, in the offline state, the random tree $T_{offline}$ is expanded with the root node of p_{start} in the joint space Φ_J of the robot. Unknown obstacles, such as cables, are not considered during the expansion process, and only fixed obstacle constraints are considered $P_{StaticObs}$. The method of expanding random trees offline is introduced in Algorithm 2.

During the working process, a tree T_{goal} is built with the cable position p_{goal} obtained by the sensor as the root node. T_{goal} is expanded using RRT* until the new node p_{new} of T_{goal} can be connected to the node of $T_{offline}$. From this node, two paths can be planned to reach p_{start} and p_{goal}. Through p_{new}, a path from p_{start} to p_{goal} can be planned. Dynamic obstacles, such as cables, are imported into the model, and whether the planned path collides with these obstacles is checked. If a collision occurs, Algorithm 4 is used to replan the path.

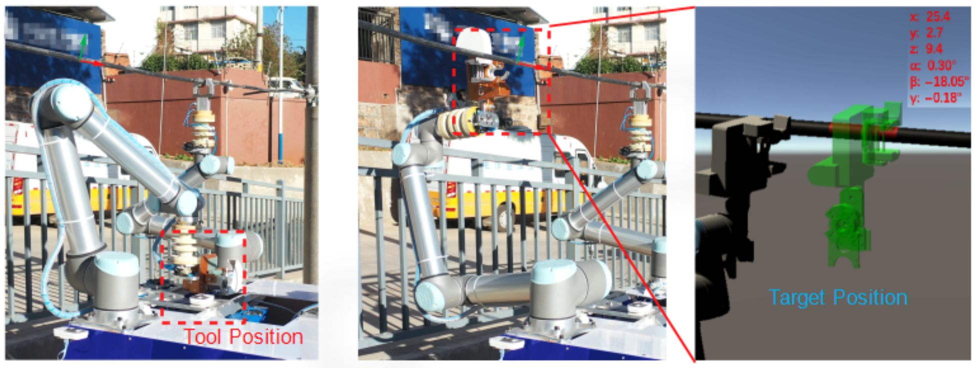


Figure 15. The beginning and end of motion planning.

4.1. *Searching Path from the Target*

In power operations, the target may be cables, clamps, terminals, etc. The positions of these targets are obtained by the depth camera installed in the robot system. Planning the path to the target can be searched by expanding $T_{offline}$, but this method generates a large number of useless random points. Using RRT-connect for reference, we take the target as the root to build and expand the search tree, as shown in Figure 16.

During the expansion process, the distance and collisions between the new node and the nodes on $T_{offline}$ are detected (line 7 in Algorithm 5). The points with a distance less than *segmenlen* and with no collisions are marked as joint points. Two paths leading to p_{goal} and p_{start} can be planned through the joint node. The two paths are combined to obtain a path from the starting point to the target. To obtain a better path, it is necessary to continuously expand T_{goal} and obtain a certain number (n) of joint nodes. In addition, the *Path* with the lower path cost is chosen.

In this way, T_{goal} requires only a small number of nodes to establish a connection with $T_{offline}$ and to obtain a path.

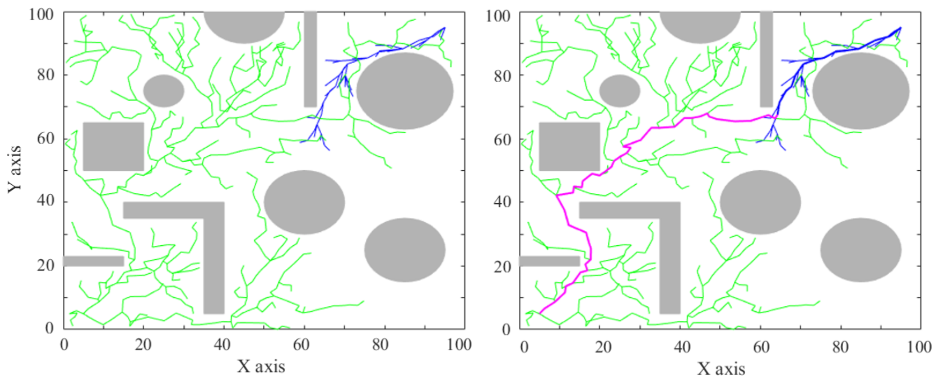


Figure 16. Tree expanded from the gsoal and the planned path. ((**left**) The green $T_{offline}$ and the blue T_{goal}, (**right**) the magenta part in $T_{offline}$ and the blue part in T_{goal} are combined to obtain a path from the starting point to the target).

