

Article

On Fast Jerk-Continuous Motion Functions with Higher-Order Kinematic Restrictions for Online Trajectory Generation

Burkhard Alpers 

Department of Mechanical Engineering, Aalen University, D-73430 Aalen, Germany;
burkhard.alpers@hs-aalen.de

Abstract: In robotics and automated manufacturing, motion functions for parts of machines need to be designed. Many proposals for the shape of such functions can be found in the literature. Very often, time efficiency is a major criterion for evaluating the suitability for a given task. If there are higher precision requirements, the reduction in vibration also plays a major role. In this case, motion functions should have a continuous jerk function but still be as fast as possible within the limits of kinematic restrictions. The currently available motion designs all include assumptions that facilitate the computation but are unnecessary and lead to slower functions. In this contribution, we drop these assumptions and provide an algorithm for computing a jerk-continuous fifteen segment profile with arbitrary initial and final velocities where given kinematic restrictions are met. We proceed by going systematically through the design space using the concept of a varying intermediate velocity and identify critical velocities and jerks where one has to switch models. The systematic approach guarantees that all possible situations are covered. We implemented and validated the model using a huge number of random configurations in Matlab, and we show that the algorithm is fast enough for online trajectory generation. Examples illustrate the improvement in time efficiency compared to existing approaches for a wide range of configurations where the maximum velocity is not held over a period of time. We conclude that faster motion functions are possible at the price of an increase in complexity, yet which is still manageable.

Keywords: online trajectory generation; fast motion function; fifteen segment profile; jerk continuity; snap restriction



Citation: Alpers, B. On Fast Jerk-Continuous Motion Functions with Higher-Order Kinematic Restrictions for Online Trajectory Generation. *Robotics* **2022**, *11*, 73. <https://doi.org/10.3390/robotics11040073>

Academic Editor: Zhangguo Yu

Received: 9 June 2022

Accepted: 7 July 2022

Published: 12 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Moving quickly from one motion state (position, velocity, acceleration, . . .) to another one is essential for efficiency and quick reactions to unforeseen events in robotics as well as in other areas such as automated manufacturing machines [1–3]. This has led to numerous suggestions of motion functions that take into account restrictions on velocity, acceleration and even higher-order derivatives [1]. The meanwhile classical approach using a “bang-zero-bang” discontinuous acceleration profile, which switches from maximum to zero to minimum constant values, turned out to be unsuitable in many cases, since it cannot be realized by controllers and induces unwanted vibrations. Therefore, very often, motion functions are required to have at least a continuous acceleration. An algorithm for computing such a function with piecewise constant jerk (“seven segment profile”) was presented by the author in [4], where restrictions on velocity, acceleration, and jerk and boundary conditions regarding velocity and acceleration can be prescribed. If there are stronger requirements on vibration reduction (e.g., for restricting position errors), higher-order continuity of the motion function has proven to be effective in experimental settings [5–12]. In this contribution, we present an algorithm for computing a jerk-continuous motion function (as opposed to the piecewise-constant, discontinuous jerk function designed in [4]) in one dimension, such that symmetric restrictions regarding velocity, acceleration, jerk, and snap (the derivative of the jerk, abbreviated as sn in the sequel) are fulfilled and boundary

conditions for position and velocity can be arbitrarily prescribed. We term this function as “fast” because, at any time, for at least one of the kinematic restrictions, the maximum or minimum is reached. Moreover, we show that the algorithm for computing this function is fast enough for online trajectory generation where a computation should be performed within a controller cycle [13,14].

Since the task of designing a fast motion function is so elementary, a considerable number of research articles have appeared, particularly in the last two decades. This work can roughly be organized into two classes: in the first one, the task has been formulated as a restricted optimization problem, and algorithms from this field have been applied (e.g., [15–25]). This approach has the advantage that other aspects such as position errors regarding a given path or energy consumption can be easily included, and even a multi-objective optimization can be performed [22]. Moreover, optimization approaches often employ quintic spline functions, which guarantees continuity up to the snap function (fourth derivative). Yet, for online trajectory generation, the optimization algorithms are too slow. Only the contribution by Valente et al. [18] reports a sufficiently fast optimization procedure, but their three-segment profile could also be computed directly and leaves much space for improving on time efficiency. This leads to a second kind of approach, which constructs the motion function directly based on the given restrictions and boundary conditions. For the subsequent literature overview, we restrict ourselves to those approaches which design a jerk-continuous motion function where restrictions at least regarding velocity, acceleration and jerk can be prescribed [11,12,26–34].

In order to provide a well-structured overview, we apply the following criteria in our description: we first consider the kind of task regarding the boundary conditions of the motion. In many tasks, point-to-point motions are to be designed (e.g., in pick-and-place tasks) where initial and final velocities and accelerations are zero. These are also called rest-to-rest motions and are abbreviated as RR in Table 1. In automated manufacturing, one often has a situation where initial and final velocities are given and higher-order derivatives are zero (cf. [17,35,36]), e.g., when a tool moves with constant velocity for performing a certain task and then has to quickly move to another place to again perform a task with constant velocity. Using an abbreviation from a guideline of the German Association of Engineers [37], we term this task GG in Table 1 (for more information on the tasks, see [4]). Note that the task GG comprises the task RR, since the velocities are allowed to be zero. In robotics, there are also situations where the initial and final velocities and accelerations are arbitrary, e.g., when one has to react to unforeseen events (see [13]). This kind of task is abbreviated as BB in Table 1, where BB+ denotes that higher-order derivatives greater than 2 can also be prescribed. Again, the task BB contains the task GG. As can be seen in Table 1, most of the recent articles are restricted to RR, since this task can become already quite complicated when a fast motion function using the limitations as far as possible is required. The second criterion consists of the type of jerk function. Many of the designed functions in Table 1 work with a fifteen-segment approach, which is the natural extension of the seven-segment profile for motion functions with continuous acceleration. Some authors [11,27,34] also allow more generally for $2^n - 1$ segments when restrictions up to the n -th derivative are to be taken into account. Other approaches achieve smoother jerk functions using trigonometric and zero pieces (having less discontinuities in the snap), but this restricts the possibilities to stay at the maximum value for some period of time. The next criterion is the continuity level of the motion function, i.e., up to which derivative continuity is guaranteed. All functions considered in this summary are required to be at least jerk-continuous, i.e., the continuity level must be at least 3. As can be seen from Table 1, most of the reviewed articles guarantee just the minimum order of 3. In the functions proposed by Lee and Choi [29] and Nguyen et al. [27] (part b), the snap is also continuous, which, on the other hand, has the disadvantage of a slower motion. In the approach by Ezair et al. [34], Nguyen et al. [27] (part a) and Biagiotti and Melchiorri [11,33], the continuity order can be freely chosen (we denote this by adding a “+” in Table 1), but orders higher than 3 lead to an increase in computational effort, which can make the suitability for online

trajectory generation questionable. Fang et al. [30] propose the sigmoid function for going from one constant segment to another one, where all derivatives are continuous. Again, this leads to slower motion functions, as there is always a tradeoff between smoothness and time efficiency. The next criterion is concerned with the kind of kinematic restrictions that can be taken into account. As stated above, we only consider proposals where restrictions on velocity, acceleration and jerk can be guaranteed. Only the designs that work with fifteen segments using linear pieces (or the sigmoid function) also allow us to prescribe a restriction on the snap. This way, it is possible to avoid jerk functions that are “theoretically” continuous but have such a steep increase in jerk that the intended property of vibration reduction is not achieved. In the proposals by Ezair et al. [34], Nguyen et al. [27] (part a) and Biagiotti and Melchiorri [11], one can choose up to which derivative restrictions can be described (denoted by a “+” in Table 1).

Table 1. Jerk-Continuous Motion Functions with Restrictions.

Author/Year	Boundary Conditions	Type of Jerk Function	Continuity Level	Kinematic Restrictions	Constant Phases	Additional Assumptions
Lambrechts et al. 2005 [26]	RR	Piecewise linear, 15 segments	3	v, a, j, sn	v, a, j	j = 0 at max./min. velocity
Nguyen et al. 2008 a [27]	RR	Piecewise linear, 15 segments or higher order	3+	v, a, j, sn +	v, a, j, sn +	j = 0 at max./min. velocity +
Nguyen et al. 2008 b [27]	RR	Piecewise trigon. and zero, 7 segments	4	v, a, j	v, a	j = 0 at max./min. velocity
Lee/Choi 2015 [29]	RR	Piecewise trigon. and zero, 7 segments	4	v, a, j	v, a	j = 0 at max./min. velocity; sn = 0 at max./min. acceleration
Fang et al. 2019 [30]	RR	Piecewise sigmoid and constant, 15 segments	∞	v, a, j, sn	v, a, j, sn	j = 0 at max./min. velocity; sn = 0 at max./min. acceleration
Wang et al. 2019 [31]	RR	3rd degree poly., constant, 3rd degree poly.	3	v, a, j	v	v_{max} is reached
Fang et al. 2020 [12]	RR	Piecewise trigon. and constant, 15 segments	3	v, a, j	v, a, j	j = 0 at max./min. velocity
Wu et al. 2022 [32]	RR	Piecewise trigon. and constant, 15 segments	3	v, a, j	v, a, j	j = 0 at max./min. velocity; j_{max} is reached
Biagiotti/Melchiorri 2019 [11]	RR	Piecewise linear, 15 segments or higher order	3+	v, a, j, sn +	v, a, j, sn +	algorithms are based on assumptions
Biagiotti /Melchiorri 2020 [33]	RR	Piecewise trigon. and zero, 7 segments	3+	v, a, j +	v, a +	j = 0 at max./min. velocity
Biagiotti et al. 2008 [1]	GG	Piecewise linear, 15 segments	3	v, a, j, sn	v, a, j, sn	all maximal/minimal values are reached
Fan et al. 2012 [28]	GG	Piecewise linear, 15 segments	3	v, a, j, sn	v, a, j, sn	j = 0 at max./min. velocity
Own 2022	GG	Piecewise linear, 15 segments	3	v, a, j, sn	v, a, j, sn	none
Ezair et al. 2015 [34]	BB+	Piecewise linear, 15 segments or higher order	3+	v, a, j, sn +	v, a, j, sn +	j = 0 at max./min. velocity +

The final two criteria deal with the question of whether the proposed shape or algorithm affects the time optimality of the motion function. The first one states for which

kinematic functions periods of constant maximum or minimum value are possible. If this is not the case, the resulting functions are often sub-optimal because they do not take full advantage of the given restrictions. Only the designs working with a piecewise linear function with fifteen segments (or more) provide the possibility of having constant phases for maximum/minimum velocity, acceleration, jerk and snap. If there are only three segments as in Wang et al. [31], only the velocity can be held at a constant extremal value. Secondly, there are assumptions in nearly all proposals that are often implicit in the jerk functions and are only seldom stated explicitly. Biagiotti and Melchiorri [1] point out that the provided formulae are only valid when all extremal values (for v , a , j) are taken (this requires a minimum distance to be travelled). In the function proposed by Fang et al. [29], the extremal jerk must occur, which, again, is not necessary for small distances. All but one of the approaches include an assumption that is recognized as making the function potentially sub-optimal by Ezair et al. [34] and Biagiotti and Melchiorri [11]: they assume that when the velocity reaches its extremum in the middle piece (in functions with fifteen segments, this is segment eight), the jerk goes back to zero. This is only necessary when the extremum is the maximum/minimum possible velocity, and this is held for a time period of length greater than zero. If the distances to be travelled are not so large that this is the case, then the assumption leads to an avoidable loss in time efficiency. In the approaches by Fang et al. [30], Nguyen et al. [27] (part b) and Lee and Choi [29], additionally, the shape of the function requires the snap to be zero when the extremum of the acceleration is reached, which is also unnecessary. Moreover, this applies to higher-order derivatives in Ezair et al. [34] and Nguyen et al. 2008 a [27]. The reason for making this assumption is quite obvious: it leads to a mathematically better tractable problem with manageable complexity. Only in the approach by Biagiotti and Melchiorri [19] (in the task RR) is a way to remedy this problem suggested by staying at the maximum/minimum jerk level, but we will give an example showing that, still, their algorithm includes assumptions that prevent them from finding a time-optimal solution in several configurations.

The design presented in this contribution improves what can be found in the literature in the following ways:

- We design a jerk-continuous motion function with better time efficiency by avoiding any additional assumption, particularly that the jerk must go down to zero at the intermediate extremal velocity. This improves the currently available approaches relating to the rest-to-rest (RR) task as well as the work on the GG task (arbitrary initial and final velocities). Using randomly generated input data (restrictions, distance, initial and final velocities), we demonstrate that there are a variety of different possible configurations, and hence we show that the task has an inherent level of complexity that cannot be sidestepped without staying sub-optimal in some situations. Therefore, one cannot expect that simple algorithms will optimally cover all situations.
- The algorithm presented in this contribution is also computationally efficient and hence suitable for Online Trajectory Generation (OLG), which is needed to quickly react to unforeseen events in human–robot interactions.

We do not term our approach “optimal” because we do not have mathematical proof for this (for such kinds of proof see [28,34] where, for example, in [28] the proof is performed by assuming that there is a motion function needing less time, which leads to a contradiction based on a theorem from the field of differential equations; the proof depends on the additional assumption that the jerk is zero at an intermediate velocity). Since, at any time, at least one restriction is adopted, we term our function “fast”.

Our approach has two main areas of real-world application: first, to improve on time efficiency and hence throughput in automated production and, at the same time, to control vibrations and positioning errors by guaranteeing jerk continuity and restriction; secondly, to allow for quick reaction in case of unforeseen events in robotics, particularly when human interaction is involved and computing a new motion function must be performed within a controller cycle (cf. [13,25]).

The paper is structured as follows: Section 2 provides some basic concepts and visualizations that guide our work, and we split the task into two major cases depending on whether or not the maximum jerk can be reached before the maximum acceleration (the distinction is already made in Fan et al. [28]). In Sections 3 and 4, the motion models that can occur in cases I and II are stated, and criteria for switching between models are investigated depending on the intermediate critical velocities and jerk values being reached. In Section 5, we outline our algorithm for setting up the fifteen-segment profile by finding the suitable model. We present our implementation in Matlab®, which we validated using a huge number of randomly chosen tasks. Stability and time efficiency aspects important for online trajectory generation are discussed. In Section 6, we compare our results with those of Fan et al. [28], Ezair et al. [34] and Biagiotti and Melchiorri [11] by presenting some numerical experiments in example configurations. Section 7 summarizes the results, highlights the improvements and outlines directions of future work. All formulae and algorithms needed to reproduce our work have been placed in the appendices.

2. Basic Concepts and Situations

This contribution is concerned with finding a fast jerk-continuous motion function when the following six values are given: sn_{max} , j_{max} , a_{max} , v_{max} (each >0) such that $-sn_{max} \leq sn(t) \leq sn_{max}$, $-j_{max} \leq j(t) \leq j_{max}$, $-a_{max} \leq a(t) \leq a_{max}$, $-v_{max} \leq v(t) \leq v_{max}$; initial velocity: v_A ; final velocity: v_E ; distance: Δs (arbitrary real numbers). Initial and final accelerations and jerks are assumed to be zero. Without loss of generality, we only consider the case $v_A \leq v_E$, since otherwise, we can simply change the positive direction of the distance. We solved this task using a fifteen-segment profile, as shown in Figure 1, for the rest-to-rest (RR) situation. Note that all fifteen segments are only needed when the maximum and minimum jerk, acceleration and velocity values are taken for a time period of length larger than zero.

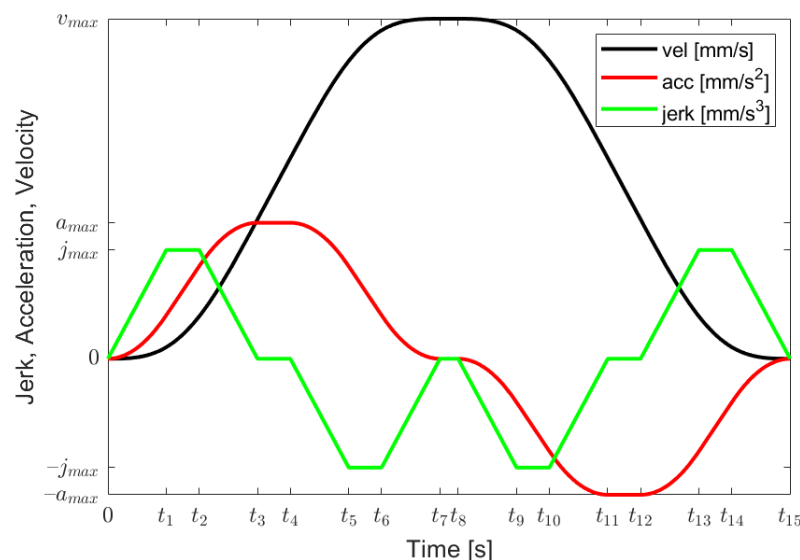


Figure 1. Fifteen-segment motion profile.

Broquere et al. [38] introduced the so-called velocity-acceleration plane to visualize seven-segment motion functions. This visualization has proved to be very helpful in [4] in order to systematically go through the space of all possible situations. In this contribution, we proceed analogously, but the plane has more structure to it. Figure 2a depicts the area of the plane within which a motion can be placed as a parametric curve $(v(t), a(t), t = 0 \dots t_{end})$. The boundary curve consists of pieces where the extremal values of snap, jerk and acceleration, respectively, are taken. We apply the following color coding: light blue, light green and light red denote the maximum values of snap, jerk and acceleration, respectively. The “dark” versions of the colors denote the minimum values. In order

to avoid over-coloring, we draw the boundary curve in black and only apply the color coding for the parametric motion curves. This can be seen in Figure 2b, where an example curve is displayed, which has only nine segments. Such a curve must run from left to right through the upper half-plane and from right to left through the lower one. Note that the parametric curves do not have any kinks, as is the case for only acceleration-continuous motion functions with piecewise constant jerk, as considered in [4,38].

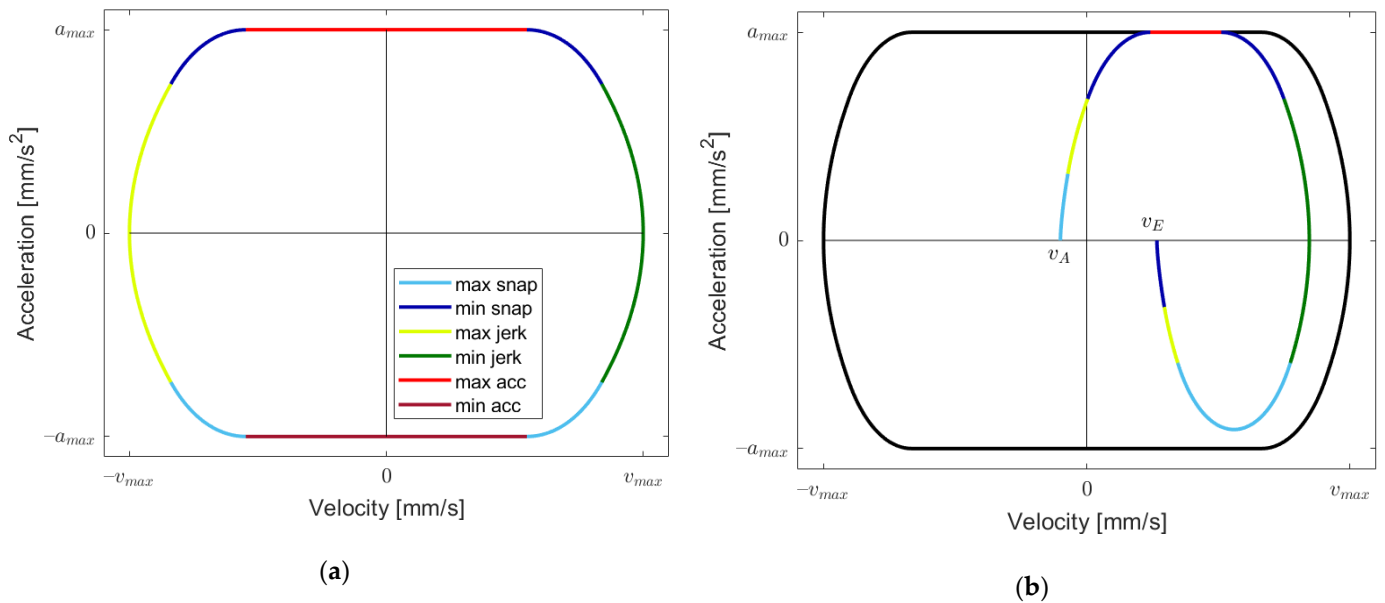


Figure 2. Velocity-acceleration plane. (a) Boundary curve; (b) parametric motion curve.

