A Dynamic Real-Time Motion Planning Method for Multi-robots with Collision Avoidance

Yonghong Zhang, Huan Zhao(✉), Congcong Ye, and Han Ding

State Key Laboratory of Digital Manufacturing Equipment and Technology Huazhong, University of Science and Technology, Wuhan 430074, Hubei, People's Republic of China
huanzhao@hust.edu.cn

Abstract. Collision avoidance is the major concern for the multi-robots operation. However, few literatures can generate a collision free path as well as a smooth motion profile at the same time. To solve this problem, this paper presents an integrated motion planning scheme for two manipulators working in a shared workspace. In this method, first, a collision free path is calculated in the path planning phase. Then, the smooth trajectory is generated by using a dynamic nonlinear filter. Both the path and the trajectory are calculated directly, thus the computation load is low and the approach can be applied in a real-time manner. Simulation results indicate that the proposed method is effective to realize collision avoidance for industrial robots working in a common workspace.

Keywords: Motion planning · Collision avoidance · Real-time · Trajectory generation

1 Introduction

Factory automation has been widely increasing through the use of industrial robots. Many industrial tasks are performed efficiently with the use of two manipulators systems, which increases productivity and reduces the amount of space needed in the factory. These systems execute mainly two types of operations: one type is that all manipulators cooperate for a given task, such as carrying an object; and another is that each manipulator accomplishes its own task independently in a common workspace, such as picking and placing. The basic requirement of fulfilling their tasks is that a robot must be able to move from its start point to goal and avoid possible collisions. A widely used technique is zone blocking method [1]: when one robot enters into the common workspace, a flag is activated to prevent other robots from entering the workspace. The method is not efficient because all robots can not work in the shared space at the same time.

Being a key challenge of robotics and automation engineering, path planning to avoid potential collisions has been studied extensively in the last two decades. Probabilistic Road Map [2, 3] and Rapid Random Trees [4, 5], as well as all their variants,

Y. Huang et al. (Eds.): ICIRA 2017, Part II, LNAI 10463, pp. 59–69, 2017.
DOI: 10.1007/978-3-319-65292-4_6

are the most popular. Because such algorithms avoid the hard problem of building high-dimensional configuration space explicitly by sampling the configuration space.

The common characteristic of collision avoidance approaches in [2–5] is that obstacles are static. Two important factors limit the direct application of these collision avoidance approaches on robot coordination. The first factor is that a change in the environment can invalidate the previously planned path. The second factor is that they cannot respond to dynamic environment immediately. Hence, collision avoidance for multi robots working in the shared space is more complex and need specially treated.

Afaghani et al. [6–8] proposed a method to avoid collisions using time scheduling of the execution time of commands. If a collision were to occur, one robot could move while the other robots should stop and wait. The goal, however, cannot reach if the path is obstructed by other robots. To address this problem, Montaño A et al. [9, 10] proposed a coordination method in discretized coordination space [11]. The coordination is achieved through adjustment of movements of each robot along its original planned path. However, each robot must have information about the next movements of the other robots. Cheng [12] proposed a method to achieve coordination by adjusting the geometric paths of robots. Wang et al. [13] presents a collision avoidance method in configuration space. To reduce the computation time, only the first three joints of the robot are used for collision avoidance. Li et al. [14] presents a method tailored to path planning problems in changing environments where many moving obstacles crowed together. A roadmap is built using uniform random sampling method, and then A* algorithm is arranged to search for a feasible path. The aim of research in [12–14] is to find a collision-free path, which is a geometric representation of a plan to move from a start to a target. Motion planning has to produce an executable trajectory for a robot, and not merely a geometrical path. Chiddarwar and Babu [15] present a conflict free coordinated path planning for multiple robots in configuration-time space. The coordination strategy is moving to a safety position and waits there until conflict has been resolved. The smoothness of motion is not considered. Vannoy et al. [16] introduces an adaptive motion planning approach based on evolutionary computation. Although the smoothness of motion can be guaranteed, the trajectory is generated though iterative search process, which increases the calculation load.