4.2. *Dynamic Obstacle*

For each operation, the position of the lifting platform relative to the overhead line changes greatly. During the operation, additional obstacles will be added, such as the drainage line as shown in Figure 17. These obstacles are not considered when constructing the offline tree. However, the full wiring tree in the planning space can help algorithm bypass obstacles and quickly complete the planning. As shown in Figure 18a, blue dynamic obstacles p_{dyn_obs} are dynamic obstacles in the environment. When p_{dyn_obs} appears on the planning path, the planned path is unavailable and the tree needs to be rewired. Algorithm 5 introduces the method of rewiring and planning the path.

First, section $[p_s\colon\ p_e]$ of the path that collides with the obstacle is found and is taken as the divider to remove part of the canopy of $T_{offline}$; the remaining part is T_{cut}. Some branches of T_{cut} may go through the dynamic obstacle. Detecting these branches consumes many resources, so no detection is performed. The partial path Path (p_e: end) is retained on the target side as the new tree (T_{dyn}). Algorithm 4 is used to expand (T_{dyn}) and plan the path. It is determined whether the planned path passes through dynamic obstacles, and if so, a new joint node is selected to obtain the path.

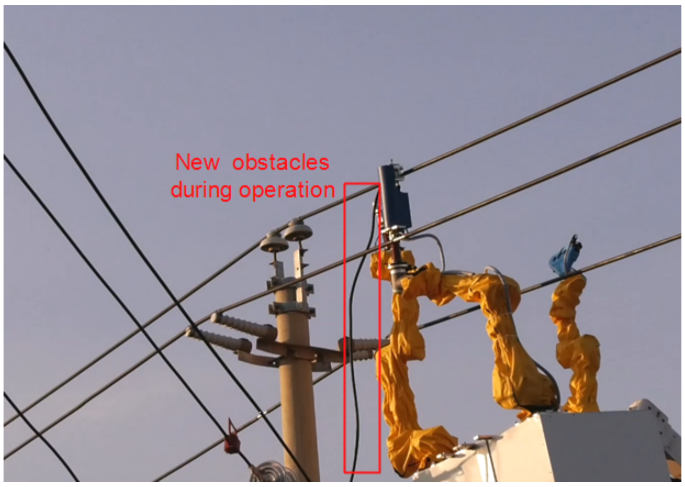


Figure 17. The newly installed drainage wire affects robot movement.

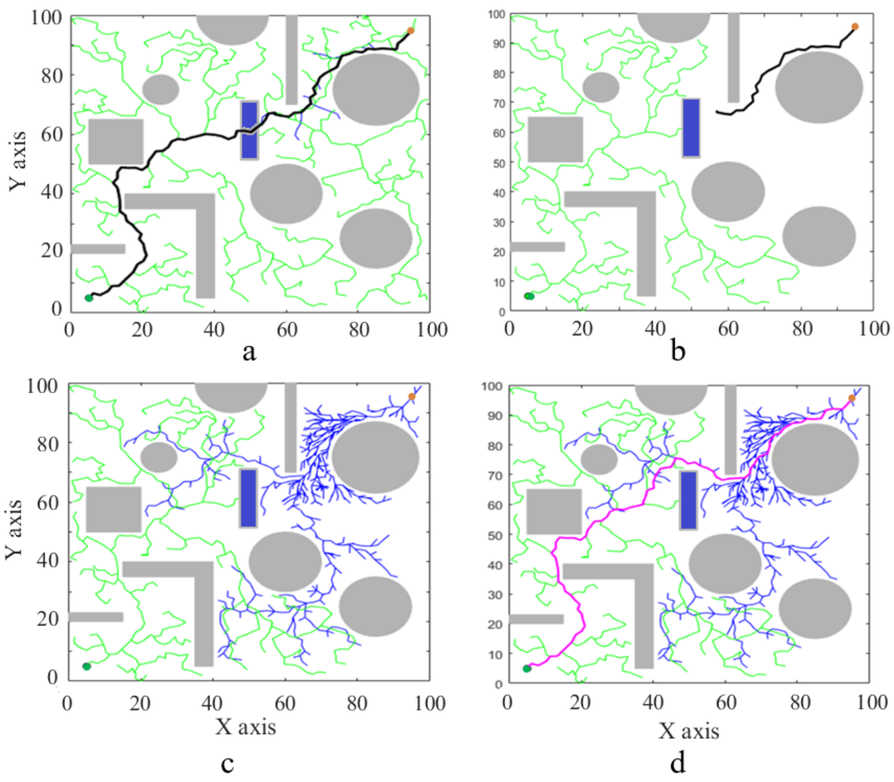


Figure 18. Replanning for dynamic obstacles. ((**a**) Dynamic obstacles appear on the planned path, (**b**) cut $T_{offline}$ according to the obstacle position, and (**c**) expand T_{dyn} from the left path (black part in (**b**)), (**d**) a new path is selected to avoid dynamic obstacles).

