Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 7405831, 12 pages
https://doi.org/10.1155/2017/7405831

Research Article

The Optimal Collision Avoidance Trajectory Planning of Redundant Manipulators in the Process of Grinding Ceramic Billet Surface

Shipu Diao,[1,2] Xindu Chen,[1,2] Lei Wu,[1,2] Mingjiang Yang,[1,2] and Junhui Liu[1]

[1] *School of Electromechanical Engineering, Guangdong University of Technology, Guangzhou, China*
[2] *Computer Integrated Manufacturing System Key Lab of Guangdong Province, Guangzhou, China*

Correspondence should be addressed to Xindu Chen; chenxindu@gdut.edu.cn

Received 9 July 2017; Accepted 23 November 2017; Published 14 December 2017

Academic Editor: Marcello Pellicciari

The intelligent manufacturing system (IMS) is widely used in the surface machining of the workpiece. In the process of ceramic surface grinding, the intelligent machine (manipulator) in IMS is required to automatically plan the collision avoidance trajectory in a complex environment. This paper presents an optimal trajectory planning method of the use of redundant manipulators in the surface grinding of ceramic billet, which is based on trajectory evaluation. The collision avoidance trajectory can be optimized, taking into account several parameters in the trajectory, including the length of the collision avoidance path, the weighted sum of the strokes of all joints, and the duration of the collision avoidance trajectory. Firstly, get the planning task. Secondly, set the planning parameters and obtain a number of collision avoidance trajectories. Finally, the evaluation function is used to evaluate the collision avoidance trajectories and get the optimal collision avoidance trajectory. The performance of the proposed optimal collision avoidance trajectory planning method is validated in different evaluation functions.

1. Introduction

More and more robotic manipulators are used for industrial applications such as grinding, assembling, welding, and handling, to replace the boring work done by workers in harsh environments. In order to meet the quality of ceramic products, the inner and outer surfaces of its billet usually need to be polished. The manipulators can be taught by the worker to grind the outer surface, but this process is extremely time-consuming and requires relevant experience. The inner surface grinding completely relies on handwork because the narrow space is difficult to realize the teaching approach. In the use of intelligent manufacturing on the surface of ceramic billet grinding, it is necessary to require the manipulators to automatically plan the collision avoidance trajectory in a complex environment.

The collision avoidance trajectory planning of the manipulator is divided into two parts: preprocessing and postprocessing. The preprocessing uses the collision avoidance planner to create collision avoidance paths of the planning scenario. The postprocessing mainly completes the interpolation of the collision avoidance path, introduces the time parameter, calculates the inverse kinematics solver, detects the collision of the environment and the joint, and finally obtains the collision avoidance trajectory which can be executed by the manipulator. In a given time to repeat the planning, a lot of collision avoidance trajectories are obtained, and it requires appropriate evaluation methods to select the best collision avoidance trajectory.

In order to get the optimal collision avoidance trajectory, a framework of optimal collision avoidance trajectory planning method is proposed in this paper, which is divided into three parts:

(A) Identify current planning tasks.

(B) Set the planning time, planner, and other parameters. A number of collision avoidance trajectories are obtained after collision avoidance path planning, trajectory planning and collision detection and stored in a container.

(C) Calculate the value of the parameters in the collision avoidance trajectory, and then use the evaluation function to evaluate the collision avoidance trajectory after normalization process. The optimal collision avoidance trajectory is returned to the manipulator controller for execution.

The main contributions of this paper are as follows:

(i) We propose a collision avoidance trajectory evaluation function that can consider several different parameters at the same time and show how to calculate all the parameters in the collision avoidance trajectory evaluation function.

(ii) In order to save energy, we do weight processing according to the drive motor power consumption of the different joints and put it into the trajectory evaluation function.

(iii) We demonstrate that, with the use of a sample-based collision avoidance path planner, the length of the collision avoidance path, the weighted sum of the strokes of all joints, and the duration of the collision avoidance trajectory are all subject to the normal distribution.

(iv) We demonstrate that the setup time for replanning has no significant impact on the performance of the optimal trajectory.

The rest of this paper is arranged as follows: Section 2 introduces the mathematical basis of the optimal collision avoidance planning, including the evaluation function, the calculation of the parameters of the evaluation function, the TRAC-IK inverse kinematics solver, and the collision detection method; Section 3 presents an overall framework of optimal collision avoidance planning based on trajectory evaluation; in Section 4, we set different evaluation functions of collision avoidance trajectory and test the optimal collision avoidance trajectory planning method proposed in this paper; Section 5 discusses the influence of the replanning time on the optimal trajectory and the statistical law of the parameters of the collision avoidance trajectory; this paper concludes in Section 6.

2. Related Work

The trajectory planning problem of the manipulator can be handled in different ways, and several categories of these techniques are proposed in [1]. According to the information processing in the system, it is divided into local and global methods. Local methods use incomplete information of the environment and gradually build the trajectory, which can achieve rapid collision detection and trajectory planning. In contrast, the global methods use all the information in the workspace, and an optimization criterion can be used to search the best collision avoidance trajectory.

When this problem is treated as an optimization problem, it is necessary to define the criteria related to the trajectory planning, such as the torque minimization of the robot, the energy, and the speed. For this problem, a lot of technology has been developed, and the main difference is the source of the problem and the algorithm used to solve it. Garg and Kumar [2] proposed torque minimization and used genetic algorithms to solve this problem. Lin [3] proposed another technique in which they use the kinematics of the robot to avoid the calculation of the inverse Jacobian matrix. In order to solve this problem, they used the perturbation method. Chettibi et al. [4] proposed to minimize the power of the actuator, the time of the actuator, and the power consumption of the joints of the manipulators in the joint space and use the nonlinear optimization to solve the problem. da Graça Marcos et al. [5] used a genetic algorithm to obtain a solution in the joint space and evaluated it by means of the position error of the end effector and the minimized fitness function of speed, acceleration, torque, and energy. Menasri et al. [6] proposed to solve the problem based on the use of bilevel optimization and metaheuristics, divide the path into small displacements, and make full use of the redundancy of the manipulator to search out the optimal collision avoidance configuration of the robot in the joint space. Another method is proposed in [7], which uses genetic algorithms to find a solution in the joint space. In order to evaluate the current solution, the error position of the end effector in Cartesian space needs to be calculated. To find the next solution, the inverse Jacobian matrix is used to convert the position error to the error of the joint variable. Of course, other algorithms can be found to solve this problem by using particle swarm optimization [8], direct variational method [9], and other methods that consider this problem as multiobjective optimization problems [10].

Another category dealing with this problem is based on the probabilistic approach. Their general idea is to build a graph between the initial configuration and the termination configuration in a very abstractly defined state space. Each node of the graph is found by using a different state sampler (e.g., uniform, gauss, and obstacle based). We can find the PRM method [11–13], SPARS method [14–16], RRT method [13, 17–20], SBL method [21], EST method [22], KPIECE method [23], and SyCLOP method [24].

Taking into account all these ideas above, we propose a new optimal trajectory planning method. It also uses probabilistic methods and optimization methods to solve the problem, which we call the trajectory evaluation method. In this method, we first use the probability method to get a lot of collision avoidance trajectories and then use the appropriate trajectory evaluation function to get the best collision avoidance trajectory. The weighted sum of all the joints used in the trajectory evaluation function is to take into account the energy saving problem of the manipulator during the execution of the collision avoidance trajectory.