This paper presents a dynamic motion planning for two industrial robots working in a shared workspace. In this method, a collision free path is obtained in path planning stage, and then the smooth trajectory is generated by employing a dynamic nonlinear filter. Unlike motion planning in discretized configuration space described in Refs. [13–15], the motion commands are generated by the dynamic nonlinear filter, which produces a trajectory with continuous velocity. Compared with Ref. [16],the path and trajectory are calculated directly instead of iteration, which decreases the computation load significantly.

The rest of this paper is organized as follows: Sect. 2 proposes the collision avoidance algorithm; Sect. 3 provides simulation results followed by the corresponding analysis; the paper is concluded in Sect. 4.

2 Dynamic Motion Planning for Robots with Collision Avoidance

Figure 1 is Architecture of the proposed method. The approach comprises of two major distinct phases, path planning and online trajectory generation. In path planning phase, SSV technique is used to detect the collision. When collision will occur, a new path is calculated immediately. In trajectory generation phase, a dynamic nonlinear filter is used to generate the smooth trajectory for each robot.

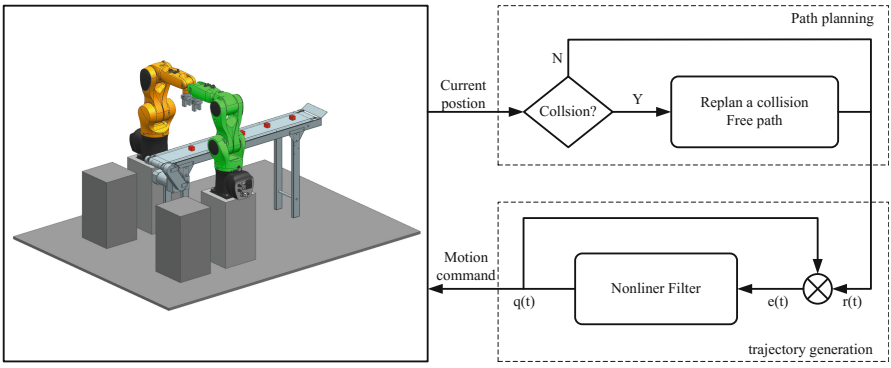

Fig. 1. Architecture of the proposed collision avoidance approach.

2.1 Path Planning

Modelling the manipulator plays an important role in collision avoidance. The model should be precise while being based on a simple mathematical representation for preventing high calculation cost. In this paper, SSV [17] method is used to determine detect collision. It is a safe and effective technique and easier to find the shortest distance, because cylinders and spheres are used to approximate the actual robot component geometry.

In SSV technique, each link is represented as a finite line $\vec{S}_n$, which is spanned through the two adjacent points $\mathbf{P}_n$ and $\mathbf{P}_{n+1}$, as shown in Fig. 2(a). Besides, a radius r_n is introduced which is bigger than the maximum distance between the line and the surface of the related robot component. As described above, $\vec{S}_n$ can be described as

$$\vec{S}_n(\mu) = \mathbf{P}_n + \mu(\mathbf{P}_{n+1} - \mathbf{P}_n) \tag{1}$$

where μ is a variable between 0 and 1.

Once the robot is modeled as shown in Fig. 2(b), the collision detection process between the robots will be realized in Cartesian coordinates. The first two components of robot are not considered because link1 is fixed and the movement range of link2 is small. As current joint angles are obtained from robot controller, $\vec{S}_n$ can be calculated

using forward kinematics. As shown in Fig. 3, $\mathbf{P}_i = (X, Y, Z)$ is a point on the link $\vec{S}_n(\mu_n)$, $\mathbf{P}_j = (U, V, W)$ is a point on the link $\vec{S}_m(\mu_m)$

$$\begin{aligned} &X = x_n + \mu_n(x_{n+1} - x_n),\ Y = y_n + \mu_n(y_{n+1} - y_n),\ Z = z_n + \mu_n(z_{n+1} - z_n) \\ &U = x_m + \mu_m(x_{m+1} - x_m),\ V = y_m + \mu_m(y_{m+1} - y_m),\ W = z_m + \mu_m(z_{m+1} - z_m) \end{aligned} \tag{2}$$