4.3. Optimized Planning Path

In order to fully wire in the plan space, the step size during expansion is short, which leads to more planned path segments. Therefore, the path needs to be smoothed to improve the execution efficiency of the robot. Smoothing is done by taking a small section of the path, testing whether the beginning and end points of the path can be directly connected, and removing the intermediate points to generate a new path. A path with fewer segments can be obtained by multiple operations, as shown in Figure 19. In the simulation, it is found that the sampling order has some influence on the generated smoothing results.

In the simulation, it is found that the sampling order and sampling radius have an effect on the smoothing results. In Figure 19, the following three sampling orders are used: sequential sampling, maximum distance sampling, and minimum distance sampling. Sequential sampling rewires waypoints one by one in the order of the points on the path. The maximum distance sampling rewires the waypoints with a large safe distance first. The minimum distance sampling rewires the waypoints with a small safe distance first.

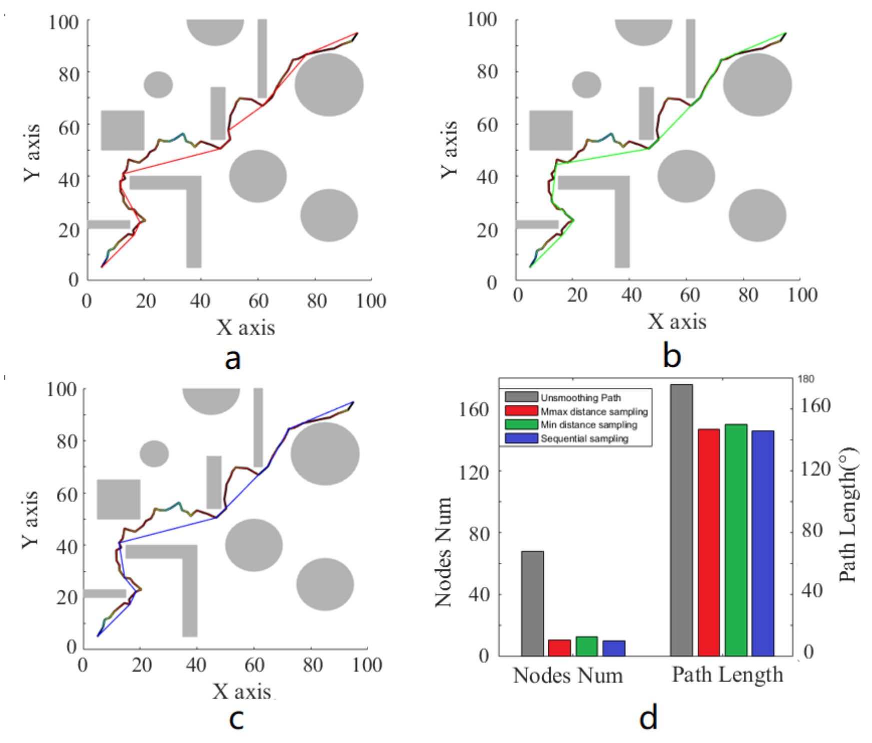


Figure 19. Smoothing results of different sampling methods. ((**a**) maximum distance sampling, (**b**) minimum distance sampling, and (**c**) sequential sampling, and (**d**) the number of nodes and path length changes after smoothing).

To evaluate the sampling approach, we evaluate the smoothing results using three metrics: the average distance of obstacles, path length, and number of nodes. Fifteen different combinations of sampling methods are used to smooth the 20 planning paths. In Table 1, 20 planning paths are smoothed using 15 different combinations of sampling methods. In the table, a, b and c indicate the maximum distance sampling, minimum distance sampling and sequential sampling, respectively. Sampling method ab smooths the path with maximum distance sampling first and then minimum distance sampling. The simulation results are shown in Figure 20. The results show that: the number of nodes is less in scheme 10; the path length is smaller in scheme 11, but the average safety distance is smaller in this scheme; the average safety distance of the path is larger in scheme 2. Therefore, scheme 2 is usually selected for smoothing based on safety factors.

