2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 23-27, 2022, Kyoto, Japan

Jerk-continuous Online Trajectory Generation for Robot Manipulator with Arbitrary Initial State and Kinematic Constraints

Haoran Zhao*[1], Nihal Abdurahiman[2], Nikhil Navkar[2], Julien Leclerc[1], and Aaron T. Becker[1]

***Abstract*— This work presents an online trajectory generation algorithm using a sinusoidal jerk profile. The generator takes initial acceleration, velocity and position as input, and plans a multi-segment trajectory to a goal position under jerk, acceleration, and velocity limits. By analyzing the critical constraints and conditions, the corresponding closed-form solution for the time factors and trajectory profiles are derived. The proposed algorithm was first derived in Mathematica and then converted into a C++ implementation. Finally, the algorithm was utilized and demonstrated in ROS & Gazebo using a UR3 robot. Both the Mathematica and C++ implementations can be accessed at `https://github.com/Haoran-Zhao/Jerk-continuous-online-trajectory-generator-with-constraints.git`**

I. Introduction

Due to their outstanding agility and adaptability, industrial robot arms have proliferated to reduce cost, increase throughput, and ensure safety of personal and property. To achieve movement tasks, trajectory generation is a critical topic that has been developing since the first robot released. Moreover, trajectory generation is an important section in almost every robotics textbook [1]. Trajectories are often planned in the task space (Cartesian space) [2]–[4] because movements in the task space are more intuitive and trajectories can be easily observed or adjusted after planning. However, it is often challenging to avoid kinematic singularities and respect kinematic and dynamic constraints in task space planning. Another leading method is to plan a trajectory in the joint space [5]–[7], which can provide a non-singular solution and meet both kinematic and dynamic constraints. Both methods can ensure that the robot passes through desired waypoints in the task space. However, the latter method cannot guarantee a straight path because the inverse kinematic conversion causes a non-linear relationship between Cartesian space and joint space.

Most published work on interpolating methods for trajectory planning with waypoints use polynomial interpolation [8]–[11]. Using higher degree polynomials can improve accuracy and satisfy higher level constraints. For instance, a cubic polynomial can satisfy velocity constraints, a quintic can satisfy acceleration constraints, and a 7th order polynomial can satisfy jerk constraints [10]–[13]. However, higher degree polynomials can also cause oscillation known as Runge's phenomenon and lead to unstable convergence [12], [14].

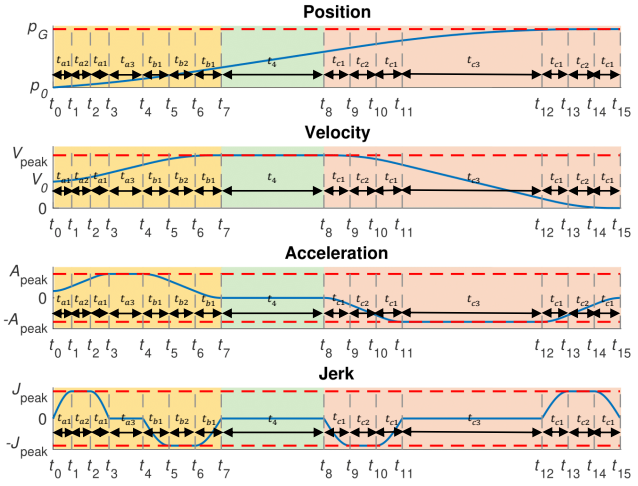

Fig. 1. Kinematic profile of a 15-phase jerk-continuous trajectory. See video overview at `https://youtu.be/aqUpL7RO9BY`.

To address these issues, piecewise polynomials are used in many practical situations. Splines are a class of piecewise functions that are defined by multi-order polynomials. They are simple to construct and they can accurately approximate complicated shapes [15], [16]. When compared to polynomial fits, splines yield similar results to lower degree polynomials and avoid Runge's phenomenon. Trajectory planning algorithm can be further categorized according to their computation cost as either online or offline algorithms. Polynomial based algorithms are often computed offline with given waypoints. Often optimization methods or iterative computation are implemented to find the control sequence [11], [12], [17]. However, offline computation can not react to sensor or interface inputs.

To achieve online trajectory planning, most presented work uses a trigonometric kinematic profile to provide a closed-form solution. This reduces computational burden [2], [13], [17]. In [13], an online methodology to generate smooth trajectories using sinusoidal jerk profiles was proposed. The trajectory planner takes an initial and final position as input, and plans the trajectory to respect maximum jerk, acceleration, and velocity constraints. Because the jerk profiles are designed using a sinusoidal pattern (so that they are differentiable away from 0), the magnitude of the jerk is bounded by the amplitude, and the integrated acceleration and velocity can also be bounded by carefully designing the

[1]H. Zhao, J. Leclerc, and A. T. Becker are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX.
[2]N. Abdurahiman and N. V. Navkar are with the Department of Surgery, Hamad Medical Corporation, Doha, Qatar.
*Corresponding author. Email: zhaohaorandl@gmail.com

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-7927-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IROS47612.2022.9981086

time factors of the trajectory segments corresponding to the inputs. This methodology solves the trajectory problem by splitting it into phases. The duration of each phase is called a *time factor*. The closed-form solution for the time factors and the trajectory profile were well explained in [13]. By analyzing the closed-form equation of the accumulated final displacement of the planned trajectory, the peak jerk varies correspondingly with a cube of $1/K$, where K is the ratio of the general trajectory minimum time and individual degree of freedom minimum time. Therefore, the ratio K is the key condition to achieve phase synchronization of all degree-of-freedom by changing the time factors and the magnitude of peak jerk of each degree of freedom. Moreover, the phase synchronization relies on a symmetric trajectory profile, which is only possible if the acceleration and velocity is initially zero. These constraints on intial conditions limit the efficiency of the motion conducted by the planned trajectory because the robot must stop at each waypoint and cannot incorporate new information from the sensors or interface while implementing the motion. To address these issues, the current acceleration and velocity should also be taken as inputs.