The distance between $\mathbf{P}_i$ and $\mathbf{P}_j$ can be calculated as follows

$$d = ||\mathbf{P}_i\mathbf{P}_j|| = \sqrt{f(\mu_n, \mu_m)} = \sqrt{(X-U)^2 + (Y-V)^2 + (Z-W)^2} \tag{3}$$

The minimal distance $d_{\min}$ between two links can be obtained by solving

$$\frac{\partial f}{\partial \mu_n} = 0,\ \frac{\partial f}{\partial \mu_m} = 0 \tag{4}$$

It is necessary to replan path for two robots if only $d_{\min}$ is less than the safety distance

$$d_{\min} < r_n + r_m = d_{safe} \tag{5}$$

For multiple robots path planning, prioritization scheme is a much more efficient technique [18, 19]. In this paper, the priority for each robot is assigned as follows:

(1) When one robot is completing a more important task than another robot, it has a higher priority;
(2) It has a higher priority if one robot is closer to its goal than another robot.

Figure 4 is an illustration of priority assignment. The priority of robot A is higher than robot B. Robot A can move along its previous path if there is no collision after robot B modifies its path. Otherwise both robots should modify their path to avoid collision. The modification only has to be good enough for a short period before the next cycle time since it will be corrected constantly. Based on this, a collision-free position of end-effector in the next cycle time can be defined as fellows

$$\mathbf{P}_{t+\triangle t} = \mathbf{P}_t + l \cdot \boldsymbol{\tau} \tag{6}$$

where $\mathbf{P}_t$ is the current position of end-effector. $\boldsymbol{\tau}$, from $\mathbf{P}_i^{\min}$ to $\mathbf{P}_j^{\min}$, is the unit vector of line segment $\mathbf{P}_i^{\min}\mathbf{P}_j^{\min}$, as shown in Fig. 3. $\mathbf{P}_i^{\min}$ and $\mathbf{P}_j^{\min}$ are the points on their respective robot links with minimal distance to each other. l is the distance between $\mathbf{P}_{t+\triangle t}$ and $\mathbf{P}_t$,and it should larger than the difference between $d_{\min}$ and d_{safe}. Here, it is assumed that $l = 1.5\left|d_{safe} - d_{\min}\right|$. It is assumed that the pose of end-effector at $\mathbf{P}_{t+\triangle t}$ is the same as $\mathbf{P}_t$.

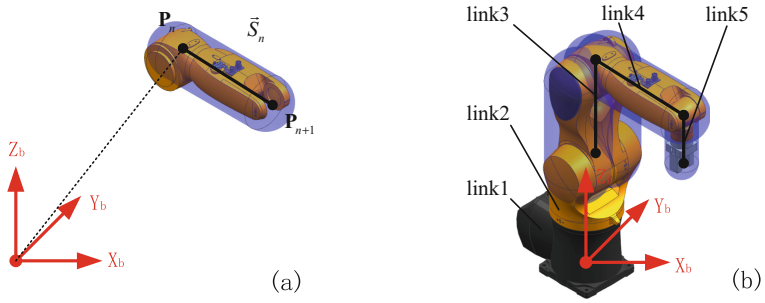

Fig. 2. Modelling a robot using SSV. (a) Description of a single robot link. (b) Description of a robot.