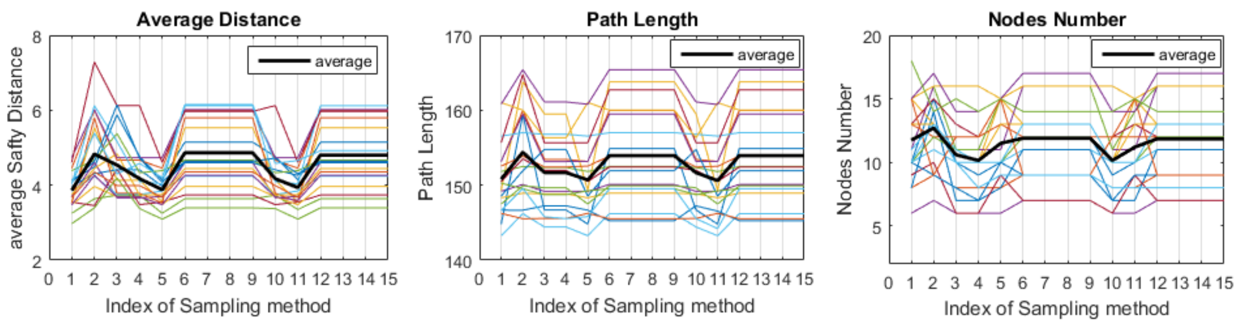


Figure 20. Comparison of the smoothing results of different schemes.

Table 1. Effect of the sampling method on smoothing results.

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Sampling method	a	b	c	ab	ac	bc	ba	ca	cb	abc	acb	bca	bac	cab	cba

1. Maximum distance sampling is used; 2. Minimum distance sampling is used; 3. Sequential sampling is adopted; 4. Use maximum distance sampling first, then sequential sampling, etc., are adopted.

5. Simulation and Experiment

5.1. Simulation

Figure 21 shows the experimental platform. The left panel is the physical platform, and the right panel is the 3D online simulation environment, which has the function of collision detection. The robot motion planning results can be verified in a simulation environment.

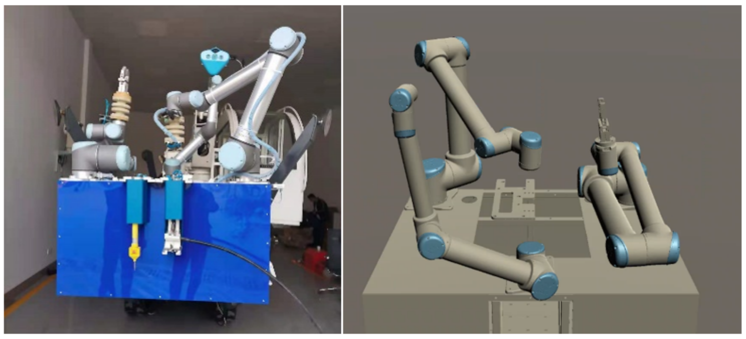


Figure 21. Test platform and simulation verification environment.

To evaluate the performance of the algorithm, we randomly set 20 groups of cable poses for path planning using different algorithms. The planning space is a 6D joint space, and the range of motion of each joint is 360°. The step size of both algorithms is 2°, and the greedy strategy coefficient $\alpha = 0.1$. The rewire radius of the RRT* and the offline method is 5°. The same smoothing method is used to smooth the paths. The test results are shown in Table 2.

In terms of computational efficiency, the algorithm is evaluated with the number of samples and the number of collisions detected. Compared with the other two algorithms, RRT*-offline can complete trajectory planning with a small amount of sampling. In the experiment, FCL [29] was used for collision detection, and the environmental model was simplified to improve the detection efficiency. RRT-connect and RRT*-offline take similar amounts of time, but in the absence of dynamic obstacles, RRT*-offline is faster.

Table 2. Comparison of of different algorithms for motion planning of 6-DOF robots.

Algorithm	**RRT***	**RRT-Connect**	**RRT*-Offline**
Sample times	1823	648	214
Collision detection times	20,860	1003	1235
Calculation time (ms)	6326	578	395
Path length (°)	323.4	430.7	194.5
Path segment number	12.1	15.5	9.3
Success rate	85% (4000)	100%	100%
Path availability	65%	45%	95%

The path length is the sum of the modules of each path section in the joint space. The planned paths of each algorithm are smoothed using the same smoothing method. In comparison, the length and segment number of the RRT*-offline planning path are less than those of the other two algorithms.

In 20 rounds of planning, both RRT-connect and RRT*-offline completed the planning, and only RRT* did not complete 3 planning tasks after 4000 rounds of sampling. Path availability refers to whether the planned trajectory meets expectations. Most of the results of RRT*-connect meet the preset preferences. The planning results of RRT* and RRT-connect are highly uncertain, and some of the results do not meet the preferences.

5.2. Test in Simulated Field

In order to verify the reliability of the system function and the planning algorithm, a large number of simulation experiments were carried out in the training field, as shown in Figure 22. The purpose is to verify the reliability of the robot's path planning in real scenarios. The operation process is recorded through video and database, which are used to evaluate the planning effect. For the path planning results, the evaluation indicators include: reliability (no collision), stability (whether the planning is completed), and optimality of the results (whether there are redundant trajectories). A total of 16 rounds of tests were carried out, and the results are shown in Table 3.

