Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Collision-free Trajectory Planning Algorithm for Manipulators in Unstructured Environment

Zhaolei Hou
[1]*College of Information Science and Engineering, Northeastern University*
[2]*State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences*
[3]*Institues for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences*
Shenyang, Liaoning Province, China
houzhaolei@sia.cn

Yong Jiang and Hao Wang
[1]*State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences*
[2]*Institues for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences*
Shenyang, Liaoning Province, China
{jiangyong & wanghao1}@sia.cn

***Abstract* - In this article, we propose a collision-free trajectory planning method for the manipulator that satisfies the robot's operational requirements in unstructured environment. The algorithm considers the collision-free trajectory requirements of the end effector and the manipulator body. The reactive trajectory planning algorithm in the local environment is used to superimpose the control quantity on the velocity layer, which avoids the mechanical dynamics model in the calculation process, Parameter acquisition and computational complexity issues. The method is simulated by MATLAB and further implemented on the UR manipulator. The simulation and experimental results show the real-time and effectiveness of the algorithm.**

Index Terms - Manipulator, Trajectory planning, Collision-free, Unstructured Environment.

I. Introduction

The need for collaborative robots to be in harmony with people and to work with people requires robots to have the ability to sense the environment and plan their movements. The ability of the robot to sense the surrounding environment and obtain the appropriate information is the basis for the safety of the robot and humans. On this basis, the robot needs to respond to environmental information, such as stopping the movement or planning a new trajectory. In order to solve the above problems, the researchers proposed a series of planning algorithms. The global planning algorithms mainly include randomly exploring randomized trees, graph search algorithms, and grid map methods. The local planning method is mainly the artificial potential field method and related modification.

The working space of the manipulator is relatively fixed. Generally, the global planning algorithms used by the robot, such as Randomly Exploring Randomized Trees and graph search algorithms, can plan a new path in a limited time, but the solution efficiency is low, and the global environment information needs to be known. The local planning method does not need to obtain global information, and relies on local information to constrain the movement of the robot arm, thereby obtaining the next motion instruction. Therefore, compared with the global programming algorithm, the local trajectory planning algorithm more satisfies the motion planning requirements of the robot arm. Because the mechanical arm is a series of hinged multi-rigid body, there is mutual coupling between the connecting rod and the connecting rod, and the traditional motion planning algorithm can not directly realize the overall obstacle avoidance of the mechanical arm. At present, there are roughly the following methods for the overall obstacle avoidance of the mechanical arm. One is to simply superimpose the amount of control [1]. However, the obstacle avoidance of the connecting rod under the manipulator will lead to the movement of the upper connecting rod, which may lead to oscillations and even failure to avoid obstacles. The second is to make use of the redundant space of the manipulator. Tasks with low priority work in the redundant space of the main task [3][9][13]. The third is to transform the planning problem into an optimization problem [4][10]. First give a desired speed or pose, and then try to make the actual speed or pose close to the expected value under the given constraints.

An improved robot trajectory planning method is proposed in [1]. The author introduces the boundaries of Cartesian spatial components to optimize the attracting field function. By introducing speed feedforward, the stability of the manipulator is proved by Lyapunov theory. The control amount of each joint of the robot arm is directly linearly superimposed. In [2], the control amount is applied to the velocity layer, and the chattering phenomenon near the obstacle is eliminated by introducing a spring damping system. The parameters of the spring damping system are optimized using the fuzzy control method. In [3], an improved robotic obstacle avoidance algorithm based on joint space redundancy is designed. The authors designed a manual parallel system that allows the robotic arm to smoothly return to the target pose after the obstacle is removed. The proposed dimension reduction method makes better use of the redundancy of the robot. The literature [4] modifies the direction of the repulsion vector, avoiding the problem that the repulsion of the operator hinders the traction force when the end effector is pulled. The constraint condition ensures that the internal product of the velocity vector and the repellent unit vector at the control point is positive. In [5], an obstacle avoidance algorithm based on two repulsive potential fields is designed. The end trajectory is set to the highest priority, and the obstacle avoidance task of the robot arm is set as the secondary task and works in the zero space of the main task. This method tries to ensure the completion of the end of the manipulator actuator task, but the reliability of the obstacle avoidance cannot be guaranteed. Collisions may occur when the manipulator does not guarantee that the end effector is not affected. Document [6] achieves a smooth transition between different priority tasks by releasing the degrees of freedom that high-priority tasks do not use. The proposed method proves the stability of the algorithm by using the Lyapunov method. Literature [7] proposed a collision avoidance method based on distance modulation. The method uses the velocity and distance of the obstacle to calculate the modulation matrix of the dynamic system. Literature [8] evaluated the distance between the manipulator and the obstacle in depth space, and then generated a repulsion vector for obstacle avoidance.