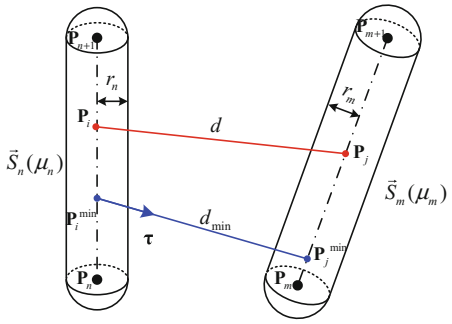

Fig. 3. Distance between two links.

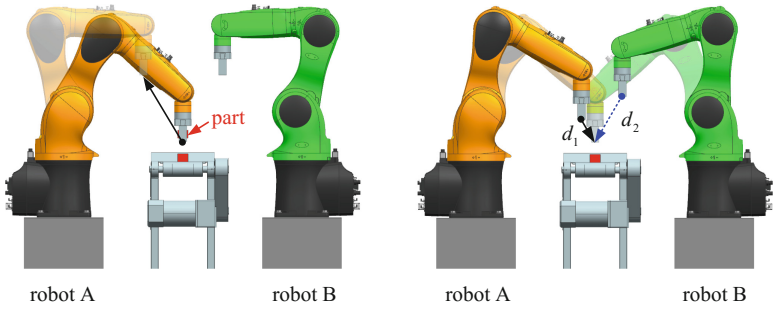

Fig. 4. An illustration of priority assignment. (a) Robot A holds a part. (b) Robot A is closer to its goal point.

As a result, the new path is moving the robot from $\mathbf{P}_t$ to goal $\mathbf{P}_{goal}$ via $\mathbf{P}_{t+\triangle t}$, as shown in Fig. 5.

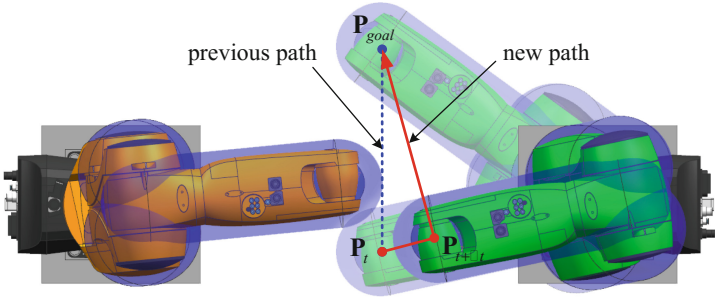

Fig. 5. Illustration of path modification when collision occurs.

2.2 Online Trajectory Generation

As a collision free path is obtained in path planning phrase, the task of motion planning is generating a smooth trajectory online while respecting drive limits. Online trajectory generation methods must enable the path (re-)calculation during the robot motion in order to avoid collision. This means that a robot moves along a path that has not necessarily been computed completely, and which may change during the movement. Instead of defining a function satisfying the desired conditions as [16], a dynamic nonlinear filter is employed to generate a smooth motion profile [20]. This trajectory generator is able to transform in real-time reference $r(t)$, received as input, in a smooth output $q(t)$ which satisfies the following constraints:

$$\begin{aligned} |\dot{q}_k| &\leq v_{\max} \\ |\ddot{q}_k| &\leq a_{\max} \end{aligned} \tag{7}$$

The scheme of the trajectory generator is shown in Fig. 6. $r(t)$ is the desired potion of a path. At each interpolating period $t_k = kT_s$, the variable structure controller C_2 receives $r(t)$ and and the current values of position q_k, velocity $\dot{q}_k$, and computes the value of the control action u_k. This control variable u_k corresponds to the desired acceleration, which must be integrated two times to obtain the position profile. The integration is performed by means of a rectangular approximation for the velocity

$$\dot{q}_k = \dot{q}_{k-1} + T_s u_k \tag{8}$$

while the trapezoidal approximation is used for the position

$$q_k = q_{k-1} + \frac{T_s}{2}(\dot{q}_k + \dot{q}_{k-1}) \tag{9}$$

the control variable u_k is computed at each interpolating period $t_k = kT_s$ as

