**Novel Trajectory Generation Method using Convolution Operation**

Geon Lee  
Hanyang University  
eamarin@hanyang.ac.kr

Youngjin Choi  
Hanyang University  
cyj@hanyang.ac.kr

**Abstract** — A novel trajectory generation method is proposed by making active use of performance specifications such as maximum speed, acceleration and jerk of a given system. The suggested method has an advantage of less computational burden thanks to a recursive form of convolution sum in discrete-time domain.

**Keywords** — trajectory generation, system limits.

1. Introduction

An efficient trajectory generation method is one of the important elements in achieving a good motion control performance. A number of methods using polynomial types have been proposed in such a way to satisfy given boundary conditions. On the other hand, Khalsa has proposed a method using convolution sum which could simply generate the trajectory satisfying a given maximum speed in [1]. Also, a discrete-time convolution sum is able to reduce a computational burden because it can be expressed in terms of recursive form in [2,3]. Having this advantage, a new trajectory generation method is proposed by making active use of maximum speed, maximum acceleration, and maximum jerk of a given system in this paper.

2. Convolution Method

A convolution operation between two discontinuous functions produces a continuous function as a result; furthermore, a continuously differentiable S-curve function can be obtained by applying successive convolution operations to the discontinuous function such as staircase form. For instance, if the convolution is performed with a given staircase function, \( y_0(t) \), and a rectangular function with unit area, \( h(t) \), as shown in Fig 1, we have a continuous trapezoidal function, \( y_1(t) \). Furthermore, if the convolution is one more performed with this trapezoidal function, \( y_1(t) \), and the rectangular function with unit area, \( h_2(t) \), we can obtain a continuously differentiable S-curve function, \( y_2(t) \), as illustrated in Fig 1.

Also, the maximum values \((v_0, v_1, \cdots, v_n)\) of resultant functions \((y_0, y_1, \cdots, y_n)\) by applying successive convolutions to the rectangular function with unit area \((h_0, h_1, \cdots, h_n)\) have the boundedness property of \(v_0 \geq v_1 \geq \cdots \geq v_n\). Here, we should note that the non-zero values of \(n\)th function, \(y_n(t)\), are defined in the extended domain of \(0 \leq t \leq t_0 + t_1 + \cdots + t_n\). As we can see in Fig. 1, \(n\)th function, \(y_n(t)\), becomes equal to a value of \(v_f\) at the instant of \(t = t_0 + t_1 + \cdots + t_n\). On the other hand, if we take \(n\)-order time derivative for \(n\)th function, \(y_n(t)\), then we have the discontinuous function composed of rectangular forms. Here, let us denote the maximum absolute value of \(n\)-order time derivative of \(y_n(t)\) as \(v_{\text{max}}^{(n)}\). Thus we can say that the \(n\)th function, \(y_n(t)\), which is obtained from \(n\) times successive convolution operations, has the finite maximum value as \(v_n \leq v_{n-1} \leq v_{n-2} \leq \cdots \leq v_0\) and the final value \(v_f\) at time \(t = \sum_{k=0}^{n} t_k\), and its \(n\)-order-time derivative is also bounded as \(v_{\text{max}}^{(n)}\).

![Fig. 1. Convolution of staircase function](image)

For practical use, the convolution integral in continuous-time domain is inappropriate when hardware implementation is required, because it includes a numerical integration. However, the convolution sum in discrete-time domain is preferable because a recursive formula can be obtained as suggested in [2,3]:

\[
\begin{align*}
\sum_{k=0}^{n} y_k(t) & = \sum_{k=0}^{n} h_k(t) y_0(t) \\
\sum_{k=0}^{n} h_k(t) y_0(t) & = \sum_{k=0}^{n} h_k(t) y_k(t) \\
\sum_{k=0}^{n} h_k(t) y_k(t) & = \sum_{k=0}^{n} h_{k+1}(t) y_k(t)
\end{align*}
\]
\[ y_n(k) = \frac{y_{n-2}(k) - y_{n-1}(k-m_n)}{m_n} + y_n(k-1) \]  

(1)

where \( k \) and \( m_n \) are the positive integers satisfying \( k = \lfloor t/T_s \rfloor \) and \( m_n = \lfloor nT_s/T \rfloor \), respectively, with sampling time \( T_s \) and Gauss floor function \( \lfloor x \rfloor \) to denote the largest integer not greater than \( x \). As we can see in Eq. (1), one convolution sum just requires two additions and one division. The less computational burden is one of the advantages obtained by using the discrete-time convolution sum.

