Geon Lee
Korean Institute of Science and Technology (KIST),
Seoul, 136-791, Republic of Korea
e-mail: 022454@kist.re.kr

Jinhyun Kim
Department of Mechanical Engineering,
Seoul National University of Science and Technology,
Seoul, 139-743, Republic of Korea
e-mail: jinhyun@seoultech.ac.kr

Youngjin Choi[1]
Department of Electronic Systems Engineering,
Hanyang University,
Ansan, 426-791, Republic of Korea
e-mail: cyj@hanyang.ac.kr

Convolution-Based Trajectory Generation Methods Using Physical System Limits

This paper proposes two novel convolution-based trajectory generation methods using physical system limits such as maximum velocity, maximum acceleration, and maximum jerk. Convolution is a mathematical operation on two functions of an input function and a convoluted function, producing an output function that is typically viewed as a modified version of input function. Time duration parameters of the convoluted functions with a unit area are determined from the given physical system limits. The convolution-based trajectory generation methods to be proposed in this paper have three advantages; first, a continuously differentiable trajectory is simply obtained by applying successive convolution operations; second, a resultant trajectory is always generated satisfying the given physical system limits; third, the suggested methods have low computational burden thanks to recursive form of convolution operation. The suggested methods consider both zero and nonzero initial/terminal conditions. Finally, the effectiveness of the suggested methods is shown through numerical simulations. [DOI: 10.1115/1.4007551]

1 Introduction

Motion systems are mainly composed of a desired trajectory generator, a tracking controller to follow the trajectory, and a target system including actuators. Conventionally, the controller plays a role in compensating quickly and accurately for an error generated during tracking and the trajectory generator makes the desired trajectory that the target system wants to follow. To be a motion control system with high performance, the trajectory generator is as important as the controller because the trajectory to be followed should be generated within physical limits of actuator system. For this reason, a lot of researches as regards the trajectory generation have been done for a few decades. The most important element in the trajectory generation is to make the position (or configuration) trajectory of S-curve shape function that is differentiable at least till either acceleration-level or jerk-level. Especially, a jerk trajectory bounded in the physical jerk limit of the actuator specification will reduce damages to the control system from unforeseen vibrations or overshoots as well as improve the accuracy or speed when the system is under tracking [1]. Also, the smaller the jerk is limited, the smoother the trajectory is generated [2]. For a special purpose such as surgical robot that makes contact with patients, the jerk-level trajectory bounded with an arbitrary value must be considered rather than just the bounded acceleration-level trajectory; moreover the jerk-level is required to be kept as small as possible [3,4].

The control system has its own actuator such as electric motor, hydraulic motor, pneumatic actuator and so on. Also, the actuator has its own physical limits such as maximum velocity, maximum acceleration, and maximum jerk. To realize the control system with higher performance, the physical system limits should be considered when the desired trajectory is generated [2,5,6]. Also, Ref. [7] is considered not only kinematic parameters but also torque limitation. The trajectory generated over the physical system limits is not only impossible to be followed by the controller, but also gives damages to the system due to overload of the actuator. If we make use of the trajectory generator without considering the system limits, we may spend a lot of time to find experimentally a range of the available trajectory. Last but not least, the economical cost for realizing the trajectory generator is an important element for extending application ranges to industrial and household control systems. If the control system including the trajectory generator can be implemented in a cheap processor, it is able to reduce the economical cost in realizing the real-time control system [8,9].

Generally, the desired jerk trajectory has been generated using higher order polynomial method above third order [10,11]. The polynomial method is very useful because it can establish individual functions of velocity, acceleration, jerk, and so forth by changing its order. However, the conventional polynomial method has a disadvantage that it cannot satisfy the given physical system limits. As an alternative, several methods have been suggested to satisfy the physical system limits by using the polynomial method. The most intuitive method makes the desired trajectory divide into many segments. For example, the trajectory generation including arbitrary jerk limit requires sixteen segments using forth order polynomial functions [12]. To generate the trajectory satisfying the given physical system limits, we should spend much time to design the segmented desired trajectories. Also, many other methods to remedy this disadvantage have been proposed with the purpose of lower computational burden in Refs. [12–15]. Besides aforementioned methods, both ZSPOT (Zero States Polynomial-like Trajectory) and ASPOT (Arbitrary States Polynomial-like Trajectory) have been proposed with the aim of reducing the computational load in the discrete-time domain, respectively, in Refs. [8,16]. Also, the efficient trajectory generation method using LSPB (linear segment with parabolic blends) has been proposed in Refs. [2,7]. The LSPB method is able to generate the time-optimal trajectory like the bang-singular-bang method. Moreover, the time-optimal trajectory generation method with the purpose of low computational burden has been proposed in Refs. [9,10].

On the other hand, a convolution-based trajectory generation method has been suggested in Refs. [5,17], which does not use the polynomial any more. The convolution-based method is able to generate an S-curve trajectory within the allowable physical system limits by applying successive convolution operations [18,19]. Also a recursive form of convolution operation can reduce the computational loads drastically. However, the conventional convolution-based trajectory generation method has been developed under only zero initial and terminal conditions. In this paper, we are to extend it to more general case including nonzero initial and terminal conditions as well as to establish it on the theoretically substantial basis.

[1]Corresponding author.

Contributed by the Dynamic Systems Division of ASME for publication in the JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENT, AND CONTROL. Manuscript received November 29, 2010; final manuscript received June 22, 2012; published online October 29, 2012. Assoc. Editor: Warren E. Dixon.

This paper is organized as follows; Sec. 2 deals with several properties, conventional convolution-based trajectory generation for the case of zero initial and terminal conditions, and the recursive form of convolution sum to reduce the computational burden for real-time issue. Section 3 proposes two novel convolution-based trajectory generation methods for the case of nonzero initial and terminal conditions; the one (method I) makes use of the symmetric property of acceleration and deceleration for nonzero initial and terminal velocity conditions and the other (method II) reduces the elapsed time taken to move the same distance than the former. Section 4 shows a few simulations results to confirm the validity of the suggested methods. Finally, Sec. 5 draws the conclusions.

2 Properties of Convolution Operations

Prior to suggest novel convolution-based trajectory generation method, this section reviews several properties of convolution operations. The convolution operation has mainly been used in the linear time-invariant system to obtain an output function when both an impulse response of the system and an input function are given. Here we assume that the impulse response of the system, $h(t)$, is a rectangular function having a unit area. Then, the output can be obtained as a form of smoother function than the input by applying the input to the system, only if the input is a piecewise continuous function. Using this property, it is possible to generate the smooth desired trajectory by applying successive convolution operations.