$$C_2: \begin{cases} z_k = \frac{1}{T_s}\left(\frac{e_k}{T_s} + \frac{\dot{e}_k}{2}\right), \quad \dot{z}_k = \frac{\dot{e}_k}{T_s} \\ m = floor\left(\frac{1+\sqrt{1+8|z_k|}}{2}\right) \\ \sigma_k = \dot{z}_k + \frac{z_k}{m} + \frac{m-1}{2}\text{sign}(z_k) \\ u_k = -a_{\max}\frac{1+sign(\dot{q}_k sign(\sigma_k)+v_{\max}-T_s a_{\max}}{2} sat(\sigma_k) \end{cases} \tag{10}$$

where $e_k = (q_k - r_k)/a_{\max}$, $\dot{e}_k = \dot{q}_k/a_{\max}$ are the normalized tracking errors for position and velocity, z_k and $\dot{z}_k$ are the system state variables, $floor(\cdot)$ is the function 'integer part', $sign(\cdot)$ is the sign function, and $sat(\cdot)$ is a saturation function defined by

$$sat(x) = \begin{cases} -1, & x < -1 \\ x, & -1 \le x \le 1 \\ +1, & x > 1 \end{cases} \tag{11}$$

More details about the controller C_2 can be found in [20]. When the reference $r(t)$ is a step displacement, the output motion is perfectly equivalent to the trapezoidal trajectory.

One obvious advantage of the proposed collision avoidance algorithm is the smoothness of trajectory. Unlike motion planning in discretized configuration space as [13–15], motion commands are generated by the dynamic nonlinear filter, which produce a trajectory with continuous velocity. Another advantage of the proposed collision avoidance algorithm is its low calculation. The collisionfree path and the trajectory with smooth velocity profile are calculated directly rather than through an iterative search [16].

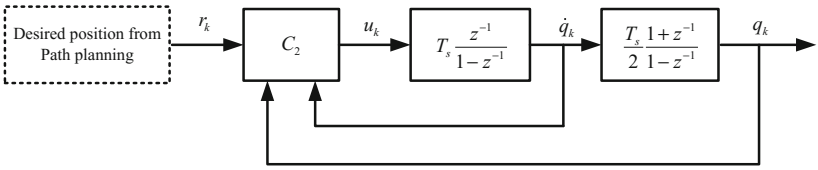

Fig. 6. Second order filter for online trajectory generator.

3 Simulation Validation

The proposed algorithm is evaluated using two KUKA robots (KR6R700), as shown in Fig. 7. In this environment, two robots are fixed on a board, with a distance between the central axes of the robots' bases 900 mm. The global coordinate system is located in the base of robot A. Each robot is controlled independently. The parameters used in the simulation are list in Table 1. In detail, robot A, holding a workpiece,

moves from **P** to $\mathbf{P}_A$, while robot B moves from $\mathbf{P}_B$ to **P**. Apparently, it is highly possible that the collision of two manipulators occurs during the work process among the workspace.

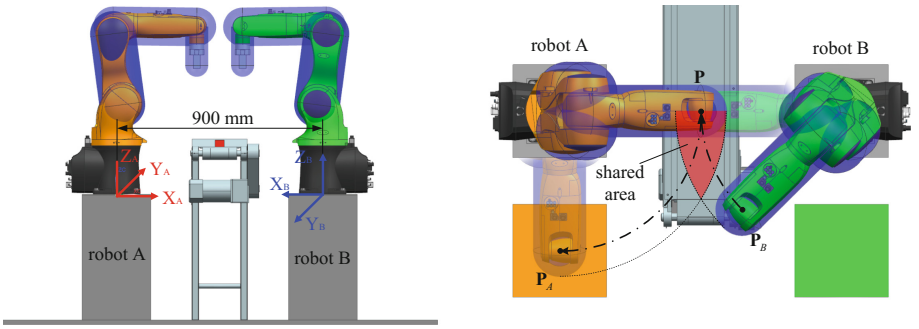

Fig. 7. Simulation system with two robots. (a) The snapshot of this simulation. (b) Initial and target position of robot A and B.