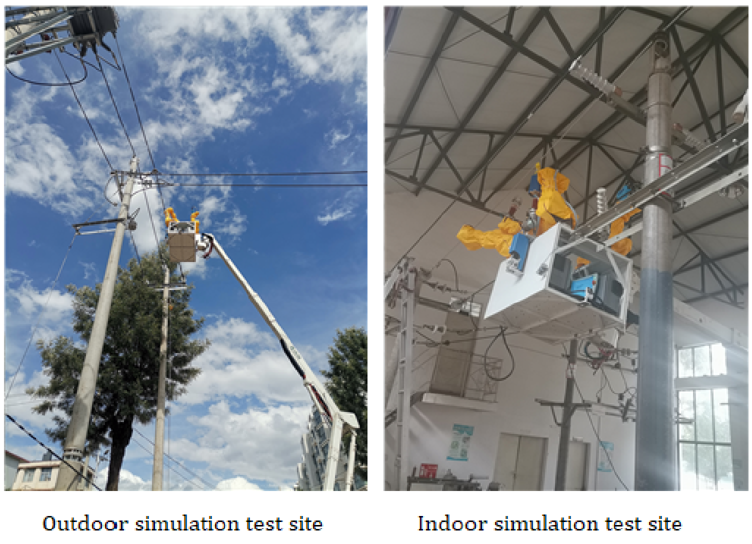


Figure 22. Simulated experiment scene.

Table 3. Simulation results at the test site.

	Wire Stripping			Clamp Installation		
Planning Algorithm	**RRT* -Offline**	**RRT -Connect**	**RRT***	**RRT* -Offline**	**RRT -Connect**	**RRT***
Number of collisions	0	0	0	0	0	0
Potentially dangerous path	0	3	4	4	7	5
success of first planning	15	16	15	14	10	7
Result optimality	1.52	7.61	2.44	1.71	5.23	3.20

The results show that all kinds of algorithms can effectively avoid obstacles when the safe distance is set reasonably (safe distance > 5 cm). A potentially dangerous path means that the robot is too close to a charged object, such as the cross arm, during the movement. Experiments show that RRT*-offline can effectively avoid this risk through dynamic sampling.

The stability of the algorithm is evaluated by whether the first plan is successful. All planning algorithms for wire stripping have a high success rate, but during the clamp installation operation, the success rate of RRT-connect and RRT* drops sharply due to the existence of the cable. The main problem is that the cable's constraints on the robot's movement cannot be solved. RRT*-offline avoids this problem through dynamic area sampling.

The optimality of the result refers to the ratio of the path length of the first planning result to the shortest path length of the 10 times planning under the simulation conditions. The smaller the value, the closer the planning result is to the optimal solution. Due to the limitation of the number of iterations (5000), the planning effect of RRT* algorithm is inferior to RRT*-offline, and the performance of RRT-connect algorithm is the worst.

The experiment shows that the path planning constraints designed according to the task characteristics have achieved the expected results. As shown in Figure 23a–c, posture constraints are added to the robot path planning, the angle between the direction of gravity and the closing direction of the fixed parts is always greater than 90 degrees, so as to prevent it from closing under the action of gravity. As shown in Figure 23d–f, the longer tool size results in a larger load torque on the robot. Through the torque constraint, the robot is in a better output attitude and the load capacity is improved. As shown in Figure 23g–i, under the effect of preference constraints, the robot body is far away from the end to avoid cable winding.

5.3. *Field Test*

Two field operations were carried out in August 2021, including a drainage operation in Figure 24 and a broken drainage operation in Figure 25. The two field tests were mainly to verify whether path planning can improve work efficiency. We compared the efficiency of manual operations, teleoperation operations, and path planning.

The time for the system to complete three drainage is 40 min, and the manual operation time for the same task is about 30 min (including wearing protective equipment). In order to ensure the safety of operation, each trajectory of the robot is executed only after the safety is confirmed manually, which takes a lot of time (25–30 min according to the video recording).