Now, aforementioned properties are used to generate desired trajectories. Assume that one rectangular function is defined in the time-domain \( 0 \leq t \leq t_0 \) with its maximum velocity \( v_{\text{max}} \), and another rectangular function is defined in the time-domain \( 0 \leq t \leq t_1 \) with the height \( 1/t_1 \). If the convolution for these two rectangular functions is performed, then we have the trapezoidal velocity function defined in \( 0 \leq t \leq t_0 + t_1 \) in which the maximum acceleration is defined as \( a_{\text{max}} = \frac{v_{\text{max}}}{t_1} \) because the velocity increases or decreases by \( v_{\text{max}} \) for an interval \( 0 \leq t \leq t_1 \). Also, if we apply the convolution with another rectangular function defined in the time-domain \( 0 \leq t \leq t_2 \) with the height \( 1/t_2 \), then we can get the S-curve velocity function defined in \( 0 \leq t \leq t_0 + t_1 + t_2 \) in which the maximum acceleration is defined as \( a_{\text{max}} = \frac{v_{\text{max}}}{t_1} \) and the maximum jerk as \( j_{\text{max}} = \frac{a_{\text{max}}}{t_2} \).

Here, let us denote \( v^{(1)}_{\text{max}} = a_{\text{max}} \) and \( v^{(2)}_{\text{max}} = j_{\text{max}} \), then we can determine the time-domain parameters \( t_1, t_2, \cdots, t_n \) for convolutions as follows:

\[ t_n = \frac{v^{(n-1)}_{\text{max}}}{v^{(n)}_{\text{max}}} \quad \text{for} \quad n \geq 1 \]  

(2)

Especially, in the case of non-zero final velocity as shown in Fig.1, decreasing quantity of velocity for the interval \( 0 \leq t \leq t_1 \) is no longer \( v_{\text{max}} \), but either \( v_{\text{max}} - v_f \) or \( v_{\text{max}} + v_f \) with the final velocity \( v_f \). Thus first time-domain parameter \( t_1 \) should be remedied as follows:

\[ t_1 = \frac{\max(v_{\text{max}} - v_f, v_{\text{max}} + v_f)}{v^{(1)}_{\text{max}}} \]  

(3)

where the concrete form of \( v_0 \) will be suggested later. In addition, if we want to assign the non-zero final acceleration, similar remedy should be done when determining the second time-domain parameter \( t_2 \). Now, we will just consider the case of non-zero final velocity. As we can see in Fig. 1, the area until the time \( t = \sum_{k=0}^{n} t_k \), \( S_n \), is obtained as following form:

\[ S_n = v_0 t_0 + \frac{v_f}{2} \sum_{k=0}^{n} t_k \]  

(4)

where the area \( S_n \) implies the given target distance to be travelled and \( S_0 = S_1 = \cdots = S_n \), if \( v_f = 0 \). Actually, the minimum distance \( S_n^* \) which is determined by the given system specifications \( (v_{\text{max}}, v^{(1)}_{\text{max}}, v^{(2)}_{\text{max}}, \cdots, v^{(n)}_{\text{max}}) \) and non-zero final velocity \( v_f \) can be assumed as following form:

\[ S_n^* = \frac{v_f}{2} \left( \frac{v^{(1)}_{\text{max}}}{v_{\text{max}}} + \frac{v^{(2)}_{\text{max}}}{v_{\text{max}}} + \cdots + \frac{v^{(n-1)}_{\text{max}}}{v_{\text{max}}} \right) \]  

(5)

If \( S_n > S_n^* \) then \( v_0 = v_{\text{max}} \), if \( S_n > S_n^* \) then \( v_0 = -v_{\text{max}} \), and if \( S_n = S_n^* \), then \( v_0 = 0 \) in the Fig. 1. Here, \( v_0 \) can be expressed as a short form:

\[ v_0 = \text{sgn}(S_n - S_n^*) v_{\text{max}} \]  

(6)

Also, we can determine zeroth time-domain parameter \( t_0 \) using (4) and (6) as follow:

\[ t_0 = \frac{\text{sgn}(S_n - S_n^*)}{v_{\text{max}}} \left( S_n - \frac{v_f}{2} \sum_{k=1}^{n} t_k \right) \]  

(7)

where we should note that \( t_0 = 0 \), when \( S_n = S_n^* \).

![Fig. 2. Simulation result](image)

### 3. Simulation Result

Let us assume that the system specifications are given as the following data:

<table>
<thead>
<tr>
<th>( S_1 )</th>
<th>( v_{\text{max}} )</th>
<th>( a_{\text{max}} )</th>
<th>( j_{\text{max}} )</th>
<th>( v_f )</th>
<th>( T_s )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 4 ) [m]</td>
<td>( 4 ) [m/s]</td>
<td>( 8 ) [m/s²]</td>
<td>( 32 ) [m/s³]</td>
<td>( 1 ) [m/s]</td>
<td>( 1 ) [ms]</td>
</tr>
</tbody>
</table>

According to the algorithm suggested in the previous section, we determine the number of convolutions as \( n = 2 \) because the maximum jerk is given as \( v^{(2)}_{\text{max}} = j_{\text{max}} \). The remaining parameters are also determined according to the suggested algorithm. We can confirm through simulation result in Fig.2 that the trajectory is generated by making active use of the system specifications such as
maximum velocity, maximum acceleration and maximum jerk. The simulation was performed by using MATLAB.

4. Conclusion Remark

This paper has proposed a new trajectory generation method using a convolution sum and system specifications. The effectiveness of the suggested algorithm was shown through simulation.

References