Optimal Motion Trajectory Generation and Real-Time Trajectory Modification for an Industrial Robot Working in a Rectangular Space

Naoki UCHIYAMA∗∗∗, Kazunori MORI∗∗∗, Kazuhiko TERASHIMA∗∗∗, Toru SAEKI∗∗∗, Toshio KAMIGAKI∗∗∗ and Hirotoshi KAWAMURA∗∗∗

** Department of Mechanical Engineering, Toyohashi University of Technology
1-1 Hibarigaoka, Tempaku, Toyohashi, Aichi, 441-8580 Japan
E-mail: uchiyama@tut.jp

*** Toyohashi Plant, Sinfonia Technology Co., Ltd.
150 Motoyashiki, Mitsuyacho, Toyohashi, Aichi, 441-3195 Japan

Abstract
This paper considers the time-optimal motion and obstacle avoidance trajectory generation for industrial robots. Although there are many previous works that had a similar objective, the generation of the trajectory for a general type of robot is still challenging because high computation time is required. This research focuses on a common type of robot that has a simple structure and moving environment. This limitation allows us to generate the trajectory in a much simpler and practical manner. In addition, oscillation suppression can be considered in the trajectory generation in the proposed approach. The real-time modification of the optimal trajectory is often required from a practical point of view. To avoid the recalculation of the trajectory that requires high computation time, we propose a real-time modification method of the trajectory. Simulation and experimental results demonstrate the effectiveness of the proposed methods.

Key words: Motion Control, Vibration Control, Controller, Manipulator, Mechatronics

1. Introduction
To generate an optimal trajectory is one of the fundamental problems for controlling robotic systems. Regarding industrial robots, there have been many studies conducted during the past three decades. In particular, to find ways for the generation of a time-optimal trajectory is crucial to reduce production takt time and improve production efficiency. The problem of the generation of a time-optimal trajectory along a given geometric path has been considered in[10]–[12]. The problem of the suppression of a jerk to obtain a smooth trajectory has been considered in[61]–[63]. Optimizing the energy used to control robots is also dealt with in[9].

Another important issue for the trajectory generation of robotic manipulators is obstacle avoidance. Two well-known approaches to this problem are the configuration-space-based method[10] and potential-function-based method[11]–[13], which have been the subject of many studies[14]–[18]. Other studies have considered both the optimal and collision-avoidance trajectory generation[19]–[22]. However, the generation of the optimal and collision-avoidance trajectory for general types of robotic manipulators is still challenging because high computation time is required. Therefore, some researches took a more practical approach and focused on specific types of robots, instead of trying to solve the problem with the general types of robots[23].

In a practical implementation, not only the previously mentioned problems must be solved but also oscillation suppression is required, because these problems damages the robot
manipulator mechanically and residual oscillation increases the takt time and reduces production efficiency.

This study presents a trajectory generation method for industrial robots that considers obstacle avoidance, oscillation suppression, and motion time. To the best of the authors’ knowledge, there are no methods that have considered above mentioned three criteria, probably, because of the required computation time. This study focuses on a robotic manipulator that moves only in a horizontal plane and within a rectangular space. This type of robot is often used in industries that require a dust-proof environment. This limitation allows us to develop a practical implementation of the trajectory generation algorithm that does not require too much computation time. Some researchers have focused on only the geometric trajectory generation for robotic manipulators, whereas their dynamics were not considered. However, because there are some limitations on the actuator power, the dynamics of the manipulator should be considered. We consider simple robot dynamics to achieve a practical implementation of a trajectory generation algorithm that suppresses oscillation and to accommodates the actuator limitations.

Another required function is the real-time modification of the optimal trajectory because the relative position of the transported object to the manipulator end-effector changes slightly and is different from the ideal case. The manipulator has to modify the given trajectory in real-time so that the transported object arrives at the exact desired position even when the relative position to the end-effector changes. The real-time regeneration of the optimal trajectory is usually difficult because high computation time is required. This paper also presents a real-time method to modify the trajectory generated off-line without increasing the motion acceleration that excites the oscillation of the robotic manipulator.

This paper is organized as follows. First, we formulate a problem of the practical trajectory generation for a robotic manipulator working in a rectangular space. Next, this problem is converted to a discrete-time problem, which allows us to use a conventional nonlinear programming technique. A real-time method to modify the optimal trajectory is also presented. Simulation and experimental verification results of the proposed methods are demonstrated, although only simulations were conducted in some existing studies. Because these results are obtained with a robotic manipulator and working environment actually used in industry, the effectiveness is clearly confirmed. Conclusions are given in the last section.