Suppose that $x(t)$ is an arbitrary input function defined in time duration of $0 \le t \le t_x$, $h(t)$ is a convoluted rectangular function having the unit area defined in time duration of $0 \le t \le t_h$, namely, $h(t) = 1/t_h$, for $0 \le t \le t_h$, and $y(t)$ is an output function produced by the convolution operation on two functions $x(t)$ and $h(t)$. Here, we should note that two functions $x(t)$ and $h(t)$ are zeros outside the defined time durations. Also let us denote that x_m and y_m are the maximum values of $x(t)$ and $y(t)$, respectively, then the convolution operations have several properties as follows;

Property 1. The output $y(t)$ is defined in time duration of $0 \le t \le t_x + t_h$, which is the sum of both time durations of the input $x(t)$ and the convoluted function $h(t)$.

Property 2. The area of output $y(t)$ is always equal to that of input $x(t)$.

Property 3. The maximum absolute value of output y_m is always smaller than or equal to that of input x_m. Especially, if $x(t)$ maintains x_m constantly for the time duration t_h or more, then y_m is equal to x_m.

First, it is easy to prove the property 1 using the formal definition of convolution operation as follows:

$$\begin{aligned} y(t) &= \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau \\ &= \frac{1}{t_h}\int_0^{t_h} x(t-\tau)d\tau \\ &= \begin{cases} \dfrac{1}{t_h}\displaystyle\int_0^{t_h} x(t-\tau)d\tau, & 0 \le t \le t_x + t_h \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{1}$$

This property comes out from convolution operation itself. Extending this property to successive convolution applications, we can know that the result function is defined in the total sum of time durations of the input and the convoluted functions.

Second, suppose that $X(s)$, $H(s)$, and $Y(s)$ are the Laplace transforms of $x(t)$, $h(t)$, and $y(t)$, respectively, then the area of $y(t)$ is denoted by $Y(s)/s$ and the convolution operation on two functions $x(t)$ and $h(t)$ implies the multiplication of $X(s)$ and $H(s)$. Now, by using the final value theorem, we can easily prove the property 2

$$\begin{aligned} \lim_{s\to 0} s\left(\frac{Y(s)}{s}\right) &= \lim_{s\to 0} X(s)H(s) \\ &= \lim_{s\to 0}\left(X(s)\frac{1}{t_h s}(1-e^{-t_h s})\right) \\ &= \lim_{s\to 0} X(s) \end{aligned} \tag{2}$$

where we should note that the Laplace transform of $h(t) = 1/t_h$ for $0 \le t \le t_h$ is $H(s) = (1 - e^{-t_h s})/(t_h s)$ and l'Hôpital's rule was used in Eq. (2). Above equation implies that the area of input is always equal to that of output. This completes the proof of the property 2. Moreover, the invariant area principle is always true only if the convoluted functions have the unit area.

Third, let us assume that $y(t)$ has the maximum value y_m at any time t_m, then we can get the following relation from the Eq. (1):

$$\begin{aligned} y(t_m) = y_m &= \frac{1}{t_h}\int_0^{t_h} x(t_m - \tau)d\tau \\ &\le \frac{1}{t_h}\int_0^{t_h} x_m d\tau = x_m \end{aligned} \tag{3}$$

where we can know that above inequality is always true because we chose the maximum value x_m among all values of function $x(t)$. Namely, the maximum value of the output is smaller than or equal to that of the input. Also, if $x(t) = x_m$ for $0 < t_h \le t_x$, then $y_m = x_m$. These complete the proof of the property 3. In other words, the maximum value of the output cannot exceed that of the input only if the convoluted functions have the unit area. In the following Sec. 2.1, we will illustrate a convolution-based trajectory generation method under the condition of the zero states (zero initial and terminal velocities).

2.1 Convolution-Based Trajectory Generation: Zero States. For the sake of simplicity, we will consider a single-axis motion control system as a target system. Also, suppose that the system has the limits such as the maximum velocity, denoted by $v_{\max}$, the maximum acceleration, denoted by $v^{(1)}_{\max}$, and the maximum jerk, denoted by $v^{(2)}_{\max}$. Without loss of generality, the system limit for nth order differentiation of velocity function is denoted by $v^{(n)}_{\max}$. Also, assume that the system moves given distance, S, then we can make the input function $y_0(t)$ using the maximum velocity $v_{\max}$ as follows:

$$y_0(t) = \begin{cases} v_0, & 0 \le t \le t_0 \\ 0, & \text{otherwise} \end{cases} \tag{4a}$$

$$v_0 = \text{sgn}(S)v_{\max} \quad \text{and} \quad t_0 = \frac{|S|}{v_{\max}} \tag{4b}$$

where v_0 and t_0 imply the signed maximum velocity and time duration of the rectangular input function as shown in Fig. 1.

Now let us define the first convoluted function $h_1(t)$ as the rectangular function with the time duration $0 \le t \le t_1$, then we can get the trapezoidal function $y_1(t)$ produced by the convolution operation on two rectangular functions $y_0(t)$ and $h_1(t)$ as shown in Fig. 1. By the property 1, the output function $y_1(t)$ is defined in the time duration $0 \le t \le t_0 + t_1$. By the property 2, the distance to be moved S is not changed by the convolution. By the property 3, the maximum absolute value $|v_1|$ of $y_1(t)$ becomes smaller than or equal to the maximum velocity $v_{\max}$ of the given system. From the trapezoidal function $y_1(t)$ of the Fig. 1, we can know that each time duration for acceleration and deceleration is t_1, respectively, thus the maximum acceleration of $y_1(t)$ becomes $v_{\max}/t_1$. To make active use of the physical limit about maximum acceleration $v^{(1)}_{\max}$, let us determine the time duration of first convoluted function $h_1(t)$ to be $t_1 = v_{\max}/v^{(1)}_{\max}$. Moreover, if $y_0(t)$ maintains the signed maximum v_0 constantly for the time

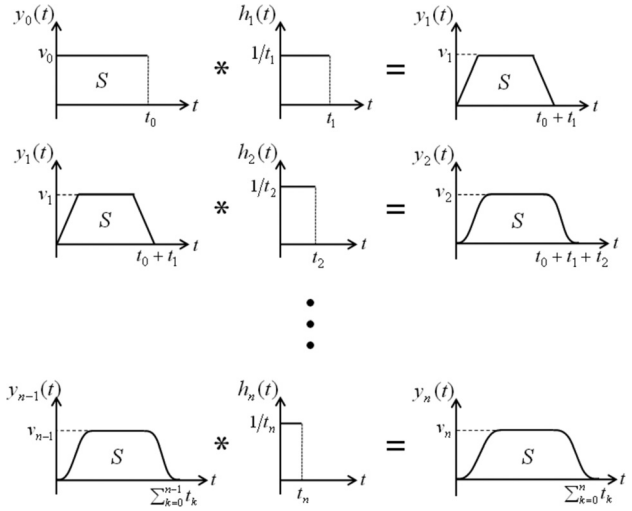

Fig. 1 Convolution-based trajectory generation method: zero states

interval t_1 or more, namely, if $t_0 \geq t_1$ in the Fig. 1, then $v_1 = v_0$ by the property 3.