In this work, we present an algorithm that extends the methodology in [13] by enabling non-zero initial acceleration and velocity. The proposed algorithm can also handle cases when the initial velocity or acceleration are beyond the maximum constraints, or cause the system to overshoot the goal position. A brief review of the previous work is presented in Section II-A. The extended algorithms for generating a trajectory with non-zero initial acceleration and velocity are illustrated in Section II-B. Case studies are discussed in Section III, with ROS&Gazebo simulation results. Section IV concludes with a discussion on the current work and potential future work. The closed-form equations of the proposed algorithms are published in Mathematica and C++ [18].

II. Methodology

An online trajectory generation algorithm with sinusoidal jerk pattern under kinematic constraints is presented in this section. The main concepts from [13] are briefly summarized in Section II-A to make this paper understandable and self-contained. Next, the extension that enables non-zero initial acceleration and velocity is presented in Section II-B. All the mathematical computation and critical constraints will be discussed to find the closed-form solution of the time factors which are used to generate the trajectory segments.

A. Jerk-continuous trajectory generation with zero initial kinematic inputs

A three-phase sinusoidal jerk profile trajectory planning method was first proposed in [19], and it was adopted and implemented for industrial robot and manipulator tools [20]–[22]. The three-phase trajectory consists of the symmetric sequential connection of an acceleration phase, a constant velocity phase and a deceleration phase. Although the trajectory is designed to utilize peak jerk, the main weakness is that modern robots cannot maintain jerk and acceleration at the maximum values. Fang [13] proposed another sinusoidal jerk profile planner with a fifteen-phase sinusoidal jerk profile to address this weakness. The idea is to maintain the constant peak velocity phase as long as possible by keeping the acceleration and jerk quantities in their saturation state for the longest possible time. The jerk is continuous, and there are eight partial sine jerk phases, four constant jerk phases, two constant acceleration phases, and one constant velocity phase. This is similar to Fig. 1, but the duration of all phases are defined by T_1, T_2, T_3 and T_4. The sinusoidal jerk profile for each segment is defined as the following piecewise function:

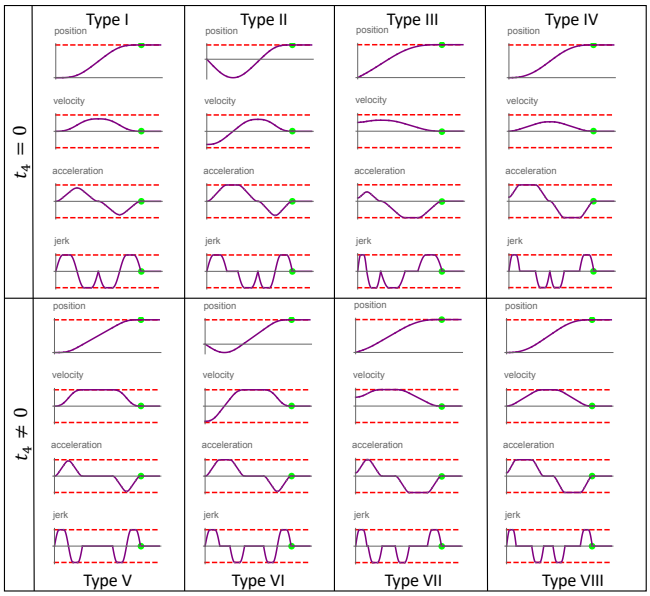

Fig. 2. Kinematic profiles for the eight types of 15-phase jerk-continuous trajectories.

$$J(t) = \begin{cases} J_{\text{peak}} \sin \frac{\pi \tau_i}{2T_1} & t_0 \le t < t_1, t_{12} \le t < t_{13} \\ J_{\text{peak}} & t_1 \le t < t_2, t_{13} \le t < t_{14} \\ J_{\text{peak}} \sin \frac{\pi}{2}(1 + \frac{\tau_i}{T_1}) & t_2 \le t < t_3, t_{14} \le t \le t_{15} \\ 0 & t_3 \le t < t_4, t_7 \le t < t_8, \\ & t_{11} \le t < t_{12} \\ -J_{\text{peak}} \sin \frac{\pi \tau_i}{2T_1} & t_4 \le t < t_5, t_8 \le t < t_9 \\ -J_{\text{peak}} & t_5 \le t < t_6, t_9 \le t < t_{10} \\ -J_{\text{peak}} \sin \frac{\pi}{2}(1 + \frac{\tau_i}{T_1}) & t_6 \le t < t_7, t_{10} \le t < t_{11} \end{cases} . \tag{1}$$

The $\tau_i = t - t_i$ values represent the relative time in the time range of the ith interval, and $i = 1, 2, \ldots, 15$. The corresponding acceleration, velocity and displacement profiles can be derived sequentially by integrating the lower-level profiles. Equations for each segment can be found in [13].