This paper presents a mechanical arm obstacle avoidance method in a non-structural environment. The obstacle avoidance task has a higher priority than the end effector task, thus ensuring that the robot arm does not collide due to the work task. The control amount is applied to the velocity layer, which avoids the solution of the kinematic model, and the smoothing of the motion trajectory is processed by the smoothing factor.

The rest of this article is organized as follows. The second part describes the acquisition of obstacle information in three-dimensional space, including the extraction of the manipulator and the detection of obstacles. The third part describes the control strategy of the robot arm. The fourth part experiments the effectiveness of the algorithm. The last part is summarized.

II. Obstacle Detection

The detection of obstacles is mainly to obtain the distance between the robot arm and the obstacle. Before performing the test, it is first necessary to separate the robot arm from the environment to prevent the robot arm from seeing itself as an obstacle. Through the solution of the positive kinematics of the manipulator, the position of each joint of the manipulator in the working space can be obtained. Combined with the LCCP algorithm, the manipulator can be extracted from the three-dimensional point cloud. The detection of the obstacle distance is searched using an octree structure.

A. Manipulator Separation

The UR manipulator has six degrees of freedom, and its three-dimensional model and coordinate system are shown in Fig.1. The D-H matrix analysis method is applied to the manipulator to establish the link coordinate system of each joint. Assuming that the rotation angles of the joints from the pedestal to the end effector in the joint space are sequentially, the transformation matrix between the links is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1) $$

In the formula, ${}^{i-1}_{i}T$ is a coordinate transformation matrix of the link i with respect to the link $i-1$. For convenience of representation, $\cos\theta_i$ is abbreviated as $c\theta_i$ and $\sin\theta_i$ is abbreviated as $s\theta_i$.

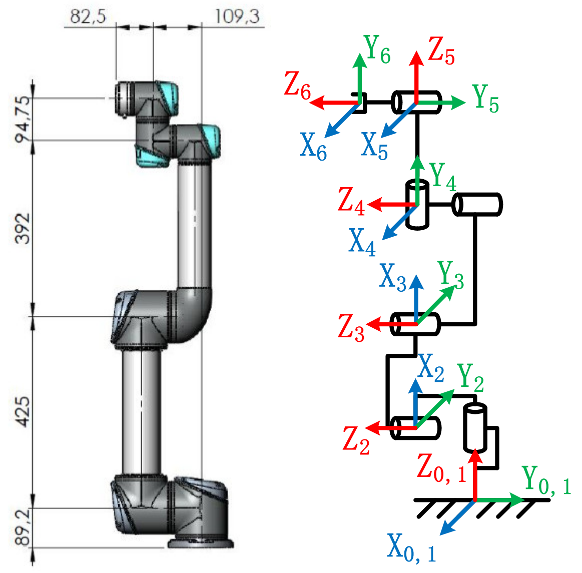

Fig.1 The three-dimensional model (a) and the coordinate system (b).

Assuming that the coordinates of the next point in the i coordinate system of the link are $P_i = (x_i \quad y_i \quad z_i \quad 1)^T$, the position of P_i in the base coordinate system can be obtained according to equation (2). The coordinates of the P are converted from the base coordinate system to the camera coordinate system to obtain the position of the robot arm in the world coordinate system.

$$ P = {}^{0}_{1}T\,{}^{1}_{2}T \cdots {}^{i-3}_{i-2}T\,{}^{i-2}_{i-1}T\,{}^{i-1}_{i}TP_i \qquad (2) $$

Locally Convex Connected Patches is a non-learning bottom-up segmentation algorithm based on 3D point cloud [11]. The algorithm is roughly divided into two parts. First, a surface patch adjacency graph is created using Voxel Cloud Connectivity Segmentation, and then spatial information and normal information are used for re-clustering.