Let us apply the convolution operation once more using the trapezoidal velocity function $y_1(t)$ as the input. Similarly we define the second convoluted function $h_2(t)$ as the rectangular function with time duration $0 \leq t \leq t_2$, then we can get the S-curve function $y_2(t)$ produced by the convolution of $y_1(t)$ and $h_2(t)$ as shown in Fig. 1. By the property 1, the output function $y_2(t)$ is defined in time duration $0 \leq t \leq t_0 + t_1 + t_2$. By the property 2, the distance S is also not changed through the convolution. By the property 3, the maximum absolute value $|v_2|$ of $y_2(t)$ becomes smaller than or equal to the maximum velocity $v_{\max}$ of the given system. For the similar reason for the process of obtaining t_1, to make active use of the physical limit about maximum jerk, we determine the time duration to be $t_2 = v_{\max}^{(1)}/v_{\max}^{(2)}$. Also, if $y_1(t)$ maintains the maximum v_1 constantly for the time duration t_2 or more, namely, if $t_0 - t_1 \geq t_2$ and $t_1 \geq t_2$ in $y_1(t)$ of the Fig. 1, then $v_2 = v_1$ by the property 3. Furthermore, if $t_0 \geq t_1 + t_2$ and $t_1 \geq t_2$ are satisfied, then we can say that the S-curve velocity function is an optimal trajectory to move the given distance by making active use of the given physical system limits such as the maximum jerk, the maximum acceleration and maximum velocity. Similarly, the results took more convolution operations stay within properties mentioned above and have certain rules especially for t_k. In other words, we can notice that relationship between t_k and the number of the convolution k has a certain rule, from the convolution process mentioned above explanation. Moreover, although solving any optimal problems with performance index, the convolution method with zero initial and zero terminal conditions result in the trajectory very similar to time-optimal trajectory to move the given distance by utilizing the given physical system limits such as the maximum velocity and maximum acceleration mentioned in Ref. [6].

In the convolution method, t_k stands for a minimum time duration with respect to the number of the convolution k, which is how long it takes to reach $v_{\max}^{(k-1)}$ from its initial value with $v_{\max}^{(k)}$. Take the case of a result performing three times convolution for example. Because it takes $v_{\max}/v_{\max}^{(1)}$ for velocity to reach its maximum velocity with maximum acceleration, the minimum time duration that the velocity increase from zero to maximum velocity, t_1 is $v_{\max}/v_{\max}^{(1)}$. Similarly, because it takes $v_{\max}^{(1)}/v_{\max}^{(2)}$ for acceleration to reach its maximum acceleration with maximum jerk, the minimum time duration that the acceleration increase from zero to maximum acceleration, t_2 is $v_{\max}^{(1)}/v_{\max}^{(2)}$. Without loss of generality, we can extend above procedures to the smoother S-curve velocity function within the allowable physical system limits such as $v_{\max}^{(n)}, \ldots, v_{\max}^{(2)}, v_{\max}^{(1)}$, and $v_{\max}$. Here, only design parameters to be considered are the time durations of the convoluted functions, which should be determined as follows:

$$t_k = \frac{v_{\max}^{(k-1)}}{v_{\max}^{(k)}} \quad \text{for} \quad k = 0, 1, 2, \cdots, n \tag{5}$$

where $v_{\max}^{(0)} = v_{\max}$ and $v_{\max}^{(-1)} = |S|$. Also, in order to generate the optimal trajectory by making active use of all the physical system limits such as $v_{\max}^{(n)}, \ldots, v_{\max}^{(2)}, v_{\max}^{(1)}$, and $v_{\max}$, the following inequality conditions as regards the time durations should be satisfied:

$$t_l \geq \sum_{k=l+1}^{n} t_k \qquad \text{for} \qquad l = 0, 1, 2, \ldots, n \tag{6}$$

In summary, for given the distance S and the physical system limits $v_{\max}^{(n)}, \ldots, v_{\max}^{(2)}, v_{\max}^{(1)}, v_{\max}$, if we determine the time durations by using Eq. (5), then the S-curve velocity function generated by the suggested convolution-based method is always within the allowable physical limits or at their boundary. Moreover, if the determined time durations satisfy the inequality conditions of Eq. (6), then the generated S-curve trajectory must be optimal in viewpoint that it makes active use of the physical limits, not within the allowable physical limits. As mentioned before, the convolution-based trajectory generation method is effective because it satisfies automatically the physical system limits, but the convolution operations require much computational burden. The following Sec. 2.2 suggests a recursive form of convolution operation to reduce it drastically.

2.2 Recursive Form of Convolution Operation. The convolution operation can be expressed by two forms such as convolution integral in the continuous time domain and convolution sum in the discrete-time domain. Actually, the implementation of convolution integral must be inappropriate for the digital motion control systems, especially for real-time issue. Thus, the convolution sum is considered with nth convoluted function having unit area, $h_n[k] = 1/m_n$ for $0 \leq k \leq m_n - 1$, as follows:

$$\begin{aligned} y_n[k] &= \sum_{l=-\infty}^{\infty} h_n[l] y_{n-1}[k-l] \\ &= \frac{1}{m_n} \sum_{l=0}^{m_n-1} y_{n-1}[k-l] \\ &= \frac{1}{m_n} \left(y_{n-1}[k] + y_{n-1}[k-1] + \cdots + y_{n-1}[k-m_n+1] \right) \end{aligned} \tag{7}$$

In addition, $y_n[k-1]$, preceding value of $y_n[k]$, is expressed as the following form:

$$y_n[k-1] = \frac{1}{m_n} \left(y_{n-1}[k-1] + y_{n-1}[k-2] + \cdots + y_{n-1}[k-m_n] \right) \tag{8}$$

Subtracting Eq. (8) from (7), we can obtain a recursive form of the convolution sum as follows:

$$y_n[k] = \frac{y_{n-1}[k] - y_{n-1}[k-m_n]}{m_n} + y_n[k-1] \tag{9}$$

As we can see in the Eq. (9), the recursive form of convolution sum is very effective because it requires just two additions and one division for the convoluted function having unit area. In Eqs. (7)–(9), k and m_n are positive integers satisfying $k = [t/T_s]$ and $m_n = [t_n/T_s]$, respectively, with sampling time T_s and Gauss floor function $[x]$ to denote the largest integer not greater than x.

An error can be caused by the Gauss floor function according to the size of sampling time in the convolution sum. As an alternative, Ref. [20] has proposed the compensation method for the error. In this paper, however, this error will be neglected by adding the assumption that sampling time is enough small. Till now we dealt with several properties of the convolution operation, convolution-based trajectory generation method under the condition of zero states (zero initial and terminal velocities), and the recursive form of convolution sum for real-time implementation issue. Section 3 will propose the more general convolution-based trajectory generation method including the condition of nonzero states (nonzero initial and terminal velocities).