As is conducted in [28], we distinguish between two major cases depending on whether or not the maximum jerk is reached when applying the maximum and minimum snap in order to get from acceleration zero to maximal acceleration. For the convenience of the reader, we briefly summarize some of the results developed in [28]. The maximum jerk is not reached (Case I) if and only if $j_{max} > \sqrt{a_{max} \cdot sn_{max}}$ holds. We term the other case as Case II in the sequel. We describe a point in the velocity–acceleration plane by a triple (v, a, j) , where the third “coordinate” provides the jerk value of the point (v, a) . A straightforward computation (cf. [28]) yields the time needed for going from $(v_A, 0, 0)$ to $(v_E, 0, 0)$ in the velocity–acceleration plane by accelerating and decelerating as much as possible (see Figure 3a), abbreviated as $T(v_A, v_E)$. The results need further sub-cases and are given in Table 2.

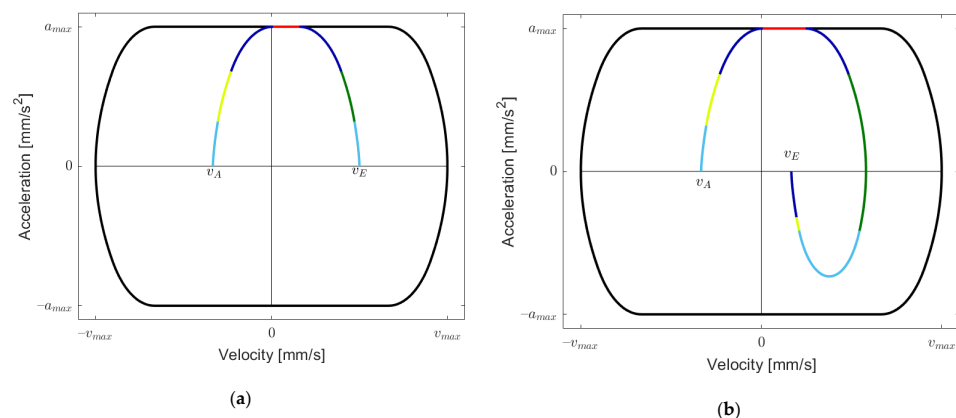


Figure 3. Parametric motion curves. (a) Going from $(v_A, 0, 0)$ directly to $(v_E, 0, 0)$; (b) going from $(v_A, 0, 0)$ to $(v_m, 0, j_m)$ and back to $(v_E, 0, 0)$.

Table 2. Results by Fan et al. [28].

Cases	Additional Conditions	Durations
Case I (j_{max} not reached)		
Sub – Case : a_{max} is not reached	$v_E - v_A < 2sn_{max} \left(\frac{a_{max}}{sn_{max}} \right)^{3/2}$	$T(v_A, v_E) = 4 \cdot \sqrt[3]{\frac{v_E - v_A}{2sn_{max}}}$
Sub – Case : a_{max} is reached	$v_E - v_A \geq 2sn_{max} \left(\frac{a_{max}}{sn_{max}} \right)^{3/2}$	$T(v_A, v_E) = 4 \cdot \sqrt{\frac{a_{max}}{sn_{max}}} + \frac{v_E - v_A - 2sn_{max} \left(\frac{a_{max}}{sn_{max}} \right)^{3/2}}{a_{max}}$
Case II (j_{max} reached)		
Sub – Case : a_{max} and j_{max} not reached	$v_E - v_A < 2sn_{max} \left(\frac{j_{max}}{sn_{max}} \right)^3$	$T(v_A, v_E) = 4 \cdot \sqrt[3]{\frac{v_E - v_A}{2sn_{max}}}$
Sub – Case : a_{max} not reached, j_{max} reached	$v_E - v_A \geq 2sn_{max} \left(\frac{j_{max}}{sn_{max}} \right)^3$	$T(v_A, v_E) = \frac{j_{max}}{sn_{max}} + \frac{\sqrt{4(v_E - v_A) \cdot j_{max} \cdot sn_{max}^2 + j_{max}^4}}{j_{max} \cdot sn_{max}}$
	and $v_E - v_A < a_{max} \left(\frac{j_{max}}{sn_{max}} + \frac{a_{max}}{j_{max}} \right)$	
Sub – Case : a_{max} reached (hence j_{max})	$v_E - v_A \geq a_{max} \left(\frac{j_{max}}{sn_{max}} + \frac{a_{max}}{j_{max}} \right)$	$T(v_A, v_E) = \frac{j_{max}}{sn_{max}} + \frac{a_{max}}{j_{max}} + \frac{v_E - v_A}{a_{max}}$

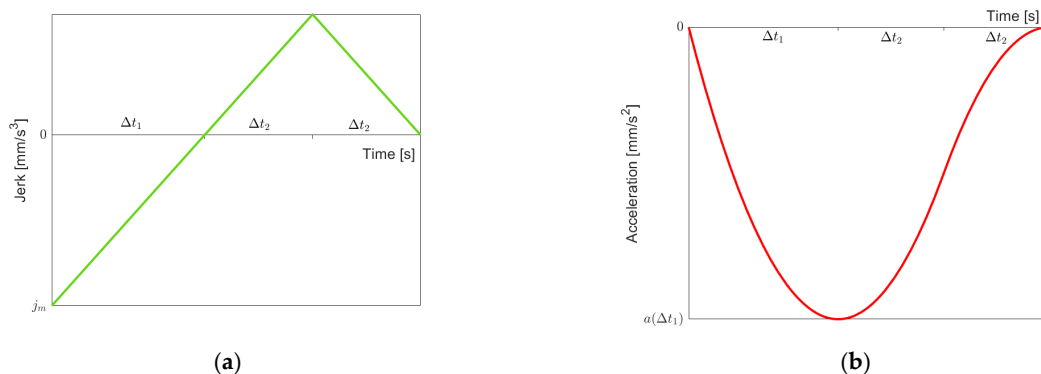
Using the durations stated in Table 2, the distance for going directly from $(v_A, 0, 0)$ to $(v_E, 0, 0)$ can simply be computed as:

$$S(v_A, v_E) = \frac{v_A + v_E}{2} \cdot T(v_A, v_E) \quad (1)$$

In [28], this consideration is actually sufficient, since it is assumed that when the velocity reaches its extremal value v_m , then the jerk goes back to zero, such that the overall motion can be split up into two parts, from $(v_A, 0, 0)$ to $(v_m, 0, 0)$ and from $(v_m, 0, 0)$ to $(v_E, 0, 0)$. This is depicted in Figure 3b.

Our basic idea is also to work with an intermediate extremal velocity v_m , but we obtain a shorter time (in cases where the maximum velocity is not taken for a non-zero period) by allowing the jerk to be non-zero. We will provide an example in Section 6. We first restrict ourselves to the situation where the required distance Δs is greater than or equal to $S(v_A, v_E)$. For finding a suitable intermediate velocity, we let v_m move from v_E to v_{max} . We choose the corresponding jerk value by finding the lowest (negative) value j_m such that it is possible to get back from $(v_m, 0, j_m)$ to $(v_E, 0, 0)$ using extremal snap. Figure 4a shows the jerk function $j(t)$ with three time intervals of lengths Δt_1 , Δt_2 and $\Delta t_3 = \Delta t_2$. This function must fulfill the two requirements: $a(\Delta t_1 + 2\Delta t_2) = 0$ and $v(\Delta t_1 + 2\Delta t_2) = v_E$. Since the snap is maximal on the first interval, we obtain

$$\Delta t_1 = -\frac{j_m}{sn_{max}}. \quad (2)$$

**Figure 4.** Function for going back. (a) Jerk (b) Acceleration.

From the first requirement, it follows that $-\frac{1}{2}j_m \cdot \Delta t_1 = sn_{max}(\Delta t_2)^2$, and hence

$$\Delta t_2 = -\sqrt{\frac{1}{2} \cdot \frac{j_m}{sn_{max}}}. \quad (3)$$

Now, we consider $a(t)$, as shown in Figure 4b. Since for $t \leq \Delta t_1$, we have $a(t) = j_m t + \frac{1}{2}sn_{max}t^2$, we obtain

$$a(\Delta t_1) = -\frac{1}{2} \cdot \frac{j_m^2}{sn_{max}}. \quad (4)$$

Now, we consider $v(t)$. For $t \leq \Delta t_1$, we have $v(t) = v_m + \frac{1}{2}j_m t^2 + \frac{1}{6}sn_{max}t^3$, and hence

$$v(\Delta t_1) = v_m + \frac{1}{2}j_m \Delta t_1^2 + \frac{1}{6}sn_{max} \Delta t_1^3. \quad (5)$$

From the second requirement, we obtain $v_E = v(\Delta t_1 + 2\Delta t_2) = v(\Delta t_1) + \frac{1}{2}a(\Delta t_1) \cdot 2\Delta t_2$. By substituting according to Equations (2)–(5), we finally obtain

$$v_E = v_m + \left(\frac{1}{3} + \frac{\sqrt{2}}{4}\right) \cdot \frac{j_m^3}{sn_{max}^2} \quad (6)$$

$$j_m = -\sqrt[3]{\frac{12sn_{max}^2 \cdot (v_m - v_E)}{4 + 3\sqrt{2}}} \quad (7)$$

From Equation (6), we can also easily compute the intermediate velocity v_m for which j_m reaches $j_{min} = -j_{max}$, which is $v_E - \left(\frac{1}{3} + \frac{\sqrt{2}}{4}\right) \cdot \frac{j_{max}^3}{sn_{max}^2}$. Note that this might be larger than v_{max} , but we neglect this for the moment. We now define the function $vm2jm$, which maps v_m to the corresponding jerk value j_m :

$$vm2jm(v_m) = \begin{cases} -\sqrt[3]{\frac{12sn_{max}^2 \cdot (v_m - v_E)}{4 + 3\sqrt{2}}} & \text{for } v_E \leq v_m \leq v_E - \left(\frac{1}{3} + \frac{\sqrt{2}}{4}\right) \cdot \frac{j_{max}^3}{sn_{max}^2} \\ -j_{max} & \text{for } v_m > v_E - \left(\frac{1}{3} + \frac{\sqrt{2}}{4}\right) \cdot \frac{j_{max}^3}{sn_{max}^2} \end{cases} \quad (8)$$

The main reasoning behind our method can be summarized as follows:

- In order to cover all possible distances larger than or equal to $S(v_A, v_E)$, we let the intermediate velocity and corresponding jerk $(v_m, 0, j_m)$ “flow” from $(v_E, 0, 0)$ via $(v_m, 0, vm2jm(v_m))$ to $(v_{max}, 0, vm2jm(v_{max}))$ where $vm2jm(v_m)$ might become constantly j_{min} at some intermediate velocity. Then, we proceed further by going from $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$. From there, one can achieve arbitrarily large distances by moving constantly with v_{max} . Since in this “flow”, we make continuous changes in v_m and j_m , the covered distances also change continuously, and thus we have full coverage of all distances $\geq S(v_A, v_E)$. One can proceed in a similar way by letting $(v_m, 0, j_m)$ run from $(v_A, 0, 0)$ to $(-v_{max}, 0, 0)$, covering all distances $\leq S(v_A, v_E)$.
- In this “flow”, we identify the occurring jerk models and the switching points when there is a model change. For this, we compute the so-called “critical” velocities and jerks where extremal values of acceleration or jerk are reached.
- For all velocity-jerk combinations where there is a model change, we compute the distances reached such that we can determine which model must be applied for the given distance. In that model, we then can compute the time periods of the segments it consists of.

The subsequent Sections 3 and 4 describe the occurring models, the computation of “critical” velocities and jerks as well as model sequences when traversing the velocity-

acceleration plane. In Section 5, we describe how to compute the time periods in the applicable model using a simple bisection method.

3. Models and Critical Velocities and Jerks in Case I

In Case I, j_{max} is not reached when going from $(v_A, 0, 0)$ to the maximal acceleration where the jerk must be zero. The maximal possible jerk value occurs when starting at a point with $a = a_{min}$ and applying maximal snap until one reaches $a = 0$. A simple computation yields the value $\sqrt{2a_{max}sn_{max}}$. If j_{max} is larger than this value, we simply set $j_{max} = \sqrt{2a_{max}sn_{max}}$.

We first assume $\Delta s \geq S(v_A, v_E)$, and we neglect the restriction on maximal velocity. In order to manage complexity, we split the task of finding suitable models into two parts: going “forth” from $(v_A, 0, 0)$ to $(v_m, 0, j_m)$ and going “back” from $(v_m, 0, j_m)$ to $(v_E, 0, 0)$. Figure 5 shows possible jerk models for going “forth” using maximal, minimal or zero snap (the notation “I.xf” refers to these models in Case I). The first four models are built using the criteria of whether j_{min} is or is not reached when going from the maximal acceleration (of this function) down to acceleration zero, and a_{max} is or is not reached. Model I.1f depicts the situation where j_{min} and a_{max} are both not reached. Model I.2f shows the motion going “forth” when j_{min} is reached but a_{max} is not reached, whereas Model I.3f is vice versa. In Model I.4f, both j_{min} and a_{max} are reached. In both models I.2f and I.4f where j_{min} is reached, $j_m > j_{min}$ still holds, which need not be the case. As in Model I.2f, in Model I.5f j_{min} is reached and a_{max} is still not reached, but now, $j_m = j_{min}$. Finally, in Model I.6f, both j_{min} and a_{max} are reached (as in Model I.4f), and $j_m = j_{min}$. Note that Models I.5f and I.6f could also be treated as special cases of Models I.2f resp. I.4f with $\Delta t_5 = 0$ resp. $\Delta t_6 = 0$. We decided to set up separate models for two reasons: reaching certain limits (here $j_m = j_{min}$) is more recognizable by having a model change, and the models can then be simpler (see the formulae in Table A1).

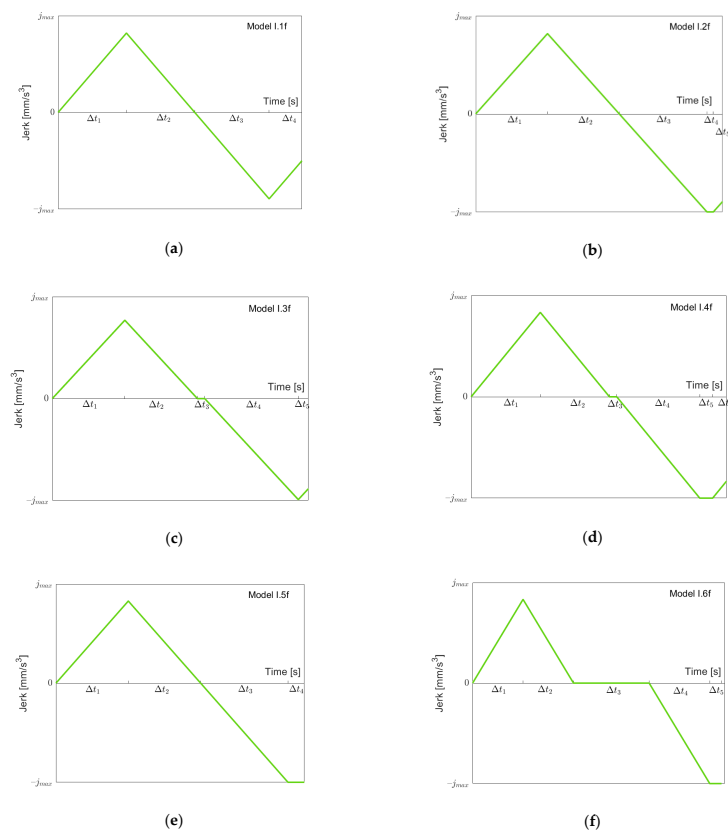


Figure 5. Jerk models in Case I going forth. (a) j_{min} , a_{max} not reached; (b) j_{min} reached, a_{max} not reached, $j_m > j_{min}$; (c) j_{min} not reached, a_{max} reached; (d) j_{min} , a_{max} reached, $j_m > j_{min}$; (e) j_{min} reached, a_{max} not reached, $j_m = j_{min}$; (f) j_{min} , a_{max} reached, $j_m = j_{min}$.

If one of these models is suitable for going from $(v_A, 0, 0)$ to $(v_m, 0, j_m)$ (conditions for this are investigated below), one has to compute the length of the time intervals Δt_i ($i = 1 \dots 4$ or 5 or 6). In order to have a simple notation in the subsequent computations, we define the point in time t_i ($i = 1 \dots \#intervals$) as $t_i = \sum_{k=1}^i \Delta t_k$. From the jerk models, one can easily produce acceleration and velocity models using the initial values 0 and v_A , respectively. We have three conditions that must be fulfilled (t_{end} denotes the sum of the time intervals):

$$j(t_{end}) = j_m, a(t_{end}) = 0, v(t_{end}) = v_m \quad (9)$$

Further conditions originate from the special shape of the model functions: In all models we have $\Delta t_1 = \Delta t_2$; additional equations can be set up when j_{min} or a_{max} are reached. It is easy to check that this results in a sufficient number of equations for computing the time intervals. These equations lead either directly to formulae for the time intervals or to a polynomial equation that has a time interval as one of its roots. We demonstrate this for Model I.1f, where the equations to be fulfilled are the following:

$$\Delta t_1 = \Delta t_2, v(t_4) = v_m, a(t_4) = 0, j(t_4) = j_m \quad (10)$$

From $a(t_4) = 0$, we obtain $(\Delta t_1)^2 sn_{max} = (\Delta t_3)^2 sn_{max} - \frac{1}{2}(\Delta t_3 - \Delta t_4)^2 sn_{max}$, and hence

$$\Delta t_1 = \sqrt{\frac{1}{2}(\Delta t_3)^2 + \Delta t_3 \Delta t_4 - \frac{1}{2}(\Delta t_4)^2}. \quad (11)$$

From $j(t_4) = j_m$, it follows that

$$\Delta t_4 = \Delta t_3 + \frac{j_m}{sn_{max}}. \quad (12)$$

By integrating the jerk function twice and using $v(t_4) = v_m$, one obtains:

$$v_m = v_A + sn_{max}(\Delta t_1)^3 + sn_{max}(\Delta t_1)^2 \Delta t_3 - \frac{1}{6}sn_{max}(\Delta t_1)^3 + \left(sn_{max}(\Delta t_1)^2 - \frac{1}{2}sn_{max}(\Delta t_3)^3 \right) \Delta t_4 - \frac{1}{2}sn_{max} \Delta t_3 (\Delta t_4)^2 + \frac{1}{6}sn_{max}(\Delta t_4)^3. \quad (13)$$

By substituting Δt_1 and Δt_4 in Equation (13) using Equations (11) and (12), we obtain an equation where only Δt_3 is unknown. This equation cannot be solved analytically, and because of the square root in Equation (11), it is not a polynomial equation. Yet, using algebraic transformations, it can be turned into a polynomial equation of degree 6, the solutions of which have to be multiplied with sn_{max}^{-1} . We used the Computer Algebra System (CAS) Maple® for doing this. The coefficients of the polynomial are given in Appendix A (Table A1, Model I.1f). Note that all solutions to the original Equation (13) are also solutions to the transformed polynomial problem but not the other way around, such that it must be checked which of the solutions solve the original problem. Nonetheless, by transforming the problem into a polynomial one, stable polynomial solvers can be employed, which find all roots such that one can avoid the problem of depending on suitable initial values in order to find feasible solutions. This strategy has been pursued at several places in our algorithms, and all resulting polynomials can be found in the appendices.