The amount of point cloud data collected by the depth camera is large, and it takes a lot of time to directly process it, which is not conducive to the robot arm avoiding obstacles in real time. The position of the arm in the three-dimensional environment can be obtained by using the positive kinematics of the arm. By setting the detection space to filter out data that has no effect on the job task, the detection efficiency of the robot arm and obstacles can be greatly improved. The segmentation result is shown in Fig.2.

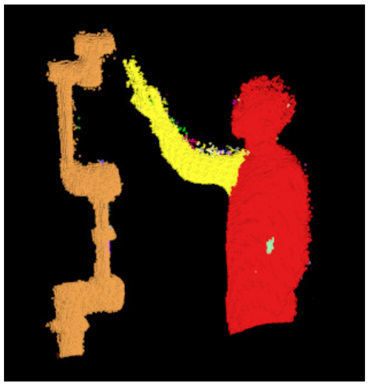

Fig.2 The result of the segmentation in the detection area.

B. *Distance Detection*

The detection of the distance between the manipulator and the obstacle is mainly to obtain the distance of the robot arm itself from one or more points closest to the obstacle. Taking the shortest distance as an example, this paper uses a series of control points on the robot arm to search and traverse the octree structure to obtain the nearest point and the closest distance. Set the control point of the arm to q_i , and the nearest point of the corresponding obstacle is b_i , then the closest distance between the arm and the obstacle is

$$d_{\min} = \min\{\|\overrightarrow{q_1 b_1}\|, \|\overrightarrow{q_2 b_2}\|, \cdots, \|\overrightarrow{q_n b_n}\|\} \tag{3}$$

Assuming that the obstacle point corresponding to $d_{\min}$ is $b_{\min}$, the unit direction vector of the nearest point is

$$\vec{n}_{\min} = \frac{\overrightarrow{q_{\min} b_{\min}}}{d_{\min}} \tag{4}$$

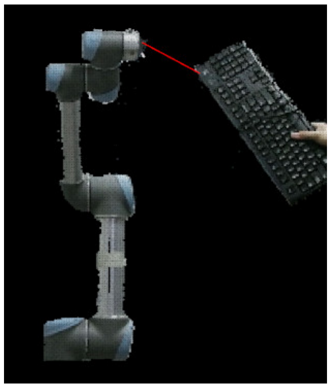

Fig.3 Detection of the nearest point

III. Control Strategy

The obstacle avoidance of the manipulator refers to the encounter of dynamic or static obstacles during the operation, and the robot arm can complete the end task while avoiding the obstacle, as shown in Fig.4. When a joint link of the robot arm enters the range of the obstacle, the obstacle will apply a repulsive force to the arm. At this time, the arm needs to adjust its configuration and try not to affect the working trajectory of the end effector.

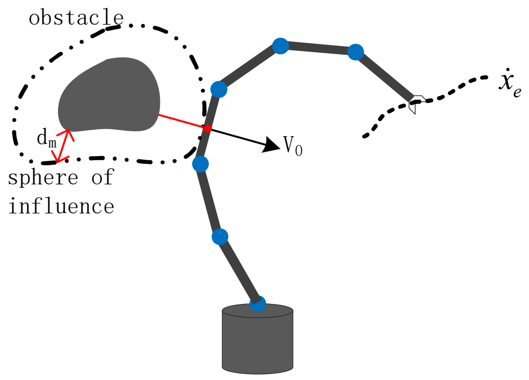

Fig.4 Repulsive effect of obstacles on the manipulator

The relationship between the speed of the end effector of the manipulator and the joint speed can be expressed by the equation (5).

$$\dot{x} = J\dot{\theta} \tag{5}$$

For a redundant robotic arm, its Jacobian matrix is not a square matrix, so the expression of joint velocity is

$$\dot{\theta} = J^{\dagger}\dot{x} + (I - J^{\dagger}J)\dot{\theta}_w \tag{6}$$

In the formula, $J^{\dagger} = J^T (JJ^T)^{-1}$ is the pseudo-inverse of the Jacobian matrix, and $\dot{\theta}_w$ is the arbitrary joint velocity. Since the joint velocity is in the redundant space, it can be used to perform secondary tasks such as obstacle avoidance.