2. Generation of the optimal trajectory

2.1. Problem formulation

We consider a robotic manipulator that moves horizontally in a rectangular space as shown in Fig. 1 whose notations are explained later. This type of region is often used for a dust-proof environment. The manipulator has to meet the following five requirements:

(a1) The manipulator has to move from an initial position to a final position as fast as possible to reduce the takt time.

(a2) The oscillation of the manipulator has to be reduced to avoid mechanical damage and to transit to the next motion, such as grasping or releasing the object.

(a3) The manipulator has to avoid colliding with the walls of the rectangular space.

(a4) The motion trajectory has to accommodate the manipulator’s actuator power limitations.

(a5) Because the relative position of the object to the manipulator’s end-effector is often changed for some reason, the planned trajectory for the manipulator must be able to be modified slightly in the real-time without changing the oscillation property or letting the robot collide with the walls.

This paper presents a method of generating the trajectory for robotic manipulators that meets the above five requirements. We consider the following manipulator dynamics for their
Journal of System Design and Dynamics

Fig. 1 Robot and rectangular working environment

simplicity:

\[ J\ddot{\theta} = G\tau \]  

where \( J \) and \( G \) are the inertia coefficient and actuator gear ratio of the \( i \)th joint of the manipulator, respectively, \( \theta_i \) and \( \tau_i \) are the position and actuator torque (force for the linear actuator) of the \( i \)th joint, respectively. Because a very large gear ratio is generally used in industrial robotic manipulators, we employ the above simple dynamics by assuming that nonlinear force terms, such as centrifugal and Coriolis force terms, are negligible for the trajectory generation. This assumption is often used for the controller design of industrial robots\(^{(24)}\). For example, \( J_i \) can be represented for the revolute joint as follows\(^{(24)}\):

\[ J_i = G_i^2 I_{mi} + I_i \]  

where \( I_{mi} \) is the actuator inertia of the the \( i \)th joint, and \( I_i \) is the load inertia applied to the \( i \)th joint.

In general, the end-effector exhibits the largest amplitude of oscillation compared to other links, therefore, to consider the oscillation property of the manipulator, we include the following end-effector dynamics:

\[ J_n\ddot{\theta}_n + K(\theta_n - \theta_a) = G_n\tau_n \]  

\[ J_a\ddot{\theta}_a - K(\theta_n - \theta_a) = 0 \]  

where \( J_a \) is the equivalent inertia of the end-effector and \( K \) is the spring constant that represents the oscillation property of the end-effector. The value \( K \) can be determined experimentally as follows:

\[ K = J_a(2\pi f_a)^2 \]  

where \( f_a \) is the fundamental natural frequency of the oscillation of the end-effector. The value of \( f_a \) can generally be measured using the acceleration sensor. Because the manipulator considered in this paper is assumed to work in a closed area such as dust-proof environment in industry, the natural frequency of oscillation can be measured in advance and generally does not change. Hence, the robustness to the variation of natural frequency \( f_a \) is not considered in this paper.

Representative two-dimensional positions \( p_j = [x_j, y_j]^T \), \( j = 1, \ldots, m \) for avoiding collision with the walls can be defined as follows:

\[ x_j = f_{xj}(\theta_i), \]  

\[ y_j = f_{yj}(\theta_i) \]
For example, we consider the following coordinates of the three positions in Fig. 1,

\[ x_1 = L_1 \cos \theta_1 \]
\[ y_1 = L_1 \sin \theta_1 \]
\[ x_2 = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \]
\[ y_2 = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) \]
\[ x_3 = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3) \]
\[ y_3 = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3) \]

where \( L_i \) is the \( i \)th link length. Conditions for avoiding collision with the walls can be represented as follows:

\[ x_{\text{min}} \leq x_i \leq x_{\text{max}} \]
\[ y_{\text{min}} \leq y_i \leq y_{\text{max}} \]

where \( x_{\text{min}} \) and \( x_{\text{max}} \) are the allowable minimum and maximum values in the \( X \)-direction, and \( y_{\text{min}} \) and \( y_{\text{max}} \) are those in the \( Y \)-direction, as shown in Fig. 1. No collision between the manipulator and walls occurs as long as Eqs. (14) and (15) are satisfied, as shown in Fig. 1.