3. The Mathematical Basis of Optimal Collision Avoidance Trajectory Planning

This section will introduce the mathematical basis of the optimal collision avoidance trajectory planning, including the evaluation function, the calculation of the parameters

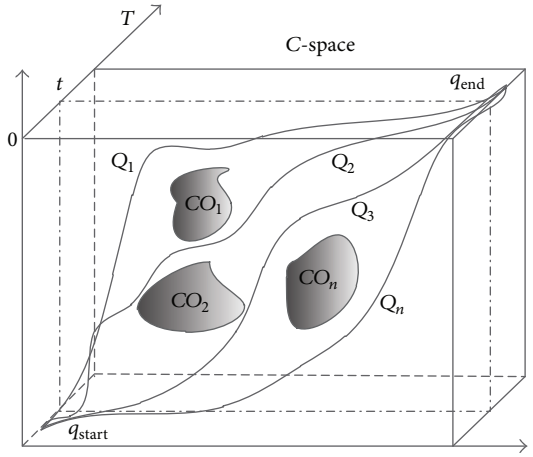


Figure 1: Replanning in a given time to obtain multiple collision avoidance trajectories. In the figure, CO_1, CO_2, and CO_n represent the different obstacles in the environment. $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{Q}_3$, and $\mathbf{Q}_n$ denote the different collision avoidance trajectories.

of the evaluation function, the TRAC-IK inverse kinematics solver, and the collision detection method.

3.1. Evaluation Function of Collision Avoidance Trajectory. Traditionally, the goal of collision avoidance planning trajectory is to find a trajectory to avoid collisions between the start configuration and the target configuration. Specifically, the start configuration vector $\mathbf{q}_{\text{start}}$ (in Cartesian space is $\mathbf{p}_{\text{start}}$) and the target configuration vector $\mathbf{q}_{\text{end}}$ (in Cartesian space $\mathbf{p}_{\text{end}}$) are defined in the manipulator's configuration space $\mathbf{C}$, where the $\mathbf{C}$'s size D is equal to the number of manipulator joints. There may be a number of obstacles in the environment corresponding to the rigid body. We assume that the duration of the collision avoidance trajectory is t and discretize it into N path points. The trajectory can be expressed as a vector $\mathbf{Q} \in R^{D \cdot N}$

$$\mathbf{Q} = \left[\mathbf{q}_1^T, \mathbf{q}_2^T, \ldots, \mathbf{q}_N^T\right]^T. \tag{1}$$

In general, any collision-free satisfies all constraints, and smooth trajectories can be considered as acceptable solutions. For the same planning task, we can get different collision avoidance trajectories, as shown in Figure 1.

In order to evaluate the proposed collision avoidance trajectories, we define the evaluation function of collision avoidance trajectory $\mathbf{Q}_i$ as $f_{\mathbf{Q}_i}$

$$f_{\mathbf{Q}_i} = A * c_{\text{length}}(\mathbf{Q}_i)^* + B * c_{\text{rotation}}(\mathbf{Q}_i)^* + C * c_{\text{duration}}(\mathbf{Q}_i)^*. \tag{2}$$

Here $i \in [1, M]$ and M represent the number of collision avoidance trajectories obtained in a given period of time. A, B, and C represent the weighting coefficients of the length of the collision avoidance path, the weighted sum of the strokes of all joints, and the duration of the collision avoidance trajectories, which are determined according to actual needs and have $A + B + C = 1$. Section 4 has a relevant test. In (2), $c_{\text{length}}(\mathbf{Q}_i)^*$, $c_{\text{rotation}}(\mathbf{Q}_i)^*$, and $c_{\text{duration}}(\mathbf{Q}_i)^*$ represent the values after the corresponding parameter Z-score normalization. The purpose of Z-score normalization is to remove the unit limit of the data and convert it into a dimensionless pure value to facilitate the mathematical operation of different units or orders of magnitude. The statistical rules for these parameters are given in Section 5.

After the original values of these parameters are Z-score normalized, the standard normal distribution can be obtained. That is, the mean is 0, the standard deviation is 1, and the conversion function is

$$X^* = \frac{X - \upsilon}{\sigma}. \tag{3}$$

In (3), X can take $c_{\text{length}}(\mathbf{Q}_i)$, $c_{\text{rotation}}(\mathbf{Q}_i)$, and $c_{\text{duration}}(\mathbf{Q}_i)$, and υ and σ are the mean and standard deviation of the corresponding parameters.

The optimal collision avoidance trajectory is the trajectory when the evaluation value F_{optimal} of the trajectory equals the minimum evaluation value $f_{\min}$ of all the trajectories:

$$F_{\text{optimal}} = f_{\min} = \min\left(f_{\mathbf{Q}_i}\right). \tag{4}$$

3.2. The Length of the Collision Avoidance Path. The three-dimensional Euclidean distance reflects the length of the collision avoidance path and is an important factor in evaluating the performance of the collision avoidance trajectory. In order to calculate the length $c_{\text{length}}(\mathbf{Q}_i)$ of the collision avoidance path $\mathbf{Q}_i$, a method of calculating the three-dimensional Euclidean distance between the two adjacent points in the collision avoidance path is used, assuming that the coordinates of the point are $(x_{\mathbf{q}_{ij}}, y_{\mathbf{q}_{ij}}, z_{\mathbf{q}_{ij}})$, then

$$c_{\text{length}}(\mathbf{Q}_i) = \sum_{j=1}^{n} \sqrt{\left(x_{\mathbf{q}_{i(j+1)}} - x_{\mathbf{q}_{ij}}\right)^2 + \left(y_{\mathbf{q}_{i(j+1)}} - x_{\mathbf{q}_{ij}}\right)^2 + \left(z_{\mathbf{q}_{i(j+1)}} - z_{\mathbf{q}_{ij}}\right)^2}. \tag{5}$$

Here j is the number of the point in the collision avoidance trajectory $\mathbf{Q}_i$, $j \in [1, n]$, and n is the total number of points in $\mathbf{Q}_i$. The meaning of these symbols appearing later in this paper is defined as the same.

3.3. The Weighted Sum of the Strokes of All Joints. In order to take into account the difference in the energy consumption of the drive motor of the different joints of the manipulator in the process of the trajectory planning, a new concept of the weighted rotation of all joints of the collision avoidance trajectory is proposed, which is called the weighted sum of the strokes of all joints. The weighted sum of the strokes of all joints of the collision avoidance trajectory $\mathbf{Q}_i$ is denoted by $c_{\text{rotation}}(\mathbf{Q}_i)$, and the weighted rotation angle of the same joint k from the current point j to the next point $j + 1$ is

$$c_{\text{rotation}}\left(\mathbf{Q}_{i_{k(j+1)}}\right) = \omega_k \left|\theta_{\mathbf{q}_{ik(j+1)}} - \theta_{\mathbf{q}_{ikj}}\right|. \tag{6}$$

Here ω_k represents the weighting coefficient of joint k, k is the joint number, and $k \in [0, 6]$. $\theta_{\mathbf{q}_{ikj}}$ represents the angle of

the kth joint corresponding to the jth point in the collision avoidance trajectory. In this paper, the weighting coefficient of joint k is calculated as

$$\omega_k = \frac{W_k}{\sum_{k=0}^{6} W_k}. \tag{7}$$

Here W_k is the power consumption of the drive motor for each joint.

Then, the weighted sum of the strokes of all joints of collision avoidance trajectory $\mathbf{Q}_i$ is

$$c_{\text{rotation}}(\mathbf{Q}_i) = \sum_{k=0}^{6}\sum_{j=1}^{n} \frac{W_k}{\sum_{k=0}^{6} W_k} \left|\theta_{\mathbf{q}_{ik(j+1)}} - \theta_{\mathbf{q}_{ikj}}\right|. \tag{8}$$

3.4. The Duration of the Collision Avoidance Trajectory. The duration of the collision avoidance trajectory can be same, but considering the difference in the collision avoidance path length obtained from the sampler-based planner, the duration is not constant when the trajectory satisfies other constraints.