Table 1. Collision avoidance parameters used in the simulation

	Robot A	Robot B
Initial position	**P** (450, 0, 500)	$\mathbf{P}_B$ (582, −318, 500)
Target position	$\mathbf{P}_A$ (0, −450, 500)	**P** (450, 0, 500)
Maximum joint velocity (rad/s)	1	1
Maximum joint acceleration (rad/s^2)	15	15

The trajectory of simulation is shown in Fig. 8. According to the proposed priority assignment described in Sect. 2, the higher priority is given to A because it holds a workpiece. The path of robot A may not be changed if there is no potential collision possibility when path modification is performed for the robot B. As a result, the trajectory of robot A is changed slightly, while the trajectory of robot B is changed evidently. The distance between links is shown in Fig. 9. The motion profiles are shown in Fig. 10. The acceleration and velocity changed immediately to avoid collision when the distance between link4 of robot A and link4 of robot B is less than safety distance. It takes a little time to change movement direction of robot B. As a result, the distance between link4 of robot A and link4 of robot B decreases in an acceptable range even if it less than the safety distance. Though the movement of joint 6 has no influence on collision avoidance, it is responsible for the pose of the end-effector, which is adjusted when avoiding collisions. Because the collision-free path is planned locally in

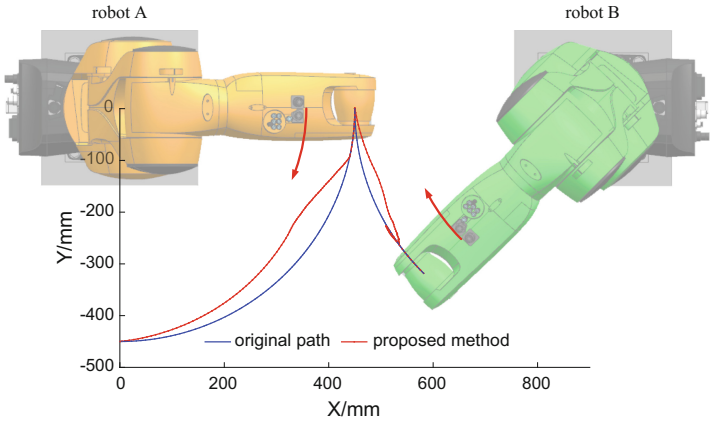

Fig. 8. Trajectory of simulation results.

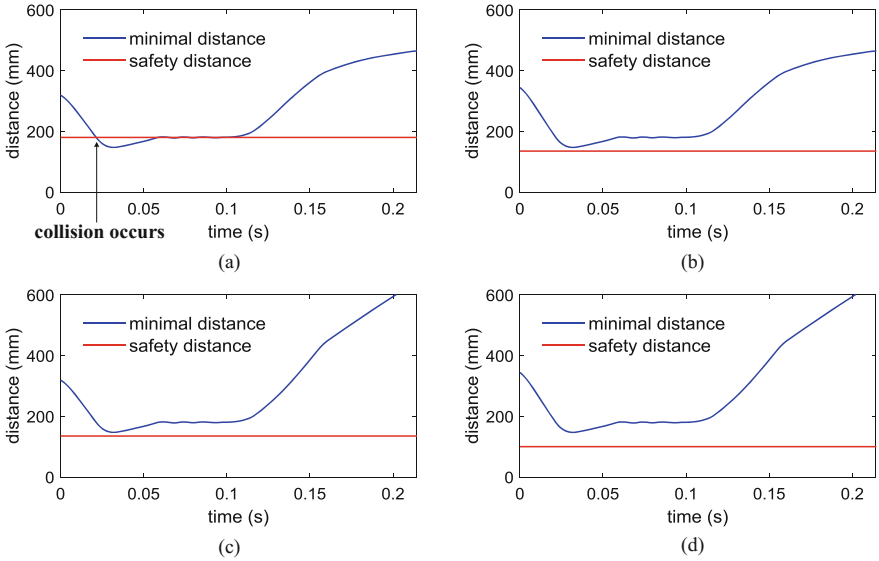

Fig. 9. Minimal distance. (a) Distance between link4 of robot A and link4 of robot B. (b) Distance between link4 of robot A and link5 of robot B. (c) Distance between link5 of robot A and link4 of robot B. (d) Distance between link5 of robot A and link5 of robot B.