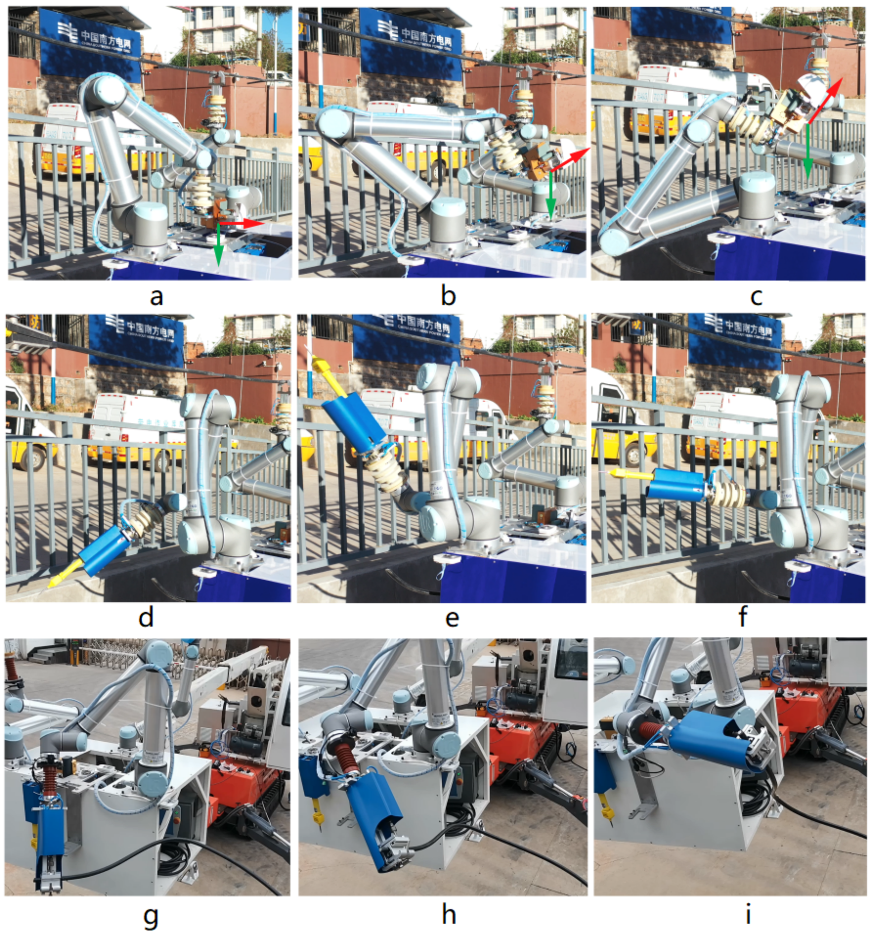


Figure 23. The impact of planning algorithms on the task process. (**a**–**c**) During the stripping operation, the non-fixed parts should be kept open during the tool removal operation. (**d**–**f**) During the movement of the long tool, each joint of the robot is in a proper state of force and the load capacity is improved. (**g**–**i**) During the installation of the cable, the robot body maintains the principle cable to prevent entanglement.)

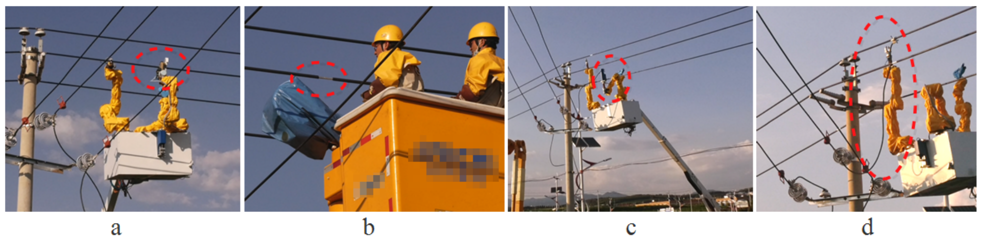


Figure 24. Field work: drainage operation on overhead lines. ((**a**) wire stripping, (**b**) inspect stripping quality, (**c**) install the clamp, and (**d**) single-phase drainage operation is complete).

In addition, we compared the time-consuming of manual operation and automatic operation in the process of fire removal. In manual operation, the operator needs to remotely control the manipulator to move the tool to the target position, and a single operation takes about 3–7 min. The automatic operation takes less than 1 min (including cable positioning process).