Considering the priority of the task at the end of the arm and the priority of the obstacle avoidance task, when the job task of the end actuator is set to high priority, if the manipulator can not complete the job under the condition of obstacle avoidance, it will fail to avoid the obstacle. Setting an obstacle avoidance task to a high priority may result in a job failure, but it guarantees the safety of the robotic arm. The expression of the obstacle avoidance of the manipulator is

$$\dot{x}_0 = J_0\dot{\theta} \tag{7}$$

Where J_0 is the Jacobian matrix at the obstacle avoidance point. Combine it with the equation (8), we can get

$$\dot{\theta}_w = (J_e(I - J_0{}^{\dagger}J_0))^{\dagger}(\dot{x}_e - J_e J_0{}^{\dagger}\dot{x}_0) \tag{8}$$

Therefore, the expression that can satisfy both the end actuator job task and the obstacle avoidance task is

$$\dot{\theta} = J_0{}^{\dagger}\dot{x}_0 + (I - J_0{}^{\dagger}J_0)(J_e(I - J_0{}^{\dagger}J_0))^{\dagger}(\dot{x}_e - J_e J_0{}^{\dagger}\dot{x}_0) \tag{9}$$

If there is an obstacle on the desired trajectory of the end of the robot arm, then the mission trajectory needs to be adjusted to prevent collision. Thus the end effector has a velocity function of $\dot{x}_e = v_a + v_o$, where v_a is the attraction speed of the end effector and v_o is the rejection speed of the obstacle. d_m is the range of action of the obstacle. The smaller the distance between the robot arm and the obstacle, the greater the repulsion speed.

$$\dot{x}_a = V_a = \begin{cases} 1 + d_a - e^{d_a - d}, & d > d_a \\ d, & d \le d_a \end{cases} \tag{10}$$

$$\dot{x}_0 = V_0 = \begin{cases} \vec{n}_0 \cdot \dfrac{1}{1 + e^{(d_0(2/d_m)-1)\gamma}}, & d_0 < d_m \\ 0, & d_0 \ge d_m \end{cases} \tag{11}$$

It is assumed that the unit direction vector of the movement of the arm is $\vec{n}_m$, the unit direction vector of the obstacle closest to the arm is $\vec{n}_p$, and the unit direction vector of the obstacle movement is $\vec{n}_{obs}$. When $\angle(\vec{n}_m, \vec{n}_{obs}) \ge 90°$, the obstacle moves away from the arm, at this time $\vec{n}_0 = \vec{n}_p$. When $\angle(\vec{n}_m, \vec{n}_{obs}) < 90°$, the obstacle moves toward the arm. If the obstacle avoidance direction of the manipulator is not adjusted, if the moving speed of the obstacle is greater than the moving speed of the arm, a collision may occur. The adjusted obstacle avoidance direction vector is to superimpose the direction vector $\vec{n}_c$ on the basis of $\vec{n}_p$, as shown in Fig.5. Where $\vec{n}_c = \vec{n}_{c1} + \vec{n}_{c2}$, $\vec{n}_{c1}$ is perpendicular to the plane formed by $\vec{n}_p$ and $\vec{n}_{obs}$, ie $\vec{n}_{c1} = \vec{n}_p \times \vec{n}_{obs}$. The direction of $\vec{n}_{c2}$ is perpendicular to the plane formed by $\vec{n}_{c1}$ and $\vec{n}_{obs}$, and the angle with $\vec{n}_p$ is less than $90°$.

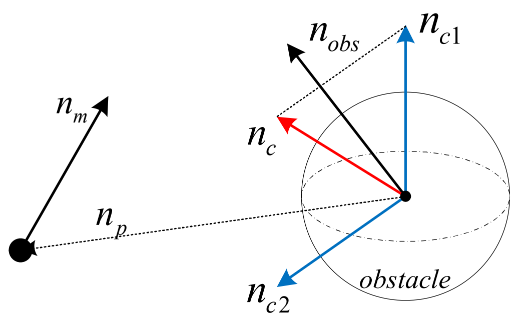

Fig.5 Unit direction vector superimposed in rejection velocity

For redundant manipulators, the degree of freedom of joint space is greater than the dimension of the task space, and the difference between the two is redundancy [12]. Thus, when the task dimension is less than the number of degrees of freedom of the robot arm, a structurally non-redundant robotic arm can be redundant for a particular task. The robot arm used in this paper is a UR cooperative robot arm with six degrees of freedom. The end effector in the workspace needs three degrees of freedom to reach the target position, and the remaining degrees of freedom can meet the obstacle avoidance requirement. The Jacobian matrix of the end operation and the Jacobian matrix at the obstacle avoidance point are respectively