The initial and final positions and orientations of the end-effector are given as follows:

\[ x(0) = x_e, \quad y(0) = y_e, \quad \phi(0) = \phi_e, \quad x(t_e) = x_e, \quad y(t_e) = y_e, \quad \phi(t_e) = \phi_e \]
\[ \dot{x}(0) = \dot{x}_e, \quad \dot{y}(0) = \dot{y}_e, \quad \dot{\phi}(0) = \dot{\phi}_e, \quad \dot{x}(t_e) = \dot{x}_e, \quad \dot{y}(t_e) = \dot{y}_e, \quad \dot{\phi}(t_e) = \dot{\phi}_e \]

where \( x(t), \ y(t) \) and \( \phi(t) \) are positions and orientations of the end-effector at time \( t \). Symbols \( x_e, \ y_e, \ \phi_e, \ x_e, \ y_e, \ \phi_e \) are the end-effector initial (X-Y) position and orientation, final position and orientation, initial velocity, and final velocity, respectively, and \( t_e \) is the time at the final position. These values are converted to the manipulator’s joint positions as follows:

\[ \theta(0) = \theta_1, \quad \theta(t_e) = \theta_e \]
\[ \dot{\theta}(0) = \dot{\theta}_1, \quad \dot{\theta}(t_e) = \dot{\theta}_e \]

where \( \theta_1, \ \dot{\theta}_1, \ \theta_e \) and \( \dot{\theta}_e \) are the initial joint position vector, final position vector, initial velocity vector and final velocity vector, respectively. For example, the following inverse kinematics is used for this conversion for the case shown in Fig. 1.

\[ \theta_1 = \frac{B}{A} \pm \cos \left( \frac{L_1^2 + B^2 + L_2^2 - L_3^2}{2L_1 \sqrt{(A^2 + B^2)}} \right) \]
\[ \theta_2 = \frac{B - L_1 \sin(\theta_1)}{A - L_1 \cos(\theta_1)} - \theta_1 \]
\[ \theta_3 = \phi_d - \theta_1 - \theta_2 \]
\[ A = x_d - L_3 \cos(\phi_d) \]
\[ B = y_d - L_3 \sin(\phi_d) \]

where \( x_d, \ y_d \) and \( \phi_d \) are the given end-effector X-Y position and orientation, respectively.

The actuator torque/force limitation can be assigned as follows:

\[ \tau_{li} \leq \tau_i \leq \tau_{ui} \]

where \( \tau_{li} \) and \( \tau_{ui} \) are the lower and upper limits of the \( i \)th joint actuator torque/force, respectively. In general, these values are difference between the rated torque/force and friction ones.

We consider the following objective function for achieving the requirements in (a1) and (a2):

\[ \min_{\theta \in \Theta_{ae}} \left\{ t_e + \alpha \sum (\theta_i - \theta_{ae})^2 \right\} \]
where the first term reduces the motion time and the second one reduces oscillation. Because the oscillation (spring effect) is generated from the relative angle in Eqs. (3) and (4), we design Eq. (26) to include the relative angle, not velocity or acceleration. However, they obviously have strict correlation, and the acceleration is evaluated in the experiment as shown in Section 3. In the next subsection, we transform the problem formulated above to effectively obtain the numerical solution.

2.2. Discrete-time representation

To obtain the numerical solution for the problem stated in subsection 2.1, we transform it into a discrete-time form. Because the initial candidate trajectory is required for the numerical calculation, we employ a linear trajectory that just connects the given initial and final joint positions \( \theta_i \) and \( \theta_f \), respectively. This trajectory is divided into \( N \) positions with a constant sampling period \( T \), and the \( k \)th position of the \( i \)th joint is represented as \( \theta_i[k] = \theta_i(kT) \). The period \( T \) is the value for the numerical generation of the optimal trajectory and is different from the actual sampling period for control and measurement. The value of \( N \) is determined from the available computational resource for solving the problem numerically. The obtained trajectory is interpolated appropriately and divided into several points with a more precise sampling period for control. Therefore, we apply the central difference approximation to Eqs. (1) and (3) rather than applying the exact discretization, as follows:

\[
\frac{\theta_i[k+1] - 2\theta_i[k] + \theta_i[k-1]}{T^2} = G_1 \tau_1[k].
\]