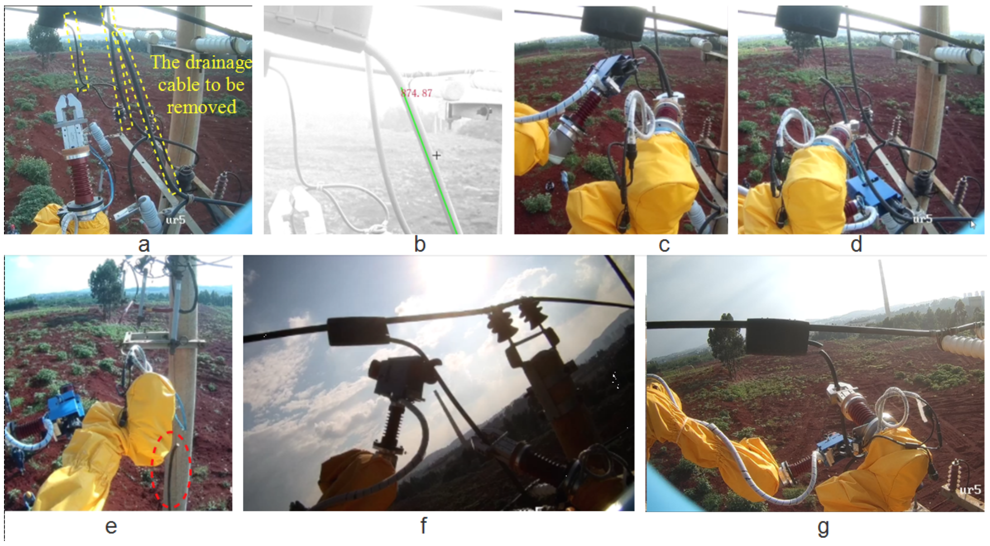


Figure 25. Field work: broken drainage operation on overhead lines. ((**a**) Operating environment, (**b**) identify and locate cables, (**c**) cut the upper end of the near side drainage cable, (**d**) cut the lower end of the near side drainage cable. (**e**) discard drainage cable, (**f**) broken drainage operation for mid-phase drainage cable, and (**g**) broken drainage operation for far side drainage cable).

5.4. Discussion

According to the simulation and experiment, RRT*-offline has better performance and can effectively solve the problem of robot motion planning in live working scenarios.

(1) Real-time performance: By generating a random tree offline, most of the calculation is removed from the real-time calculation process, which effectively improves the efficiency of real-time motion planning.
(2) Practicability: The randomness of the planning results is the main limiting factor of RRTs. Dynamic area sampling and tree pruning solve this problem to a certain extent. The expansion and pruning of offline trees are complicated tasks, but they only need to be completed once.
(3) Versatility: The research in this paper is aimed at a live working environment, and its characteristic is that most of the obstacles are fixed, the positions of a few obstacles are changed, and the target position is also changed. In similar scenarios, the algorithm in this article can be used. For example, in the flexible manufacturing process, tools and parts will change during the production process, but the overall production environment is fixed.

In the research, we also found the following problems:

(1) The process of demonstrating the planning results in a simulation environment consumes operating time and reduces operating efficiency.
(2) The offline tree cannot be displayed intuitively, and there is no good way to evaluate the offline tree.

6. Conclusions

This study presents a motion planning algorithm based on RRT for live-line maintenance robot. This method reduces the time-consuming online calculation through offline wiring. During the offline expansion, the RRT* algorithm is used to perform a large number of iterations to make the offline tree asymptotically optimal, and dynamic area sampling and pruning are used to reduce the uncertainty of the planning results. The online planning method designed for dynamic targets and obstacles can make the planning results converge quickly. The simulation and experimental results show that the method can meet the requirements of motion planning in practical scenarios of live-line working.

There are many opportunities for improvement in future works. This includes designing evaluation indicators for offline trees, optimizing offline trees through new expansion methods and conducting research on barriers to flow patterns.

Author Contributions: Conceptualization, J.F. and W.Z.; methodology, J.F. and W.Z.; software, J.F.; validation, J.F.; formal analysis, J.F.; investigation, J.F. and W.Z.; resources, W.Z.; data curation, J.F.; writing—original draft preparation, F.J.; writing—review and editing, J.F. and W.Z.; visualization, J.F.; supervision, W.Z.; project administration, W.Z.; funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This paper reports no data.

Acknowledgments: Thanks for the support and help provided by Kunming Live Work Center during the field test.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RRT	rapid exploring random tree
ALMR	Autonomous Live-line Maintenance Robot
DOF	Degrees of Freedom
FCL	Flexible Collision Library
RT-RRT*	Real time RRT*