$$\dot{x}_e = J_e \dot{\theta} \ , \ J_e = J_{1:3} \tag{12}$$

$$\dot{x}_0 = J_0 \dot{\theta} \ , \ J_0 = J_{obs,1:3} \tag{13}$$

Where $J_{1:3}$ is the first three rows of the Jacobian matrix at the end point and $J_{obs,1:3}$ is the first three rows of the Jacobian matrix at the obstacle avoidance point.

Since the point closest to the obstacle changes at any time, this may cause discontinuity in the joint speed. To reduce this effect, the expression for modifying the job task is as shown in equation (15). The γ and β in the α_h expression are adjustable parameters.

$$\alpha_h = \frac{e^{-\gamma(x-\beta)}}{e^{-\gamma(x-\beta)} + e^{\gamma(x-\beta)}} \tag{14}$$

$$\dot{\theta} = \alpha_h J_0^{\dagger} \dot{x}_0 + (I - J_0^{\dagger} J_0)(J_e(I - J_0^{\dagger} J_0))^{\dagger}(\dot{x}_e - \alpha_h J_e J_0^{\dagger} \dot{x}_0) \tag{15}$$

IV. Simulation and Experiment

In this work, simulations were performed using MATLAB, and two sets of experiments were designed to verify the effectiveness of the proposed method. The first set of experiments is the effect of static obstacles on the manipulator, and the second set of experiments is the effect of dynamic obstacles on the mechanical arm.

A. Static Obstacle

In the static obstacle experiment, the end effector of the robot arm moves from the initial point to the target point. The obstacle is placed on the desired trajectory of the robot arm. The experimental results are shown in Fig.6. As can be seen from the figure, when the manipulator approaches the obstacle, the angle of each joint is adjusted to successfully avoid obstacles. The movement of the third, fourth and sixth joints of the manipulator is shown in the figure, from which the adjustment of the trajectory of the manipulator can also be seen. The angle change of each joint is shown in Fig.7.

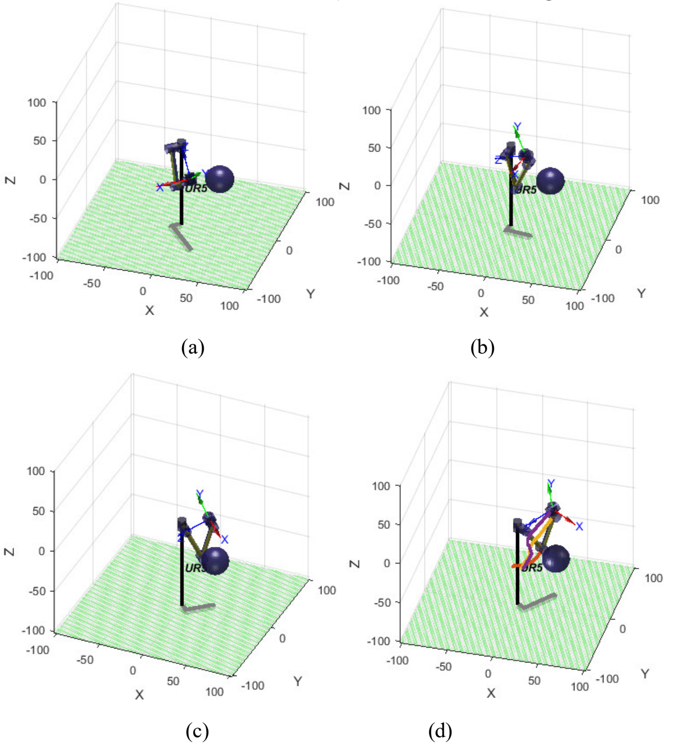

Fig.6 Simulation results of static obstacles

B. Dynamic Obstacle

In the dynamic obstacle experiment, the end effector of the robot arm moves from the initial point to the target point. The obstacle gradually approaches the robot arm from a distance. The experimental results are shown in Fig.8. As can be seen from the figure, when the obstacle approaches the arm, the arm adjusts its own trajectory to successfully avoid obstacles. The angle change of each joint is shown in Fig.9.