The duration of the collision avoidance trajectory is calculated as

$$c_{\text{duration}}(\mathbf{Q}_i) = t_{\mathbf{q}_{iN}} - t_{\mathbf{q}_{i1}}. \tag{9}$$

Here $t_{\mathbf{q}_{i1}}$ and $t_{\mathbf{q}_{iN}}$ represent the start and end times of the trajectory $\mathbf{Q}_i$.

3.5. TRAC-IK Inverse Kinematics Solver. Using TRAC-IK to calculate the inverse kinematics of the redundant manipulators not only improves the success rate of the trajectory planning but also reduces the calculation time of the inverse solver to a certain extent and gets more collision avoidance trajectories during the same time period. As with Beeson and Ames's [25] research work, we will run two IK implementations, KDL and SQP. By default, the IK search returns immediately when either of these algorithms converges to an answer.

KDL is obtained by the following iterative equation:

$$\mathbf{q}_{\text{next}} = \mathbf{q}_{\text{prev}} + \mathbf{J}^{-1}\mathbf{p}_{\text{err}}. \tag{10}$$

Here $\mathbf{J}^{-1}$ is the inverse Jacobian matrix and $\mathbf{q}_{\text{next}}$ can be used to calculate the new value of $\mathbf{p}_{\text{err}}$. When all the elements of $\mathbf{p}_{\text{err}}$ are below the stop criterion, the current vector $\mathbf{q}$ is the inverse kinematics solver.

SQP is a nonlinear optimization iterative solver, and its characteristics can be described as

$$\begin{aligned} \arg\min_{\mathbf{q}\in R^n} \quad & (\mathbf{q}_{\text{seed}} - \mathbf{q})^T (\mathbf{q}_{\text{seed}} - \mathbf{q}) \\ \text{s.t.} \quad & f_i(\mathbf{q}) \le b_i, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{11}$$

Here $\mathbf{q}_{\text{seed}}$ is the n-dimensional seed value of the joint angle, and the inequality constraint $f_i(\mathbf{q})$ is the joint limit, the Euclidean distance error, and the angular distance error.

3.6. Collision Detection. Collision detection is used in collision avoidance path planning and trajectory planning. In order to calculate the static barrier cost, Euclidean distance variation (EDT) and geometric collision detection were used. As with Ratliff et al.'s work [26], we divide the workspace into three-dimensional voxel grids and precalculate the distance between each voxel and the nearest static obstacle boundary. In addition, we approximate the shape B of the manipulator by using a set of overlapping spheres. In this case, the static barrier cost of configuring $\mathbf{q}_i$ can be found by the table in the voxel and can be calculated as follows:

$$c_s(\mathbf{q}_i) = \sum_{b\in B} \max\left(\varepsilon + r_b - d(x_b), 0\right) \left\|\dot{\mathbf{x}}_b\right\|. \tag{12}$$

Here r_b is the radius of a sphere b, x_b is the 3D point of the sphere b calculated from the kinematic model of the manipulator at configuration $\mathbf{q}_i$, $\|\dot{\mathbf{x}}_b\|$ is the signed EDT of the 3D point x, and ε is the minimum safe distance between the manipulator and the obstacle.

4. Optimal Avoidance Trajectory Planning Based on Trajectory Evaluation

After obtaining a new number of grinding targets from the scheduler, the task planner sends the planning request to the optimal collision avoidance planner, and then the optimal avoidance planner starts. The detailed procedure can be described as follows (Figure 2).

Step 1. Get the current planning task.

(i) Select *Task*4 as the current planning task and name it *Task* from the planning tasks named *Task*1, *Task*2, *Task*3, and *Task*4 received from the task planner.

(ii) Get the starting position $\mathbf{p}_{\text{start}}$ and the ending position $\mathbf{p}_{\text{end}}$ of the task *Task* in the Cartesian coordinated system.

(iii) Confirm the parameters of the manipulator and the processed billet.

Step 2. Replan in a given time.

(i) Set the planning time and planner type number *PlannerID*, and clear the number of successful planning count named *success*.

(ii) If the collision avoidance path planning named *pathplan* succeeds, start the collision avoidance trajectory planning named *trajectoryplan* and collision detecting named *collisiondetect*; otherwise duplicate *pathplan*. If both *trajectoryplan* and *collisiondetect* are successful, then save the current collision avoidance data $\mathbf{Q}$ into the container and increase *success* by one.

(iii) Repeat the steps above until the planned cost reaches the planned time.

The pseudocode of Step 2 can be found in Pseudocode 1, and its program chart can be seen in Figure 2.

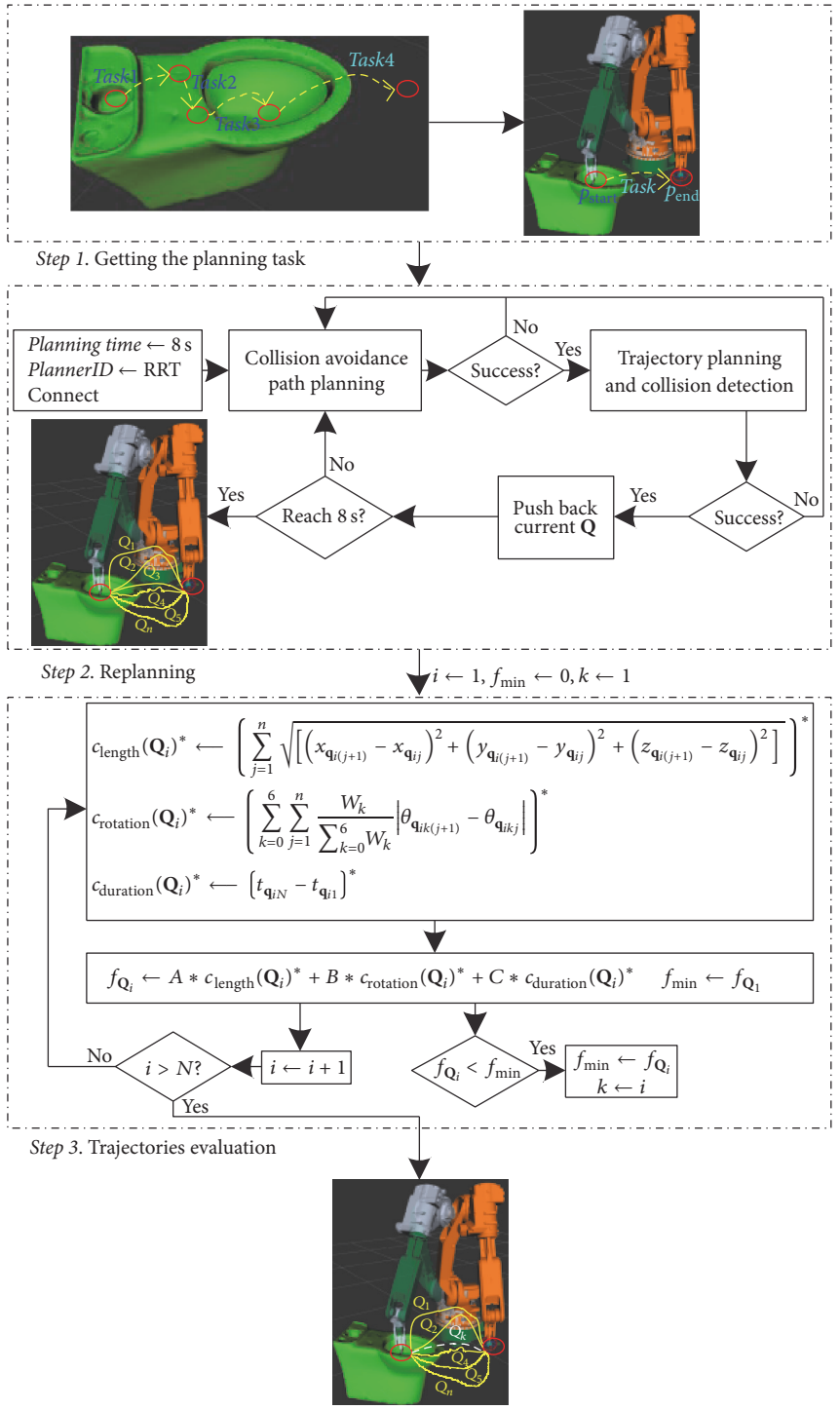


Figure 2: Procedure of the optimal collision avoidance planning.