An additional parameter α in [13] was introduced that controls the sinusoidal profile. The parameter α is defined as the ratio of the sine jerk profile duration (T_1) and the

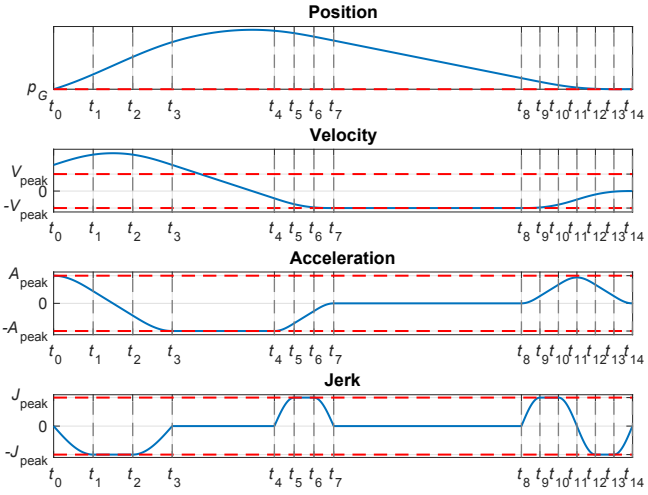

Fig. 3. Overshoot trajectory profile generated by the proposed algorithm.

total duration of sine and constant jerk profile ($T_1 + T_2$):

$$\alpha = \frac{T_1}{T_1 + T_2}, \alpha \in [0, 1]. \tag{2}$$

When α approaches 0, T_1 also approaches 0, and the sinusoidal jerk trajectory planner approximates a bang-bang jerk controller. Oppositely, when α approaches 1, T_2 approaches 0, and the 15-phase sinusoidal jerk trajectory planner approaches a 3-phase sinusoidal jerk trajectory planner. To generate a minimum time trajectory with this sinusoidal jerk pattern, the main problem can be subdivided into the following four cases:

1) Case 1: Maximum acceleration and velocity are attainable ($|A_{\text{peak}}| = A_{\max}$, $|V_{\text{peak}}| = V_{\max}$), and T_1, T_2, T_3 and T_4 are not zero.

2) Case 2: Maximum acceleration is attainable but maximum velocity is unattainable ($|A_{\text{peak}}| = A_{\max}$, $|V_{\text{peak}}| < V_{\max}$), and only T_4 is zero.

3) Case 3: Maximum acceleration is unattainable but maximum velocity is attainable ($|A_{\text{peak}}| < A_{\max}$, $|V_{\text{peak}}| = V_{\max}$), and only T_3 is zero.

4) Case 4: Maximum acceleration and velocity are unattainable ($|A_{\text{peak}}| < A_{\max}$, $|V_{\text{peak}}| < V_{\max}$), and T_3 and T_4 are zero.

The details for computing the time factors are explained well in [13], and will not be further discussed. To apply this to a multi-joint robot, the trajectory duration is computed for each degree of freedom, and the execution time is the maximum duration over all the joints.

B. Jerk-continuous trajectory generation with non-zero initial kinematic inputs

The proposed trajectory generation algorithm that takes non-zero initial kinematic inputs still consists of 15 phases as shown in Fig. 1. However, the time factors of segments τ_1 to τ_4 are replaced by the time factors t_{a1}, t_{a2}, t_{a3}, the time factors of segments τ_5 to τ_7 are replaced by the time factors t_{b1}, t_{b2}, and the time factors of segments τ_9 to τ_{15} are replaced by the time factors t_{c1}, t_{c2}, t_{c3}. This is needed because non-zero kinematic inputs break the symmetric property of the original design concept, thus, the segments τ_1 to τ_7 are no longer symmetric with the segments τ_8 to τ_{15}. Using three sets of time factors provides a greater variety of jerk and acceleration profiles to handle more complicated cases than can be handled by two sets time of time factors. For instance, if the initial velocity and acceleration are maximized, a long displacement is required. After substituting the initial conditions and $t_{a1} \sim t_{a3}$ to the first four phase segments, the sinusoidal jerk profile of the first four phases are:

$$J(t) = \begin{cases} J_{\text{peak}} \sin \frac{\pi \tau_i}{2T_{a1}} & t_0 \le t < t_1 \\ J_{\text{peak}} & t_1 \le t < t_2 \\ J_{\text{peak}} \sin \frac{\pi}{2}(1 + \frac{\tau_i}{T_{a1}}) & t_2 \le t < t_3 \\ 0 & t_3 \le t < t_4 \end{cases}, \tag{3}$$

where $t_1 - t_0 = t_{a1}, t_2 - t_1 = t_{a2}, t_3 - t_2 = t_{a1}$, and $t_4 - t_3 = t_{a3}$. The general acceleration, velocity, and position profile of each phase segment are:

$$\begin{cases} a(t) = a(t_i) + \left(\int_{t_i}^{t} J(t) + C_{i+1}^{a}(t)\, dt \right) \\ v(t) = v(t_i) + \left(\int_{t_i}^{t} a(t) + C_{i+1}^{v}(t)\, dt \right) \\ p(t) = p(t_i) + \left(\int_{t_i}^{t} v(t) + C_{i+1}^{p}(t)\, dt \right) \end{cases}, \tag{4}$$

where t_i is the end time of the previous phase segment. C_{i+1}^{*} is the compensation term of each profile to ensure the current segment starts from the end of the previous segment. The closed-form profile of each segment is listed in the Mathematica file [18]. The sinusoid profile parameter is

$$\alpha = \frac{t_{*1}}{t_{*1} + t_{*2}}. \tag{5}$$

Here $*$ presents the three time factor sets a, b, and c. As in [13], the parameter α defines t_{a2}, t_{b2} and t_{c2} according to (5) and the values t_{a1}, t_{b1} and t_{c1}, simplifying the closed-form solution. It is possible to use three ratio parameters for the three sets of time factors, but only one parameter α is used in the computation to reduce the complexity of the closed-form solution. As Fig. 1 shows, the whole trajectory can be classified as three parts, the first part is from $t_0 \sim t_7$, the second part is from $t_7 \sim t_8$, and the third part is from $t_8 \sim t_{15}$. The acceleration reaches and maintains peak magnitude at time t_3 and t_{11}, and the velocity reaches and maintains peak magnitude at time t_7 and t_8. t_4 is mainly decided by the length of the required displacement because it is the coasting time at the maximum velocity. However, not every trajectory profile can reach the maximum acceleration or velocity from the given initial states, so the time factors t_{a3}, t_{c3} and t_4 can be zero after planning. Therefore, the trajectory profile consists of time factor combinations with and without t_{a3}, t_{c3} and t_4, resulting in the eight types shown in Fig. 2. As shown in Fig. 1, Case 1 presents the first part colored in yellow, Case 2 presents the second