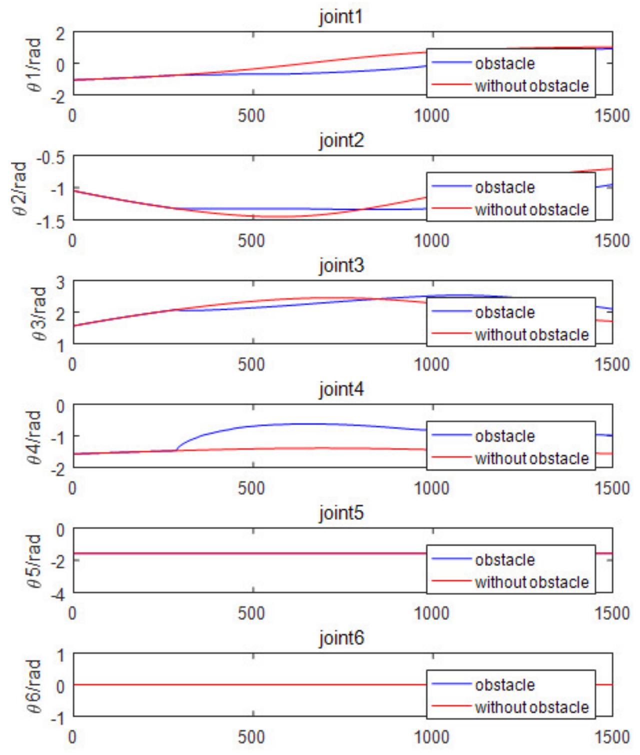

Fig.7 Angle change of joints in static obstacle experiment

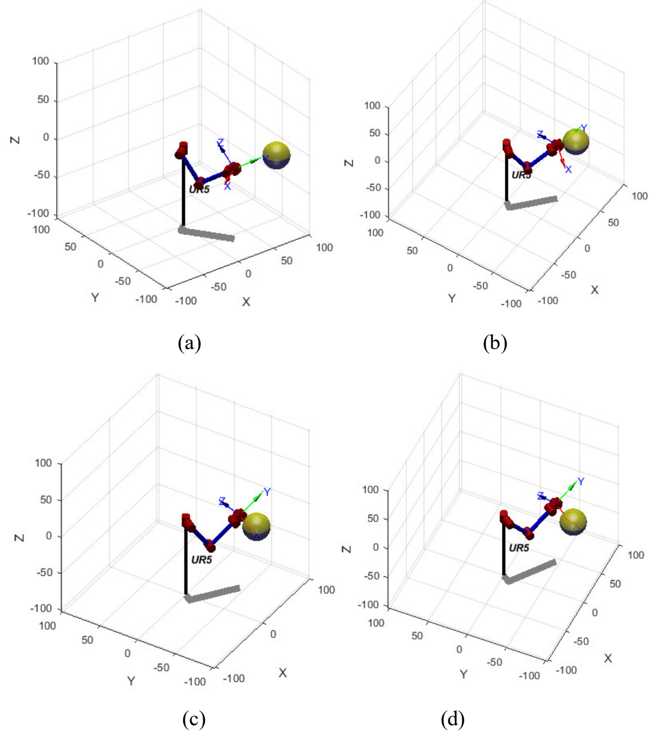

Fig.8 Simulation results of dynamic obstacles

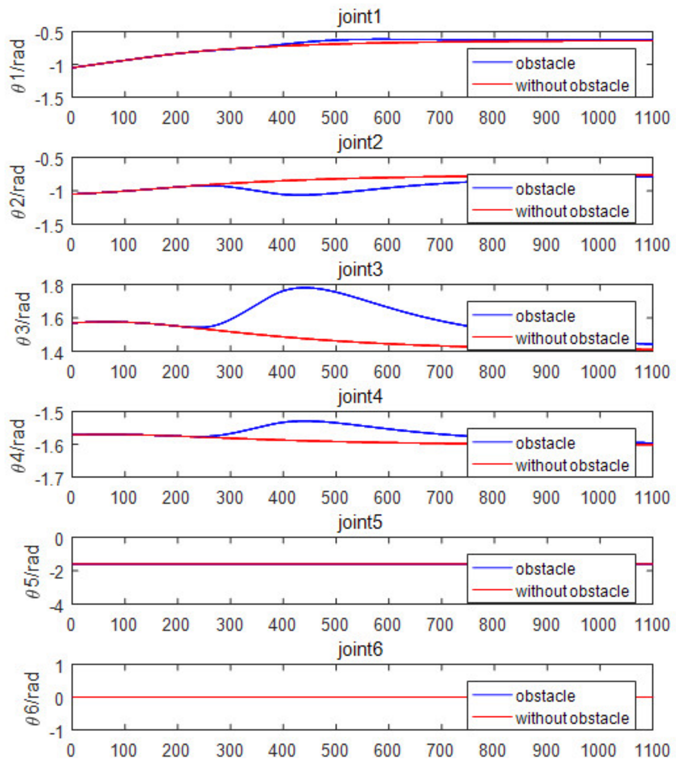

Fig.9 Angle change of joints in dynamic obstacle experiment

V. Conclusion

This paper presents a method of collision avoidance of mechanical arms in a non-structural environment. The method considers both static and dynamic obstacles. For obstacles moving toward the arm, the avoidance is done by modifying the direction vector of the obstacle avoidance. Setting the obstacle avoidance task to a high priority avoids the collision of the robotic arm in order to complete the task of the end effector. For the problem of discontinuity of joint velocity during exercise, set the smoothing factor for processing. Simulation experiments demonstrate the reliability and effectiveness of the proposed method. In the future work, more complex obstacles will be considered, including the influence of parameters such as shape and size, so that the robotic arm can adapt to more complex and varied environments.

References

[1] Wang W , Zhu M , Wang X , et al. An improved artificial potential field method of trajectory planning and obstacle avoidance for redundant manipulators[J]. International Journal of Advanced Robotic Systems, 2018, 15(5).

[2] Xu X , Hu Y , Zhai J M , et al. A novel non-collision trajectory planning algorithm based on velocity potential field for robotic manipulator[J]. International Journal of Advanced Robotic Systems, 2018, 15(4).

[3] Wang X , Yang C , Ma H , et al. Shared control for teleoperation enhanced by autonomous obstacle avoidance of robot manipulator[C]. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2015.