3 Convolution-Based Trajectory Generation Method: Nonzero States

First, in order to generate desired trajectory with nonzero terminal velocity, let us consider the input of stepwise function that is composed of the maximum velocity v_0 for $0 \le t \le t_0$ and terminal velocity v_f for $t_0 \le t \le \alpha$, where α is a sufficiently large time parameter. Actually, the value of α is dependent on the number of convolution operations to be applied, which is expressed by $\alpha = \sum_{k=0}^{n} t_k$; e.g., if $n=2$, then $\alpha = t_0 + t_1 + t_2$ and if $n=3$, then $\alpha = t_0 + t_1 + t_2 + t_3$. If the stepwise input $y_0(t)$ is convoluted with the unit area function $h_1(t)$, then we have the output $y_1(t)$ as shown in Fig. 2. Once more, if we perform the convolution operation on two functions of $y_1(t)$ and $h_2(t)$, then the S-curve function is generated as shown in Fig. 2. Here, by the property 3, the maximum velocity of output function v_n is smaller than or equal to that of the stepwise input v_0. Also, the output $y_n(t)$ goes to zero passing through v_f at the time $t = \sum_{k=0}^{n} t_k$ as shown in Fig. 2. Hence, the trajectory satisfying the nonzero terminal condition can be generated by performing the successive convolution operations until $t = \sum_{k=0}^{n} t_k$. As we can see in the Fig. 2, the convolution time duration in the case of nonzero terminal condition is equal to that of zero states suggested in the previous Sec. 2.1. Namely, the time duration of the convolution-based trajectory generation method is independent of the value of terminal velocity.

In the case of nonzero terminal condition, the distance to be moved becomes different according to the number of convolutions. Since the acceleration and deceleration during time interval of $t_0 \le t \le \sum_{k=0}^{n} t_k$ always are formed symmetrically regardless of the number of successive convolution operations, the area S_n of final velocity function $y_n(t)$ with the nonzero terminal condition is obtained as follows:

$$
\begin{aligned}
S_n &= S_0 + v_f(\alpha - t_0) - \frac{v_f}{2}\sum_{k=1}^{n} t_k \\
&= v_0 t_0 + v_f\left(\sum_{k=0}^{n} t_k - t_0\right) - \frac{v_f}{2}\sum_{k=1}^{n} t_k \\
&= v_0 t_0 + \frac{v_f}{2}\sum_{k=1}^{n} t_k
\end{aligned}
\tag{10}
$$

where we can see that the property 2 holds if the terminal velocity condition is zero.

Second, if the trajectory should have both nonzero initial and terminal velocity conditions, then the trajectory can be decomposed into a rectangular initial velocity function and the nonzero final velocity function as shown in Fig. 3. The rectangular function has a value of initial velocity v_i for $0 \le t \le \sum_{k=0}^{n} t_k$ and the nonzero final velocity function has a difference between terminal and initial conditions $v_f - v_i$ at the terminal time as shown in Fig. 3. Also, its area S_n in the case of nonzero initial and terminal conditions can be obtained as the sum of areas of two functions, S_n^t and S_n^b, as following form:

$$
\begin{aligned}
S_n &= S_n^t + S_n^b \\
&= \left(v_0 t_0 + \frac{(v_f - v_i)}{2}\sum_{k=1}^{n} t_k\right) + v_i \sum_{k=0}^{n} t_k \\
&= (v_0 + v_i) t_0 + \frac{(v_f + v_i)}{2}\sum_{k=1}^{n} t_k
\end{aligned}
\tag{11}
$$

where we can see that the property 2 holds if both initial and terminal conditions are zero. In the case of nonzero initial and terminal conditions, the distance to be moved becomes different according to the number of convolutions. Since the area S_n of the final velocity function $y_n(t)$ should be equal to the given distance S of the system, we are to modify how to determine v_0 differently from Eq. (4*b*). Accompanied by the change of v_0, other two parameters t_0 and t_1 should be also modified. The following Secs. 3.1 and 3.2 suggest how these three parameters (v_0, t_0, t_1) should be changed in order to satisfy the given distance, namely, $S_n = S$.

3.1 Method I. For given v_i and v_f, one of four possible trajectories can generally be generated according to the distance to be moved S_n as shown in Fig. 4, upper two trajectories in the Fig. 4 are for the case of $v_i < v_f$ and below ones for the case the of $v_i > v_f$. Also, the maximum velocity of the generated trajectory can be either $v_n = v_{\max}$ or $v_n = -v_{\max}$ according to the given distance. Now, let us introduce a concept of criterion distance

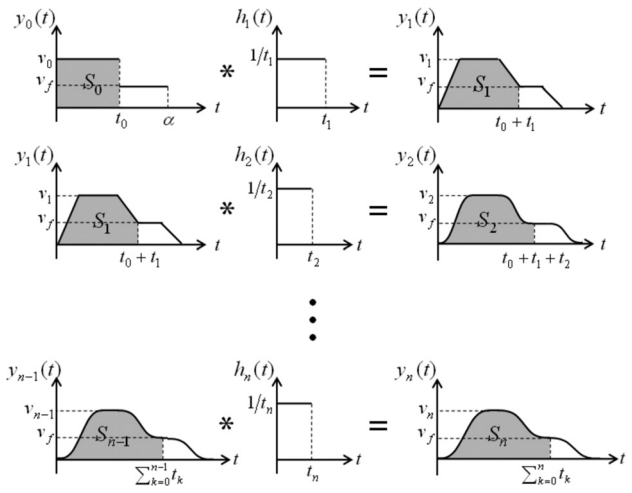

Fig. 2 Convolution-based trajectory generation method: nonzero terminal condition

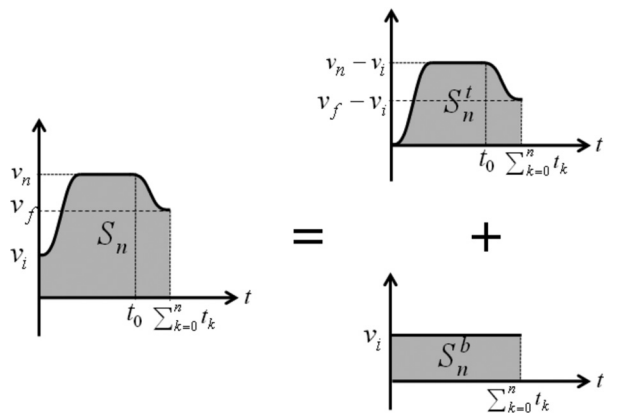

Fig. 3 Decomposition of the trajectory with nonzero states into a rectangular initial velocity function and the nonzero final velocity function