Step 3. Evaluate the performance of the collision avoidance trajectory obtained in Step 2.

(i) Set the collision number $i = 1$, the minimum trajectory evaluation value $f_{\min} = 0$, and the optimal trajectory number $k = 1$.

(ii) Loop the following steps when $i \leq success$:

(a) Calculate the length of the current collision avoidance path and standardize it.

(b) Calculate the weighted sum of the strokes of all joints of the current collision avoidance trajectory and carry out the normalization process.

(c) Calculate the duration of the current collision avoidance trajectory and standardize it.

```
Planningtime ← 8 s
PlannerID ← RRTConnect
success ← 0
do {
    if(pathplan==1)
    {
        if ((trajectoryplan==1)&&(collisiondetect==1))
        {
          push back Q;
          success++;
        }
      else
      {
        trajectoryplan;
        collisiondetect;
    }
    end if
    }
    else pathplan;
    end if
} while(t ≤ 8 s)
```

Pseudocode 1: The pseudocode of Step 2.

(d) Calculate the evaluation value E of the current collision avoidance trajectory.

(e) Determine whether i is 1. If it is, $f_{\min} = f_{\mathbf{Q}_1}$.

(f) Determine whether the evaluation value $f_{\mathbf{Q}_i}$ of the current collision avoidance trajectory is less than the current $f_{\min}$. If it is, $f_{\min} = f_{\mathbf{Q}_i}$, and $k = i$.

The pseudocode of Step can be found in Pseudocode 2, and its program chart can be seen in Figure 2.

When the steps above are completed, the optimal collision avoidance trajectory can be obtained.

The replanning in Step 2 allows us to acquire multiple collision avoidance trajectories in a given period of time. Success is to count the total number of collision avoidance trajectories acquired during a given period of time and for Step 3 to obtain the data $\mathbf{Q}$ of the collision avoidance trajectory from the container.

In Step 3, the optimal collision avoidance trajectory is found by evaluating the performance of multiple trajectories. The weighted sum of the strokes of all joints is calculated according to (8), and on the basis of the principle of power balance, the weighting factor is calculated according to (7). The normalization process is used to calculate the length $c_{\text{length}}(\mathbf{Q}_i)$ of the current collision avoidance path, the weighted sum of the stroke of all joints $c_{\text{rotation}}(\mathbf{Q}_i)$ of the current collision avoidance trajectory, and the duration $c_{\text{duration}}(\mathbf{Q}_i)$ of the current collision avoidance trajectory.

5. Experimental Verification

This section examines the effectiveness of the optimal collision avoidance trajectory planning for different collision avoidance trajectory evaluation functions.

In order to verify the algorithm proposed in this paper, the required experimental environment is constructed, which includes the computing platform for planning the optimal collision avoidance trajectory and the redundant manipulators for executing the planned optimal collision avoidance trajectory. The computing platform for planning the optimal collision avoidance trajectory is a superior computer with a CPU of Inter (R) Xeon (R) E5-2620 v3 6-core (12-thread) 2.4 GHz and 16 GB memory. The computer's operating system is Ubuntu 16.04 LTS, the middleware is ROS Kinetic, and the motion planning framework is MoveIt. Redundant manipulators that perform optimal collision avoidance trajectory is specifically designed. The body of the manipulators is constructed by adding the seventh joint to the body of HP20F manipulators of Yasukawa Electric (China) Co., Ltd. The power of the driving motor is 1200 W, 600 W, 300 W, 120 W, 80 W, 50 W, and 50 W, respectively. The manipulators' multiaxis real-time control system is based on GUC-800-TPV motion controller of Googol Technology (Shenzhen) Ltd. The control system can communicate with third-party computers via TCP/IP protocol.

The experiment scenario of this paper is shown in Figure 3, which includes the computing platform of the optimal collision avoidance trajectory planning in Figure 3(a) and the execution platform of the optimal collision avoidance trajectory in Figure 3(b). The relative position of the manipulators and the workpiece are the same in the computing platform and the execution platform of the collision avoidance trajectory. We can see some of the grinding areas on the surface of the ceramic billet, and the planned optimal collision avoidance trajectory can be seen in Figure 3(a). The experimental verification process is as follows: Step 1: through the 3D visual inspection system, the relative position of the manipulators and the workpiece in the processing site are detected from Figure 3(b); Step 2: after obtaining the relation data of the relative positions mentioned above, the computing platform of Figure 3(a) uses the method proposed in this paper to plan the optimal collision avoidance trajectory; Step 3: the collision avoidance trajectory data obtained in Step 2 is sent to the actual machining environment in Figure 3(b) via TCP/IP protocol.

5.1. Experiments of Optimal Collision Avoidance Trajectory Planning Based on Trajectory Evaluation. Set the Planning time to 8 s, the PlannerID to RRTConnect, and use single-threaded planning. In these experiments: the position of the start point of the selected subtask is $\mathbf{P}_{\text{start}} = (-0.33973, 0.81281, 0.21828)$, and the orientation is $\mathbf{O}_{\text{start}} = (0.98068, -0.018394, 0.13372, 0.1416)$; the position of the end point is $\mathbf{P}_{\text{end}} = (-1.33973, 0.81281, 0.38828)$, and the orientation is $\mathbf{O}_{\text{end}} = (0.98068, -0.018394, 0.13372, 0.1416)$. After running the algorithm proposed in this paper, 61 sets of collision avoidance trajectories are obtained in 8 s, and different evaluation functions are set as below to find the optimal collision avoidance trajectory.