In all other models in Case I “going forth”, we similarly get the respective equations but those can be solved directly where again a CAS is helpful. Sometimes, one obtains more than one solution, but it is always possible to find among those the only one that provides a real positive value. The results are shown in Table A1 (Appendix A).

Figure 6 depicts the possible models for going “back” from $(v_m, 0, j_m)$ to $(v_E, 0, 0)$. Clearly, before (or at the same moment as) the acceleration goes down to a_{min} , the jerk j_m reaches its minimal value j_{min} . Therefore, we have a simple sequence of the three models I.1b, I.2b and I.3b, as shown in Figure 6a–c when we proceed from $(v_E, 0, 0)$ to $(v_{max}, 0, vm2jm(v_{max}))$. Note that not all models must be present depending on the value

of v_{max} . When we then proceed from $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$, as described in the previous section, one of the Models I.4b to I.7b might occur (Figure 6d–g). Models I.4b, I.5b and I.7b simply add a “front piece” starting at j_m , which goes up to zero. Model I.6b is needed, since when starting with Model I.7b and reducing j_m , it is possible that first, the minimal jerk becomes larger than j_{min} before the minimal acceleration becomes larger than a_{min} . Note that when reaching $(v_{max}, 0, 0)$, the model must be symmetric, since one starts and ends with acceleration zero. Again, one could treat Models I.1b, I.2b and I.3b as special cases of Models I.4b, I.5b and I.7b with $\Delta t_1 = 0$, but, for the same reasons as stated above, for going forth, we treat them separately. From the jerk models, one can easily produce acceleration and velocity models using the initial values 0 and v_m , respectively. We have three conditions that must be fulfilled (t_{end} denotes the sum of the time intervals):

$$j(0) = j_m, a(t_{end}) = 0, v(t_{end}) = v_E \quad (14)$$

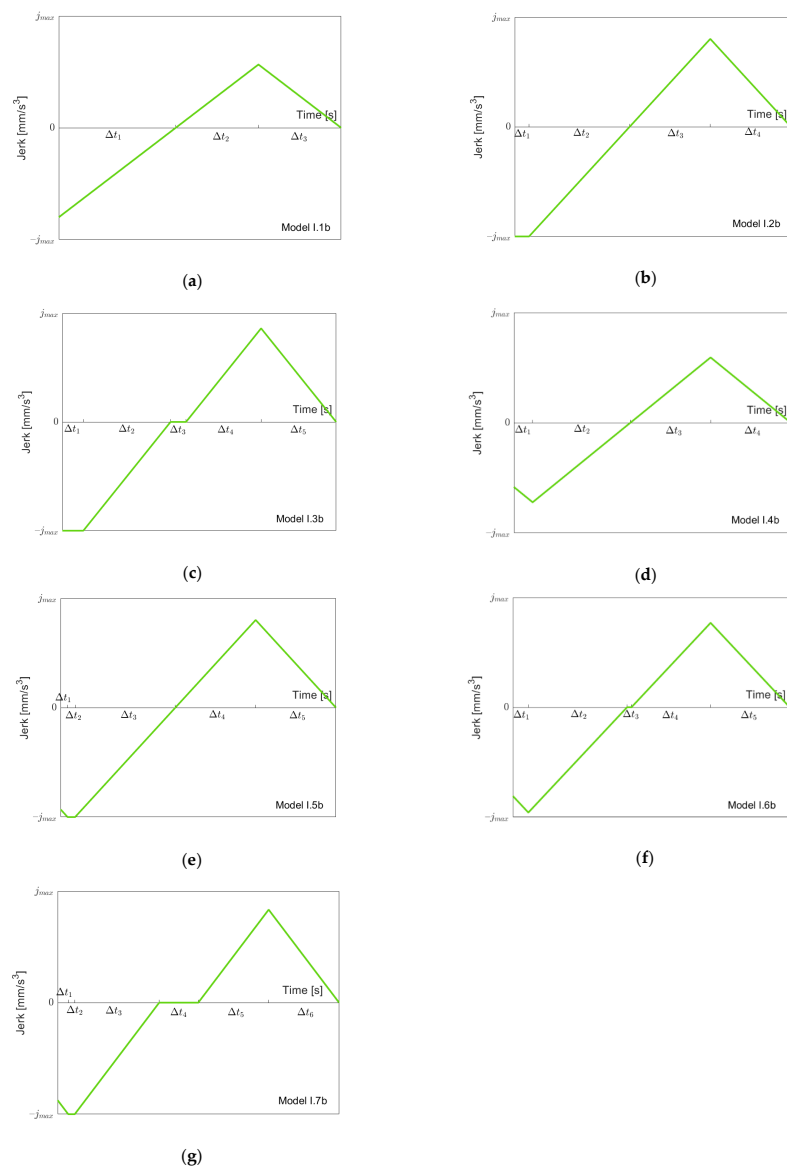


Figure 6. Jerk models in Case I going back. (a) $j_m > j_{min}$; (b) $j_m = j_{min}$, a_{min} not reached; (c) $j_m = j_{min}$, a_{min} reached; (d) $v_m = v_{max}$, j_{min} , a_{min} not reached; (e) $v_m = v_{max}$, j_{min} reached, a_{min} not reached; (f) $v_m = v_{max}$, j_{min} not reached, a_{min} reached; (g) $v_m = v_{max}$, j_{min} , a_{min} reached.

Further conditions originate from the special shape of the model functions: in all models, the final two time intervals must have equal length. Moreover, in Models I.2b, I.3b, I.5b and I.7b, there is an interval where the jerk goes from its minimum to zero. In Models I.3b, I.6b and I.7b, the minimal acceleration is reached. Finally, there is a relation between the start jerk j_m and the first two resp. three time intervals in Models I.4b, I.6b resp. I.5b, I.7b (note that in Model I.1b, v_m and j_m are related via Equation (8), upper part, so the system of equations is not overdetermined). It is easy to check that this leads to a sufficient number of conditions to determine the time intervals in all models going “back”. The corresponding equations lead either directly to formulae for the time intervals or to a polynomial equation of a degree of at most four, which has a time interval as one of its roots (cf. the exemplary computation of time intervals for the “going forth” model I.1f above). We provide the respective expressions in Appendix A, Table A2.

Next, we address the question in which possible sequences the models occur when $(v_m, 0, j_m)$ runs from $(v_E, 0, 0)$ to $(v_{max}, 0, v_{m2jm}(v_{max}))$, as stated in the previous section. For this, we investigate the values of v_m for which, in the going forth situation, the minimal jerk reaches the value $j_{min} = -j_{max}$ (denoted by v_{mjminA}), and the maximal acceleration reaches the value a_{max} (denoted by v_{mamaxA}). In the sequel, we present the procedure for which the algorithm can be found in Appendix B, Algorithm A1.

- If the inequality $v_E - v_A \geq 2sn_{max} \left(\frac{a_{max}}{sn_{max}} \right)^{3/2}$ holds (cf. Table 1), then a_{max} is already reached for $v_m = v_E$, so we have $v_{mamaxA} = v_E$. In order to find v_{mjminA} , we set $j(t_4) = j_{min}$ in Model I.3f. In the equations given for the time intervals in Appendix A, Table A1, for this model, now, j_m is unknown, but the additional equation $j(t_4) = j_{min}$ allows us to compute this quantity: since $\Delta t_4 = \frac{j_{max}}{sn_{max}}$, $\Delta t_5 = \frac{j_{max} + j_m}{sn_{max}}$, and $a_{max} = \frac{1}{2}j_{max}\Delta t_4 + \frac{j_{max} - j_m}{2}\Delta t_5$, we obtain the formula presented in line 3 for j_m . The corresponding value for v_m can be determined using Equation (6). This gives v_{mjminA} , and we are done.
- Otherwise, we consider Model I.1f and investigate the condition $j(t_3) = j_{min}$ i.e., $\Delta t_3 = \frac{j_{max}}{sn_{max}}$. Again, j_m is an additional unknown. If $v_A = v_E$, we have $j_m = j_{min}$ because of symmetry. Otherwise, by substituting Δt_1 and Δt_4 in Equation (13) using Equations (11) and (12), we obtain an equation where, now, only j_m is unknown. This equation can be transformed into a polynomial one of degree 6, the coefficients of which are given in line 8 of algorithm A1. We pick the real root for which $j_{min} \leq j_m \leq 0$ holds.
- Next, we have to check whether j_{min} is really reached before a_{max} . For this, we compute $a(t_2)$ in Model I.1f, which gives $\frac{2j_{max}^2 - j_m^2}{2sn_{max}}$, and we check whether $a(t_2) \leq a_{max}$ holds. If that is the case, then j_{min} is reached first, and we compute v_{mjminA} according to Equation (6) (cf. line 14). Then, we set $a(t_2) = a_{max}$ in Model I.2f. This leads to the formula for j_m presented in line 15. If $j_m \geq j_{min}$, then Model I.2f was applicable, and we obtain v_{mamaxA} according to Equation (6), as presented in line 17. Otherwise, we have to apply Model I.5f (i.e., j_m has already reached j_{min}), and we obtain v_{mamaxA} according to line 18.
- Otherwise (i.e., $a(t_2) > a_{max}$), a_{max} is reached first, and we have to set $a(t_2) = a_{max}$ in Model I.1f. This additional equation allows us to compute j_m as the real root of the polynomial of degree 4 presented in line 21, for which $0 \geq j_m \geq j_{min}$ holds. Then, again, we obtain v_{mamaxA} according to Equation (6), as given in line 23. By setting $j(t_4) = j_{min}$ in Model I.3f, we obtain the value for j_m when j_{min} is reached, and line 25 computes the corresponding value for v_{mjminA} .

The algorithm illustrates a strategy often pursued in this work: By going stepwise through the available models one ends up with an applicable model in which one can compute the desired value either directly or by computing roots of a polynomial.

Note that “critical” velocities v_{mjminA} , v_{mamaxA} might be larger than v_{max} . We will look at the possible configurations below. Next, we investigate at which value of v_m in the going

“back” part the minimal jerk and the minimal acceleration are reached (denoted by v_{mjminE} and by v_{maminE} , resp.). In the “going back” situation, we always have $v_{mjminE} \leq v_{maminE}$, and we obtain v_{mjminE} using Equation (6) with $j_m = j_{min}$ and v_{maminE} by setting $a(t_2) = a_{min}$ in Model I.2b (see Table A5).) Figure 7 depicts the possible relative positionings of v_{max} , v_{mjminA} and v_{mamaxA} on the v_m -scale and provides information on which model to apply when v_m lies within a certain interval left of v_{max} . In Figure 7b,d,e, Models I5.f resp. I.6f have to be chosen once v_{mjminE} is reached. Note that the interval of possible values for v_m starts at v_E and ends at v_{max} , so if, for example, $v_E = v_{mamaxA}$ in Figure 7c or e, then Model I.1f does not occur.

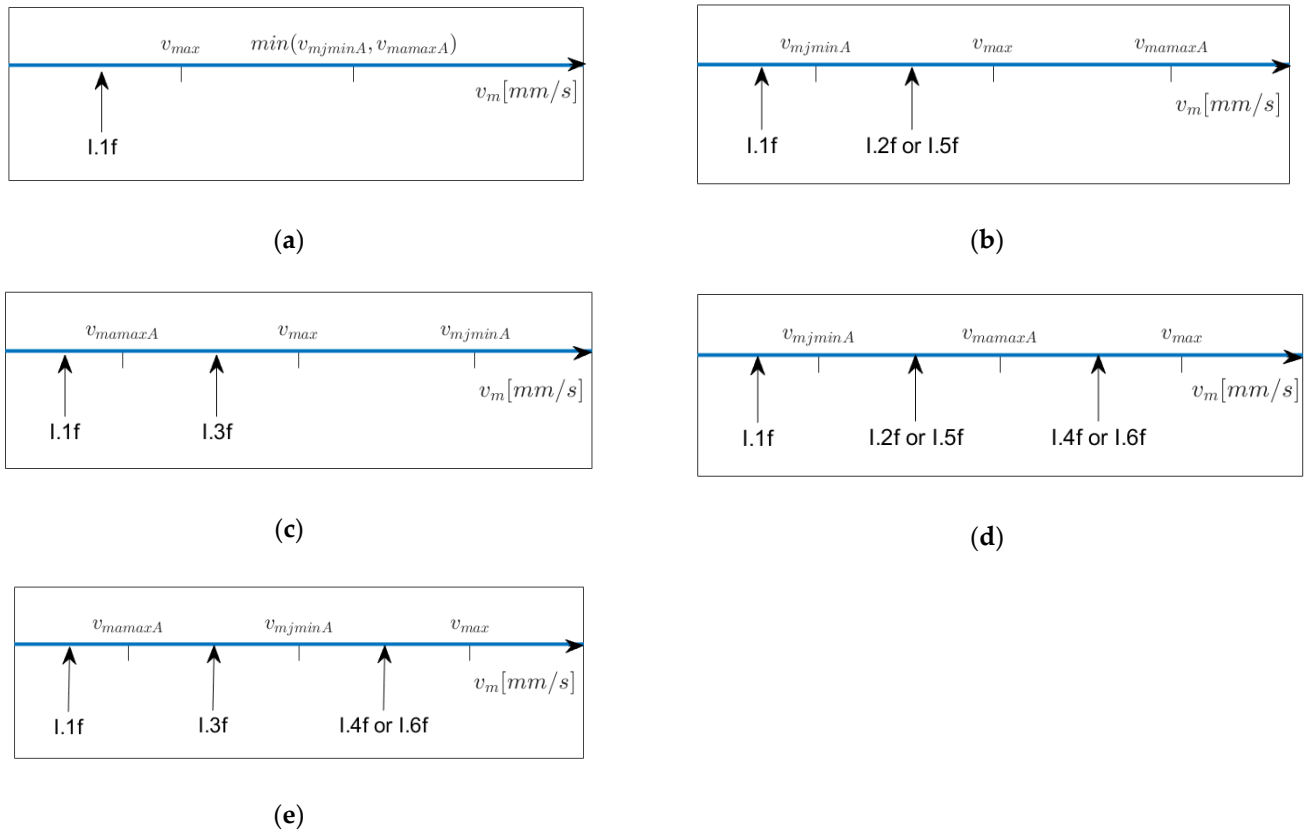


Figure 7. Relative positioning of v_{max} , v_{mjminA} and v_{mamaxA} on the v_m scale and corresponding models.

Figure 8 depicts the possible relative positionings of v_{max} , v_{mjminE} and v_{maminE} on the v_m scale and provides information on which model to apply when v_m lies within a certain interval left of v_{max} .

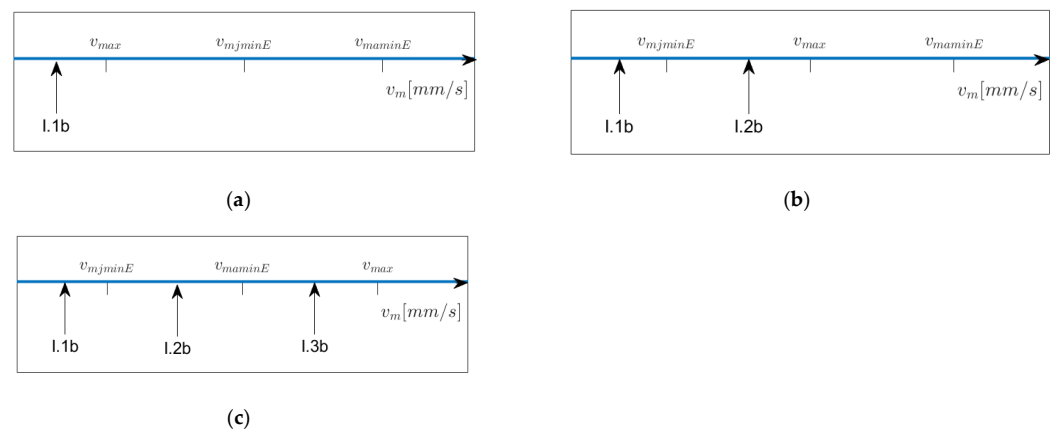


Figure 8. Relative positioning of v_{max} , v_{mjminE} , and v_{maminE} on the v_m -scale and corresponding models.

So, we are able to specify the applicable model when v_m runs from from $(v_E, 0, 0)$ to $(v_{max}, 0, vm2jm(v_{max}))$ in Case I. The remaining task consists of finding the correct model when we go from $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$, i.e., we keep $v_m = v_{max}$ constant and let j_m run from $vm2jm(v_{max})$ to 0. The additional challenge here originates from the possibility that already-reached extremal values j_{min}, a_{max} (for going forth) resp. j_{min}, a_{min} (for going back) might be no longer reached. For example, when j_{min} was reached (going forth) and j_m moves to zero again then, at some value of j_m , the minimal jerk has to be above j_{min} since we are in Case I. Figure 9 demonstrates this in the velocity acceleration plane where we omit the boundary curve for better visibility of interesting curve sections: we have $v_{max} = 35$, $a_{max} = 10$, $j_{max} = 12$, $sn_{max} = 10$ for the restrictions, $v_A = 10$, $v_E = 20$, $v_m = v_{max}$ and $j_m = vm2jm(v_{max}) = j_{min}$, resulting in a distance $\Delta s = 166.45$ (in our examples, the units are always mm, $\frac{mm}{s}$, $\frac{mm}{s^2}$, $\frac{mm}{s^3}$ and $\frac{mm}{s^4}$, respectively, and we omit these in the sequel for brevity). We recognize Model I.6f for going forth and I.2b for going back in Figure 9a,b. When we now enlarge j_m to -10.58 , we obtain a distance of 167, and we see in Figure 9c that the models switched to I.4f and I.4b, respectively. When we now consider the final jerk value zero, we obtain as distance 201.19, and Figure 9d shows that the going forth model again switched to I.3f, whereas the going back model does not change. The three curves in Figure 9b–d are close to each other, but by zooming, we see that we obtain higher curvature when moving to $j_m = 0$. In this case, the maximal acceleration is still reached, but when one moves v_A closer to v_E such that the part with constant acceleration a_{max} is very short, another model switch might occur. This illustrates the inherent complexity, which cannot simply be sidestepped.

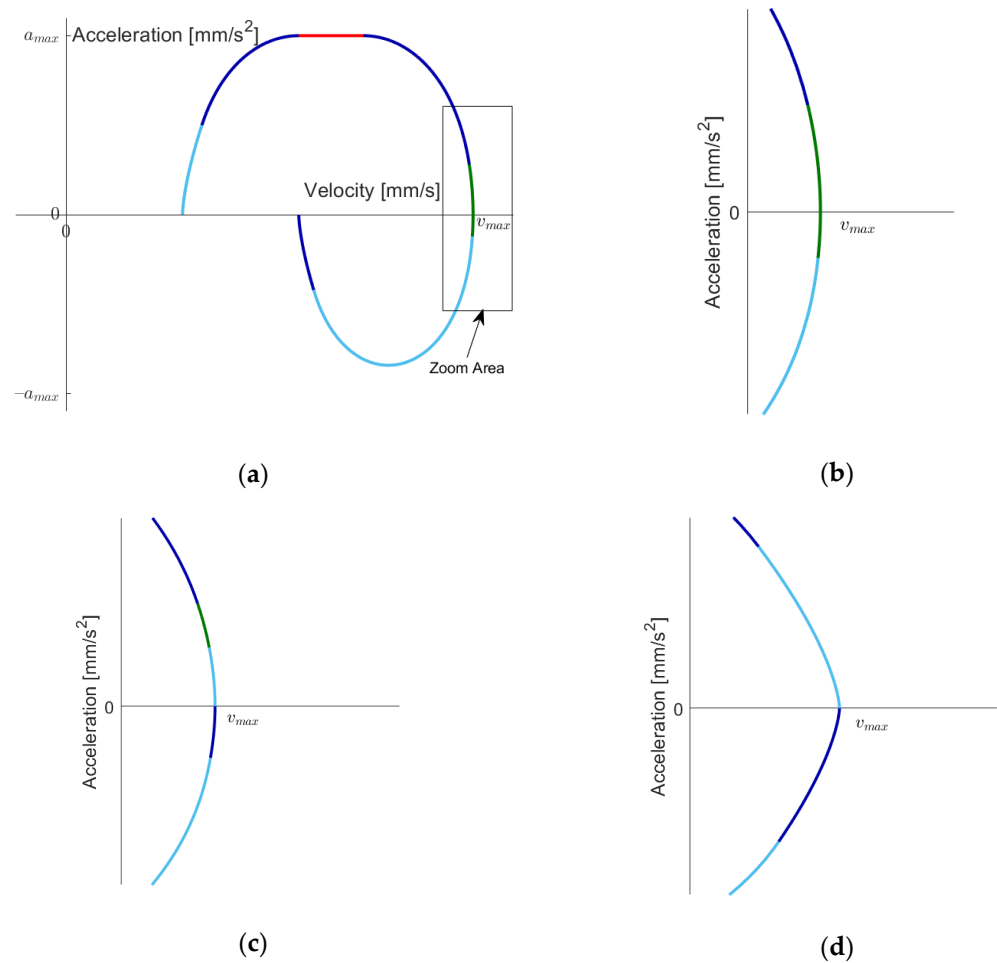


Figure 9. Going from $j_m = vm2jm(v_{max}) = j_{min}$ to $j_m = 0$. (a) $j_m = j_{min} = -12$, $\Delta s = 166.45$; (b) zoom into (a); (c) zoom into $j_m = -10.58$, $\Delta s = 167$; (d) zoom into $j_m = 0$, $\Delta s = 201.19$.