part colored in green, and Case 3 presents the third part colored in red. For Case 1, after substituting the time factors $t_{a1}, t_{a2}, t_{a3}, t_{b1}, t_{b2}$, and integrating from $t_0 \sim t_7$, the equations for the final acceleration, velocity and position at t_7 are:

$$a(t_7) = a_0 + \frac{4J_{\max} s_a t_{a1}}{\pi} + J_{\max} s_a t_{a2} - \frac{4J_{\max} s_a t_{b1}}{\pi} - J_{\max} s_a t_{b2} \tag{6}$$

$$v(t_7) = \frac{1}{2\pi} 2A_{peak}\pi(t_{a3} + 2t_{b1} + t_{b2}) + J_{\max} s_a((2t_{a1} + t_{a2})(4t_{a1} + \pi t_{a2}) - 4t_{b1}^2 - 4t_{b1}t_{b2} - \pi t_{b2}^2 - 2t_{b1}(2t_{b1} + \pi t_{b2})) + 2\pi(a_0(2t_{a1} + t_{a2}) + v_0) \tag{7}$$

$$p(t_7) = p_0 + \frac{8J_{\max} s_a t_{b1}^3}{\pi^3} + \frac{1}{6}(3A_{peak}(t_{a3} + 2t_{b1} + t_{b2})^2 + 3a_0(2t_{a1} + t_{a2})(2t_{a3} + 2t_{a1} + t_{a2} + 4t_{b1} + 2t_{b2}) + \frac{1}{\pi^3} J_{\max} s_a(48(-2t_{a1}^3 + t_{b1}^3) + 24\pi(-t_{a1}^2 t_{a2} + t_{b1}^2 t_{b2}) + \pi^3(3t_{a3}t_{a2}(2t_{a1} + t_{a2}) + 3t_{a2}(2t_{a1} + t_{a2})t_{b1} + 3t_{a1}t_{a2}(t_{a1} + 2t_{b1}) + t_{a2}^2(t_{a2} + 3(t_{a1} + t_{b1})) + 3t_{a2}(2t_{a1} + t_{a2})t_{b2} - 3t_{b1}^2 t_{b2} - 3t_{b1}t_{b2}^2 - t_{b2}^3) - 6\pi^2(-4t_{a1}(2t_{a1} + t_{a2})t_{b1} + 2t_{b1}^3 + 2t_{b1}^2 t_{b2} + t_{b1}t_{b2}^2 + 2t_{b1}^2(t_{b1} + t_{b2}) - t_{a1}(2t_{a1} + t_{a2})(2t_{a3} + 2t_{a1} + t_{a2} + 2t_{b2}))) + 6(t_{a3} + 2t_{a1} + t_{a2} + 2t_{b1} + t_{b2})v_0) \tag{8}$$

Case 2 is the coasting phase at maximum velocity, so the acceleration $a(t_8)$ and velocity $v(t_8)$ are the same as $a(t_7)$ and $v(t_7)$, and position $p(t_8) = p(t_7) + v(t_7)t_4$. For Case 3, after substituting the time factors t_{c1}, t_{c2}, t_{c3}, and t_4, and integrating from $t_0 \sim t_7$, the equation of the velocity and position at t_{15} are:

$$v(t_{15}) = \frac{J_{\max} s_v\left(8t_{c1}^2 + 2(2+\pi)t_{c1}t_{c2} + \pi t_{c2}^2\right) + \pi v(t_8)}{\pi} \tag{9}$$

$$p(t_{15}) = p(t_7) + \frac{1}{2}A_{peak}(2t_{c1} + t_{c2} + t_{c3})^2 + \frac{1}{2}J_{\max} s_v t_{c2}(4t_{c1}^2 + t_{c2}(t_{c2} + t_{c3}) + 2t_{c1}(2t_{c2} + t_{c3})) + \frac{1}{\pi}J_{\max} s_v t_{c1}(8t_{c1}^2 + 2t_{c2}(t_{c2} + t_{c3}) + 4t_{c1}(2t_{c2} + t_{c3})) + (4t_{c1} + 2t_{c2} + t_{c3} + t_4)v(t_7) \tag{10}$$

The s_a and s_v in the above equations are the sign of the jerk profile of Case 1 and Case 3. Because the acceleration profile of Case 3 is symmetric about t_{c3}, and the initial acceleration $a(t_8) = 0$, the final acceleration is $a(t_{15}) = 0$. First, the solutions for a trajectory that does not contain the maximum velocity coasting phase ($t4 = 0$) is discussed.

Type I: ($t_{a3} = 0, t_4 = 0, t_{c3} = 0$). For this type, the A_{peak} of Case 1 and Case 3 are assumed to be less than $A_{\max}$, thus the relationship between t_{a1} and t_{b1} can be found by setting (6) to zero.

$$t_{b1} = t_{a1} + \frac{(a_0\pi\alpha)}{J_{\max}\pi s_a + 4J_{\max} s_a\alpha - J_{\max}\pi s_a\alpha} \tag{11}$$

Then substitute (5) and (7) into (10) and (9), then t_{a1} and t_{c1} can be found by setting the final displacement (10) equal to P_{goal} and final velocity (9) equal to zero. However, the closed-form solution of t_{a1} and t_{c1} can not be directly solved because each equality equation contains second or third order powers of t_{a1} and t_{c1}. This issue is addressed by collecting coefficients of t_{c1} using (9) after substitution, then collecting coefficients of t_{a1} using (10) after substitution, then rewriting these two equality equations with new coefficients. The coefficients were solved in the general form using Mathematica. This results in a sextic equation in t_{a1}. All real, positive roots for t_{a1} are then substituted into an equation to solve for t_{c1}, and only the values of t_{a1} that result in a positive, real t_{c1} are retained.