time, there are still collisions in the nearly future. From Fig. 10, it is seen that the velocity profile is smooth and continuous. Besides, both the velocity and acceleration are under the constraint of maximum values.

Simulation results show that the presented method can realize collision avoidance effectively for industrial robots working in a common workspace.

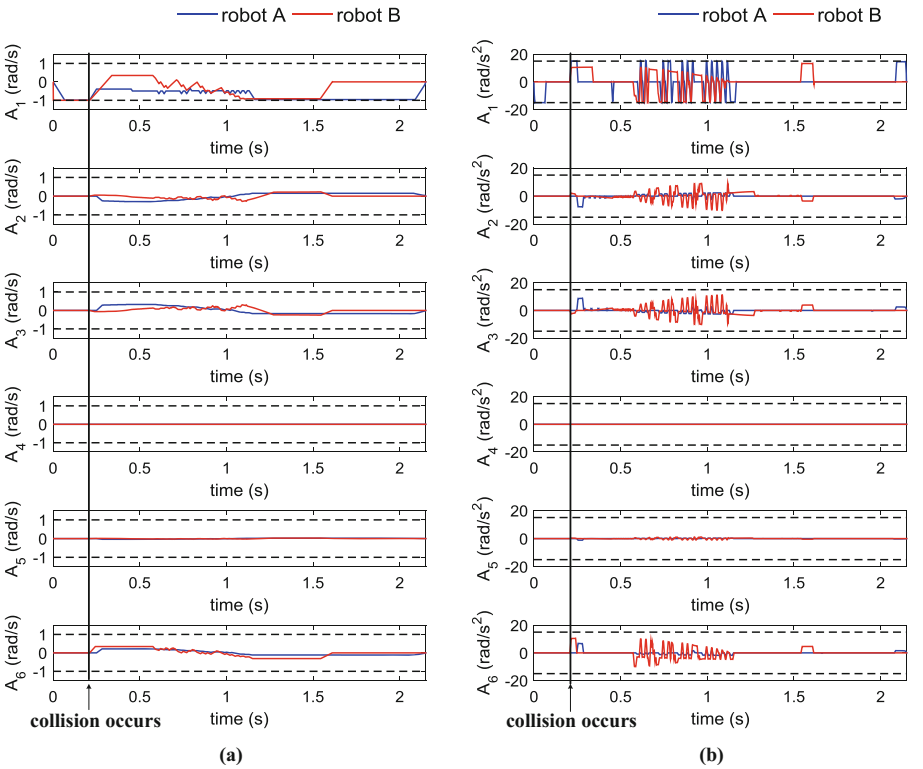

Fig. 10. Motion profiles. (a) Joint velocity. (b) Joint acceleration.

4 Conclusion

In this paper, a real-time motion planning method is proposed to achieve collision avoidance. With the proposed method, the path may not be changed for the robot with higher priority while path modification is performed for the robot with lower priority if there are some potential collision possibilities. Meanwhile, a second order dynamic nonlinear filter is used to generate smooth trajectory regarding the drive constraints. Simulation results indicate that the presented method can realize collision avoidance effectively for industrial robots working in a common workspace. Furthermore, the path as well as the trajectory is calculated directly, thus the computation load is low and the approach can be applied in real time.

Acknowledgement. This work was supported by the National Natural Science Foundation of China under Grant Nos. 51535004, 51323009, 51375196 and 51405175.