Next, we find out at which values for j_m certain model switches occur. We set $v_m = v_{max}$ and proceed similarly as when determining the critical velocities: in the models, we add an additional equation depending on which extremal acceleration or jerk has been reached, and we determine j_m as an additional unknown; the latter can be computed directly or via solving a polynomial equation. If the solution leads to non-admissible values for j_m or to negative values for time intervals, then there is no feasible solution. Since the computations are straightforward and can be easily performed using a CAS, we just give the results that are needed for implementation. The procedure is described in more detail in Algorithm A3. We first consider the going forth part and refer to Figure 7 regarding the possible positionings of v_{max} . If we have situation (a), Model I.1f will be applied for all jerk values. If situation (b) applies, we proceed as follows:

- In Model I.1f, set $v_m = v_{max}$ and $j(t_3) = j_{min}$ and compute j_m by proceeding as stated in Algorithm A3, line 2, denoted by $j_{mswitch}$. There must be a solution, since we must end up with a symmetric model, and Model I.2f is not symmetric.
- For $v_{m2}j_m(v_{max}) \leq j_m < j_{mswitch}$, apply Model I.2f, for $j_{mswitch} \leq j_m \leq 0$ Model I.1f.

If situation (c) applies, the procedure is as follows:

- In Model I.1f, set $v_m = v_{max}$ and $a(t_2) = a_{max}$ and compute j_m by proceeding as stated in Algorithm A3, line 5, denoted by $j_{mswitch}$ in case of existence. If there is no such solution, the suitable model is always Model I.3f. This is also the case when $v_E = v_{mamaxA}$ holds (cf. the explanation for Figure 5).
- Otherwise, for $v_{m2}j_m(v_{max}) \leq j_m < j_{mswitch}$, apply Model I.3f, for $j_{mswitch} \leq j_m \leq 0$ Model I.1f.

If situation (d) applies, perform the following:

- In Model I.4f, set $v_m = v_{max}$ and $\Delta t_5 = 0$ (i.e., the phase of constant minimal jerk has duration zero). Compute j_m and substitute the value into the expression given for Δt_3 in Table A1, Model I.4f. If $\Delta t_3 \geq 0$ holds (i.e., the solution is admissible), set $j_{mswitch1} = j_m$, and the next model is I.3f. Otherwise, set $\Delta t_3 = 0$ in Model I.4f (i.e., the phase of constant maximal acceleration has duration zero) and compute j_m by proceeding as stated in Algorithm A3, lines 12/13, also denoted by $j_{mswitch1}$. If there is a real solution with $v_{m2}j_m(v_{max}) \leq j_{mswitch1} \leq 0$, the next model is I.2f. Since Model I.4f is not symmetric, a valid $j_{mswitch1}$ must exist.
- If Model I.3f is next, set $v_m = v_{max}$ and $a(t_2) = a_{max}$ in Model I.1f and compute j_m as in situation (c), denoted by $j_{mswitch2}$. This need not exist.
- If Model I.2f is next, set $v_m = v_{max}$ and $j(t_3) = j_{min}$ in Model I.1f and compute j_m as in situation (b), denoted by $j_{mswitch2}$. This, then, must exist, since I.2f is not symmetric.
- Now, for $j_{min} \leq j_m < j_{mswitch1}$, take Model I.4f. For $j_{mswitch1} \leq j_m < j_{mswitch2}$, apply Model I.3f or I.2f, resp. If no valid $j_{mswitch2}$ exists, apply Model I.3f for $j_{mswitch1} \leq j_m \leq 0$. Otherwise, for $j_{mswitch2} \leq j_m \leq 0$, Model I.1f is suitable.

Finally, if situation (e) applies, we have the same procedure as for situation (d). If $v_E = v_{mamaxA}$, there is no $j_{mswitch2}$ and we switch directly from Model I.4f to I.3f.

Next, we consider the going back situation and refer to Figure 8. If we have situation (a), Model I.4b will be used for all jerk values. If situation (b) applies, we proceed as follows:

- In Model I.5b, set $v_m = v_{max}$ and $\Delta t_2 = 0$ and compute j_m , denoted by $j_{mswitch}$ by proceeding as stated in Algorithm A4, line 2. There must be a solution, since we must end up with a symmetric model, and Model I.5b is not symmetric.
- For $j_{min} \leq j_m < j_{mswitch}$, apply Model I.5b; for $j_{mswitch} \leq j_m \leq 0$, Model I.4b.

If situation (c) applies, perform the following:

- In Model I.7b, set $v_m = v_{max}$ and $\Delta t_2 = 0$ (i.e., the phase of constant minimal jerk has duration zero), and compute j_m according to Table A2. Substitute the value in the formula given for Δt_4 (Model I.7b), and check if $\Delta t_4 \geq 0$. If this is the case, set $j_{mswitch1} = j_m$, and the next model is I.6b. Otherwise, set $\Delta t_4 = 0$ in Model I.7b (i.e., the phase of constant maximal acceleration has duration zero), and compute j_m according

to Algorithm A4, line 10, also denoted by $j_{mswitch1}$. If there is a real solution, the next model is I.5b. Since Model I.7b is not symmetric, a valid $j_{mswitch1}$ must exist.

- If Model I.6b is next, set $v_m = v_{max}$ and $\Delta t_3 = 0$, and compute j_m according to Algorithm A4, line 15, denoted by $j_{mswitch2}$. This need not exist.
- If Model I.5b is next, set $v_m = v_{max}$ and $\Delta t_2 = 0$, and compute j_m according to Algorithm A4, line 2, denoted by $j_{mswitch2}$. This, then, must exist, since I.5b is not symmetric.
- Now, for $j_{min} \leq j_m < j_{mswitch1}$, take Model I.7b. For $j_{mswitch1} \leq j_m < j_{mswitch2}$, apply Model I.6b or I.5b, resp. If no valid $j_{mswitch2}$ exists, apply Model I.6b for $j_{mswitch1} \leq j_m \leq 0$. Otherwise, for $j_{mswitch2} \leq j_m \leq 0$, Model I.4b is suitable.

Now, for all triples $(v_m, 0, j_m)$, we can determine the model to be applied in Case I when $S(v_A, v_E) \leq \Delta s \leq S(v_A, v_{max}) + S(v_{max}, v_E)$ holds. Moreover, one can stay at $(v_{max}, 0, 0)$ for a period of arbitrary length, which can easily be included between the going back and the going forth part.

When $\Delta s < S(v_A, v_E)$, we can take intermediate velocities that lie left of v_A , and by reflection and reversion, we can rewrite the problem as one where $\Delta s \geq S(v_A, v_E)$: Assume, v_A, v_E (still with $v_A \leq v_E$), and Δs are given where $\Delta s < S(v_A, v_E)$ holds. Then, consider the reflected configuration (denoted by a bar): $\bar{v}_A = -v_E, \bar{v}_E = -v_A$, and $\Delta \bar{s} = -\Delta s$. Since $\Delta \bar{s} \geq S(\bar{v}_A, \bar{v}_E)$, we can perform the procedure explained above. Then, we reverse the solution by reflecting it again, i.e., we consider

$$j(t) = -\bar{j}(t_{end} - t), a(t) = \bar{a}(t_{end} - t), v(t) = -\bar{v}(t_{end} - t), s(t) = \bar{s}(t_{end} - t) - s(t_{end}) \quad (15)$$

Figure 10 illustrates this (we omit the boundary and color coding for simplicity reasons). The left magenta part shows the solution to the reflected problem, and then by reflection (and thus reversion), we obtain the solution to the original problem in black. Thus, we cover all possible distances.

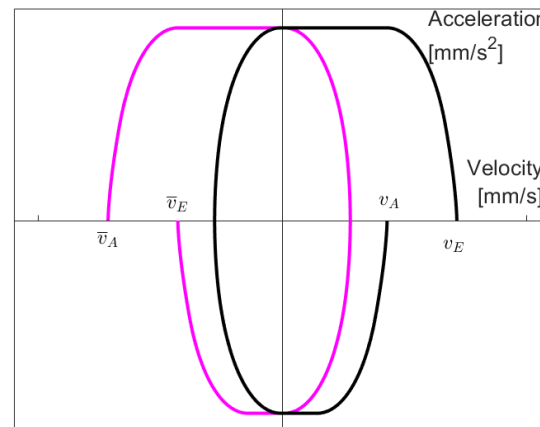


Figure 10. Reflection of the problem and of the solution to the reflected problem.

4. Models and Critical Velocities and Jerks in Case II

In Case II, we proceed analogously to Case I. Again, we take an intermediate velocity v_m and a corresponding jerk j_m and go “forth” from $(v_A, 0, 0)$ to $(v_m, 0, j_m)$ and “back” from $(v_m, 0, j_m)$ to $(v_E, 0, 0)$. Figure 11 shows possible jerk models for going “forth” (the notation “II.xf” describes these models in Case II) using maximal, minimal or zero snap. In Case II, there are three interesting switching points: j_{min} is reached when going down from the maximal acceleration; j_{max} resp. a_{max} is reached when going up from $(v_A, 0, 0)$. Note that Models II.1f, II.2f and II.3f are identical to Models I.1f, I.2f and I.5f. In the other models, the maximum jerk j_{max} is reached, which is not possible in Case I.

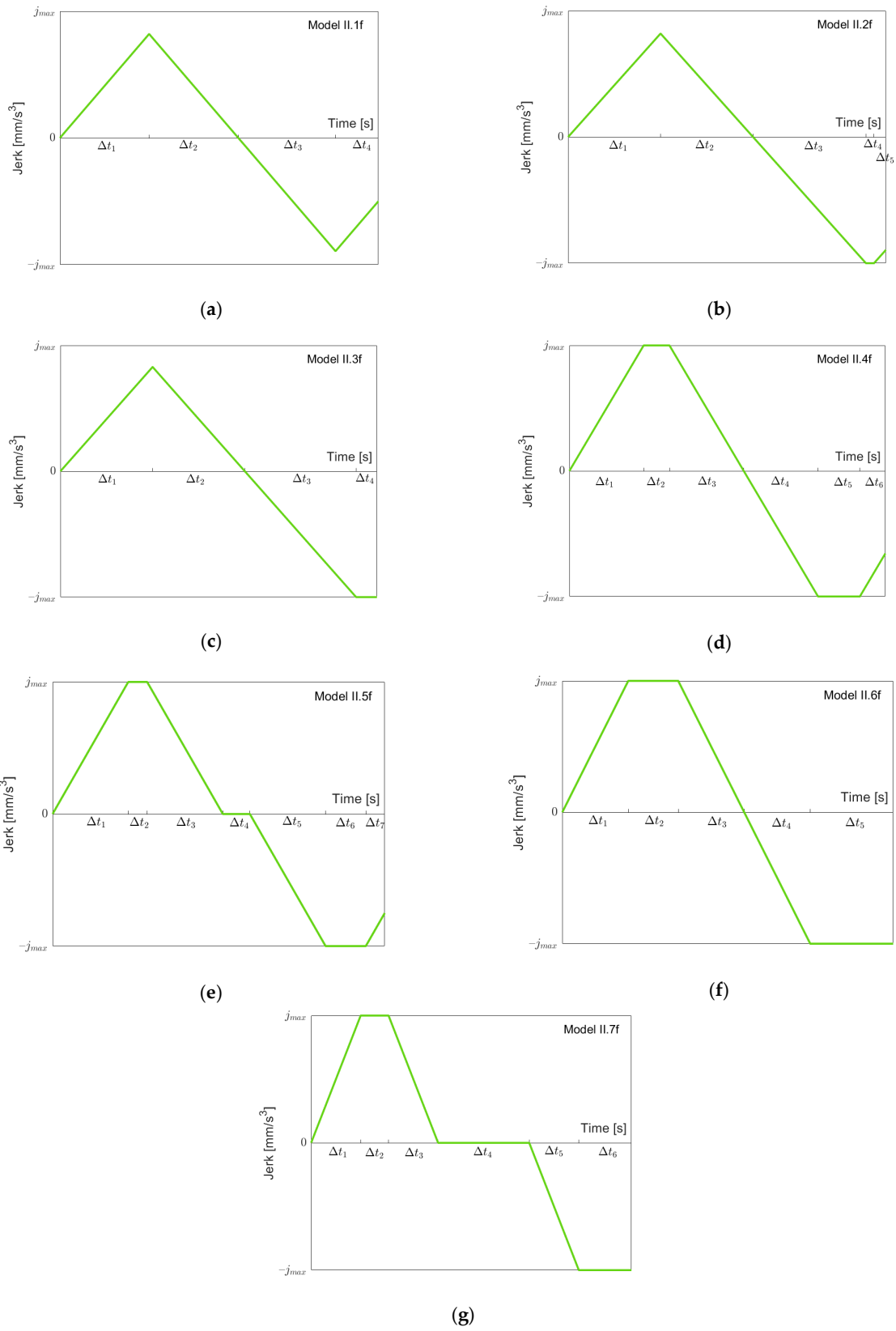


Figure 11. Jerk models in Case II going forth. (a) j_{min} , j_{max} and a_{max} not reached; (b) j_{min} reached, j_{max} , a_{max} not reached, $j_m > j_{min}$; (c) j_{min} reached, j_{max} , a_{max} not reached, $j_m = j_{min}$; (d) j_{min} and j_{max} reached, $j_m > j_{min}$; a_{max} not reached; (e) j_{min} , j_{max} and a_{max} reached, $j_m > j_{min}$; (f) j_{min} and j_{max} reached, a_{max} not reached; $j_m = j_{min}$. (g) j_{min} , j_{max} and a_{max} reached, $j_m = j_{min}$.

If one of these models is suitable for going from $(v_A, 0, 0)$ to $(v_m, 0, j_m)$, one has to compute the length of the time intervals Δt_i ($i = 1 \dots 4, 5, 6$ or 7). Again, the conditions stated in Equation (9) for Case I carry over to Case II. Further conditions can be identified by looking at the special shape of a jerk model. If an interval starts or ends at j_{max} or j_{min} , it has to have a length $\frac{j_{max}}{sn_{max}}$. In Models II.1f, II.2f and II.3f, the first two intervals have the same length. Moreover, in Models II.5f and II.7f, $a(t_3) = a_{max}$. It is easy to check that these conditions provide the required number of equations for determining the Δt_i in the respective model. The results are given in Appendix A, Table A3.

Figure 12 depicts the possible models for going “back” from $(v_m, 0, j_m)$ to $(v_E, 0, 0)$ in Case II. Models II.1b and II.2b are identical to I.1b resp. I.2b. In Model II.3b, j_{max} is reached, and finally, a_{min} is reached in Model II.4b. We have a simple sequence of the four models, as shown in Figure 12a–d, when we proceed from $(v_E, 0, 0)$ to $(v_{max}, 0, vm2jm(v_{max}))$. Note that not all models must be present depending on the value of v_{max} . When we then go further from $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$, one of the Models II.5b to II.8b might occur (Figure 6e–h). These models simply add a “front piece” starting at j_m . When reaching $(v_{max}, 0, 0)$, the model must be symmetric.

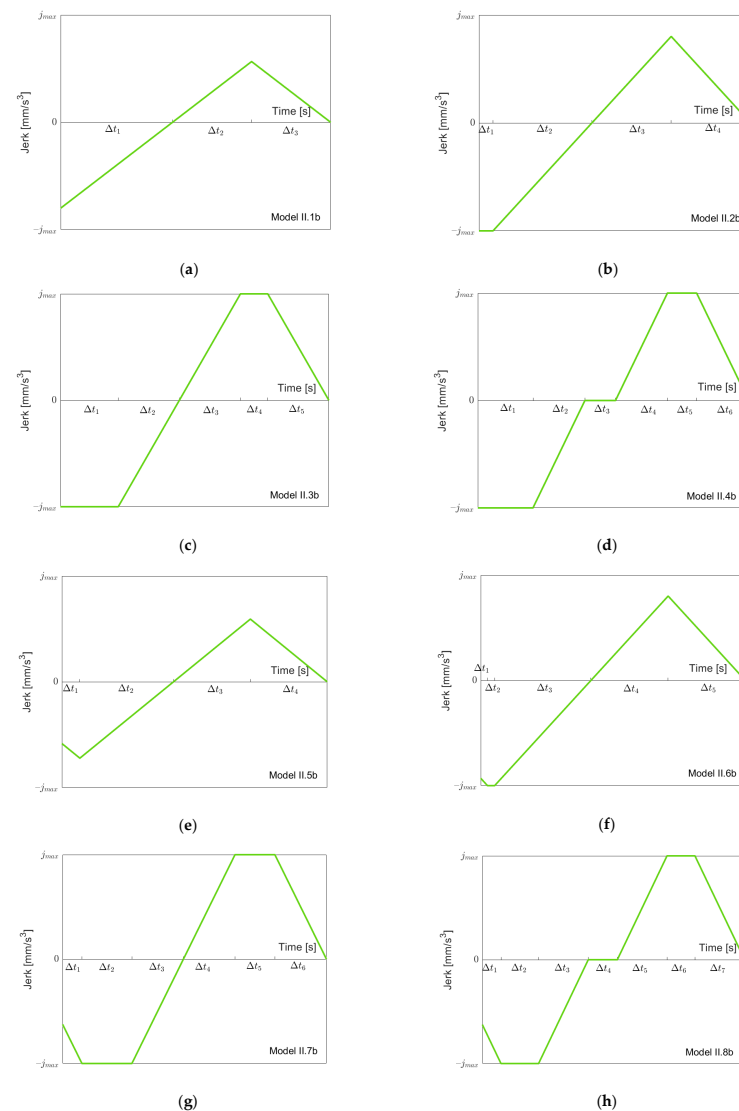


Figure 12. Jerk models in Case II going back. (a) $j_m > j_{min}$; (b) $j_m = j_{min}$, a_{min} not reached; (c) $j_m = j_{min}$, j_{max} reached, a_{min} not reached; (d) $j_m = j_{min}$, j_{max} and a_{min} reached; (e) $v_m = v_{max}$, j_{min} , j_{max} , a_{min} not reached; (f) $v_m = v_{max}$, j_{min} reached, j_{max} , a_{min} not reached; (g) $v_m = v_{max}$, j_{min} and j_{max} reached, a_{min} not reached; (h) $v_m = v_{max}$, j_{min} , j_{max} and a_{min} reached.

For computing the length of the time intervals, as in Case I, Equation (14) provides some conditions. Further necessary conditions can be derived similarly, as stated for the going forth part in a straightforward manner, so we omit this. The results are given in Appendix A, Table A4.