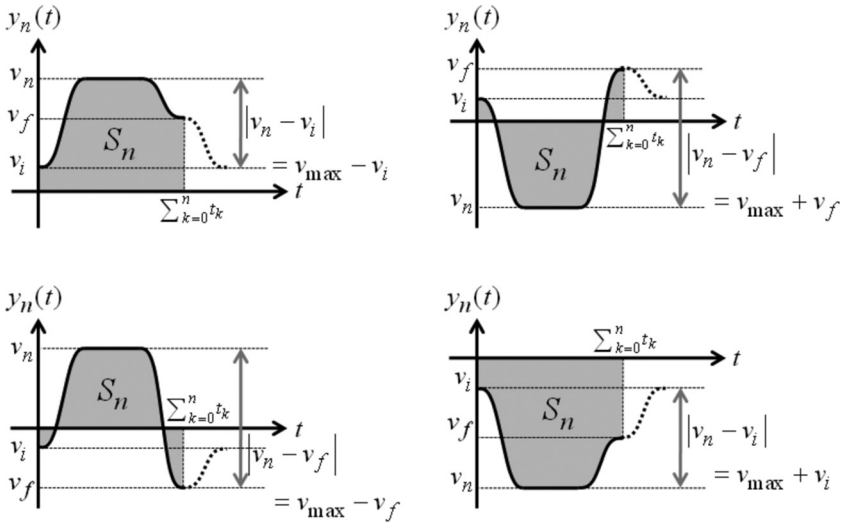

Fig. 4 Four possible trajectories according to the given distance, initial and terminal conditions

denoted by S_n^*, namely, two trajectories shown in left ones of the Fig. 4 are generated using $v_n = v_{\max}$ when $S_n > S_n^*$ and two trajectories shown in right ones of the Fig. 4 are generated using $v_n = -v_{\max}$ when $S_n < S_n^*$. In order to find out the criterion distance S_n^*, if we approach t_0 to zero in the Fig. 2, then we have Fig. 5. Figure 5 shows an inevitable distance for moving from zero to terminal velocity under the case of zero initial velocity. In the case of nonzero initial velocity, corresponding decomposition like the Fig. 3 should be considered. Moreover, since the S_n^* is dependent on the value of t_1, we take the value of S_n^* from the minimum t_1^* determined as a small value between $|v_n - v_i|/v_{\max}^{(1)}$ and $|v_n - v_f|/v_{\max}^{(1)}$. More detail, since v_n can be either $v_{\max}$ or $-v_{\max}$, we have possible eight cases of the minimum t_1^* according to the relationship between v_i and v_f as shown in Table 1. Fortunately, the possible eight cases can be expressed by one equation as the following form:

$$t_1^* = \frac{v_{\max} - \mathrm{sgn}(v_i v_f)\min(|v_i|, |v_f|)}{v_{\max}^{(1)}} \tag{12}$$

Once the value of t_1^* is determined, the criterion distance can be obtained from Eq. (11) by $t_0 \to 0$ as follows:

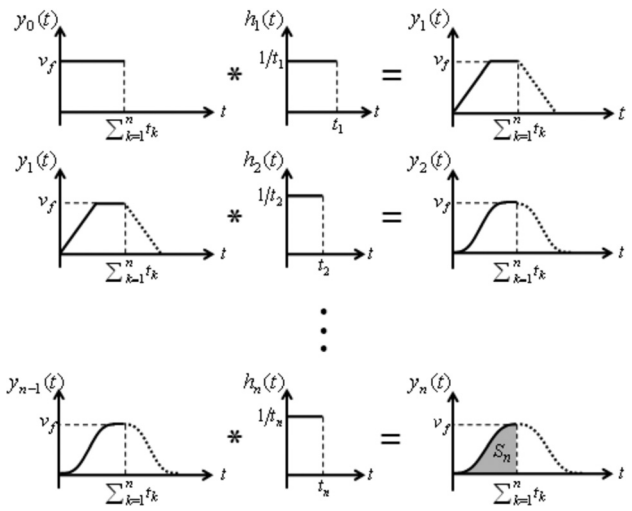

Fig. 5 Criterion distance S_n^* obtained as $t_0 \to 0$ in the Fig. 2, in the case of zero initial velocity

Table 1 The value of t_1^* according to relationship between v_i and v_f

	$v_i \geq 0$			$v_i < 0$		
	$\lvert v_i\rvert \geq \lvert v_f\rvert$	$\lvert v_i\rvert < \lvert v_f\rvert$		$\lvert v_i\rvert \geq \lvert v_f\rvert$	$\lvert v_i\rvert < \lvert v_f\rvert$
$v_f \geq 0$	$\frac{v_{\max} - v_f}{v_{\max}^{(1)}}$	$\frac{v_{\max} - v_i}{v_{\max}^{(1)}}$	$v_f \geq 0$	$\frac{v_{\max} + v_f}{v_{\max}^{(1)}}$	$\frac{v_{\max} - v_i}{v_{\max}^{(1)}}$
$v_f < 0$	$\frac{v_{\max} - v_f}{v_{\max}^{(1)}}$	$\frac{v_{\max} + v_i}{v_{\max}^{(1)}}$	$v_f < 0$	$\frac{v_{\max} + v_f}{v_{\max}^{(1)}}$	$\frac{v_{\max} + v_i}{v_{\max}^{(1)}}$

$$S_n^* = \frac{v_f + v_i}{2}\left(t_1^* + \sum_{k=2}^{n} t_k\right) \tag{13}$$

Since the generated final velocity function should be bounded as the maximum velocity of the given system, the value of v_0 can be either $v_{\max} - v_i$ when $S_n > S_n^*$ or $-v_{\max} - v_i$ when $S_n < S_n^*$. Also, the value of v_0 can take any value when $S_n = S_n^*$ because t_0 will be zero in that case. Thus, the value of v_0 can be expressed by either

$$v_0 = \mathrm{sgn}(S_n - S_n^*)v_{\max} - v_i \text{ or } v_0 = \frac{1}{\mathrm{sgn}(S_n - S_n^*)}v_{\max} - v_i \tag{14}$$

If $S_n > S_n^*$, then we can see that the value of t_1 should take the large value between $(v_{\max} - v_f)/v_{\max}^{(1)}$ and $(v_{\max} - v_i)/v_{\max}^{(1)}$ from left two trajectories in the Fig. 4, on the other hand, if $S_n < S_n^*$, then the value of t_1 should take the large value between $(v_{\max} + v_f)/v_{\max}^{(1)}$ and $(v_{\max} + v_i)/v_{\max}^{(1)}$ from right two trajectories in the Fig. 4. Thus, the value of t_1 is expressed as following form:

$$t_1 = \frac{\max\left(v_{\max} - \mathrm{sgn}(S_n - S_n^*)v_i,\quad v_{\max} - \mathrm{sgn}(S_n - S_n^*)v_f\right)}{\frac{1}{2}\left(1 + \mathrm{sgn}(S_n - S_n^*)^2\right)v_{\max}^{(1)}} \tag{15}$$

where we should note that we take $t_1 = 2v_{\max}/v_{\max}^{(1)}$ when $S_n = S_n^*$. Finally, the value of t_0 can be obtained by applying Eqs. (14) to (11) as follows:

$$t_0 = \frac{\mathrm{sgn}(S_n - S_n^*)}{v_{\max}}\left(S_n - \frac{v_f + v_i}{2}\sum_{k=1}^{n} t_k\right) \tag{16}$$