(27)

\[
\frac{\theta_i[k+1] - 2\theta_i[k] + \theta_i[k-1]}{T^2} = G_2 \tau_2[k].
\]

(28)

\[
\frac{\theta_i[k+1] - 2\theta_i[k] + \theta_i[k-1]}{T^2} + K(\theta_i[k] - \theta_i[k]) = G_3 \tau_3[k].
\]

(29)

Boundary conditions in Eqs. (18) and (19), respectively, are represented as follows:

\[
\theta[0] = \theta_i, \quad \frac{\theta[1] - \theta[0]}{T} = \dot{\theta}_i,
\]

(30)

\[
\frac{\theta[N] - \theta[0]}{T} = \dot{\theta}_f.
\]

(31)

Conditions for obstacle avoidance in Eqs. (14) and (15) are described as follows:

\[
x_{\min} \leq x_i[k] \leq x_{\max}
\]

(32)

\[
y_{\min} \leq y_i[k] \leq y_{\max}
\]

(33)

\[
k = 0, \ldots, N.
\]

The actuator torque/force limitation in Eq. (25) is represented as follows:

\[
\tau_{li} \leq \tau_i[k] \leq \tau_{ui}, \quad k = 0, \ldots, N,
\]

(34)

If the velocity limitation is required, the following backward difference approximation can be considered:

\[
\dot{\theta}_i \leq \frac{\theta_i[k+1] - \theta_i[k]}{T} \leq \dot{\theta}_{ui}, \quad k = 0, \ldots, N
\]

(35)

From the relationship, \( \dot{\theta}_e = NT \), we can consider the following objective function on behalf of Eq. (26):

\[
\min_{\theta, \tau} \left\{ T + \alpha \sum_{k=1}^{N} (\theta_i[k] - \theta_a[k])^2 \right\}
\]

(36)

Solving the trajectory generation problem to satisfy the requirements in (a1)-(a5) results in \( \theta[k], \tau[k] \) and \( T \) to minimize Eq. (36) under the constraints in Eqs. (27)-(35). A conventional nonlinear programming technique can be applied to solve this problem. We employed the SQP (Sequential Quadratic Programming) method using the BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm to update the Hessian like function (25).
2.3. Real-time trajectory modification

In practice, when the transported object is placed in the end-effector, the relative positional error occurs. In this case, the manipulator trajectory should be modified in the real-time so that the final position of the transported object is placed at the exact desired position. Because it is difficult to regenerate the optimal trajectory by solving the nonlinear programming problem in the real-time, we propose a method to slightly modify the optimal trajectory, such that the oscillation property is not changed much. This study assumes that the relative positional error of the transported object to the end-effector is known. The desired final value for joint positions $\theta_{new}(t_f)$ to compensate for the positional error can be obtained by using the inverse kinematics of the manipulator from the known error information.

In addition, we consider only small error magnitude because the change of relative position between the transported object and the end-effector is generally very small (a few percent of motion distance). Hence, this study does not consider the collision due to the real-time trajectory modification. The collision can be avoided by including some margin in Eqs. (14) and (15) in the original trajectory generation.

We define the following scaling parameter $\beta$ for modifying the trajectory:

$$\beta = \frac{\theta_{new}(t_f) - \theta(0)}{\theta(t_f) - \theta(0)} \quad (37)$$

Using the scaling parameter $\beta$, we propose to generate a new joint position vector at the $k$th sampling period, $\theta_{new}[k]$, as follows:

$$\theta_{new}[k] = \beta(\theta[k] - \theta(0)) + \theta(0) \quad (38)$$

In the above equation, the optimal trajectory is scaled such that the initial position $\theta(0)$ is unchanged. It is possible to fix the final position $\theta(t_f)$ instead of $\theta(0)$.

Using this scaling process, we can maintain the oscillation property of the optimal trajectory. The velocities of the trajectories before and after modification can be compared as follows with the backward difference approximation:

$$\dot{\theta}[k] = \frac{\theta[k] - \theta[k - 1]}{T} \quad (39)$$

$$\dot{\theta}_{new}[k] = \frac{\theta_{new}[k] - \theta_{new}[k - 1]}{T} = \frac{\beta(\theta[k] - \theta[k - 1])}{T} = \beta \dot{\theta}[k] \quad (40)$$

It can be confirmed that velocities are multiplied only by $\beta$ for all sampling instances after the modification.