Next, we address the question of in which possible sequences these models occur when $(v_m, 0, j_m)$ runs from $(v_E, 0, 0)$ to $(v_{max}, 0, vm2jm(v_{max}))$. For this, we consider the critical velocities starting with the going forth situation. The value of v_m for which the minimal jerk reaches the value $j_{min} = -j_{max}$ is denoted by v_{mjminA} ; the value for which the maximal jerk reaches j_{max} is denoted by v_{mjmaxA} . Moreover, the intermediate velocity at which a_{max} is reached is called v_{mamaxA} . Case II is simpler than Case I in that we always have the following inequality:

$$v_{mjminA} \leq v_{mjmaxA} \leq v_{mamaxA} \quad (16)$$

Using an analogous notation, we obtain the following for the going back situation:

$$v_{mjminE} \leq v_{mjmaxE} \leq v_{maminE}. \quad (17)$$

The algorithm and formulae for finding these velocities are given in Appendix B, Algorithm A2 and Table A6. In Case II, the line of argumentation is similar to that in Case I. Equations (16) and (17) make the model sequencing very simple depending on where v_{max} lies. Figure 13 shows, for the going forth part, the possible positions of v_{max} and the corresponding models that hold up to v_{max} . As in Case I, within the second, third and forth interval, the model changes (to II.3f, II.6f, II.7f, resp.) if v_{mjminE} is reached. Additionally, as in Case I, the first model to be applied when going from $(v_E, 0, 0)$ to $(v_{max}, 0, vm2jm(v_{max}))$ need not be II.1f, since v_E might, for example, already be identical to v_{mamaxA} if the difference between v_E and v_A is large enough. Figure 14 shows the same for the going back part.

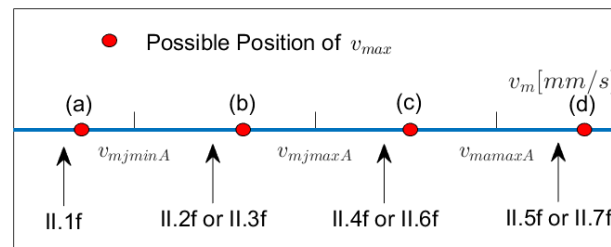


Figure 13. Possible positions of v_{max} and models for “going forth”.

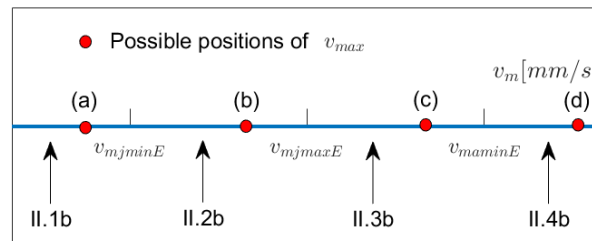


Figure 14. Possible positions of v_{max} and models for “going back”.

Finally, we determine the applicable model when we go from $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$. First, we consider the going forth situation. As in Case I, we have to identify the critical jerk values j_m where a model switch takes place. Our procedure is similar to that conducted in Case I. We refer to Figure 13. If the position of v_{max} is as marked as (a) there is no switch, i.e., the model is still II.1f. If we have position (b), we proceed as follows:

- In Model II.1f, set $v_m = v_{max}$ and $j(t_3) = -j_{max}$, and compute j_m by proceeding as stated in Algorithm A5, line 2, denoted by $j_{mswitch}$. There must be a solution, since we must end up with a symmetric model, and Model II.2f is not symmetric.
- For $vm2jm(v_{max}) \leq j_m < j_{mswitch}$, apply Model II.2f; for $j_{mswitch} \leq j_m \leq 0$, Model II.1f. If situation (c) applies, the procedure is as follows:
- In Model II.4f, set $v_m = v_{max}$ and $\Delta t_2 = 0$ (i.e., the phase of constant maximal jerk has duration zero), and compute j_m by proceeding as stated in Algorithm A5, line 5, denoted by $j_{mswitch1}$ in case of existence. If there is such a solution, the next model is II.2f. If not, Model II.4f is suitable for all jerk values.
- If Model II.2f is next, set in this model $v_m = v_{max}$ and $\Delta t_4 = 0$, and compute j_m by proceeding as stated in Algorithm A5, line 2, denoted by $j_{mswitch2}$. This, then, must exist, since Model II.2f is not symmetric. The next model is II.1f. So, for $vm2jm(v_{max}) \leq j_m < j_{mswitch1}$, take Model II.4f; for $j_{mswitch1} \leq j_m < j_{mswitch2}$, apply Model II.2f; and for $j_{mswitch2} \leq -j_m \leq 0$, Model I.1f.

If situation (d) applies, perform the following:

- In Model II.5f, set $v_m = v_{max}$ and $\Delta t_4 = 0$ (i.e., the phase of constant maximal acceleration has duration zero). Compute j_m by proceeding as stated in Algorithm A5, line 13, denoted by $j_{mswitch1}$. If there is such a solution, the next model is II.4f. Otherwise, Model II.5f is suitable for all jerk values.
- If Model II.4f is next, set $v_m = v_{max}$ and $\Delta t_2 = 0$ in this model, and compute j_m by proceeding as stated in Algorithm A5, line 5, denoted by $j_{mswitch2}$, in case of existence. If there is such a solution with $j_{mswitch1} \leq j_{mswitch2} \leq 0$, the next model is II.2f. If not, apply Model II.5f for $vm2jm(v_{max}) \leq j_m < j_{mswitch1}$ and Model II.4f for $j_{mswitch1} \leq j_m \leq 0$.
- If Model II.2f is next, set $v_m = v_{max}$ and $\Delta t_4 = 0$ in this model, and compute j_m by proceeding as stated in Algorithm A5, line 2, denoted by $j_{mswitch3}$. This, then, must exist, since II.2f is not symmetric. Now, for $vm2jm(v_{max}) \leq j_m < j_{mswitch1}$, apply Model II.5f; for $j_{mswitch1} \leq j_m < j_{mswitch2}$, Model II.4f; for $j_{mswitch2} \leq j_m < j_{mswitch3}$, Model II.2f; and for $j_{mswitch3} \leq j_m \leq 0$, Model II.1f.

Next, we consider the going back situation and refer to Figure 14. If we have situation (a), Model II.5b is suitable for all jerk values. Note that, in situation (b–d), we have $vm2jm(v_{max}) = j_{min}$, which makes the inequalities a bit more compact. If situation (b) applies, we proceed as follows:

- In Model II.6b, set $v_m = v_{max}$ and $\Delta t_2 = 0$, and compute j_m by proceeding as stated in Algorithm A6, line 2, denoted by $j_{mswitch}$. There must be a solution, since we must end up with a symmetric model, and Model II.6b is not symmetric.
- For $j_{min} \leq j_m < j_{mswitch}$, apply Model II.6b; for $j_{mswitch} \leq j_m \leq 0$, Model II.5b.

If situation (c) applies, perform the following:

- In Model II.7b, set $v_m = v_{max}$ and $\Delta t_5 = 0$ (i.e., the phase of constant maximal jerk has duration zero). Compute j_m by proceeding as stated in Algorithm A6, line 5, denoted by $j_{mswitch1}$, in case of existence. If there is such a solution, the next model is II.6b. Otherwise, Model II.7b is suitable for all jerk values.
- If Model II.6b is next, set $v_m = v_{max}$ and $\Delta t_2 = 0$, and compute j_m by proceeding as stated in Algorithm A6, line 2, denoted by $j_{mswitch2}$. This, then, must exist, since Model II.6b is not symmetric. The next model then is II.5b. For $j_{min} \leq j_m < j_{mswitch1}$, apply Model II.7b; for $j_{mswitch1} \leq j_m < j_{mswitch2}$, Model II.6b; and for $j_{mswitch1} \leq j_m \leq 0$, Model II.5b.

In situation (d), we can perform the procedure of situation (c):

- In Model II.8b, set $v_m = v_{max}$ and $\Delta t_4 = 0$, and compute j_m by proceeding as stated in Algorithm A6, line 13, denoted by $j_{mswitch1}$. If this does not exist, Model II.8b is suitable for all jerk values.

- If it does exist, the next model is II.7b, and we proceed as in (c), obtaining a total of up to three critical jerk values.

The remarks concerning the situation $\Delta s < S(v_A, v_E)$ given for Case I also apply to Case II.

5. Implementation and Validation

We assume that the following data are given:

- The kinematic restrictions v_{max} , a_{max} , j_{max} , sn_{max} .
- The initial and final velocities v_A , v_E , where $v_A \leq v_E$.
- The distance $\Delta s \geq S(v_A, v_E)$.

The main task now consists of finding the intermediate velocity and corresponding jerk $(v_m, 0, j_m)$, such that the covered distance equals Δs where $(v_m, 0, j_m)$ “flows” from $(v_E, 0, 0)$ via $(v_{max}, 0, vm2jm(v_{max}))$ to $(v_{max}, 0, 0)$ and stays there for an arbitrary period of time. Algorithm 1 sketches how this task can be achieved. Briefly explained, the algorithm determines all critical velocities and computes for each of them the distance w covered using the appropriate models. If the given distance Δs is exceeded, one knows the interval of critical velocities (including v_E or v_{max}) within which v_m must lie. A simple bisection method can then be applied to obtain a guaranteed solution. If Δs even exceeds the distance one obtains for $(v_{max}, 0, vm2jm(v_{max}))$, then the critical jerks are determined. These might be differently sequenced, e.g., if we have $j_{mswitch_f}$ going forth and $j_{mswitch_b}$ going back, the ordering might be $j_{mswitch_f} \leq j_{mswitch_b}$ or the other way around. For each possible ordering, and for each critical jerk from $vm2jm(v_{max})$ to 0, the covered distance is computed using the appropriate models. If the given distance Δs is exceeded, one knows the interval of critical jerks (including 0 and $vm2jm(v_{max})$) within which j_m must lie. Again, a simple bisection method can then be applied to obtain a guaranteed solution. If Δs even exceeds the distance one obtains for $(v_{max}, 0, 0)$ as intermediate velocity and jerk, one takes the models to be applied for $(v_{max}, 0, 0)$ and inserts a segment with constant maximal velocity, such that $\Delta s = w$.

We implemented the main algorithm and the auxiliary algorithms given in Appendices B and C as Matlab[®] functions using the Matlab[®] programming language. In these functions, the only calls to genuine mathematical facilities offered by Matlab[®] are the ones to Matlab’s polynomial solver called “roots”. Therefore, when implementing the algorithms in another environment, only a polynomial solver needs to be programmed or imported from a mathematics library.

We set up the following test scenario in order to check whether viable results are provided:

- v_{max} , a_{max} , j_{max} , sn_{max} are chosen randomly between 5 and 100 ($\frac{mm}{s}$, $\frac{mm}{s^2}$, $\frac{mm}{s^3}$ and $\frac{mm}{s^4}$, respectively; in the sequel, we will omit the units).
- v_A is chosen randomly between v_{min} and v_{max} , v_E between v_A and v_{max} .
- Δs is chosen randomly between $S(v_A, v_E)$ and $S(v_A, v_{max}) + S(v_{max}, v_E)$.

We generated 10^8 test configurations and checked and stopped the bisection iteration when the resulting distance was reached with a maximal error of 10^{-3} (i.e., 1 μm). We also checked the error regarding v_E and $a(t_{end}) = 0$. In addition, we counted whether all possible combinations for positioning v_{max} depicted in Figures 7 and 8 for Case I and in Figures 13 and 14 for Case II are covered, as well as all feasible combinations of numbers of $j_{mswitch}$ values. Tables 3 and 4 provide the values for Case I; Tables 5 and 6 those for Case II. Those entries that are marked with a star are not feasible. We will demonstrate the reasoning behind this by giving two examples:

- When, in Case I, “going forth”, we have situation (c) (see Figure 7), then, because of $v_{mjminA} \leq v_{mjminE}$ (see Section 3), only situation (a) is possible for “going back”.
- Assume, in Case II, that we have two $j_{mswitch}$ values when “going forth”, so we must have situation (c). Then, three $j_{mswitch}$ values for going back are only possible when we have situation (d) for “going back”, but then, because of $v_{mamaxA} \leq v_{maminE}$, we must also have situation (d) for “going forth”, which is a contradiction. Note that not all

values in the upper triangle of Table 6 need to be zero: for example, we can have the combination 0–1 in case of situation (c) “going forth” and situation (b) “going back”.

Algorithm 1. Main Algorithm.

Input: $v_A, v_E, v_{max}, a_{max}, j_{max}, sn_{max}, \Delta s$

Output: $(v_m, 0, j_m)$, models going forth and going back, corresponding Δt_i

```

1: Determine the critical velocities using the algorithms/formulae presented in Appendix B
2: Determine the situation regarding the positioning of  $v_{max}$ 
3: Determine  $S(v_A, v_E)$  according to Equation (1) and Table 1
4: If  $\Delta s = S(v_A, v_E)$  Then
5:    $(v_m, 0, j_m) = (v_E, 0, 0)$ 
6:   Determine the going forth model according to Table 1 and the  $\Delta t_i$  according to Table A1 resp. A3
7: Else
8:   For Each critical velocity from  $(v_E, 0, 0)$  to  $(v_{max}, 0, vm2jm(v_{max}))$  (including  $v_{max}$ )
9:     Compute the distance  $w$  reached when going via this intermediate velocity from  $(v_A, 0, 0)$  to  $(v_E, 0, 0)$ 
10:    If  $\Delta s > w$  Then
11:      Pick next critical velocity (or  $v_{max}$ )
12:    Else
13:       $v_m$  must lie between the critical velocity reached before (or  $v_E$ ) and the actual critical velocity.
14:      Determine the models going forth and going back according to Figures 7 and 8 resp. 13, 14.
15:      Apply a bisection method for determining  $v_m$  and compute  $\Delta t_i$  according to Appendix A.
16:    Return
17:  End If
18: End For Each % End to line 8
19: Determine the critical jerks using the algorithms presented in Appendix C
20: For Each possible sequence of critical jerks
21:   For Each critical jerk from  $vm2jm(v_{max})$  to 0 (including 0)
22:     Compute the distance  $w$  reached when going from  $(v_A, 0, 0)$  to
      $(v_E, 0, 0)$  via  $(v_{max}, 0, j_{mswitch})$ 
23:     If  $\Delta s > w$  Then
24:       Pick next critical jerk (or 0)
25:     Else
26:        $j_m$  must lie between the critical jerk reached before (or  $vm2jm(v_{max})$ ) and the actual critical
       jerk, and the models to be applied are known according to the algorithms in Appendix C.
27:       Apply a bisection method for determining  $j_m$  and compute  $\Delta t_i$  according to Appendix A.
28:     Return
29:   End If
30: End For Each % End to line 21
31: End For Each % End to line 20
32: % now  $\Delta s > S(v_A, v_{max}) + S(v_{max}, v_E)$  must hold
33: Return  $(v_{max}, 0, 0)$  and the models to be applied for  $(v_{max}, 0, 0)$ , compute  $\Delta t_i$  according to Appendix A,
    and insert a segment with constant maximal velocity such that  $\Delta s = w$ .
34: End If % End to line 4
35: Return

```

Table 3. Coverage of situations in Case I.

	I.b (a)	I.b (b)	I.b(c)
I.f (a)	68,162,974	*	*
I.f (b)	2,565,497	3,135,975	*
I.f (c)	4,472,190	*	*
I.f (d)	689,239	2,251,849	4,733,898
I.f (e)	2,564,185	2,123,533	9,246,440

The left column contains the different situations for positioning v_{max} for going forth; the upper row contains the different situations for positioning v_{max} when going back. An asterisk * denotes that the combination is not possible.

Table 4. Coverage of combinations of $j_{mswitch}$ values for $v_m = v_{max}$ in Case I.

# $j_{mswitch}$	0	1	2
0	2,044,3058	*	*
1	1,598,261	4,048,933	77,431
2	39,112	54,537	3063

The left column contains the number of $j_{mswitch}$ values for going forth; the upper row contains the number of $j_{mswitch}$ values for going back (# stands for number). An asterisk * denotes that the combination is not possible.

Table 5. Coverage of situations in Case II.

	II.b (a)	II.b (b)	II.b (c)	II.b (d)
II.f (a)	21,589,590	*	*	*
II.f (b)	4,578,466	6,321,463	*	*
II.f (c)	4,569,027	8,493,669	33,378,435	*
II.f (d)	1,156,690	2,154,669	6,511,882	11,191,542

The left column contains the different situations for positioning v_{max} for going forth; the upper row contains the different situations for positioning v_{max} when going back. An asterisk * denotes that the combination is not possible.

Table 6. Coverage of combinations of $j_{mswitch}$ values for $v_m = v_{max}$ in Case II.

# $j_{mswitch}$	0	1	2	3
0	13,516,719	2,448,930	72,459	236
1	1,291,857	1,620,708	346	3
2	33,538	57,046	1202	*
3	145	291	3	2

The left column contains the number of $j_{mswitch}$ values for going forth; the upper row contains the number of $j_{mswitch}$ values for going back (# stands for number). An asterisk * denotes that the combination is not possible.

The data presented in Tables 3–6 demonstrate a satisfactorily coverage of all possible combinations of situations and numbers of $j_{mswitch}$ values.

Table 7 shows the maximum error regarding distance, final velocity and final acceleration, as well as the mean and maximum number of iterations performed by the bisection method in Cases I and II. The maximum error regarding distance is due to our stopping criterion for the bisection method. All errors regarding the final velocity above 10^{-5} were recorded. The number of occurrences was below 10, and the maximum value is given in the table. No errors for $a(t_{end}) = 0$ larger than 10^{-12} were recorded. It is clear that the boundary set for the maximum admissible error in Δs has a considerable influence on the number of iterations performed by the bisection method, but one cannot relate the values directly, since the bisection is not performed for distance intervals but for velocity or jerk intervals.

Table 7. Maximum error and number of iterations in Cases I and II.

	Case I	Case II
Error Δs (mm)	$\leq 10^{-3}$	$\leq 10^{-3}$
Error v_E (mm/s)	$\leq 1.62 \times 10^{-5}$	$\leq 1.5 \times 10^{-5}$
Error $a(t_{end})$ (mm/s ²)	$\leq 10^{-12}$	$\leq 10^{-12}$
Mean number of iterations	13.97	13.75
Standard deviation	2.67	2.72
Maximum number of iterations	48	48

All runs were completed successfully. In initial trials, occasionally, a run did not succeed because of numerical problems. The latter were removed by applying the following measures:

- If certain velocity values are very close to each other (i.e., $< 10^{-5}$), they are set as equal, e.g., if $v_E \approx v_A$, set $v_E = v_A$; if $v_E \approx v_{max}$, set $v_E = v_{max}$; if $v_m \approx v_E$, set $v_m = v_E$.
- If in the bisection method, the middle of the considered interval coincides with a boundary because of rounding, bisection is stopped, and the middle is taken.
- If $v_E = v_A$, models I.1f and II.1f are simplified by removing the fourth time interval, which must be zero.
- When finding and checking the zeros of polynomials, small imaginary parts are neglected, and inequalities and equations of solutions are checked using an admissible error margin.

To be suitable for online trajectory generation, it should be possible to complete the computation within one controller cycle. As in [4] (cf. the discussion presented there), we use 1ms as length of such a cycle. Time data were generated using the respective “tic” and “toc” facilities offered by Matlab®, which start and stop a timer and provide information on the time spent between the calls. Since the time needed by Matlab® varies considerably when one makes several computations using the same data set, we apply the same approach as in [4]: for each random configuration, we make 1000 runs and take the minimum time. We randomly generated 100.000 configurations. The time data are given in Table 8 for Cases I and II; the underlying hardware consisted of a desktop PC running Windows 10 with a Xeon E5-1630v4 processor working at 3.7 GHz and 32GB RAM. Although the mean values and standard deviations suggest that, in general, a boundary of 1ms can be achieved, the maximum values show that, occasionally, computation might take longer. Using the profiling tool in Matlab®, one can see that finding the roots of polynomials takes about 50% of the overall computation time. In the bisection iterations, only polynomials of degree 4 occur. Whereas Matlab® implements a general polynomial solver, there are special ones for degree four that are very efficient, needing only 0.5 μ s per run (cf. the discussion and references in [4]). Even for 50 bisection iterations in models I.1f and I.4f resp. II.1f and II.5f, with two polynomials per iteration, this would reduce the time needed considerably, such that completion within 1 ms can be achieved (see also the discussion on code optimization in [4]).