Type II: ($t_{a3} = 0, t_4 = 0, t_{c3} \neq 0$) Because t_{c3} is not zero, the acceleration of the third part of the trajectory can reach $A_{\max}$. The acceleration reachable by one jerk bump can be presented as (6) without the last two terms on the right. Moreover, $a(t_8)$ is zero, and t_{c1} is

$$t_{c1} = \frac{A_{\max}\pi\alpha}{J_{\max} s_v(\pi + 4\alpha - \pi\alpha)}. \tag{12}$$

The relationship between t_{b1} and t_{a1} is the same as (11). The relationship between t_{a1} and t_{c3} can be found by setting (9) equal to zero.

$$t_{c3} = \frac{-1}{A_{\max}}\left(v_0 + \frac{2a_0 t_{a1}(1+\alpha)}{\alpha} + \frac{a_0^2\pi(1+\alpha)}{2J_{\max} s_a(\pi + 4\alpha - \pi\alpha)} + \frac{A_{\max}^2\pi(1+\alpha)}{J_{\max} s_v(\pi + 4\alpha - \pi\alpha)} + \frac{J_{\max} s_a t_{a1}^2(\pi + 4\alpha + (4-\pi)\alpha^2)}{\pi\alpha^2}\right) \tag{13}$$

Finally, t_{a1} is computed by substituting all terms into (10), generated a quartic equation, where t_{a1} is the positive real root of this equation.

Type III: ($t_{a3} \neq 0, t_4 = 0, t_{c3} = 0$) the acceleration in Case 1 can reach $A_{\max}$, so t_{a1} and t_{b1} can be directly solved using (6). The closed-form solutions are:

$$t_{a1} = \frac{(a_0 - A_{\max})\pi\alpha}{J_{\max} s_a(\pi\alpha - 4\alpha - \pi)} \tag{14}$$

$$t_{b1} = \frac{A_{\max}\pi\alpha}{J_{\max} s_a(4\alpha + \pi - \pi\alpha)} \tag{15}$$

The relationship between t_{a3} and t_{c1} are found by setting (9) to zero.

$$t_{a3} = \frac{1}{2A_{\max} J_{\max}\pi s_a(\pi(\alpha-1) - 4\alpha)\alpha^2}\left((2A_{\max}^2 - a_0^2)\pi^2\alpha^2(1+\alpha) + 2J_{\max}\pi s_a v_0\alpha^2(\pi + 4\alpha - \pi\alpha) + 2J_{\max}^2 s_a s_v t_{c1}^2(1+\alpha)(\pi + 4\alpha - \pi\alpha)^2\right) \tag{16}$$

Similar to *Type II*, t_{c1} can be solved by substituting all terms into (10), and t_{a1} is the minimum positive real root of this quartic equation.

Type IV: ($t_{a3} \neq 0, t_4 = 0, t_{c3} \neq 0$) Because both Case 1 and Case 3 can reach $A_{\max}$, the t_{c1}, t_{a1} and t_{b1} can be computed using (12), (14), and (15). The relationship

between t_{c3} and t_{a3} can be found using the final velocity equality equation:

$$t_{c3} = \frac{1}{2A_{\max 2} J_{\max} s_a s_v (\pi\ (\alpha - 1) - 4\alpha)} \Big(2A_{\max 2}^2 \pi s_a (1+\alpha) - a_0^2 \pi s_v (1+\alpha) + 2 s_v \Big(A_{\max 1}^2 \pi (1+\alpha) + A_{\max 1} J_{\max} s_a t_{a3} (\pi + 4\alpha - \pi\alpha) + J_{\max} s_a v_0 (\pi + 4\alpha - \pi\alpha) \Big) \Big). \tag{17}$$

The term $A_{\max 2}$ is the maximum acceleration reached in Case 1, and $A_{\max 2}$ is the maximum acceleration reached in Case 3. The time t_{a3} is computed by substituting all terms into the final displacement equation and setting it equal to P_{goal}. Unlike with *Type II* and *Type III*, after substitution the equation is a quadratic equation, and t_{a3} is the root. Sometimes both roots can be positive real numbers, so they can both be used as candidate solutions. Usually only the larger root is the correct answer.

The method to solve *Type V*∼*Type VIII* are similar to *Type I*∼*Type IV*, but t_4 is not zero. Because t_4 is the duration time of the coasting phase at maximum velocity, t_4 can be easily solved after all the other time factors are computed. First, let t_4 be equal to zero and substitute all time factors into the final displacement (10). The difference between p_{goal} and the displacement computed from the equation p_{compute} is the distance traveled with maximum velocity. Thus, t_4 can be solved with