For the acceleration, we can reach a similar conclusion with the backward difference approximation, as follows:

$$\ddot{\theta}[k] = \frac{\theta[k] - 2\theta[k - 1] + \theta[k - 2]}{T^2} \quad (41)$$

$$\ddot{\theta}_{new}[k] = \frac{\theta_{new}[k] - 2\theta_{new}[k - 1] + \theta_{new}[k - 2]}{T^2}$$

$$= \frac{\beta(\theta[k] - 2\theta[k - 1] + \theta[k - 2])}{T^2}$$

$$\approx \beta \ddot{\theta}[k] \quad (43)$$

Because the value of $\beta$ in Eq. (37) represents the small difference of the transported object position, $|\beta - 1|$ is generally very small. Therefore, the modified trajectory has the same velocity and acceleration property as those of the optimal trajectories, and oscillation magnitude might not change. Equations (40) and (43) may be employed to consider the allowable oscillation magnitude for the design of manipulator system.

3. Trajectory Generation and Experimental Results

3.1. Effectiveness of the oscillation suppression term

We verify the effectiveness of the oscillation suppression term $\alpha$ in Eq. (26) by experiment. The normalized link length of the manipulator in Fig. 1 with respect to a certain distance
is $L_1 = 45\%$, $L_2 = 45\%$ and $L_3 = 35\%$. The position and orientation of the end-effector are defined as Eqs. (12) and (13), and $\phi = \theta_1 + \theta_2 + \theta_3$, respectively. Hereafter, the values of positions, orientations, motion time and acceleration are normalized with respect to certain values. The normalized initial and final positions and orientations are given as $x_s = 35.0\%$, $y_s = 50.5\%$, $x_e = 9.0\%$, $y_e = -82.0\%$, $\phi_s = 0\%$ and $\phi_e = -90.0\%$. The value of the fundamental natural frequency $f_a$ of the end-effector was obtained experimentally using an acceleration sensor attached as shown in Fig. 2. To avoid the collision of the transported object with walls, we modify the conditions in Eqs. (32) and (33), as follows:

$$\begin{align*}
x_{\text{min}} + r &\leq x_n[k] \leq x_{\text{max}} - r \\
y_{\text{min}} + r &\leq y_n[k] \leq y_{\text{max}} - r
\end{align*}$$

(44)  

(45)

where $r$ is the radius of circular transported object, and $r = 15\%$. The number of discrete points $N$ is set to 30, and the obtained trajectory is interpolated with the sampling period 0.01 s. The obtained trajectory was applied to the manipulator as pulse commands for actuators.

Fig. 2 Acceleration sensor and coordinate frame

Fig. 3 Generated end-effector trajectories
We generate optimal trajectories under the conditions $\alpha = 0 \text{ s/rad}^2$ and $\alpha = 0.01 \text{ s/rad}^2$ and compared the results. Figures 3 (a) and (b) show the trajectory generation results. Motion time of the generated trajectories are $t_e = 1.15 \text{ s}$ for $\alpha = 0 \text{ s/rad}^2$ and $t_e = 1.16 \text{ s}$ for $\alpha = 0.01 \text{ s/rad}^2$. We have applied the two optimal trajectories and measured the oscillation of the end-effector using the acceleration sensor, although the relative angle is considered in Eq. (26) (The relative angle sensor is not available in the current experimental setup). Because there is strict correlation between the relative angle and acceleration, the oscillation property may be evaluated by using the acceleration sensor. Figures 4 (a) and (b) show the experimental results and clearly confirm the effectiveness of considering the oscillation property in the optimal trajectory generation. Although motion time did not change much, a large difference exists in oscillation profiles. Because the actuator torque limitation is considered in Eq. (25), the applied trajectory does not exceed it in the experiment.
3.2. Generation of optimal trajectories

We verify the applicability of the proposed method to the practical industrial problem. We consider the problem of the generation of the optimal trajectory for 22 initial and final positions as shown in Tables 1 and 2. The PA stands for a position aligner that detects the relative positional error between the transported object and the end-effector, and this error information can be used for the real-time trajectory modification in Section 2.3. The other trajectory generation conditions are the same as those in Fig. 3 (b). Optimal trajectories can be successfully generated, as shown in Fig. 5-12, in which the end-effector trajectories are plotted.