(1) When $A = 1.0$, $B = 0$, and $C = 0$, the evaluation function of the trajectory is

$$f_{\mathbf{Q}_i} = 1.0 * c_{\text{length}}(\mathbf{Q}_i)^* + 0 * c_{\text{rotation}}(\mathbf{Q}_i)^* + 0 * c_{\text{duration}}(\mathbf{Q}_i)^*. \tag{13}$$

$i \leftarrow 1$
$f_{\min} \leftarrow 0$
for i 1 to *success*
{

$$c_{\text{length}}(\mathbf{Q}_i)^* \longleftarrow \left(\sum_{j=1}^{n}\sqrt{\left(x_{\mathbf{q}_{i(j+1)}} - x_{\mathbf{q}_{ij}}\right)^2 + \left(y_{\mathbf{q}_{i(j+1)}} - x_{\mathbf{q}_{ij}}\right)^2 + \left(z_{\mathbf{q}_{i(j+1)}} - z_{\mathbf{q}_{ij}}\right)^2}\right)^*;$$

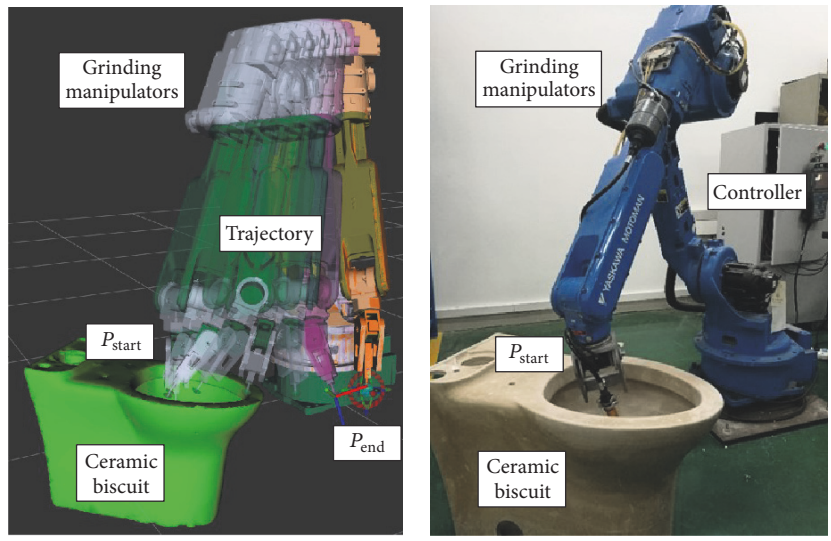
$$c_{\text{rotation}}(\mathbf{Q}_i)^* \longleftarrow \left(\sum_{k=0}^{6}\sum_{j=1}^{n}\frac{W_k}{\sum_{k=0}^{6} W_k}\left|\theta_{\mathbf{q}_{ik(j+1)}} - \theta_{\mathbf{q}_{ikj}}\right|\right)^*;$$

$$c_{\text{duration}}(\mathbf{Q}_i)^* \longleftarrow \left(t_{\mathbf{q}_{iN}} - t_{\mathbf{q}_{i1}}\right)^*;$$

$$f_{\mathbf{Q}_i} \longleftarrow A * c_{\text{length}}(\mathbf{Q}_i)^* + B * c_{\text{rotation}}(\mathbf{Q}_i)^* + C * c_{\text{duration}}(\mathbf{Q}_i)^*;$$

$f_{\min} \leftarrow f_{\mathbf{Q}_1}$;
if $(f_{\mathbf{Q}_i} < f_{\min})$
{
$f_{\min} \leftarrow f_{\mathbf{Q}_i}$;
$k \leftarrow i$;
}
end if
end for

Pseudocode 2: The pseudocode of Step 3.



(a) The computing platform of the optimal collision avoidance trajectory planning

(b) The execution platform of the optimal collision avoidance trajectory

Figure 3: Experiment scenario.

(2) When $A = 0$, $B = 1.0$, and $C = 0$, the evaluation function of the trajectory is

$$f_{\mathbf{Q}_i} = 0 * c_{\text{length}}(\mathbf{Q}_i)^* + 1.0 * c_{\text{rotation}}(\mathbf{Q}_i)^* + 0 * c_{\text{duration}}(\mathbf{Q}_i)^*. \tag{14}$$

(3) When $A = 0$, $B = 0$, and $C = 1.0$, the evaluation function of the trajectory is

$$f_{\mathbf{Q}_i} = 0 * c_{\text{length}}(\mathbf{Q}_i)^* + 0 * c_{\text{rotation}}(\mathbf{Q}_i)^* + 1.0 * c_{\text{duration}}(\mathbf{Q}_i)^*. \tag{15}$$

(4) When $A = 0.3$, $B = 0.3$, and $C = 0.4$, the evaluation function of the trajectory is:

$$f_{\mathbf{Q}_i} = 0.3 * c_{\text{length}}(\mathbf{Q}_i)^* + 0.3 * c_{\text{rotation}}(\mathbf{Q}_i)^* + 0.4 * c_{\text{duration}}(\mathbf{Q}_i)^*. \tag{16}$$

The statistical results of the evaluation value for all collision avoidance trajectories using (13)–(16) are shown in Figures 4–7, respectively. In addition, the parameters recorded in Table 1 also include the joint angles such as theta0, theta1, theta2, theta3, theta4, theta5, and theta6, the length of the

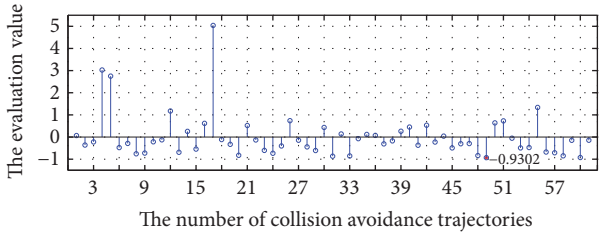


Figure 4: Evaluation of all collision avoidance trajectories when $A = 1.0$, $B = 0$, and $C = 0$.

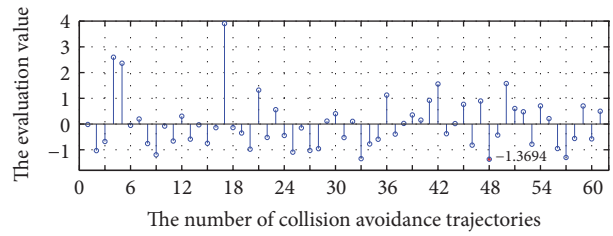


Figure 5: Evaluation of all collision avoidance trajectories when $A = 0$, $B = 1.0$, and $C = 0$.

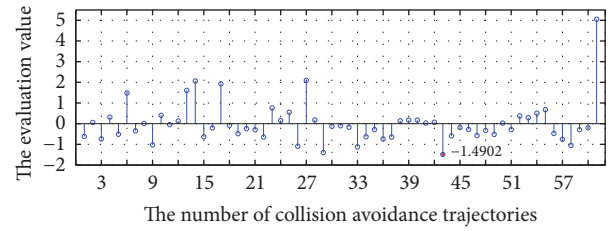


Figure 6: Evaluation of all collision avoidance trajectories when $A = 0$, $B = 0$, and $C = 1.0$.

collision avoidance path, the duration of all joint weights, and the duration of the trajectories above.

5.2. The Analysis of Experimental Results. Through the experiments above, we get the optimal collision avoidance trajectory when setting different trajectory evaluation functions. In this section, we will analyze the results of different trajectory evaluation functions.

When we take $A = 1.0$, $B = 0$, and $C = 0$ in the evaluation function, we expect the shortest collision avoidance trajectory. Now we analyze the characteristics of the collision avoidance trajectory according to the evaluation function set here. From Figure 4, we will find that all the trajectory evaluation values are unequal, and the trajectory of the smallest trajectory evaluation value which can be found is $\mathbf{Q}_{49}$. The trajectory evaluation value corresponding to this trajectory is $f_{\mathbf{Q}_{49}}$, and according to (4), we can get $F_{\text{optimal}} = f_{\mathbf{Q}_{49}} = -0.9302$. In addition, it can be seen from Table 1 that when $A = 1.0$, $B = 0$, and $C = 0$, the length of the collision avoidance trajectory $\mathbf{Q}_{49}$ is 1044.08 mm. This length is 50.1% shorter than the length of the collision avoidance trajectory obtained when $A = 0$, $B = 0$, and $C = 1.0$ in the trajectory evaluation function, which is a great improvement in trajectory performance over the trajectory length. Executing the collision avoidance trajectory $\mathbf{Q}_{49}$ on the redundant manipulators shows that the actual path length is really short. This proves that when the coefficients of the evaluation function are set as $A = 1.0$, $B = 0$, and $C = 0$, we get the length-optimal collision avoidance trajectory, and the grinding tool of the grinding manipulators can get the shortest moving distance after the trajectory is executed.