$$t_4 = \frac{|p_{\text{goal}} - p_{\text{compute}}|}{V_{\max}} \tag{18}$$

With the given inputs, the planner computes the time factors according to the eight types, and the results satisfy:
(i) All time factors should be non-negative real numbers.
(ii) V_{peak} results calculated by (7), $v(t_7)^a$ and $v(t_7)^b$ must equal each other.
(iii) A_{peak} results calculated by (6), $a(t_7)^a$ and $a(t_7)^b$ must equal each other.
(iv) V_{peak} computed using (7) must not exceed the maximum velocity magnitude.
(v) A_{peak} of *Case 1* and *Case 3* must not exceed the maximum acceleration magnitude.
(vi) Final acceleration $a(t_{15})$ is zero.
(vii) Final velocity calculated using (9) is zero.
(viii) Final position calculated with (10) is the target position.
The threshold used in the program is $1\text{e}{-06}$. The jerk-continuous trajectory with arbitrary initial state and kinematic constraints is minimized by using one of the eight profile types. Theoretically, the eight profile types cover all the trajectory cases, however, experiments show the algorithm may not always find the optimal solution when it can first accelerate or brake down to $\pm V_{\max}$ and then coast. Thus, a pre-check is implemented into the algorithm to address this issue. First, the displacement p_{ta} needed to ramp up or down from initial state to maximum $\pm V_{\max}$ is calculated, then the displacement p_{tb} needed to brake from $\pm V_{\max}$ to zero is computed. If the total distance of p_{ta} and p_{tb} is in same direction as p_{goal} and less than p_{goal}, then there is a coasting phases with $V_{\max}$. The time factor t_4 can be calculated using (18). Because this scenario is one of *Type V*∼*Type VIII*, the methods to solve other time factors is reusable. Finally, the process to compute the time factor and profile of the sinusoidal jerk continuous trajectory with given initial state under constraints is shown as Algorithm 1, which is programmed in Mathematica and C++ [18].

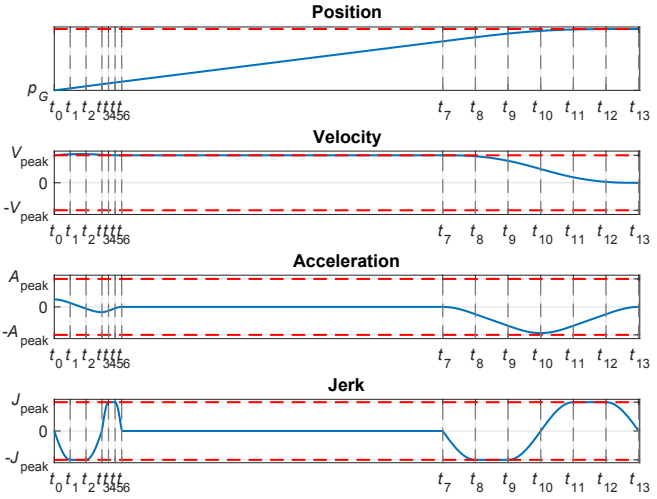

Fig. 4. Trajectory profile coasting at maximum velocity.

Algorithm 1: SINUSOIDAL JERK-CONTINUOUS ONLINE TRAJECTORY GENERATION

Data: $a_0, v_0, p_0, p_G, \alpha, J_{\max}, a_{\max}, v_{\max}$
Pre-check if solution can coast at $V_{\max}$;
if *Yes* **then**
  Generate trajectory profile
else
  Candidates ← Compute time factor of all 8 types;
  TimeFactors ← CheckIfValid[*Candidates*];
  ProfileGenerator[*TimeFactors*];
end
Return: $\langle jerk, accel, vel, pos \rangle$

III. CASE STUDIES

The sinusoidal jerk continuous trajectory generation algorithm proposed in this paper is not limited to the scenarios discussed in the previous section shown as Fig. 2. It can also address the *overshooting problem*, where the initial velocity and/or acceleration cause the system to overshoot either $V_{\max}$ or the target position p_G. Here we presented an extreme case which cannot be solved by the original algorithm presented in [13]. As shown in Fig. 3, the initial position and target position are the same, the initial velocity exceeds $V_{\max}$, and the initial acceleration is $A_{\max}$. The trajectory planned by Algorithm 1. First the acceleration and velocity are ramped down to $-A_{\max}$ and $-V_{\max}$, then it coasts with $-V_{\max}$, and

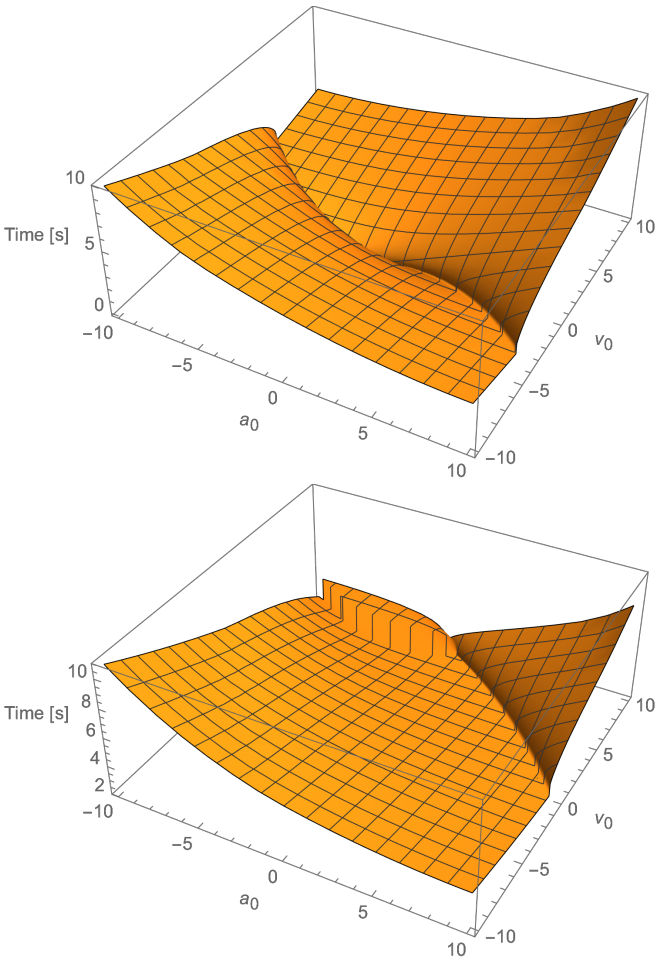

Fig. 5. Minimum time for a change in position of 0 (top) and 10 (bottom) over different initial accelerations and velocities. While the time is continuous with no position change, with a non-zero position change there is a discontinuity.

finally it arrives at the target position with zero acceleration and velocity. This example also demonstrates that if the initial velocity or acceleration causes the velocity or acceleration to break the kinematic constraints, the proposed algorithm can bring the kinematics back to the safe range.