Table 8. Computational effort in Cases I and II.

	Case I	Case II
Mean time per run	276 μ s	154 μ s
Standard deviation	144 μ s	140 μ s
Maximum time	831 μ s	732 μ s

6. Numerical Comparison with Related Work

In this section, we compare our results with two approaches from the literature that also apply a fifteen-segment profile with piecewise constant snap and claim to provide a time-optimal solution. For the point-to-point motion (abbreviated as RR, “rest-to-rest”)

with initial and final velocity equal to zero, we consider the algorithm by Biagiotti and Melchiorri [11], and for the constant-velocity-to-constant-velocity situation (abbreviated as GG) with arbitrary initial and final velocities and zero acceleration and jerk, we consider the algorithm by Fan et al. [28], who assume zero jerk at the intermediate velocity v_m . Ezair et al. [34] make the same assumption and hence should provide identical results, but they explicitly recognize the sub-optimality.

The algorithm presented by Biagiotti/Melchiorri (“CheckConstraintsT1” in [11]) is based upon rectangular filter functions and has four time parameters $T_1, \dots, T_4$ in case of piecewise constant snap functions. It is strikingly simple and hence seems to contradict the complexity of the situation as presented in this contribution. We first compare the results of a configuration of Case I: $v_A = v_E = 0$, $v_{max} = 30$, $a_{max} = 10$, $j_{max} = 12$, $sn_{max} = 10$. We let Δs run from $S(v_A, v_E)$ to $S(v_A, v_{max}) + S(v_{max}, v_E)$, compute $T_1, \dots, T_4$ according to [11] and sum up to obtain the overall time t_{sum} . Then, we compute the time intervals Δt_i as specified in this article and sum up to obtain the overall time t_{sum} for our approach. Figure 15a depicts the two quite similar curves, but the plot of the improvement of our approach compared to [11] in Figure 15b (in %) reveals that, over the whole range of distances, there is a reduction in time, which goes up to about 1%.

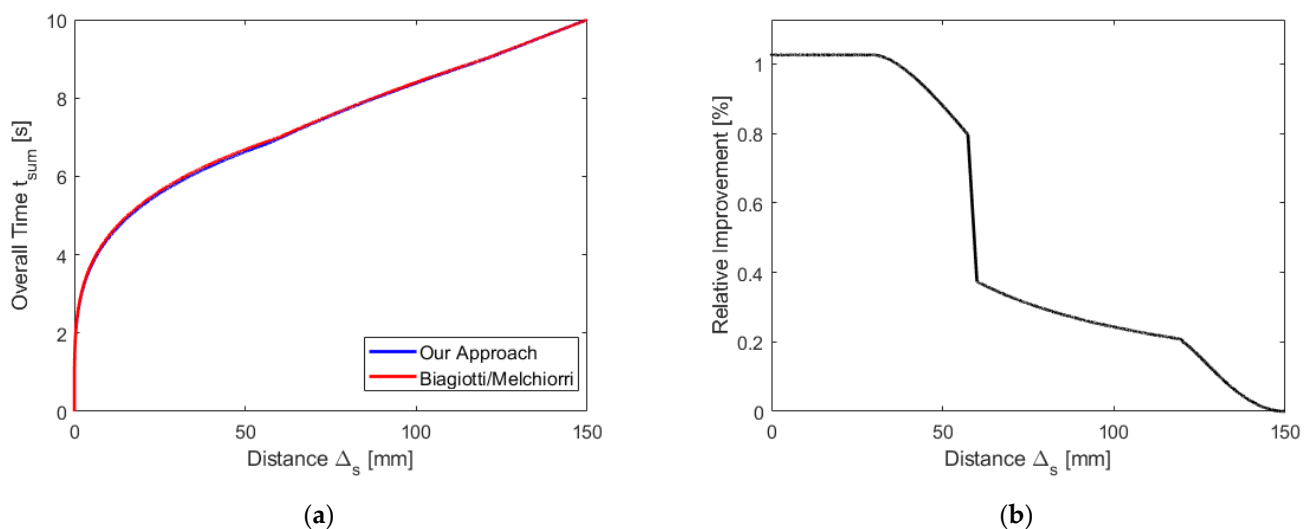


Figure 15. (a) Overall time needed for going over the distance Δs in [11] and in our approach. (b) Relative gain provided by our approach.

In order to better understand this difference, we picked one specific distance $\Delta s = 25$, with which we obtained the following results: the algorithm in [11] results in $T_1 = 2.4103$, $T_2 = 1.6069$, $T_3 = T_4 = 0.80343$, summing up to $t_{sum} = 5.624s$; our algorithm gives as models to be applied I.1f and I.1b with symmetric time interval lists. For model I.1f, we have $\Delta t_1 = \Delta t_2 = 0.8152$, $\Delta t_3 = 1.1528$, $\Delta t_4 = 0$, resulting in $t_{sum} = 5.566s$. We compare the jerk functions provided by both approaches in Figure 16. This shows that our approach takes full advantage of the lower jerk limit $j_{min} = -j_{max}$, whereas the algorithm of Biagiotti and Melchiorri [11] seems to enforce that the maximum and minimum value of the resulting jerk function have the same absolute value. This additional assumption might lead to a sub-optimal result. Actually, in the given configuration, we have Case I, so the maximum and minimum values of the resulting jerk function in our approach will have different absolute values over the full range of considered distances. Hence, our approach always results in a shorter time, as long as $(v_m, 0, j_m)$ has not reached $(v_{max}, 0, 0)$. This shows that the complexity of the situation demonstrated in this article cannot simply be sidestepped. Yet, it is fair to state that the approach by [11] can be extended to also include frequency restrictions, which makes it particularly interesting for vibration reduction.

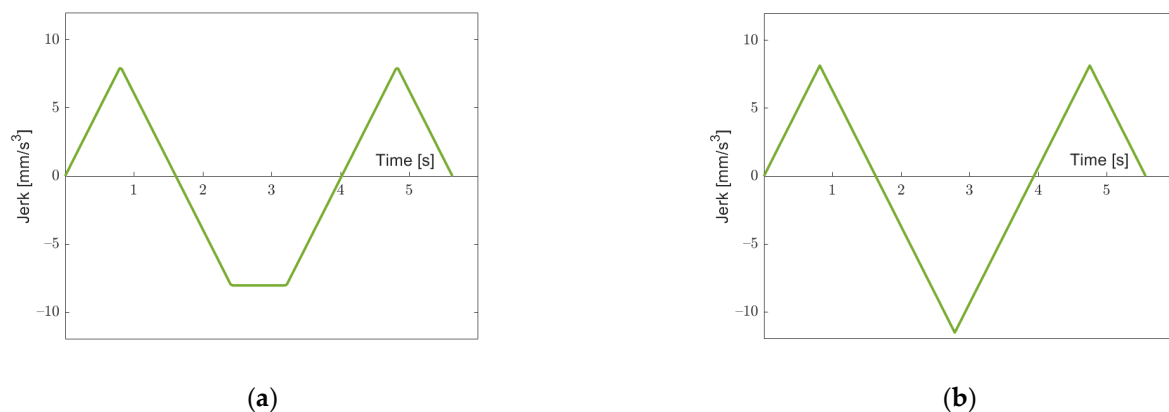


Figure 16. (a) Jerk function provided by [11]. (b) Jerk function and in our approach.

The following configuration serves to compare our approach with the one by Fan et al. [28]: $v_A = -5, v_E = 10, v_{max} = 30, a_{max} = 10, j_{max} = 13, sn_{max} = 10$ (Case I). Again, we let Δs run from $S(v_A, v_E)$ to $S(v_A, v_{max}) + S(v_{max}, v_E)$ and compute the overall time provided by our approach and by Fan et al., which is shown in Figure 17a. Here, the difference is well-recognizable. Figure 17b shows the relative improvement realized by our approach (in %), which goes up to 4.6%. If $v_A = v_E$ holds, we even found examples with a relative gain of up to 7%, which is due to the fact that the “base time” for going from v_A directly to v_E is zero. Numerical experiments also show that if sn_{max} is higher, the gain is reduced, since one can quickly get back from j_{max} to zero.

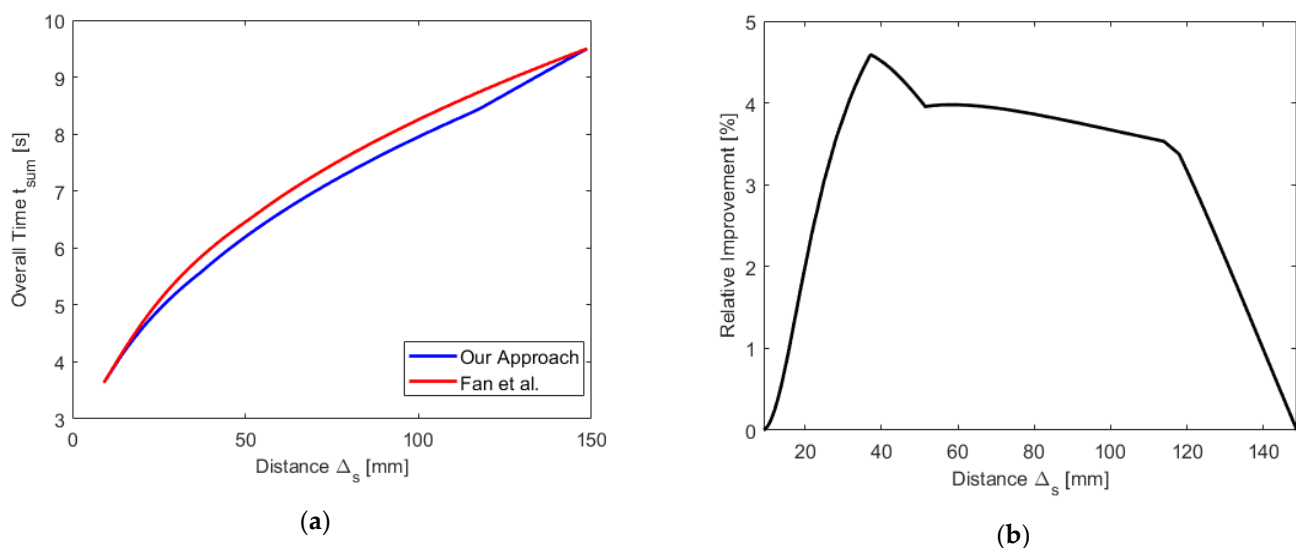


Figure 17. (a) Overall time needed for going over the distance Δs in [28] and in our approach. (b) Relative gain provided by our approach.

Again, it is fair to state that the algorithm by Ezair et al. [34] providing the same results as Fan et al. [28] has a wider range of applicability, since arbitrary boundary values for acceleration and jerk can also be prescribed, and it can be applied to higher orders.

7. Conclusions

In this contribution, we describe how a fast jerk–continuous motion function can be found when restrictions on velocity, acceleration, jerk and snap as well as initial and final values for velocity are given. We apply a fifteen-segment motion function with piecewise constant snap and ensure that, at each point in time, at least one restriction is reached, because of which we call the function fast. Note that, as shown in [4], this does

not guarantee optimality. The research literature published so far has mainly dealt with point-to-point motions, where initial and final velocities are assumed to be zero, and the jerk becomes zero when the velocity takes its extremal value. We drop this assumption, which makes the situation much more complex, as was shown in the previous sections. Yet, the additional complexity is manageable by suitably structuring the problem in the following way:

- We distinguished two major cases depending on whether the maximum /minimum jerk value can be reached when going from acceleration zero to maximum/minimum acceleration.
- We picked an intermediate velocity v_m with a corresponding jerk value and split up the motion in the velocity–acceleration plane into a going forth and a going back part.
- We identified all possible jerk models for going forth and back
- We identified the switching points for models by computing critical velocities and (at $v = v_{max}$) jerks.

By providing the corresponding algorithms, formulae and polynomials, a re-implementation is possible. Our own implementation in Matlab[®] showed that, in large test runs, a solution could always be found, and the complexity is necessary. Moreover, the investigation of time efficiency revealed that the computation should be achievable within a control cycle of 1 ms, where there is still considerable potential for improvement using dedicated quartic solvers.

We compared our approach with other motion functions suggested in the literature that apply the same fifteen-segment profile, since only those approaches allow us to stay at each restriction over a time interval of non-zero length. Numerical experiments revealed that a time improvement of up to 7% could be achieved compared to the algorithm by Fan et al., which also includes the case where initial and final velocities are non-zero. The algorithm by [11] is restricted to the point-to-point motion, and it is the only one that drops the additional assumption that the jerk must be zero when reaching the extremal velocity value. Yet, here, our algorithm showed an improvement of up to 1%, since the algorithm in [11] includes other assumptions, which, for certain configurations, make the algorithm sub-optimal.

The main progress compared to approaches that can be found in the literature can be summarized as follows:

- We take full advantage of restrictions on velocity, acceleration, jerk, and snap, and by dropping any additional assumption reducing optimality we provide the fastest currently available jerk-continuous motion function for rest-to-rest tasks and for tasks where initial and final velocities are given.
- In spite of the complexity caused by dropping any additional assumption our algorithm is still computationally efficient and hence suitable for Online Trajectory Generation.

Our improvement in time efficiency has its price, because it leads to a considerable extension of complexity. It certainly depends on the application whether the shorter processing time is worth this effort. Moreover, our approach has all the limitations that come with the application of fifteen-segment motion functions: the restrictions on jerk and snap are usually not provided in specifications of electronic drives but must be determined experimentally or taken from former experience. Another limitation of our approach consists of not taking into account the dynamics of manipulators, whereas in approaches working with optimization procedures, these can be easily included in the form of restrictions.

Given that the complexity our approach already has for the point-to-point and constant velocity-to-constant velocity task (termed RR and GG above), it might be worth investigating whether a general solution with arbitrary initial and final velocity, acceleration and jerk can be achieved. Further investigation is required to determine whether a reduction in the general case to the situation treated in this contribution is possible, as in [4].

Funding: The article processing charge was funded by the Baden-Württemberg Ministry of Science, Research and Culture and the Aalen University of Applied Sciences in the funding programme Open Access Publishing.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Formulae and Polynomials for Computing Time Intervals

Table A1. Formulae and polynomials for computing time intervals in Case I going forth.

Model	Δt_i	Formulae and Polynomials
I.1f	Δt_1	$\sqrt{\frac{2\Delta t_3^2 \Delta s n_{max}^2 - j_m^2}{2s n_{max}^2}}$
	Δt_3	$\frac{1}{s n_{max}} \cdot \text{roots}([36j_m^2, 48j_m^3 - 144s n_{max}^2 v_A + 144s n_{max}^2 v_m, -18j_m^4, -48j_m^5 + 144j_m^2 s n_{max}^2 v_A - 144j_m^2 s n_{max}^2 v_m, -17j_m^6 + 48j_m^3 s n_{max}^2 v_A - 48j_m^3 s n_{max}^2 v_m - 72s n_{max}^4 v_A^2 + 144s n_{max}^4 v_A v_m - 72s n_{max}^4 v_m^2])$
	Δt_4	$\Delta t_3 + \frac{j_m}{s n_{max}}$
I.2f	Δt_1	$-\frac{j_{max}}{2s n_{max}} + \frac{\sqrt{9j_{max}^2 + 6\sqrt{9j_m^4 + 24j_m^3 j_{max} + 18j_m^2 j_{max}^2 - 72j_{max} s n_{max}^2 v_A + 72j_{max} s n_{max}^2 v_m}}}{6s n_{max}}$
	Δt_3	$\frac{j_{max}}{s n_{max}}$
	Δt_4	$\frac{2\Delta t_1^2 s n_{max}^2 + j_m^2 - 2j_{max}^2}{2s n_{max} j_{max}}$
	Δt_5	$\frac{j_m + j_{max}}{s n_{max}}$
I.3f	Δt_1	$\sqrt{\frac{a_{max}}{s n_{max}}}$
	Δt_3	$\frac{(-6a_{max} s n_{max} + 3j_m^2) \sqrt{4a_{max} s n_{max} + 2j_m^2 - 12s n_{max}^2 a_{max}} \sqrt{\frac{a_{max}}{s n_{max}}} + 12s n_{max}^2 (-v_A + v_m) + 4j_m^3}{12s n_{max}^2 a_{max}}$
	Δt_4	$\Delta t_5 - \frac{j_m}{s n_{max}}$
	Δt_5	$\frac{j_m + \frac{\sqrt{4a_{max} s n_{max} + 2j_m^2}}{2}}{s n_{max}}$
I.4f	Δt_1	$\sqrt{\frac{a_{max}}{s n_{max}}}$
	Δt_3	$\frac{((-24\sqrt{\frac{a_{max}}{s n_{max}}} a_{max} - 24v_A + 24v_m) j_{max} - 12a_{max}^2) s n_{max}^2 - 12a_{max} j_{max}^2 s n_{max} + 3j_m^4 + 8j_m^3 j_{max} + 6j_m^2 j_{max}^2}{24s n_{max}^2 j_{max} a_{max}}$
	Δt_4	$\frac{j_{max}}{s n_{max}}$
	Δt_5	$\frac{2a_{max} s n_{max} + j_m^2 - 2j_{max}^2}{2s n_{max} j_{max}}$
	Δt_6	$\frac{j_m + j_{max}}{s n_{max}}$
I.5f	Δt_1	$-\frac{j_{max}}{2s n_{max}} + \frac{\sqrt{9j_{max}^2 + 6\sqrt{3j_{max}^4 - 72j_{max} s n_{max}^2 v_A + 72j_{max} s n_{max}^2 v_m}}}{6s n_{max}}$
	Δt_3	$\frac{j_{max}}{s n_{max}}$
	Δt_4	$\frac{2\Delta t_1^2 s n_{max}^2 - j_{max}^2}{2s n_{max} j_{max}}$
I.6f	Δt_1	$\sqrt{\frac{a_{max}}{s n_{max}}}$
	Δt_3	$\frac{((-24\sqrt{\frac{a_{max}}{s n_{max}}} a_{max} - 24v_A + 24v_m) j_{max} - 12a_{max}^2) s n_{max}^2 - 12a_{max} j_{max}^2 s n_{max} + j_{max}^4}{24s n_{max}^2 j_{max} a_{max}}$
	Δt_4	$\frac{j_{max}}{s n_{max}}$
	Δt_5	$\frac{2a_{max} s n_{max} - j_{max}^2}{2s n_{max} j_{max}}$

In all models $\Delta t_2 = \Delta t_1$ holds.

Table A2. Formulae and polynomials for computing time intervals in Case I going back.