This section has suggested how these three parameters (t_0, t_1, v_0) should be changed for satisfying $S_n = S$ in the case of nonzero initial and terminal condition. As a result, the suggested method I can be summarized as follows:

For given physical system limits, initial/terminal velocities and distance such as $v_{\max}$, $v_{\max}^{(1)}, \cdots, v_{\max}^{(n)}$, v_i, v_f, and S_n

(1) Determine t_2, t_3,…, t_n using Eq. (5).
(2) Calculate the criterion distance, S_n^*, using Eqs. (12) and (13).
(3) Obtain v_0, t_1, and t_0 using Eqs. (14)–(16), respectively.
(4) Perform the convolutions of Eq. (9) as many as the number of n at each sampling time, and add v_i on the result of the preformed convolutions.
(5) Calculate the position profile and the acceleration profile through the integration and the differentiation, if necessary.

3.2 Method II. In the previous method I, an interval of velocity increment and that of velocity decrement are always the same because they are determined by $t_0, t_1, \ldots, t_n$. In other words, the method I is impossible to set up the acceleration and deceleration separately. Thus, the acceleration and deceleration in each interval would be also not the same when the initial velocity and the final velocity are not the same. However, in order for more efficient trajectory, the maximum acceleration should be utilized in both intervals. In the both intervals, Eq. (9) implies the Euler integration if the condition of Eq. (6) is satisfied. First term of Eq. (9) means increasing quantity of velocity per sampling time, and second term implies total velocity calculated until previous step. Indeed, since the maximum acceleration is determined by the convolution performed at first time, it is possible to set the acceleration separately by modifying t_1 with respect to the time. Let us denote the interval of velocity increment as t_1^+ and the interval of velocity decrement as t_1, then t_1^+ and t_1 in order to make same acceleration in both intervals are obtained as following form, respectively:

$$t_1^+ = \frac{v_{\max} - \mathrm{sgn}(S_n) v_i}{v_{\max}^{(1)}} \tag{17a}$$

$$t_1 = \frac{v_{\max} - \mathrm{sgn}(S_n) v_f}{v_{\max}^{(1)}} \tag{17b}$$

Moreover, in order to apply the same acceleration value to both intervals, the first-time convolution form of Eq. (9) is also changed as below

$$y_1(k) = \frac{y_0(k) - y_0(k - m_1^+)}{m_1^*(k)} + y_1(k-1) \tag{18}$$

where, m_1^+ and $m_1^*(k)$ denote $[t_1^+/T_s]$ and $[t_1^c(t)/T_s]$, respectively, and $t_1^c(t)$ is the function with respect to time constructed by t_1^+ and t_1 as follows:

$$t_1^c(t) = \begin{cases} t_1^+, & 0 \le t < t_0 \\ t_1, & t_0 \le t \end{cases} \tag{19}$$

Also, the condition of t_0 is expressed as follows:

$$t_0 \ge \max\left(t_1^+, t_1\right) + \sum_{k=2}^{n} t_k \tag{20}$$

In this method, the distance to be moved can be obtained by the area of the polygon as suggested in Fig. 6. Since the accelerations in intervals of velocity increment and decrement are generated as skew-symmetric form when the conditions of Eqs. (6) and (20) are satisfied, A and A′, and B and B′ in Fig. 6 have the same area, respectively. Thus, the moving distance, S_n, obtained by performing the convolutions until $\sum_{k=0}^{n} t_k$, is calculated from Fig. 6 as follows:

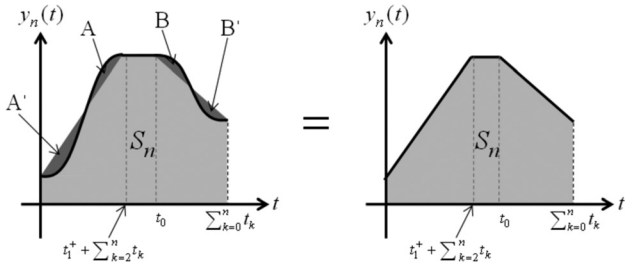

Fig. 6 The distance to be moved is equal to the area of polygon

$$S_n = (v_0 + v_i)t_0 + \frac{v_0}{2}\left(t_1 - t_1^+\right) + \frac{v_f + v_i}{2}\left(t_1 + \sum_{k=2}^{n} t_k\right) \tag{21}$$

where the sign of S_n and that of $v_0 - v_i$ are always the same, that is, S_n is positive when $v_0 = v_{\max} - v_i$ and negative when $v_0 = -v_{\max} - v_i$. Therefore, the value of v_0 can take $v_{\max} - v_i$ or $-v_{\max} - v_i$, and it is expressed by either

$$v_0 = \mathrm{sgn}(S_n) v_{\max} - v_i \quad \text{or} \quad v_0 = \frac{1}{\mathrm{sgn}(S_n)} v_{\max} - v_i \tag{22}$$

where above two equations has the same meaning. Substituting Eq. (22) into (21), we can obtain t_0 as follows:

$$t_0 = \frac{\mathrm{sgn}(S_n)}{v_{\max}}\left(S_n - \frac{v_f + v_i}{2}\sum_{k=1}^{n} t_k + \frac{v_i}{2}\left(t_1 - t_1^+\right)\right) - \frac{1}{2}\left(t_1 - t_1^+\right) \tag{23}$$

In summary, the suggested method II is able to be utilized only when Eqs. (6) and (20) are satisfied, and the trajectory making use of the maximum acceleration in both intervals of velocity increment and decrement can be obtained by using the modified convolution of Eq. (18). This section has suggested how these three parameters (t_0, t_1, v_0) should be changed for making active use of the maximum acceleration value in both intervals of acceleration and deceleration. As a result, the suggested method II can be summarized as follows:

For given system limits, initial/terminal velocities and distance such as $v_{\max}$, $v_{\max}^{(1)}, \cdots, v_{\max}^{(n)}$, v_i, v_f, and S_n

(1) Determine t_2, $t_3, \cdots, t_n$ using Eq. (5).
(2) Calculate t_1^c, v_0, and t_0 using Eqs. (19)–(23), respectively.
(3) Check whether both Eqs. (6) and (20) are satisfied, if those are satisfied, then go next step, otherwise, the method II fails.
(4) Perform the convolutions of Eq. (9) as many as the number of n at each sampling time, and add v_i on the result of the preformed convolutions, especially, if $n = 1$, perform the convolution of Eq. (18).
(5) Calculate the position profile and the acceleration profile through the integration and the differentiation, if necessary.