[4] Lin H C , Fan Y , Tang T , et al. Human guidance programming on a 6-DoF robot with collision avoidance[C]. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016.

[5] LEITE A C, ALMEIDA-ANTONIO T B, FROM P J, et al. Control and obstacle collision avoidance method applied to human-robot

interaction[C]. IEEE International Workshop on Advanced Robotics and its Social Impacts, Lyon, 2015: 1-8.

[6] Petrič T , Žlajpah, Leon. Smooth continuous transition between tasks on a kinematic control level: Obstacle avoidance as a control problem[J]. Robotics and Autonomous Systems, 2013, 61(9):948-959.

[7] Saveriano M , Lee D . Distance based Dynamical System Modulation for Reactive Avoidance of Moving Obstacles[C]. IEEE International Conference on Robotics and Automation. IEEE, 2014.

[8] Flacco F , Kroger T , Luca A D , et al. Depth space approach to human-robot collision avoidance[J]. Proceedings-IEEE International Conference on Robotics and Automation, 2012:338-345.

[9] E. Zhang, J. Yu and M. Li. Research on autonomous obstacle avoidance algorithm of teleoperation manipulator in unstructured unknown environment[C]. IEEE International Conference on Robotics and Biomimetics (ROBIO), 2017.

[10] 祁若龙,周维佳,王铁军. 一种基于遗传算法的空间机械臂避障轨迹规划方法[J]. 机器人,2014,36(3):263-270.

[11] Stein S C, Schoeler M, Papon J, et al. Object Partitioning Using Local Convexity[C]// Computer Vision and Pattern Recognition. IEEE, 2014:304-311.

[12] Žlajpah, Leon, Nemec B . Kinematic control algorithms for on-line obstacle avoidance for redundant manipulators[C]. IEEE/RSJ International Conference on Intelligent Robots & Systems. IEEE, 2002.

[13] Kaldestad K B , Haddadin S , Belder R , et al. Collision avoidance with potential fields based on parallel processing of 3D-point cloud data on the GPU[C]. IEEE International Conference on Robotics & Automation. IEEE, 2014.