Model	Δt_i	Formulae and Polynomials
I.1b	Δt_1	$-\frac{j_m}{sn_{max}}$
	Δt_2	$-\sqrt{2}\frac{j_m}{2sn_{max}}$
	Δt_3	Δt_2
I.2b	Δt_1	$\frac{2\Delta t_3^2 sn_{max}^2 - j_{max}^2}{2sn_{max} j_{max}}$
	Δt_2	$\frac{j_{max}}{sn_{max}}$
	Δt_3	$-\frac{j_{max}}{2sn_{max}} + \frac{\sqrt{9j_{max}^2 + 6\sqrt{3}j_{max}^4 - 72j_{max}sn_{max}^2 v_E + 72j_{max}sn_{max}^2 v_m}}{6sn_{max}}$
	Δt_4	Δt_3
	Δt_5	Δt_3
I.3b	Δt_1	$\frac{2a_{max}sn_{max} - j_{max}^2}{2sn_{max} j_{max}}$
	Δt_2	$\frac{j_{max}}{sn_{max}}$
	Δt_3	$\frac{((-24\sqrt{\frac{a_{max}}{sn_{max}}}a_{max} - 24v_E + 24v_m)j_{max} - 12a_{max}^2)sn_{max}^2 - 12a_{max}j_{max}^2sn_{max} + j_{max}^4}{24sn_{max}^2 j_{max} a_{max}}$
	Δt_4	$\sqrt{\frac{a_{max}}{sn_{max}}}$
	Δt_5	Δt_4
I.4b	Δt_1	$\frac{j_m + \frac{\sqrt{4\Delta t_3^2 sn_{max}^2 + 2j_m^2}}{2}}{sn_{max}}$
	Δt_2	$\Delta t_1 - \frac{j_m}{sn_{max}}$
	Δt_3	$\frac{1}{sn_{max}} \cdot \text{roots}([36j_m^2, -48j_m^3 + 144sn_{max}^2 v_E - 144sn_{max}^2 v_m, 18j_m^4, 0, -j_m^6 - 48j_m^3 sn_{max}^2 v_E + 48j_m^3 sn_{max}^2 v_m + 72sn_{max}^4 v_E^2 - 144sn_{max}^4 v_E v_m + 72sn_{max}^4 v_m^2])$
	Δt_4	Δt_3
	Δt_5	Δt_3
I.5b	Δt_1	$\frac{j_m + j_{max}}{sn_{max}}$
	Δt_2	$\frac{2\Delta t_4^2 sn_{max}^2 + j_m^2 - 2j_{max}^2}{2sn_{max} j_{max}}$
	Δt_3	$\frac{j_{max}}{sn_{max}}$
	Δt_4	$-\frac{j_{max}}{2sn_{max}} + \frac{\sqrt{9j_{max}^2 + 6\sqrt{9}j_m^4 + 24j_m^3 j_{max} + 18j_m^2 j_{max}^2 - 72j_{max}sn_{max}^2 v_E + 72j_{max}sn_{max}^2 v_m}}{6sn_{max}}$
	Δt_5	Δt_4
I.6b	Δt_1	$\frac{j_m + \frac{\sqrt{4a_{max}sn_{max} + 2j_m^2}}{2}}{sn_{max}}$
	Δt_2	$\Delta t_1 - \frac{j_m}{sn_{max}}$
	Δt_3	$\frac{(6a_{max}sn_{max} - 3j_m^2)\sqrt{4a_{max}sn_{max} + 2j_m^2} - 12sn_{max}^2 a_{max}\sqrt{\frac{a_{max}}{sn_{max}}} + 12sn_{max}^2(-v_E + v_m) + 4j_m^3}{12sn_{max}^2 a_{max}}$
	Δt_4	$\sqrt{\frac{a_{max}}{sn_{max}}}$
	Δt_5	Δt_4
I.7b	Δt_1	$\frac{j_m + j_{max}}{sn_{max}}$
	Δt_2	$\frac{2a_{max}sn_{max} + j_m^2 - 2j_{max}^2}{2sn_{max} j_{max}}$
	Δt_3	$\frac{j_{max}}{sn_{max}}$
	Δt_4	$\frac{((-24\sqrt{\frac{a_{max}}{sn_{max}}}a_{max} - 24v_E + 24v_m)j_{max} - 12a_{max}^2)sn_{max}^2 - 12a_{max}j_{max}^2sn_{max} + 3j_m^4 + 8j_m^3 j_{max} + 6j_m^2 j_{max}^2}{24sn_{max}^2 j_{max} a_{max}}$
	Δt_5	$\sqrt{\frac{a_{max}}{sn_{max}}}$
	Δt_6	Δt_5

Table A3. Formulae and polynomials for computing time intervals in Case II going forth.

Model	Δt_i	Formulae and Polynomials
II.1f		Same as Model I.1f
II.2f		Same as Model I.2f
II.3f		Same as Model I.5f
II.4f	Δt_1	$\frac{j_{max}}{sn_{max}}$
	Δt_2	$\frac{-3j_{max}^2 + \sqrt{18j_m^3 + 48j_m^3 j_{max} + 36j_m^2 j_{max}^2 + 36j_m^4 j_{max} - 144j_{max} sn_{max}^2 v_A + 144j_{max} sn_{max}^2 v_m}}{2sn_{max} j_{max}}$
	Δt_3	Δt_1
	Δt_4	Δt_1
	Δt_5	$\Delta t_2 + \frac{j_m^2}{2sn_{max} j_{max}}$
	Δt_6	$\frac{j_m + j_{max}}{sn_{max}}$
	Δt_7	$\frac{j_m + j_{max}}{sn_{max}}$
II.5f	Δt_1	$\frac{j_{max}}{sn_{max}}$
	Δt_2	$\frac{a_{max} sn_{max} - j_{max}^2}{sn_{max} j_{max}}$
	Δt_3	Δt_1
	Δt_4	$\frac{((-24v_A + 24v_m)j_{max} - 24a_{max}^2)sn_{max}^2 - 24a_{max}j_{max}^2 sn_{max} + 3j_m^4 + 8j_m^3 j_{max} + 6j_m^2 j_{max}^2}{24sn_{max}^2 j_{max} a_{max}}$
	Δt_5	Δt_1
	Δt_6	$\frac{2a_{max} sn_{max} + j_m^2 - 2j_{max}^2}{2sn_{max} j_{max}}$
	Δt_7	$\frac{j_m + j_{max}}{sn_{max}}$
II.6f	Δt_1	$\frac{j_{max}}{sn_{max}}$
	Δt_2	$\frac{-18j_{max}^2 + \sqrt{42j_{max}^4 - 144j_{max} sn_{max}^2 v_A + 144j_{max} sn_{max}^2 v_m}}{12sn_{max} j_{max}}$
	Δt_3	Δt_1
	Δt_4	Δt_1
	Δt_5	$\Delta t_2 + \frac{j_{max}}{2sn_{max}}$
	Δt_6	$\frac{j_{max}}{sn_{max}}$
	Δt_7	$\frac{j_{max}}{sn_{max}}$
II.7f	Δt_1	$\frac{j_{max}}{sn_{max}}$
	Δt_2	$\frac{a_{max} sn_{max} - j_{max}^2}{sn_{max} j_{max}}$
	Δt_3	Δt_1
	Δt_4	$\frac{((-24v_A + 24v_m)j_{max} - 24a_{max}^2)sn_{max}^2 - 24a_{max}j_{max}^2 sn_{max} + j_{max}^4}{24sn_{max}^2 j_{max} a_{max}}$
	Δt_5	Δt_1
	Δt_6	$\frac{2a_{max} sn_{max} - j_{max}^2}{2sn_{max} j_{max}}$
	Δt_7	$\frac{j_{max}}{sn_{max}}$

Table A4. Formulae and polynomials for computing time intervals in Case II going back.

Model	Δt_i	Formulae and Polynomials
II.1b		Same as Model I.1b
II.2b		Same as Model I.2b
II.3b	Δt_1	$\frac{-12j_{max}^2 + \sqrt{42j_{max}^4 - 144j_{max} sn_{max}^2 v_E + 144j_{max} sn_{max}^2 v_m}}{12sn_{max} j_{max}}$
	Δt_2	$\frac{j_{max}}{sn_{max}}$

Table A4. Cont.

Model	Δt_i	Formulae and Polynomials
	Δt_3	Δt_2
	Δt_4	$\Delta t_1 - \frac{j_{max}}{2sn_{max}}$
	Δt_5	Δt_2
II.4b	Δt_1	$\frac{2a_{max}sn_{max}-j_{max}^2}{2sn_{max}j_{max}}$
	Δt_2	$\frac{j_{max}}{sn_{max}}$
	Δt_3	$\frac{((-24v_E+24v_m)j_{max}-24a_{max}^2)sn_{max}^2-24a_{max}j_{max}^2sn_{max}+j_{max}^4}{24sn_{max}^2j_{max}a_{max}}$
	Δt_4	Δt_2
	Δt_5	$\frac{a_{max}sn_{max}-j_{max}^2}{sn_{max}j_{max}}$
	Δt_6	Δt_2
	II.5b	Same as Model I.4b
II.6b		Same as Model I.5b
II.7b	Δt_1	$\Delta t_3 + \frac{j_m}{sn_{max}}$
	Δt_2	$\Delta t_5 + \frac{j_m^2}{2sn_{max}j_{max}}$
	Δt_3	$\frac{j_{max}}{sn_{max}}$
	Δt_4	Δt_3
	Δt_5	$\frac{-3j_{max}^2+\sqrt{18j_m^4+48j_m^3j_{max}+36j_m^2j_{max}^2+36j_m^4}-144j_{max}sn_{max}^2v_E+144j_{max}sn_{max}^2v_m}/6}{2sn_{max}j_{max}}$
	Δt_6	Δt_3
	II.8b	
	Δt_1	$\frac{j_m+j_{max}}{sn_{max}}$
	Δt_2	$\frac{2a_{max}sn_{max}+j_m^2-2j_{max}^2}{2sn_{max}j_{max}}$
	Δt_3	$\frac{j_{max}}{sn_{max}}$
	Δt_4	$\frac{((-24v_E+24v_m)j_{max}-24a_{max}^2)sn_{max}^2-24a_{max}j_{max}^2sn_{max}+3j_m^4+8j_m^3j_{max}+6j_m^2j_{max}^2}{24sn_{max}^2j_{max}a_{max}}$
	Δt_5	Δt_3
	Δt_6	$\frac{a_{max}sn_{max}-j_{max}^2}{sn_{max}j_{max}}$
	Δt_7	Δt_3

Appendix B. Algorithms and Formulae for Computing Critical Velocities

Algorithm A1. Algorithm for computing critical velocities in Case I going forth.

Input: $v_A, v_E, a_{max}, j_{max}, sn_{max}$

Output: v_{mjminA}, v_{mamaxA}

```

1:  If  $v_E - v_A \geq 2sn_{max} \left( \frac{a_{max}}{sn_{max}} \right)^{3/2}$  Then, % cf. Table 1
2:   $v_{mamaxA} = v_E$ 
3:   $j_m = -\sqrt{2j_{max}^2 - 2a_{max}sn_{max}}$  % for reaching  $j_{min}$ 
4:   $v_{mjminA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$  % according to Equation (6)
5:  Return
6:  End If
7:  If  $v_A \neq v_E$ 
    Find candidates for  $j_m$  by determining the roots of the polynomial with coefficient vector (highest power first):
8:  
$$\left[ 1, -2\sqrt{2}j_{max}, j_{max}^2, 2\sqrt{2}(j_{max}^3 + v_A sn_{max}^2 - v_E sn_{max}^2), -2j_{max}^4 - 8j_{max} sn_{max}^2 (v_A - v_E), 0, \right.$$


$$\left. 8j_{max}^3 sn_{max}^2 (v_A - v_E) + 4sn_{max}^4 (v_A - v_E)^2 \right]$$

9:  Pick for  $j_m$  the real root with  $-j_{max} \leq j_m \leq 0$ 
10: Else set  $j_m = -j_{max}$ 
11: End If
12: Determine  $a(t_2)$  in Model I.1f:  $a(t_2) = \frac{2j_{max}^2 - j_m^2}{2sn_{max}}$ 
13: If  $a(t_2) \leq a_{max}$  Then, %  $j_{min}$  is reached first
14:  $v_{mjminA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$  %  $j_m$  as determined in line 9 or 10.
15: 
$$j_m = \sqrt{2} \frac{j_{max}}{2} - \frac{1}{2} \sqrt{2j_{max}^2 + 8\sqrt{2}sn_{max}^2 a_{max} \sqrt{\frac{a_{max}}{sn_{max}}} j_{max} + a_{max}^2 sn_{max}^2 + a_{max} j_{max}^2 sn_{max} + 2j_{max} sn_{max}^2 (v_A - v_E)}$$

16: If  $j_m \geq -j_{max}$  Then, %  $j_m$  is admissible
17:  $v_{mamaxA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$ 
18: Else % compute  $v_{mamaxA}$  in Model I.5f

$$v_{mamaxA} = v_A + \frac{24j_{max} sn_{max}^2 a_{max} \sqrt{\frac{a_{max}}{sn_{max}}} + 12a_{max}^2 sn_{max}^2 + 12a_{max} j_{max}^2 sn_{max} - j_{max}^4}{24j_{max} sn_{max}^2}$$

19: End If % from 16
20: Else %  $a_{max}$  is reached first
    Find candidates for  $j_m$  by determining the roots of the polynomial with coefficient vector (highest power first):
21: 
$$\left[ a_{max}, 2\sqrt{2}sn_{max} \left( \sqrt{\frac{a_{max}}{sn_{max}}} a_{max} + v_A - v_E \right), 2a_{max}^2 sn_{max}, 0, 8\sqrt{\frac{a_{max}}{sn_{max}}} a_{max} sn_{max}^3 (v_A - v_E) + 4sn_{max}^3 (v_A - v_E)^2 \right]$$

22: Pick for  $j_m$  the real root with  $-j_{max} \leq j_m \leq 0$ 
23:  $v_{mamaxA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$ 
24:  $j_m = -\sqrt{2j_{max}^2 - 2a_{max}sn_{max}}$  % for reaching  $j_{min}$ 
25:  $v_{mjminA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$ 
26: End If % from line 13
27: Return

```

Table A5. Formulae for computing critical velocities in Case I going back.

$$v_{mjminE} = v_E + \frac{j_{max}^3}{12sn_{max}^2} (3\sqrt{2} + 4)$$

$$v_{maminE} = v_E + a_{max} \sqrt{\frac{a_{max}}{sn_{max}}} + \frac{a_{max}^2}{2j_{max}} + \frac{a_{max}j_{max}}{2sn_{max}} - \frac{j_{max}^3}{24sn_{max}^2}$$

Algorithm A2. Algorithm for computing critical velocities in Case II going forth.**Input:** $v_A, v_E, a_{max}, j_{max}, sn_{max}$ **Output:** $v_{mjminA}, v_{mjmaxA}, v_{mamaxA}$

- 1: **If** $v_E - v_A \geq 2 \frac{j_{max}^3}{sn_{max}^2}$ **Then** % cf. Table 1
- 2: $v_{mjminA} = v_E, v_{mjmaxA} = v_E$
- 3: **If** $v_E - v_A \geq \frac{a_{max}^2}{j_{max}} + a_{max} \frac{j_{max}}{sn_{max}}$ **Then** % cf. Table 1
- 4: $v_{mamaxA} = v_E$
- 5: **Else** % use first Model II.4f and, if not successful, try Model II.6f
- 6:
$$j_m = \sqrt{2} \frac{j_{max}}{2} - \frac{1}{2} \sqrt{2j_{max}^2 + 8\sqrt{2a_{max}^2 sn_{max}^2} + 2a_{max}j_{max}^2 sn_{max} + 2j_{max} sn_{max}^2 (v_A - v_E)}$$
- 7: **If** $j_m \geq -j_{max}$ **Then** % j_m is admissible
- 8:
$$v_{mamaxA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$$
- 9: **Else**
- 10:
$$v_{mamaxA} = \frac{24a_{max}^2 sn_{max}^2 + 24sn_{max}j_{max}^2 a_{max} - j_{max}^4 + 24j_{max} sn_{max}^2 v_A}{24j_{max} sn_{max}^2}$$
- 11: **End If** % line 7
- 12: **End If** % line 3
- 13: **Else**
- 14: **If** $v_A \neq v_E$
- 15: Find candidates for j_m by determining the roots of the polynomial with coefficient vector (highest power first); same as in Algorithm A1 line 8

$$\left[1, -2\sqrt{2}j_{max}, j_{max}^2, 2\sqrt{2}(j_{max}^3 + v_A sn_{max}^2 - v_E sn_{max}^2), -2j_{max}^4 - 8j_{max} sn_{max}^2 (v_A - v_E), 0, \right. \\ \left. 8j_{max}^3 sn_{max}^2 (v_A - v_E) + 4sn_{max}^4 (v_A - v_E)^2 \right]$$
- 16: Pick for j_m the real root with $-j_{max} \leq j_m \leq 0$
- 17: **Else** set $j_m = -j_{max}$
- 18: **End If** % line 14
- 19:
$$v_{mjminA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$$
- 20: Determine v_{mjmaxA} from $j(t_1) = j_{max}$ in Model II.2.f:

$$j_m = \sqrt{2} \frac{j_{max}}{2} - \frac{1}{2} \sqrt{2j_{max}^2 + 8\sqrt{4j_{max}^4} + 2j_{max} sn_{max}^2 (v_A - v_E)}$$
- 21: **If** $j_m \geq -j_{max}$ **Then** % j_m is admissible
- 22:
$$v_{mjmaxA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$$
- 23: **Else** % Move to Model II.3f
- 24:
$$v_{mjmaxA} = v_A + \frac{47j_{max}^3}{24sn_{max}^2}$$
- 25: **End If** % line 7

26:	Determine v_{mamaxA} by setting $a(t_3) = a_{max}$ in Model II.4f
27:	$j_m = \sqrt{2} \frac{j_{max}}{2} - \frac{1}{2} \sqrt{2j_{max}^2 + 8\sqrt{2a_{max}^2 sn_{max}^2 + 2a_{max} j_{max}^2 sn_{max} + 2j_{max} sn_{max}^2 (v_A - v_E)}}$
28:	If $j_m \geq -j_{max}$ Then % j_m is admissible
29:	$v_{mamaxA} = v_E - \frac{j_m^3}{12sn_{max}^2} (3\sqrt{2} + 4)$
30:	Else % Model II.4f provides no solution, go to Model II.6f
31:	$v_{mamaxA} = \frac{24a_{max}^2 sn_{max}^2 + 24sn_{max} j_{max}^2 a_{max} - j_{max}^4 + 24j_{max} sn_{max}^2 v_A}{24j_{max} sn_{max}^2}$
32:	End If % line 28
33:	End If % line 1
34:	Return

Table A6. Formulae for computing critical velocities in Case II going back.