Another case with the initial velocity at the maximum velocity is shown in Fig. 4. The solution trajectory first ramps down the initial acceleration, then brings the velocity back to $V_{\max}$ and starts coasting. Finally, it stops at the goal position.

A minimum-time trajectory generation study is shown in Fig. 5. The input for generating the plot are following: $v_0 \in [-10\,\mathrm{m/s}, 10\,\mathrm{m/s}]$, $a_0 \in [-10\,\mathrm{m/s^2}, 10\,\mathrm{m/s^2}]$, $p_0 = 0\,\mathrm{m}$, $p_G = \{0\,\mathrm{m}, 10\,\mathrm{m}\}$, $J_{\max} = 10\,\mathrm{m/s^3}$, $A_{\max} = 10\,\mathrm{m/s^2}$, and $V_{\max} = 10\,\mathrm{m/s}$. As the figure shows, the x and y-axis are initial acceleration and velocity input, and the z-axis is the duration of the planned trajectory. The minimum time strongly depends on the initial state, and for some configurations the duration has a discontinuity, where the system transitions from being able to directly go to the goal position to an being obligated to overshoot.

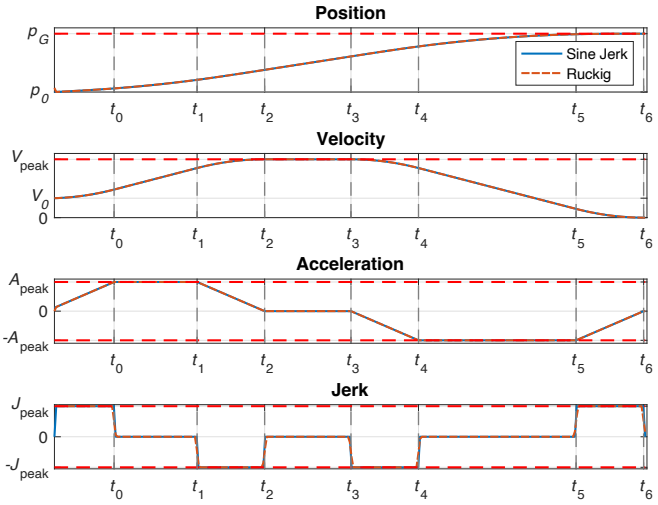

Fig. 6. Trajectory profile comparison with Ruckig [23] when $\alpha = 1\mathrm{e}{-6}$. For this small α value, the difference is tiny.

The mathematics discussed in Section II and the algorithm shown in Algorithm 1 were initially derived and implemented in Mathematica, then converted into C++. All files and codes can be accessed at [18]. To confirm that the computation results are the same after conversion into C++, a data set of 10000 random inputs were generated by MATLAB. The initial acceleration, velocity, position and target position are all bidirectional. Then the trajectory time factors of the two version were compared and mutually compared. The trajectory profiles of random picked inputs of two version were plotted and compared to ensure they are identical. The algorithm was also checked by comparing to results from in [23]. Figure 6 shows a general trajectory profile without brake phases generated by Ruckig [23] and our algorithm when $\alpha = 0.000001$, which approximates a bang-bang jerk controller. The input settings for both algorithms are: $v_0 = 1\,\mathrm{m/s}$, $a_0 = 0.35\,\mathrm{m/s^2}$, $p_0 = 0\,\mathrm{m}$, $p_G = 5\,\mathrm{m}$, $J_{\max} = 10\,\mathrm{m/s^3}$, $A_{\max} = 3\,\mathrm{m/s^2}$, and $V_{\max} = 3\,\mathrm{m/s}$. The two trajectory profiles overlapped with each other and the total time are both $2.617\,18\,\mathrm{s}$.

Finally, the proposed algorithm was implemented and simulated with a UR3 in ROS&Gazebo. Because the proposed trajectory can generate jerk, acceleration, and position profiles, MoveIt [24] jogging servo was utilized to control the robot arm UR3, and the input command is the velocity profile generated by the planner. In this simulation, as shown in Fig. 7, the robot joints followed the planned trajectory and reach the goal position. All the simulation code is available at [18].

IV. Conclusion

This paper proposed a sinusoidal, jerk-continuous online trajectory generation algorithm with non-zero initial states under kinematic constraints. The design logic was discussed in Section II, and the significant formulas are listed in Section II. All the closed-form equations can be accessed in the online Mathematica file. Moreover, the 15-phase proposed solution can also address the issue of overshooting and

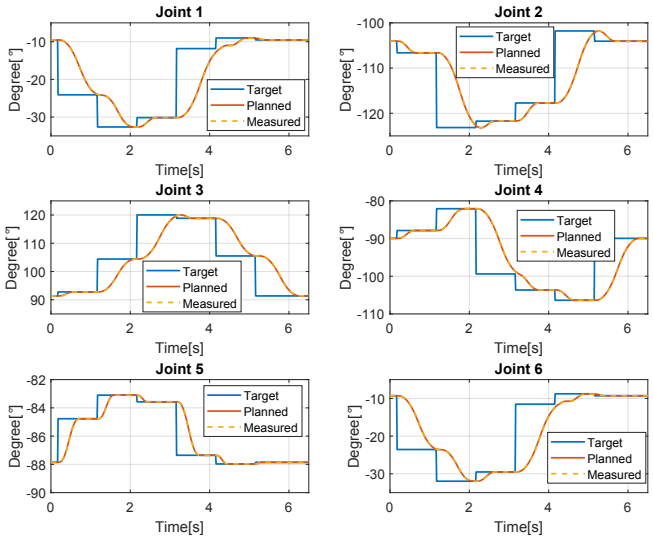

Fig. 7. The joint position recorded from the simulation in ROS&Gazebo using a UR3 robot arm.