References

1. Allan, J.F. Robotics for distribution power lines: Overview of the last decade. In Proceedings of the 2012 2nd International Conference on Applied Robotics for the Power Industry (CARPI), Zurich, Switzerland, 11–13 September 2012; pp. 96–101.
2. Boyer, M. Systems integration in telerobotics: Case study: Maintenance of electric power lines. In Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, USA, 22–28 April 1996; Volume 2, pp. 1042–1047.
3. Aracil, R.; Ferre, M. Telerobotics for aerial live power line maintenance. In *Advances in telerobotics*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 459–469.
4. Aracil, R.; Ferre, M.; Hernando, M.; Pinto, E.; Sebastian, J. Telerobotic system for live-power line maintenance: ROBTET. *Control. Eng. Pract.* **2002**, *10*, 1271–1281. [CrossRef]
5. Maruyama.; Yoshinaga. Robotic applications for hot-line maintenance. *Ind. Robot. Int. J.* **2000**, *27*, 357–365. [CrossRef]
6. Trovato, K.I.; Dorst, L. Differential a. *IEEE Trans. Knowl. Data Eng.* **2002**, *14*, 1218–1229. [CrossRef]
7. Qingxuan, J.; Gang, C.; Hanxu, S.; Shuangqi, Z. Path Planning for Space Manipulator to Avoid Obstacle Based on A* Algorithm. *J. Mech. Eng.* **2010**, *46*, 109–115.
8. Lozano-Pérez, T.; Wesley, M.A. An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM* **1979**, *22*, 560–570. [CrossRef]
9. Huang, H.P.; Chung, S.Y. Dynamic visibility graph for path planning. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, 28 September–2 October 2004; Volume 3, pp. 2813–2818. [CrossRef]
10. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous Robot Vehicles*; Springer: Berlin/Heidelberg, Germany, 1986; pp. 396–404.
11. Gilbert, E.; Johnson, D. Distance functions and their application to robot path planning in the presence of obstacles. *IEEE J. Robot. Autom.* **1985**, *1*, 21–30. [CrossRef]
12. Lanteigne, E.; Jnifene, A. Biologically inspired node generation algorithm for path planning of hyper-redundant manipulators using probabilistic roadmap. *Int. J. Autom. Comput.* **2014**, *11*, 153–161. [CrossRef]
13. Boor, V.; Overmars, M.; van der Stappen, A. The Gaussian sampling strategy for probabilistic roadmap planners. In Proceedings of the 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), Detroit, MI, USA, 10–15 May 1999; Volume 2, pp. 1018–1023. [CrossRef]
14. Qureshi, A.H.; Ayaz, Y. Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments. *Robot. Auton. Syst.* **2015**, *68*, 1–11. [CrossRef]

15. Jaillet, L.; Porta, J.M. Path planning under kinematic constraints by rapidly exploring manifolds. *IEEE Trans. Robot.* **2012**, *29*, 105–117. [CrossRef]
16. Tahir, Z.; Qureshi, A.H.; Ayaz, Y.; Nawaz, R. Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments. *Robot. Auton. Syst.* **2018**, *108*, 13–27. [CrossRef]
17. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
18. Wang, J.; Li, B.; Meng, M.Q.H. Kinematic Constrained Bi-directional RRT with Efficient Branch Pruning for robot path planning. *Expert Syst. Appl.* **2021**, *170*, 114541. [CrossRef]
19. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004. [CrossRef]
20. Kim, M.C.; Song, J.B. Informed RRT* towards optimality by reducing size of hyperellipsoid. 2015 IEEE International Conference on Advanced Intelligent Mechatronics, Busan, Korea, 7–11 July 2015; pp. 244–248.
21. Kuffner, J.J.; LaValle, S.M. RRT-connect: An efficient approach to single-query path planning. In Proceedings of the 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), San Francisco, CA, USA, 24–28 April 2000; Volume 2, pp. 995–1001.
22. Wang, W.; Li, Y. Path planning for redundant manipulator without explicit inverse kinematics solution. In Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), Guilin, China, 19–23 December 2009; pp. 1918–1923.
23. Kang, J.G.; Lim, D.W.; Choi, Y.S.; Jang, W.J.; Jung, J.W. Improved RRT-Connect Algorithm Based on Triangular Inequality for Robot Path Planning. *Sensors* **2021**, *21*, 333. [CrossRef] [PubMed]
24. Aguinaga, I.; Borro, D.; Matey, L. Parallel RRT-based path planning for selective disassembly planning. *Int. J. Adv. Manuf. Technol.* **2008**, *36*, 1221–1233. [CrossRef]
25. Naderi, K.; Rajamäki, J.; Hämäläinen, P. RT-RRT* a real-time path planning algorithm based on RRT. In Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games, Paris, France, 16–18 November 2015; pp. 113–118.
26. Yershova, A.; Jaillet, L.; Siméon, T.; LaValle, S.M. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; pp. 3856–3861.
27. Feng, J.; Zhang, W. Autonomous Live-Line Maintenance Robot for a 10 kV Overhead Line. *IEEE Access* **2021**, *9*, 61819–61831. [CrossRef]
28. DH Parameters for Calculations of Kinematics and Dynamics. Available online: https://www.universal-robots.com/articles/ur/application-installation/dh-parameters-for-calculations-of-kinematics-and-dynamics/ (accessed on 1 September 2021).
29. Pan, J.; Chitta, S.; Manocha, D. FCL: A general purpose library for collision and proximity queries. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 3859–3866.