When $A = 0$, $B = 1.0$, and $C = 0$ in the evaluation function, we expect to get the collision avoidance trajectory with the least weighted sum of the strokes of all the joints. Now we analyze the characteristics of the collision avoidance trajectory according to the evaluation function set here. From Figure 5, we will find that all the trajectory evaluation values are unequal, and the trajectory of the smallest trajectory evaluation value which can be found is $\mathbf{Q}_{48}$. The trajectory evaluation value corresponding to this trajectory is $f_{\mathbf{Q}_{48}}$. According to (4), we can get $F_{\text{optimal}} = f_{\mathbf{Q}_{48}} = -1.3694$. In addition, it can be seen from Table 1 that when $A = 0$, $B = 1.0$, and $C = 0$, the weighted sum of the strokes of all the joints of the collision avoidance trajectory $\mathbf{Q}_{48}$ is 0.97886 rad. This value is 38.8% shorter than the weighted sum of the strokes of all the joints of the collision avoidance trajectory obtained when $A = 0$, $B = 0$, and $C = 1.0$ in the trajectory evaluation function, which is a large improvement in trajectory performance over the joint travel. Executing the collision avoidance trajectory $\mathbf{Q}_{48}$ on the redundant manipulators shows that the joint drive motor with larger power of the manipulator has less rotation and the joint motor with smaller power has larger rotation. This proves that when the coefficients of the evaluation function are set as $A = 0$, $B = 1.0$, and $C = 0$, we obtain the collision avoidance trajectory with the least weighted sum of the strokes of all the joints.

When we take $A = 0$, $B = 0$, and $C = 1.0$ in the evaluation function, we expect the shortest duration of the collision avoidance trajectory. Now we analyze the characteristics of the collision avoidance trajectory according to the evaluation function set here. From Figure 6, we will find that all the trajectory evaluation values are unequal, and the trajectory of the smallest trajectory evaluation value which can be found is $\mathbf{Q}_{43}$. The trajectory evaluation value corresponding to this trajectory is $f_{\mathbf{Q}_{43}}$, and according to (4), we can get $F_{\text{optimal}} = f_{\mathbf{Q}_{43}} = -1.4902$. In addition, it can be seen from Table 1 that when $A = 0$, $B = 0$, and $C = 1.0$, the duration of the collision avoidance trajectory $\mathbf{Q}_{43}$ is 4.9002 s. This duration is 27.6% shorter than the duration of the collision avoidance trajectory obtained when $A = 0$, $B = 1.0$, and $C = 0$ in the trajectory evaluation function, which is a big improvement in trajectory performance over the trajectory duration. Executing the collision avoidance trajectory $\mathbf{Q}_{43}$ on the redundant manipulators shows that the actual trajectory duration is really short. This proves that when the coefficients of the evaluation function are set as $A = 0$, $B = 0$, and $C = 1.0$, we get the collision avoidance trajectory with the least duration.

When we take $A = 0.3$, $B = 0.3$, and $C = 0.4$ in the evaluation function, we expect the integrated optimal collision avoidance trajectory. Now we analyze the characteristics of

Table 1: Parameters of the optimal avoidance trajectory obtained from different evaluation functions.

$A/B/C$	1.0/0/0	0/1.0/0	0/0/1.0	0.3/0.4/0.5
Number	49	48	43	33
θ_0 (rad)	2.21861	0.60335	1.80834	0.89023
θ_1 (rad)	0.65410	1.36034	1.51717	0.95596
θ_2 (rad)	1.13171	1.13244	1.13161	1.1332
θ_3 (rad)	0.97200	1.42901	1.7232	1.41036
θ_4 (rad)	1.11198	0.80241	0.82238	1.31284
θ_5 (rad)	2.26693	2.97239	1.93836	1.94433
θ_6 (rad)	0.94546	1.70243	0.99593	0.63767
Length (mm)	**1044.08**	1183.91	2093.14	**1155.9**
Weighted rotation (rad)	1.56684	**0.97886**	1.59959	**0.99378**
Time (s)	6.4620	6.7684	**4.9002**	**5.4983**

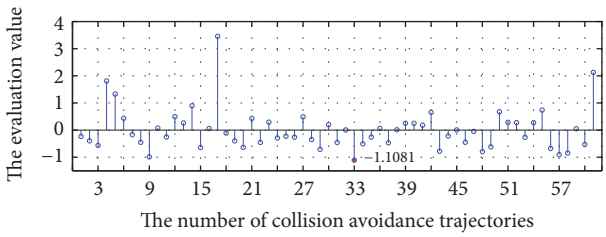


Figure 7: Evaluation of all collision avoidance trajectories when $A = 0.3$, $B = 0.3$, and $C = 0.4$.

the collision avoidance trajectory according to the evaluation function set here. From Figure 7, we will find that all the trajectory evaluation values are unequal, and the trajectory of the smallest trajectory evaluation value which can be found is $\mathbf{Q}_{33}$. The trajectory evaluation value corresponding to this trajectory is $f_{\mathbf{Q}_{33}}$, and according to (4), we can get $F_{\text{optimal}} = f_{\mathbf{Q}_{33}} = -1.1081$. In addition, it can be seen from Table 1 that when $A = 0.3$, $B = 0.3$, and $C = 0.4$, the length of collision avoidance trajectory $\mathbf{Q}_{33}$ is 1155.9 mm, the weighted sum of all joint travels is 0.99378 rad, and the duration is 5.4983 s. The length of collision avoidance trajectory is 44.8% shorter than that of $A = 0$, $B = 0$, and $C = 1.0$ in the trajectory evaluation function. The weighted sum of the strokes of all the joints is 37.9% shorter than that of $A = 0$, $B = 0$, and $C = 1.0$ in the trajectory evaluation function, and the duration was 18.8% shorter than the duration of collision avoidance trajectories obtained when $A = 0$, $B = 1.0$, and $C = 0$ in the trajectory evaluation function. Performing the collision avoidance trajectory $\mathbf{Q}_{33}$ on the redundant manipulators shows that its overall performance is the best. This proves that when the coefficients of the evaluation function are set as $A = 0.3$, $B = 0.3$, and $C = 0.4$, we get the integrated optimal collision avoidance trajectory.

6. Discussion

This section discusses some properties of the proposed optimal collision avoidance trajectory planning method, including (1) the influence of the set replanning time on the optimal trajectory; (2) the statistical law of the parameters in the collision avoidance trajectory.

In order to obtain the effect of replanning time on the relevant parameters in the collision avoidance trajectory and the characteristics of these parameters, we set the planning time as 1 s, 2 s, 3 s, 4 s, 5 s, 6 s, 7 s, and 8 s, use RRTConnect as the collision avoidance path planner, employ TRAC-IK to calculate the inverse kinematics solver, and repeat 20 times the algorithm above with a single thread.