References

1. Zhou, J., Nagase, K., Kimura, S., et al.: Collision avoidance of two manipulators using rt-middleware. In: 2011 IEEE/SICE International Symposium on System Integration (SII), pp. 1031–1036. IEEE (2011)

2. Van Den Berg, J.P., Overmars, M.H.: Roadmap-based motion planning in dynamic environments. IEEE Trans. Robot. **21**(5), 885–897 (2005)
3. Al-Mutib, K., AlSulaiman, M., Emaduddin, M., et al.: D* lite based real-time multi-agent path planning in dynamic environments. In: 2011 Third International Conference on Computational Intelligence, Modelling and Simulation (CIMSiM), pp. 170–174. IEEE (2011)
4. Kuffner, J., LaValle, S.M.: RRT-Connect An efficient approach to single-query path planning. In: IEEE International Conference on Robotics and Automation, pp. 473–479 (2000)
5. LaValle, S.M., Kuffner, Jr., J.J.: Rapidly-exploring random trees: progress and prospects (2000)
6. Afaghani, A.Y., Aiyama, Y.: On-line collision avoidance between two robot manipulators using collision map and simple escaping method. In: 2013 IEEE/SICE International Symposium on System Integration (SII), pp. 105–110. IEEE (2013)
7. Afaghani, A.Y., Aiyama, Y.: On-line collision avoidance of two command-based industrial robotic arms using advanced collision map. Int. J. Robot. Mech. (JRM) **26**(3), 321–330 (2014)
8. Afaghani, A.Y., Aiyama, Y.: Advanced-collision-map-based on-line collision and deadlock avoidance between two robot manipulators with PTP commands. In: 2014 IEEE International Conference on Automation Science and Engineering (CASE), pp. 1244–1251. IEEE (2014)
9. Montaño, A., Suárez, R.: An online coordination algorithm for multirobot systems In: 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA), pp. 1–7. IEEE (2013)
10. Rodríguez, C., Montaño, A., Suárez, R.: Optimization of robot coordination using temporal synchronization. In: 2014 IEEE Emerging Technology and Factory Automation (ETFA), pp. 1–7. IEEE (2014)
11. Shin, Y., Bien, Z.: Collision-free trajectory planning for two robot arms. Robotica **7**(03), 205–212 (1989)
12. Cheng, X.: On-line collision-free path planning for service and assembly tasks by a two-arm robot. In: 1995 IEEE International Conference on Robotics and Automation, Proceedings, vol. 2, pp. 1523–1528. IEEE (1995)
13. Wang, S., Bao, J., Fu, Y.: Real-time motion planning for robot manipulators in unknown environments using infrared sensors. Robotica **25**(02), 201–211 (2007)
14. Li, Y., Liu, H., Ding, D.: Motion planning for robot manipulators among moving obstacles based on trajectory analysis and waiting strategy. In: SICE Annual Conference. IEEE, pp. 3020–3025 (2008)
15. Chiddarwar, S.S., Babu, N.R.: Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence. Robot. Auton. Syst. **59**(7), 508–518 (2011)
16. Vannoy, J., Xiao, J.: Real-time adaptive motion planning (RAMP) of mobile manipulators in dynamic environments with unforeseen changes. IEEE Trans. Robot. **24**(5), 1199–1212 (2008)
17. Ennen, P., Ewert, D., Schilberg, D., et al.: Efficient collision avoidance for industrial manipulators with overlapping workspaces. Procedia CIRP **20**, 62–66 (2014)
18. Freund, E., Hoyer, H.: Real-time pathfinding in multirobot systems including obstacle avoidance. Int. J. Robot. Res. **7**(1), 42–70 (1988)
19. Warren, C.W.: Multiple robot path coordination using artificial potential fields. In: 1990 IEEE International Conference on Robotics and Automation, Proceedings, pp. 500–505. IEEE (1990)
20. Biagiotti, L., Melchiorri, C.: Trajectory Planning for Automatic Machines and Robots. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85629-0