We compare the motion time required for each task, as shown in Fig. 13, and confirm that longer trajectories required longer motion time in most cases. In some cases, such as LP1 to PA and LP3 to PA, required motion time is almost the same although the motion distance is different. This case occurs to decrease the rotational acceleration for reducing the oscillation in Y-axis of the acceleration sensor in Fig. 2. The required computation time is 6.85 min. on an average using a 2 GHz CPU computer. The existing method generates a motion trajectory for a two link manipulator considering obstacle avoidance in 13 min. by a mini-computer (19). Hence, the proposed method provides practical computation time even when considering not only collision avoidance but also oscillation suppression.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>List of initial and final positions and orientations of trajectories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>X [%]</td>
</tr>
<tr>
<td>LP1</td>
<td>50</td>
</tr>
<tr>
<td>LP2</td>
<td>50</td>
</tr>
<tr>
<td>LP3</td>
<td>50</td>
</tr>
<tr>
<td>LP4</td>
<td>05</td>
</tr>
<tr>
<td>LL1</td>
<td>-8</td>
</tr>
<tr>
<td>LL2</td>
<td>-8</td>
</tr>
<tr>
<td>PA</td>
<td>9</td>
</tr>
</tbody>
</table>

| Table 2  | Verified trajectories patterns |
|---|---|---|
| No. | Initial | Final | No. | Initial | Final | No. | Initial | Final |
| 1 | LP1 | PA | 9 | LL1 | LP3 | 17 | LP3 | LP1 |
| 2 | LP2 | PA | 10 | LL2 | LP3 | 18 | LP3 | LP2 |
| 3 | LP3 | PA | 11 | LP1 | LP2 | 19 | LP3 | LP4 |
| 4 | LP4 | PA | 12 | LP1 | LP3 | 20 | LP4 | LP1 |
| 5 | PA | LL1 | 13 | LP1 | LP4 | 21 | LP4 | LP2 |
| 6 | PA | LL2 | 14 | LP2 | LP1 | 22 | LP4 | LP3 |
| 7 | LL1 | LP1 | 15 | LP2 | LP3 |  |
| 8 | LL1 | LP2 | 16 | LP2 | LP4 |  |

Fig. 5 Generated trajectories (No.1 - 3)
Fig. 6  Generated end-effector trajectories (No.4 - 6)

Fig. 7  Generated end-effector trajectories (No.7 - 9)

Fig. 8  Generated end-effector trajectories (No.10 - 12)
Fig. 9  Generated end-effector trajectories (No.13 - 15)

Fig. 10  Generated end-effector trajectories (No.16 - 18)

Fig. 11  Generated end-effector trajectories (No.19 - 21)
3.3. Real-time modification of optimal trajectory

We apply the real-time path modification method to the trajectory LP4toPA in Fig. 6. The final position is changed by 1.25% and 0.5% in X and Y directions, respectively. The original and modified trajectories are shown in Fig. 14. Although we confirm in Fig. 14 that the trajectories did not have to be changed much to avoid collisions with the walls, those trajectories around the final position was changed. Figures 15 (a) and (b) shows the experimental results of the oscillation profile before and after modification. We confirm that the oscillation profile required few changes after modification.

4. Conclusions

The generation of the optimal trajectory of the robotic manipulator while avoiding collision with the environment is still a challenging problem because a large computational effort is required. This study considers not only the required motion time and obstacle avoidance that were well studied in previous research, but also the oscillation suppression and real-time modification of the optimal trajectory for a manipulator working in a two-dimensional rectangular space, which is often used in dust-proof industrial working environment. The optimal trajectory generation problem can be formulated by a conventional nonlinear programming problem with simple constraints, and it can be solved numerically within a practical amount...
of computation time. In addition, the proposed trajectory modification method can change the optimal trajectory in the real time while maintaining the oscillation property. Simulation and experimental results with a practical industrial manipulator demonstrate the effectiveness of the proposed method to generate and modify the optimal trajectory.

References

Fig. 15  Oscillation profile in original and modified trajectories


(20) F. Valero, V. Mata, A. Besa, Trajectory planning in workspace with obstacle taking into account the dynamic robot behavior, Mechanism and Machine Theory, 41, 5, 2006, 525-536.