6.1. The Impact of Replanning Time on the Optimal Trajectory. The relationship between the given planning time and the output number of collision avoidance trajectories is shown in Figure 8. The relationship between the given planning time and the length of the collision avoidance path is shown in Figure 9. The relationship between the given planning time and the weighted sum of the strokes of all joints is shown in Figure 10. The relationship between the given planning time and the duration of the collision avoidance trajectory is shown in Figure 11. Figure 8 shows that the number of collision avoidance trajectories increases linearly as the given planning time increases. Figures 9–11 show that the mean and minimum values of the duration of the collision avoidance trajectories, the length of the collision avoidance path, and the weighted sum of the strokes of all joints of the collision avoidance trajectory have not changed significantly as the given planning time increases. The mean and minimum values of the length of the collision avoidance are stable at around 2553 mm and about 776 mm, respectively. The mean sum of the weighted sums of all joints of the collision avoidance trajectory is about 1.74 rad and 0.79 rad. The mean and minimum values of the duration of the collision avoidance are stable at around 7.34 s and about 4.22 s.

An analyzing of the results above shows that using the collision avoidance trajectory planning method proposed in this paper can greatly improve the performance of the trajectory: the length of the collision avoidance path is 69.6% shorter than the mean length of the randomly generated paths; the weighted sum of the strokes of all the joints is 54.6% less than the mean of the randomly generated collision avoidance trajectories; and the duration of collision avoidance trajectory is 42.5% shorter than the mean duration of randomly generated collision avoidance trajectories.

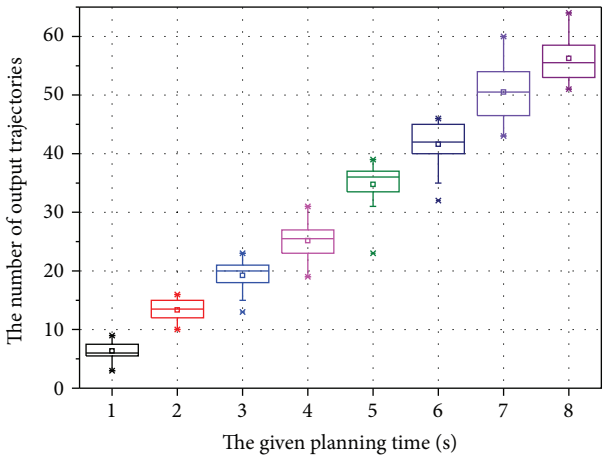


Figure 8: The relationship between the given planning time and the output number of collision avoidance trajectories.

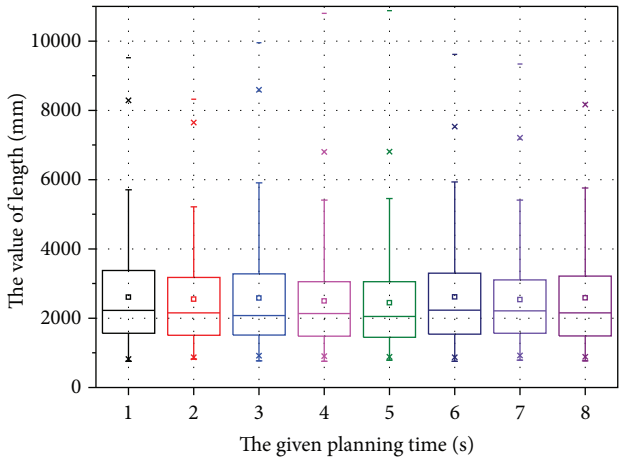


Figure 9: The relationship between the given planning time and the length of the collision avoidance path.

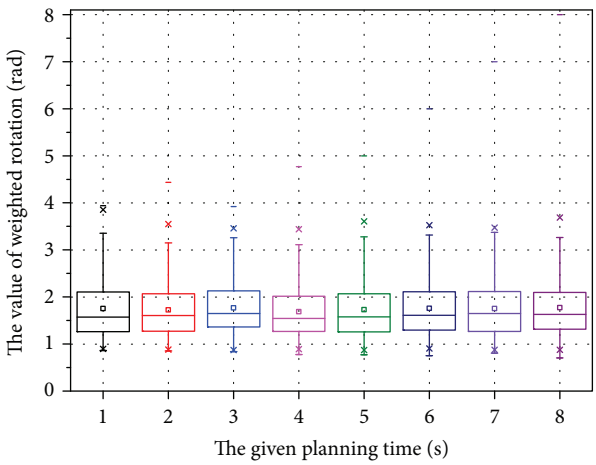


Figure 10: The relationship between the given planning time and the weighted sum of the strokes of all joints.

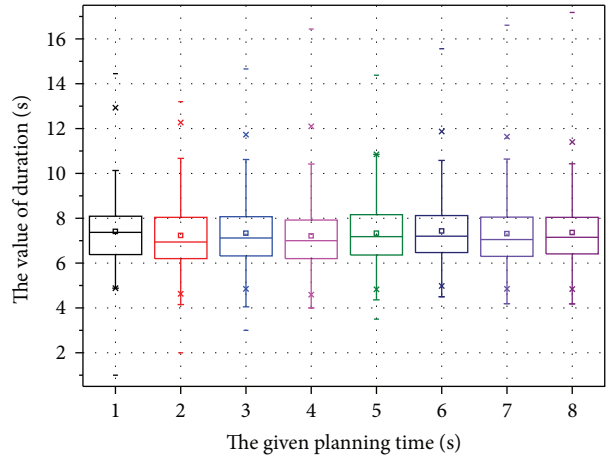


Figure 11: The relationship between the given planning time and the duration of the collision avoidance trajectory.

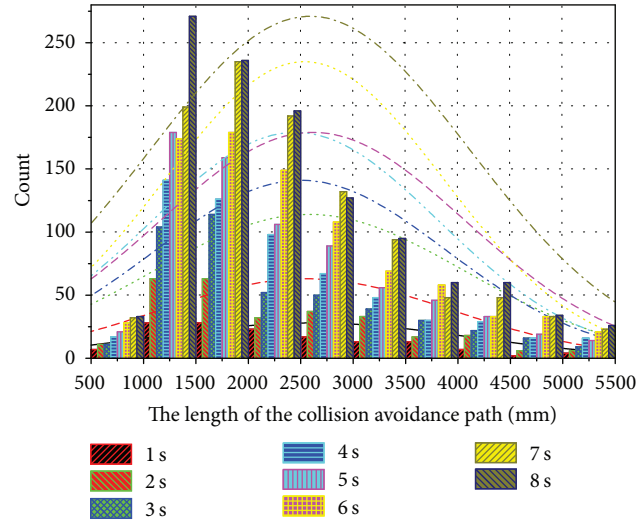


Figure 12: Distribution of the value of the collision avoidance path when given different planning time.

6.2. The Statistical Laws of the Parameters in the Collision Avoidance Trajectory. Figures 12–14 show the distribution of the collision avoidance path length, the weighted sum of the strokes of all joints, and the duration of the collision avoidance trajectories when given different planning time. From these figures we can find that the parameters above are subject to the normal distribution. The mean distribution can be seen from Figures 9–11, and we can find the specific values of the mean of these parameters in Section 6.1.

7. Conclusion

In this paper, we propose an optimal trajectory planning method which is applied to achieve the automatic planning avoidance trajectory function in the process of grinding ceramic billet surface. The core idea of this approach is to use the evaluation function to evaluate all the collision avoidance

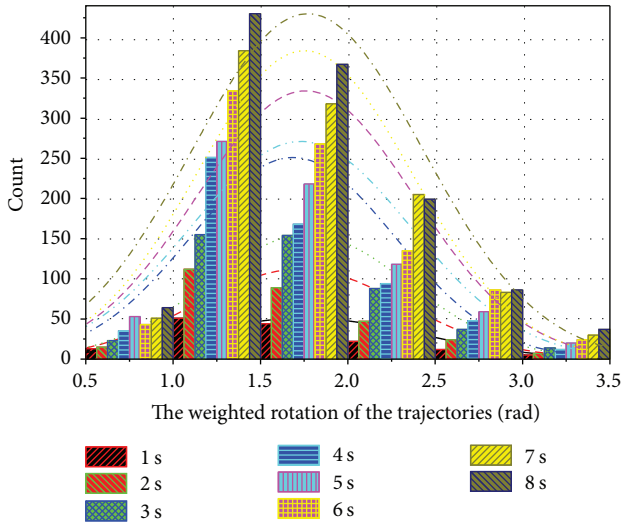


Figure 13: Distribution of weighted sums of strokes of all joints when given different planning time.