constraints violation. Furthermore, the proposed algorithm was compared with Ruckig's online trajectory planner [23] when α is close to zero. Finally, simulation results for controlling a UR3 were presented. Because the primary goal of the proposed algorithm is to generate an online trajectory with a non-zero initial state under constraints with a sine jerk profile, the planned trajectory may or may not be a minimum time trajectory. Due to constraint satisfaction challenges, the algorithm sometimes does not find a 15-phase solution. Instead the system brakes to zero velocity and acceleration, then computes the 15-phase solution to reach the target position. This occurred in 16 of the 10,000 trials.

V. Acknowledge

This work was supported by National Priority Research Program (NPRP) award (NPRP13S-0116-200084) from the Qatar National Research Fund (a member of The Qatar Foundation), the Alexander von Humboldt Foundation, and the National Science Foundation under [IIS-1553063, 1849303, 2130793]. All opinions, findings, conclusions or recommendations expressed in this work are those of the authors and do not necessarily reflect the views of our sponsors.

References

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Modelling, planning and control," *Advanced Textbooks in Control and Signal Processing. Springer,*, 2009.

[2] C. Zheng, Y. Su, and P. C. Müller, "Simple online smooth trajectory generations for industrial systems," *Mechatronics*, vol. 19, no. 4, pp. 571–576, 2009.

[3] B. Xian, M. S. de Queiroz, D. Dawson, and I. Walker, "Task-space tracking control of robot manipulators via quaternion feedback," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, pp. 160–167, 2004.

[4] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *The international journal of robotics research*, vol. 5, no. 3, pp. 72–89, 1986.

[5] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 576–584, 2010.

[6] W. Wu, S. Zhu, and S. Liu, "Smooth joint trajectory planning for humanoid robots based on b-splines," in *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2009, pp. 475–479.

[7] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318–2327, 2009.

[8] L. Biagiotti and C. Melchiorri, *Trajectory planning for automatic machines and robots*. Springer Science & Business Media, 2008.

[9] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. John Wiley & Sons, 2008.

[10] B. Konjević and Z. Kovačić, "Continuous jerk trajectory planning algorithms," in *International Conference on Informatics in Control, Automation and Robotics*, vol. 2. SCITEPRESS, 2011, pp. 481–489.

[11] P. Boscariol, A. Gasparetto, and R. Vidoni, "Planning continuous-jerk trajectories for industrial manipulators," in *Engineering Systems Design and Analysis*, vol. 44861. American Society of Mechanical Engineers, 2012, pp. 127–136.

[12] H. Liu, X. Lai, and W. Wu, "Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 2, pp. 309–317, 2013.

[13] Y. Fang, J. Qi, J. Hu, W. Wang, and Y. Peng, "An approach for jerk-continuous trajectory generation of robotic manipulators with kinematical constraints," *Mechanism and Machine Theory*, vol. 153, p. 103957, 2020.

[14] J. F. Epperson, "On the Runge example," *The American Mathematical Monthly*, vol. 94, no. 4, pp. 329–341, 1987.

[15] S. A. Bazaz and B. Tondu, "Minimum time on-line joint trajectory generator based on low order spline method for industrial manipulators," *Robotics and Autonomous Systems*, vol. 29, no. 4, pp. 257–268, 1999.

[16] A. A. Ata, "Optimal trajectory planning of manipulators: a review," *Journal of Engineering Science and technology*, vol. 2, no. 1, pp. 32–54, 2007.

[17] F. Ramos, M. Gajamohan, N. Huebel, and R. D'Andrea, "Time-optimal online trajectory generator for robotic manipulators," Eidgenössische Technische Hochschule Zürich, Institute for Dynamic Systems, Tech. Rep., 2013. [Online]. Available: https://doi.org/10.3929/ethz-a-007611532

[18] H. Zhao, "Jerk continuous online trajectory generator with constraints," 2022. [Online]. Available: https://github.com/Haoran-Zhao/Jerk-continuous-online-trajectory-generator-with-constraints

[19] S. Perumaal and N. Jawahar, "Synchronized trigonometric s-curve trajectory for jerk-bounded time-optimal pick and place operation," *International Journal of Robotics and Automation*, vol. 27, no. 4, p. 385, 2012.

[20] A. Valente, S. Baraldo, and E. Carpanzano, "Smooth trajectory generation for industrial robots performing high precision assembly processes," *CIRP Annals*, vol. 66, no. 1, pp. 17–20, 2017.

[21] H. Li, Z. Gong, W. Lin, and T. Lippa, "Motion profile planning for reduced jerk and vibration residuals," *SIMTech technical reports*, vol. 8, no. 1, pp. 32–37, 2007.

[22] L. Hongda, C. Hao, X. Xuecheng, and Z. Wansheng, "A jump motion velocity planning algorithm with continuous jerk for electrical discharge machining," *Procedia Cirp*, vol. 42, pp. 547–551, 2016.

[23] L. Berscheid and T. Kröger, "Jerk-limited real-time trajectory generation with arbitrary target states," *Robotics: Science and Systems XVII*, 2021.

[24] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a moveit! case study," *arXiv preprint arXiv:1404.3785*, 2014.