4 Simulation Results

The proposed trajectory generation methods have been implemented using the MATLAB Ver. 7.1. For the simulation, first, the algorithm is built as shown in Fig. 7. The Parameter Calculator calculates the corresponding coefficients $(v_0, t_0, t_1, \cdots, t_n)$ for generating the stepwise function after accepting inputs such as the moving distance, the system limits, the sampling time, and initial and terminal conditions. The Convolution Operator plays a role in performing the successive convolutions according to the number of n.

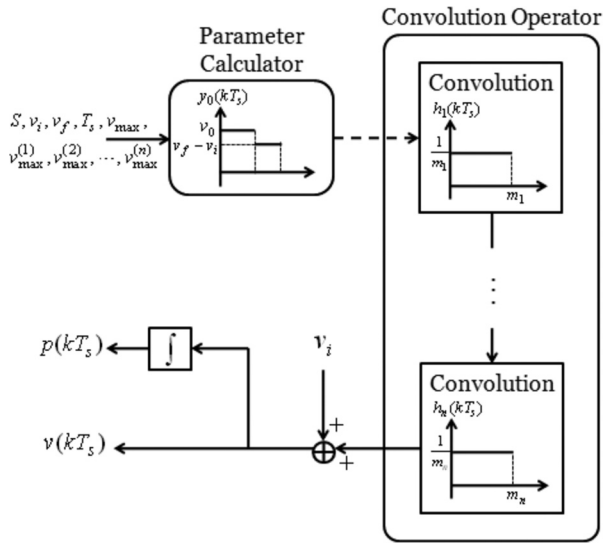

Fig. 7 Block diagram of the whole algorithm

Each convolution follows process as shown in Fig. 8, connected in serial as many as the number of n. The initial velocity is added to the output of the Convolution Operator, which completes the final velocity trajectory considering both initial and terminal conditions. If necessary, the trajectory of position can be obtained using the integration method such as the *Euler* and *Runge–Kutta*. In this paper, to obtain the position profile, Euler integration is used. The simulation performs twice convolutions in order to generate the trajectory whose jerk is bounded, and its sampling time is set to 1 ms.

4.1 Method I. For the simulation of the method I, the input parameters in order to generate the trajectory are given as shown in Table 2. Figure 9 shows the simulation results. Both results, (*a*) and (*b*) in the Fig. 9, show that the trajectories are generated within given system limits such as the maximum velocity, the maximum acceleration, and maximum jerk. Comparing (*a*) with (*b*), however, the velocity of (*b*) does not reach to the given maximum velocity, while that of (*a*) reaches to the given maximum velocity. That is because, in the case of (*b*), Eq. (6) is not satisfied. We can notice that t_0 takes 0.625 from Eq. (16) and that t_1 and t_2 take 1.75 and 0.25, respectively, so t_0 is smaller than the sum of t_1 and t_2. The method I can generate the trajectory under any input parameters, as this result tells us.

4.2 Method II. In the case of the method II, the input parameters are set as suggested in Table 3. Figure 10 shows the simulation results. If the initial velocity is equal to the terminal velocity, the method II brings same results with the method I only if the condition of Eq. (6) is satisfied, as shown in Fig. 10(*a*). On the other hand, in the case that the initial velocity is not equal to the terminal velocity, the trajectory is generated using the maximum jerk, as shown in Fig. 10(*b*). Comparing this result with Fig. 9(*b*), since the method II requires shorter time in generating the trajectories using the same input parameters; it is more efficient than the method I. However, we should notice that the method II can be applied only when Eqs. (6) and (20) are satisfied, otherwise, the method II fails.

Table 2 The input parameters for simulation of the method I; as such, system limits, distance to be moved, and initial/terminal conditions

	$v_{\max}$ (m/s)	$v_{\max}^{(1)}$ (m/s^2)	$v_{\max}^{(2)}$ (m/s^3)	v_i (m/s)	v_f (m/s)	S (m)
(a)	4	4	16	1	1	8
(b)	8	8	16	2	1	8

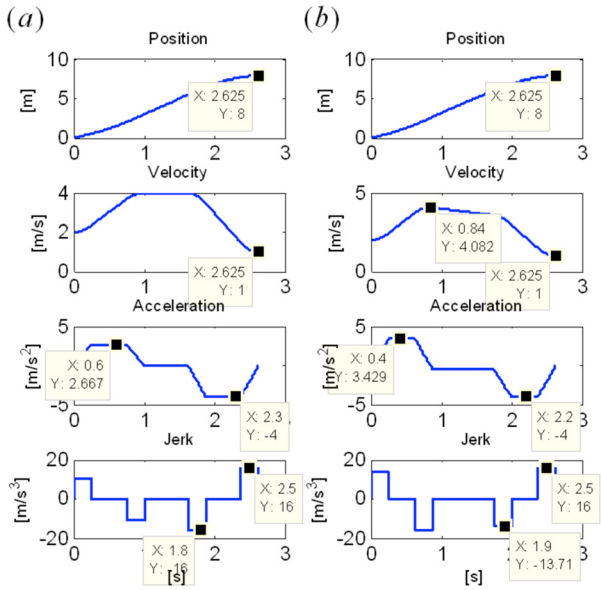

Fig. 9 Simulation results of the method I

Table 3 The input parameters for simulation of the method II

	$v_{\max}$ (m/s)	$v_{\max}^{(1)}$ (m/s^2)	$v_{\max}^{(2)}$ (m/s^3)	v_i (m/s)	v_f (m/s)	S (m)
(a)	4	4	16	1	1	8
(b)	8	8	16	2	1	8