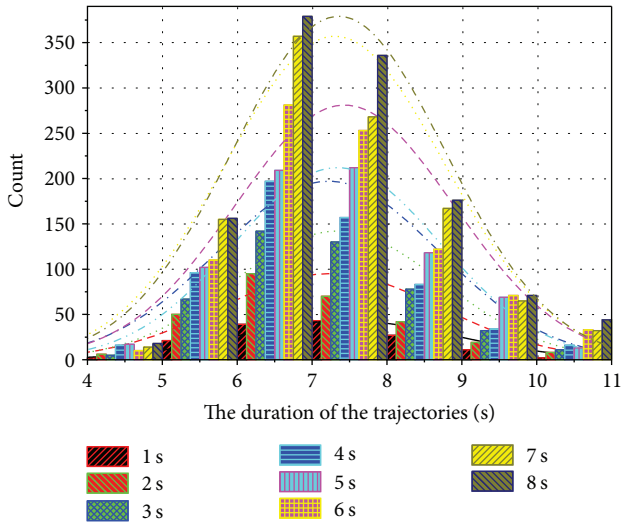


Figure 14: Distribution of the values of the duration when given different planning time.

trajectories obtained by replanning and return the optimal collision avoidance trajectory to the manipulator.

The experiments show that the proposed optimal avoidance trajectory planner can return the optimal collision avoidance trajectory when the evaluation function is different. The statistical results obtained through a large number of repeated experiments show that the three parameters in the collision avoidance trajectories are subject to the normal distribution, and the influence of the replanning time on the optimal trajectory is not very obvious.

The basis of the optimal avoidance trajectory planning includes the evaluation function of the collision avoidance trajectory and its parameter calculation, the inverse kinematics solver of the redundant manipulators, the collision avoidance path planner, and the collision detection. In this paper, the calculation of the weighted sum of the strokes of all joints in the evaluation function only takes into account the rotation of the joints and does not take into account the moments in rotation process. Therefore, it is necessary to take into account the moment when the energy saving is calculated accurately. The inverse kinematics solver of the redundant manipulators, the collision avoidance path planner, and the collision detection are based on the existing research results. If a higher computational accuracy or computational speed is required, the corresponding better performance algorithm can be selected.

The optimal avoidance trajectory planning method proposed in this paper can meet the task requirements of surface grinding. However, the process of surface grinding is very complicated; it is necessary to consider the dynamics of the manipulator and the dynamic force-position relation of the grinding process. In the future, we will research into the control of the force to make the polished surface meet the demanding process requirements.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The study was supported by Science and Technology Planning Project of Guangdong Province, China (2016B090912002 and 2015B010919001), and the National High Technology Research and Development Program of China (2013AA041001).

References

[1] J. S. Yu and P. C. Müller, "An on-line cartesian space obstacle avoidance scheme for robot arms," *Mathematics and Computers in Simulation*, vol. 41, no. 5-6, pp. 627–637, 1996.

[2] D. P. Garg and M. Kumar, "Optimization techniques applied to multiple manipulators for path planning and torque minimization," *Engineering Applications of Artificial Intelligence*, vol. 15, no. 3-4, pp. 241–252, 2002.

[3] C.-J. Lin, "Motion planning of redundant robots by perturbation method," *Mechatronics*, vol. 14, no. 3, pp. 281–297, 2004.

[4] T. Chettibi, H. E. Lehtihet, M. Haddad, and S. Hanchi, "Minimum cost trajectory planning for industrial robots," *European Journal of Mechanics - A/Solids*, vol. 23, no. 4, pp. 703–715, 2004.

[5] M. da Graça Marcos, J. A. Tenreiro Machado, and T.-P. Azevedo-Perdicoúlis, "Trajectory planning of redundant manipulators using genetic algorithms," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 7, pp. 2858–2869, 2009.

[6] R. Menasri, A. Nakib, B. Daachi, H. Oulhadj, and P. Siarry, "A trajectory planning of redundant manipulators based on bilevel optimization," *Applied Mathematics and Computation*, vol. 250, pp. 934–947, 2015.

[7] M. D. G. Marcos, J. A. Tenreiro Machado, and T.-P. Azevedo-Perdicoúlis, "A multi-objective approach for the motion planning of redundant manipulators," *Applied Soft Computing*, vol. 12, no. 2, pp. 589–599, 2012.

[8] T. Chakraborti, A. Sengupta, A. Konar, and R. Janarthanan, "Application of swarm intelligence to a two-fold optimization scheme for trajectory planning of a robot arm," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 7077, no. 2, pp. 89–96, 2011.

[9] A. Shukla, E. Singla, P. Wahi, and B. Dasgupta, "A direct variational method for planning monotonically optimal paths for redundant manipulators in constrained workspaces," *Robotics and Autonomous Systems*, vol. 61, no. 2, pp. 209–220, 2013.

[10] E. J. S. Pires, P. B. de Moura Oliveira, and J. A. T. Machado, "Manipulator trajectory planning using a MOEA," *Applied Soft Computing*, vol. 7, no. 3, pp. 659–667, 2007.

[11] L. E. Kavraki, P. Švestka, 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.

[12] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," *Proceedings-IEEE International Conference on Robotics and Automation*, vol. 1, pp. 521–528, 2000.

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

[14] A. Dobson, A. Krontiris, and K. E. Bekris, "Sparse roadmap spanners," *Springer Tracts in Advanced Robotics*, vol. 86, pp. 279–296, 2013.

[15] A. Dobson and K. E. Bekris, "Improving sparse roadmap spanners," in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation, ICRA 2013*, pp. 4106–4111, Germany, May 2013.

[16] A. Dobson and K. E. Bekris, "Sparse roadmap spanners for asymptotically near-optimal motion planning," *International Journal of Robotics Research*, vol. 33, no. 1, pp. 18–47, 2014.

[17] J. Kuffner and S. LaValle, "RRT-Connect: An efficient approach to single-query path planning ieee international conference on robotics and automation," San Francisco, USA, 2000.

[18] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2013.

[19] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *International Journal of Robotics Research*, vol. 35, no. 5, pp. 528–564, 2015.

[20] D. Devaurs, T. Simeon, and J. Cortes, "Enhancing the transition-based RRT to deal with complex cost spaces," in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation, ICRA 2013*, pp. 4120–4125, Germany, May 2013.

[21] S. Gildardo and L. Jean-Claude, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," *Robotics Research*, pp. 403–417, 2003.

[22] H. David, J-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," *European Journal of Mechanics-A/Solids*, vol. 3, pp. 2719–2726, 1997.

[23] I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," *Springer Tracts in Advanced Robotics*, vol. 57, pp. 449–464, 2010.

[24] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Motion planning with dynamics by a synergistic combination of layers of planning," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 469–482, 2010.

[25] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *Proceedings of the 15th IEEE RAS International Conference on Humanoid Robots, Humanoids 2015*, pp. 928–935, Republic of Korea, November 2015.

[26] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 489–494, Kobe, May 2009.