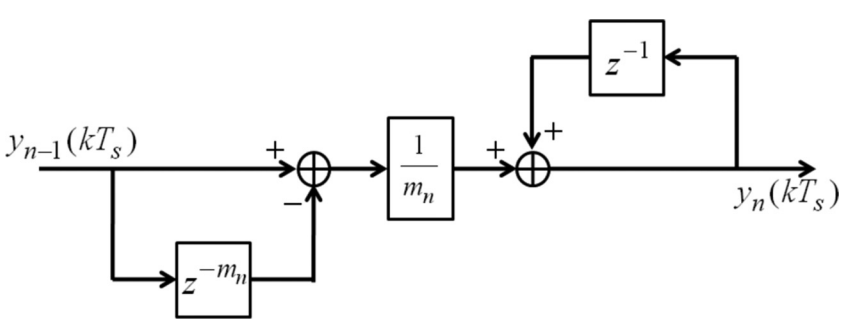

Fig. 8 Block diagram of the convolution

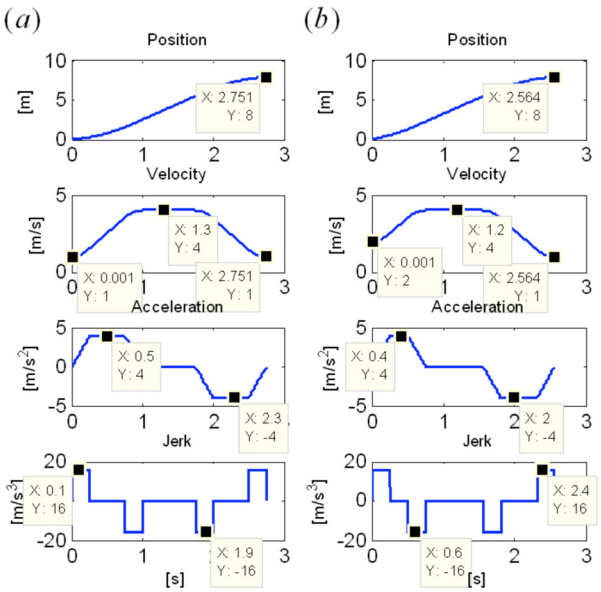

Fig. 10 Simulation results of the method II

5 Concluding Remark

The novel two trajectory generation methods making active use of physical system limits such as maximum velocity, maximum acceleration, and maximum jerk have presented in this paper. The proposed methods have utilized the recursive convolution sum, being required two additions and one division per one convolution, for practical use. Through the proposed convolution-based trajectory generation methods, we could get a continuously differentiable trajectory simply within the given physical system limits. The suggested methods were able to be applicable to both zero and nonzero initial/terminal conditions. Finally, the effectiveness of the suggested methods was shown by the numerical simulations.

Acknowledgment

This work was supported in part by the Mid-career Researcher Program through NRF grant funded by the MEST, and in part supported by the Ministry of Knowledge Economy (MKE) and Korea Institute for Advancement in Technology (KIAT) through the Workforce Development Program in Strategic Technology, and in part by the MKE under the Human Resources Development Program for Convergence Robot Specialists support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-H1502-12-1002), Republic of Korea.

References

[1] Craig, J. J., 1989, *Introduction to Robotics*, Addison-Wesley, New York.
[2] Constantinescu, D., and Croft, E. A., 2000, "Smooth and Time-Optimal Trajectory Planning for Industrial Manipulators Along Specified Paths," J. Rob. Syst., **17**, pp. 233–249.
[3] Mizoshita, Y., Hasegawa, S., and Takaishi, K., 1996, "Vibration Minimized Access Control for Disk Drives," IEEE Trans. Magn., **32**, pp. 1793–1798.
[4] Tsuji, T., Tanaka, Y., Morasso, P. G., Sanguincti, V., and Kaneko, M., 2002, "Bio-Mimetic Trajectory Generation of Robots via Artificial Potential Field With Time Base Generator," IEEE Trans. Syst., Man, Cybern Part C Appl. Rev., **32**(4), pp. 426–439.
[5] Jeon, J. W., and Ha, Y. Y., 2000, "A Generalized Approach for the Acceleration and Deceleration of Industrial Robots and CNC Machine Tools," IEEE Trans. Ind. Electron., **47**(2), pp. 133–139.
[6] Panahi, I. M. S., and Toliyat, H. A., 2006, "An Efficient Processor-Based Online Generation of Time-Optimal Trajectories," IEEE Trans. Control Syst. Technol., **14**(5), pp. 966–973.
[7] Bobrow, J. E., Dubowsky, S., and Gibson, J. S., 1985, "Time-Optimal Control of Robotic Manipulators Along Specified Paths," Int. J. Rob. Res., **4**, pp. 3–17.
[8] Ahn, K. T., Cho, J. S., and Chung, W. K., 2006, "Discrete Trajectory Formation in Comparison With the Analytical Method for Smooth Movements," Proceedings of IEEE International Conferernce on Industrial Electronics Control and Instrumentation, pp. 4462–4467.
[9] Macfarlane, S., and Croft, E. A., 2003, "Jerk-Bounded Manipulator Trajectory Planning: Design for Real-Time Application," IEEE Trans. Rob. Autom., **19**(1), pp. 42–52.
[10] Schlemmer, M., and Agrawal, S. K., 2002, "A Computational Approach for Time-Optimal Planning of High-Rise Elevators," IEEE Trans. Control Syst. Technol., **10**(1), pp. 105–111.
[11] Fu, K. S., Gonzalez, R. C., and Lee, C. S. G., 1987, *Robotics: Control, Sensing, Vision and Intelligence*, McGraw-Hill, New York.
[12] Nam, S.-H., and Yang, M.-Y., 2004, "A Study on a Generalized Parametric Interpolator With Real-Time Jerk-Limited Acceleration," Comput.-Aided Des., **36**(1), pp. 27–36.
[13] Nguyen, K. D., Chen, I.-M., and Ng, T.-C., 2007, "Planning Algorithms for S-Curve Trajectories," Proceedings of IEEE International Conference on Advanced Intelligent Mechatronics, pp. 1–6.
[14] Lambreshts, P., Boerlage, M., and Steinbuch, M., 2005, "Trajectory Planning and Feedforward Design for Electromechanical Motion Systems," Control Eng. Pract., **13**, pp. 145–157.
[15] Erkorkmaz, K., and Altintas, Y., 2001, "High Speed CNC System Design, Part I: Jerk Limited Trafectory Generation and Quintic Spline Interpolation," Int. J. Mach. Tools Manuf., **41**, pp. 1323–1345.
[16] Ahn, K. T., Chung, W. K., and Youm, Y., 2004, "Zero States Polynomial-Like Trahectory(ZSPOT) Generation," Proceedings of International Conference on the Advanced Mechatronics, pp. 120–125.
[17] Khalsa, D. S., 1990, "High Performance Motion Control Trajectory Commands Based on the Convolution Integral and Digital Filtering," Proceedings of International Conference on Intelligent Motion, pp. 54–61.
[18] Park, H. A., Ali, M. A., and Lee, C. S. G., 2010, "Convolution-Sum-Based Generation of Walking Patterns for Uneven Terrains," Proceedings of IEEE International Conference on Humanoid Robots, pp. 455–460.
[19] Kim, J.-H., 2007, "Walking Pattern Generation of a Biped Walking Robot Using Convolution Sum," Proceedings of IEEE International Conference on Humanoid Robotics, pp. 539–544.
[20] Su, K.-H., Hu, C.-K., and Cheng, M.-Y., 2006, "Design and Implementation of an FPGA-Based Motion Command Generation Chip," Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, pp. 5030–5035.