$v_{mjminE} = v_E + \frac{j_{max}^3}{12sn_{max}^2} (3\sqrt{2} + 4)$
$v_{mjmaxE} = v_E + \frac{47j_{max}^3}{24sn_{max}^2}$
$v_{maminE} = v_E + \frac{a_{max}^2}{j_{max}} + \frac{a_{max} j_{max}}{sn_{max}} - \frac{j_{max}^3}{24sn_{max}^2}$

Appendix C. Polynomials and Formulae for Computing Critical Jerks**Algorithm A3.** Computing critical jerks and models in Case I “going forth”.**Input:** $v_A, v_E, v_{max}, a_{max}, j_{max}, sn_{max}$, situation % situation according to Figure 7**Output:** (possibly empty) list of $j_{mswitch}$ values ($j_{mswitch1}, j_{mswitch2}$), list of models

1:	If situation = b Then % for situation=a there is nothing to do, see Section 3
2:	Compute roots of $\begin{bmatrix} 17, 48j_{max}, 18j_{max}^2, -48j_{max}^3 - 48sn_{max}^2 v_A + 48sn_{max}^2 v_{max}, \\ -36j_{max}^4 - 144j_{max} sn_{max}^2 v_A + 144j_{max} sn_{max}^2 v_{max}, 0, \\ 144j_{max}^3 sn_{max}^2 (v_A - v_{max}) + 72sn_{max}^4 (v_A - v_{max})^2 \end{bmatrix}$ and select the real negative j_m with $vm2jm(v_{max}) \leq j_m \leq 0$ and $v(t_{end}) = v_{max}$. For the latter to be case, the following equation must hold: $v_{max} = \frac{(-3j_m^2 + 6j_{max}^2) \sqrt{-2j_m^2 + 4j_{max}^2} + 12v_A sn_{max}^2 - 4j_m^3 - 12j_m j_{max} + 12j_{max}^3}{12sn_{max}^2}$
3:	$j_{mswitch} = j_m$ and models=(I.2f,I.1f)
4:	Else if situation = c Then
5:	Compute roots of $\begin{bmatrix} 1, 0, -18a_{max} sn_{max}, 48sn_{max}^2 a_{max} \sqrt{\frac{a_{max}}{sn_{max}}} + 48sn_{max}^2 (v_A - v_{max}), \\ -36a_{max}^2 sn_{max}^2, 0, -144 \sqrt{\frac{a_{max}}{sn_{max}}} a_{max} sn_{max}^4 (v_A - v_{max}) - 72sn_{max}^4 (v_A - v_{max})^2 \end{bmatrix}$ and select the real negative j_m with $vm2jm(v_{max}) \leq j_m \leq 0 \leq j_m \leq 0$ and $\Delta t_5 \geq 0$ (in Model I.3f) and $v(t_{end}) = v_{max}$. For the latter to be case, the following equation must hold: $v_{max} = a_{max} \sqrt{\frac{a_{max}}{sn_{max}}} + \frac{(6a_{max} sn_{max} - 3j_m^2) \sqrt{4a_{max} sn_{max} + 2j_m^2} + 12v_A sn_{max}^2 - 4j_m^3}{12sn_{max}^2}$
6:	$j_{mswitch} = j_m$ resp. empty list and models=(I.3f,I.1f) resp. models=(I.3f)
7:	Else if situation = d or situation = e Then
8:	Substitute j_m by $-\sqrt{-2a_{max} sn_{max} + 2j_{max}^2}$ in the formula for Δt_3 in Model I.4f (Table A2)
9:	If $\Delta t_3 \geq 0$ Then
10:	$j_{mswitch1} = -\sqrt{-2a_{max} sn_{max} + 2j_{max}^2}$
11:	next_model=I.3f
11:	Else % Try Model I.4f with $\Delta t_3 = 0$

	Compute the roots of
12:	$\left[3, -8j_{max}, 6j_{max}^2, 0, -24sn_{max}^2a_{max}\sqrt{\frac{a_{max}}{sn_{max}}}j_{max} - 12a_{max}^2sn_{max}^2 - 12a_{max}j_{max}^2sn_{max} - 24j_{max}sn_{max}^2(v_A - v_{max})\right]$ and select the root r for which $\Delta t_5 = \frac{2a_{max}sn_{max} - 2j_{max}^2 + r^2}{2j_{max}sn_{max}}$ is real and ≥ 0
13:	$j_{mswitch1} = -\sqrt{2\Delta t_5j_{max}sn_{max} - 2a_{max}sn_{max} + 2j_{max}^2}$ with Δt_5 taken from the previous line
14:	next_model=I.2f
15:	End If
16:	If next_model=I.3f Then
17:	Do the same as in lines 5 and set $j_{mswitch2} = j_m$ if this exists (situation=c)
18:	$j_{mswitch}list = (j_{mswitch1}, j_{mswitch2})$ resp. $j_{mswitch}list = (j_{mswitch1})$, models=(I.4f,I.3f,I.1f) resp. models=(I.4f,I.3f)
19:	Else % next_model=I.2f
20:	Do the same as in lines 2 and set $j_{mswitch2} = j_m$ (situation=b)
21:	$j_{mswitch}list = (j_{mswitch1}, j_{mswitch2})$, models==(I.4f,I.2f,I.1f)
22:	End If
23:	End If
24:	Return

Algorithm A4. Computing critical jerks and models in Case I “going back”.

Input: $v_A, v_E, v_{max}, a_{max}, j_{max}, sn_{max}$, situation % situation according to Figure 8

Output: (possibly empty) list with $j_{mswitch}$ values ($j_{mswitch1}, j_{mswitch2}$), list of models

1:	If situation = b Then % for situation=a there is nothing to do, see Section 3
2:	Compute roots of $\left[17, 48j_{max}, 18j_{max}^2, -48j_{max}^3 - 48sn_{max}^2v_E + 48sn_{max}^2v_{max}, -36j_{max}^4 - 144j_{max}sn_{max}^2v_E + 144j_{max}sn_{max}^2v_{max}, 0, 144j_{max}^3sn_{max}^2(v_E - v_{max}) + 72sn_{max}^4(v_E - v_{max})^2\right]$ and select the real negative j_m with $j_{min} \leq j_m \leq 0$ and $v(t_{end}) = v_E$. For the latter to be case, the following equation must hold: $v_E = \frac{(3j_m^2 - 6j_{max}^2)\sqrt{-2j_m^2 + 4j_{max}^2} + 12v_{max}sn_{max}^2 + 4j_m^3 + 12j_m^2j_{max} - 12j_{max}^3}{12sn_{max}^2}$
3:	$j_{mswitch} = j_m$ and models=(I.5b,I.4b)
4:	Else if situation = c Then
5:	Substitute j_m by $-\sqrt{-2a_{max}sn_{max} + 2j_{max}^2}$ in the formula for Δt_4 in Model I.7b (Table A2)
6:	If $\Delta t_4 \geq 0$ Then
7:	$j_{mswitch1} = -\sqrt{-2a_{max}sn_{max} + 2j_{max}^2}$
8:	next_model=I.6b
9:	Else % Try Model I.7b with $\Delta t_4 = 0$
	Compute the roots of
10:	$\left[3, 8j_{max}, 6j_{max}^2, 0, -24sn_{max}^2a_{max}\sqrt{\frac{a_{max}}{sn_{max}}}j_{max} - 12a_{max}^2sn_{max}^2 - 12a_{max}j_{max}^2sn_{max} - 24j_{max}sn_{max}^2(v_E - v_{max})\right]$ and select the real negative j_m with $j_{min} \leq j_m \leq 0$ and $\Delta t_2 \geq 0$ (in Model I.7b) and $v(t_{end}) = v_E$. For the latter to be case, the following equation must hold: $v_E = \frac{(-12a_{max}^2 - 24\sqrt{\frac{a_{max}}{sn_{max}}}a_{max}j_{max} + 24v_{max}j_{max})sn_{max}^2 - 12a_{max}j_{max}^2sn_{max} + 3j_m^4 + 8j_m^3j_{max} + 6j_m^2j_{max}^2}{24sn_{max}^2j_{max}}$

11:	$j_{mswitch1} = j_m$
12:	next_model=I.5b
13:	End If
14:	If next_model=I.6b Then
15:	Compute roots of $\left[\begin{array}{l} 1, 0, -18a_{max}sn_{max}, 48sn_{max}^2a_{max}\sqrt{\frac{a_{max}}{sn_{max}}} + 48sn_{max}^2(v_E - v_{max}), \\ -36a_{max}^2sn_{max}^2, 0, -144\sqrt{\frac{a_{max}}{sn_{max}}}a_{max}sn_{max}^4(v_E - v_{max}) - 72sn_{max}^4(v_E - v_{max})^2 \end{array} \right]$ and select the real negative j_m with $j_{min} \leq j_m \leq 0$ and $\Delta t_1 \geq 0$ (in Model I.6b) and $v(t_{end}) = v_E$. For the latter to be case, the following equation must hold: $v_E = \frac{-12\sqrt{\frac{a_{max}}{sn_{max}}}a_{max}sn_{max}^2 - (6a_{max}sn_{max} - 3j_m^2)\sqrt{4a_{max}sn_{max} + 2j_m^2} + 4j_m^3 + 12v_{max}sn_{max}^2}{12sn_{max}^2}$
16:	If such a j_m exists Then
17:	$j_{mswitch2} = j_m$, $j_{mswitchlist} = (j_{mswitch1}, j_{mswitch2})$ and models=(I.7b,I.6b,I.4b)
18:	Else
19:	$j_{mswitchlist} = (j_{mswitch1})$ and models=(I.7b,I.6b)
20:	End If
21:	Else % next_model=I.5b
22:	Do the same as in line 2 and set $j_{mswitch2} = j_m$ (situation = b), $j_{mswitchlist} = (j_{mswitch1}, j_{mswitch2})$ and models=(I.7b,I.5b,I.4b)
23:	End If
24:	End If
25:	Return

Algorithm A5. Computing critical jerks and models in Case II “going forth”.

Input: $v_A, v_E, v_{max}, a_{max}, j_{max}, sn_{max}$, situation % situation according to Figure 13

Output: (possibly empty) list of $j_{mswitch}$ values ($j_{mswitch1}, j_{mswitch2}, j_{mswitch3}$), list of models

1:	If situation = b Then % for situation=a there is nothing to do, see Section 4
2:	Compute roots of $\left[\begin{array}{l} 17, 48j_{max}, 18j_{max}^2, -48j_{max}^3 - 48sn_{max}^2v_A + 48sn_{max}^2v_{max}, \\ -36j_{max}^4 - 144j_{max}sn_{max}^2v_A + 144j_{max}sn_{max}^2v_{max}, 0, \\ 144j_{max}^3sn_{max}^2(v_A - v_{max}) + 72sn_{max}^4(v_A - v_{max})^2 \end{array} \right]$ and select the real negative j_m with $vm2jm(v_{max}) \leq j_m \leq 0$ and $v(t_{end}) = v_{max}$. For the latter to be case, the following equation must hold: $v_{max} = \frac{(-3j_m^2 + 6j_{max}^2)\sqrt{-2j_m^2 + 4j_{max}^2} + 12v_Asn_{max}^2 - 4j_m^3 - 12j_mj_{max} + 12j_{max}^3}{12sn_{max}^2}$ % line 2 is identical to line 2 in Algorithm A3
3:	$j_{mswitch} = j_m$ and models=(II.2f,II.1f)
4:	Else if situation = c Then
5:	Compute roots of $[3, 8j_{max}, 6j_{max}^2, 0, -48j_{max}^4 - 24j_{max}sn_{max}^2(v_A - v_{max})]$ and select the real negative root j_m with $vm2jm(v_{max}) \leq j_m \leq 0$, if existing.
6:	If such a j_m exists Then
7:	$j_{mswitch1} = j_m$
8:	Proceed as in line 2, set $j_{mswitch2} = j_m$ and models=(II.4f,II.2f,II.1f)
9:	Else
10:	$j_{mswitchlist} =$ empty list and models=(II.4f)

11:	End If
12:	Else if situation = d Then
13:	Compute roots of $[3, 8j_{max}, 6j_{max}^2, 0, -24a_{max}^2sn_{max}^2 - 24a_{max}j_{max}^2sn_{max} - 24j_{max}sn_{max}^2(v_A - v_{max})]$ and select the real negative root j_m with $vm2jm(v_{max}) \leq j_m \leq 0$, if existing.
14:	If such a j_m exists Then
15:	$j_{mswitch1} = j_m$
16:	Proceed as in line 5
17:	If such a j_m exists Then
18:	$j_{mswitch2} = j_m$
19:	Proceed as in line 2, set $j_{mswitch3} = j_m$, $j_{mswitch}list = (j_{mswitch1}, j_{mswitch2}, j_{mswitch3})$ and models=(II.5f,II.4f,II.2f,II.1f)
20:	Else
21:	$j_{mswitch}list = (j_{mswitch1})$ and Models = (II.5f, II.4f)
22:	End If
23:	Else
24:	$j_{mswitch}list = \text{empty list, and models} = (\text{II.5f})$
25:	End If
26:	End If
27:	Return

Algorithm A6. Computing critical jerks and models in Case II “going back”.

Input: $v_A, v_E, v_{max}, a_{max}, j_{max}, sn_{max}$, situation % situation according to Figure 14

Output: (possibly empty) list of $j_{mswitch}$ values ($j_{mswitch1}, j_{mswitch2}, j_{mswitch3}$), list of models

1:	If situation = b Then % for situation=a there is nothing to do, see Section 4
2:	Compute roots of $\begin{bmatrix} 17, 48j_{max}, 18j_{max}^2, -48j_{max}^3 - 48sn_{max}^2v_E + 48sn_{max}^2v_{max}, \\ -36j_{max}^4 - 144j_{max}sn_{max}^2v_E + 144j_{max}sn_{max}^2v_{max}, 0, \\ 144j_{max}^3sn_{max}^2(v_E - v_{max}) + 72sn_{max}^4(v_E - v_{max})^2 \end{bmatrix}$ and select the real negative j_m with $j_{min} \leq j_m \leq 0$ and $v(t_{end}) = v_E$. For the latter to be case, the following equation must hold: $v_E = \frac{(3j_m^2 - 6j_{max}^2)\sqrt{-2j_m^2 + 4j_{max}^2 + 12v_{max}sn_{max}^2 + 4j_m^3 + 12j_m^2j_{max} - 12j_{max}^3}}{12sn_{max}^2}$ % line 2 is identical to line 2 in Algorithm A4
3:	$j_{mswitch} = j_m$ and models=(II.6b,II.5b)
4:	Else if situation = c Then
5:	Compute roots of $[3, 8j_{max}, 6j_{max}^2, 0, -48j_{max}^4 - 24j_{max}sn_{max}^2(v_E - v_{max})]$ and select the real negative root j_m with $j_{min} \leq j_m \leq 0$, if existing.
6:	If such a j_m exists Then
7:	$j_{mswitch1} = j_m$
8:	Proceed as in line 2, set $j_{mswitch2} = j_m$ and models = (II.7b,II.6b,II.5b)
9:	Else
10:	$j_{mswitch}list = \text{empty list and models} = (\text{II.7b})$
11:	End If
12:	Else if situation = d Then
13:	Compute roots of $[3, 8j_{max}, 6j_{max}^2, 0, -24a_{max}^2sn_{max}^2 - 24a_{max}j_{max}^2sn_{max} - 24j_{max}sn_{max}^2(v_E - v_{max})]$ and select the real negative root j_m with $j_{min} \leq j_m \leq 0$, if existing.

14:	If such a j_m exists Then
15:	$j_{mswitch1} = j_m$
16:	Proceed as in line 5
17:	If such a j_m exists Then
18:	$j_{mswitch2} = j_m$
19:	Proceed as in line 2, set $j_{mswitch3} = j_m$, $j_{mswitchlist} = (j_{mswitch1}, j_{mswitch2}, j_{mswitch3})$, and models=(II.8b,II.7b,II.6b,II.5b)
20:	Else
21:	$j_{mswitchlist} = (j_{mswitch1})$ and Models=(II.8b, II.7b)
22:	End If
23:	Else
24:	$j_{mswitchlist} = \text{empty list}$ and models=(II.8b)
25:	End If
26:	End If
27:	Return

References

- Biagiotti, L.; Melchiorri, C. *Trajectory Planning for Automatic Machines and Robots*; Springer: Berlin/Heidelberg, Germany, 2008.
- Carbone, G.; Gomez-Bravo, F. (Eds.) *Motion and Operation Planning of Robotic Systems*; Mechanisms and Machine Science 29; Springer: Berlin/Heidelberg, Germany, 2015.
- Gasparetto, A.; Boscariol, P.; Lanzutti, A.; Vidoni, R. Path Planning and Trajectory Planning Algorithms: A General Overview. In *Motion and Operation Planning of Robotic Systems*; Carbone, G., Gomez-Bravo, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; pp. 3–27.
- Alpers, B. On Fast Jerk-, Acceleration- and Velocity-Restricted Motion Functions for Online Trajectory Generation. *Robotics* **2021**, *10*, 25. [\[CrossRef\]](#)
- Sencer, B.; Altintas, Y.; Croft, E. Feed optimization for five axis CNC machine tools with drive constraints. *Int. J. Mach. Tools Manuf.* **2008**, *48*, 733–745. [\[CrossRef\]](#)
- Liu, H.; Lai, X.; Wu, W. Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints. *Robot. Comput. Integr. Manuf.* **2013**, *29*, 309–317. [\[CrossRef\]](#)
- Bharathi, A.; Dong, J. A Smooth Trajectory Generation Algorithm for Addressing Higher-Order Dynamic Constraints in Nanopositioning Systems. *Proc. Manuf.* **2015**, *1*, 216–225. [\[CrossRef\]](#)
- Fan, W.; Lee, C.; Chen, J. A realtime curvature-smooth interpolation scheme and motion planning for CNC machining of short line segments. *Int. J. Mach. Tools Manuf.* **2015**, *96*, 27–46. [\[CrossRef\]](#)
- Li, Y.; Huang, T.; Chetwynd, D.G. An approach for smooth trajectory planning of high-speed pick-and-place parallel robots using quintic B-splines. *Mech. Mach. Theory* **2018**, *126*, 479–490. [\[CrossRef\]](#)
- Da Rocha, P.A.S.; De Oliveira, W.D.; De Lima Tostes, M.E. An Embedded System-Based Snap Constrained Trajectory Planning Method for 3D Motion Systems. *IEEE Access* **2019**, *7*, 125188–125204. [\[CrossRef\]](#)
- Biagiotti, L.; Melchiorri, C. Trajectory generation via FIR filters: A procedure for time-optimization under kinematic and frequency constraints. *Control Eng. Pract.* **2019**, *87*, 43–58. [\[CrossRef\]](#)
- Fang, Y.; Qi, J.; Hu, J.; Wang, W.; Peng, Y. An approach for jerk-continuous trajectory generation of robotic manipulators with kinematical constraints. *Mech. Mach. Theory* **2020**, *153*, 103957. [\[CrossRef\]](#)
- Kröger, T.; Wahl, F.M. Online Trajectory Generation: Basic Concepts for Instantaneous Reactions to Unforeseen Events. *IEEE Trans. Robot.* **2010**, *26*, 94–111. [\[CrossRef\]](#)
- Shiller, Z. Off-line and On-Line Trajectory Planning. In *Motion and Operation Planning of Robotic Systems*; Carbone, G., Gomez-Bravo, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; pp. 29–62.
- Gasparetto, A.; Zanotto, V. A new method for smooth trajectory planning of robot manipulators. *Mech. Mach. Theory* **2007**, *42*, 455–471. [\[CrossRef\]](#)
- Zhang, Q.; Li, S. Efficient computation of smooth minimum time trajectory for CNC machining. *Int. J. Adv. Manuf. Technol.* **2013**, *68*, 683–692. [\[CrossRef\]](#)
- Fan, W.; Gao, X.; Lee, C.; Zhang, K.; Zhang, Q. Time-optimal interpolation for five axis CNC machining along parametric tool path based on linear programming. *Int. J. Adv. Manuf. Technol.* **2013**, *69*, 1373–1388. [\[CrossRef\]](#)
- Valente, A.; Baraldo, S.; Carpanzano, E. Smooth trajectory generation for industrial robots performing high precision assembly processes. *CIRP Ann. Manuf. Technol.* **2017**, *66*, 17–20. [\[CrossRef\]](#)

19. Lin, J.; Somani, N.; Hu, B.; Rickert, M.; Knoll, A. An Efficient and Time-Optimal Trajectory Generation Approach for Waypoints under Kinematic Constraints and Error Bounds. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
20. Huang, J.; Hu, P.; Wu, K.; Zeng, M. Optimal time-jerk trajectory planning for industrial robots. *Mech. Mach. Theory* **2018**, *121*, 530–544. [[CrossRef](#)]
21. Zhang, S.; Zanchettin, A.; Villa, R.; Dai, S. Real-time trajectory planning based on joint-decoupled optimization in human-robot interaction. *Mech. Mach. Theory* **2020**, *144*, 103664. [[CrossRef](#)]
22. Ma, J.; Gao, S.; Yan, H.; Lv, Q.; Hu, G. A new approach to time-optimal trajectory planning with tork and jerk limits for robot. *Robot. Auton. Syst.* **2021**, *140*, 103744. [[CrossRef](#)]
23. Biagiotti, L.; Melchiorri, C. Optimization of Generalized S-curve Trajectories for Residual Vibration Suppression and Compliance with Kinematic Bounds. *IEEE/ASME Trans. Mechatron.* **2021**, *26*, 2724–2734. [[CrossRef](#)]
24. Wang, F.; Wu, Z.; Bao, T. Time-Jerk optimal Trajectory Planning of Industrial Robots Based on a Hybrid WOA-GA Algorithm. *Processes* **2022**, *10*, 1014. [[CrossRef](#)]
25. Scalera, L.; Vidoni, R.; Giusti, A. Optimal scaling of dynamic safety zones for collaborative robotics. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA 2021), Xi'an, China, 31 May–4 June 2021; pp. 3822–3828.
26. Lambrechts, P.; Boerlage, M.; Steinbuch, M. Trajectory planning and feedforward design for electromechanical motion systems. *Contr. Eng. Pract.* **2005**, *13*, 145–157. [[CrossRef](#)]
27. Nguyen, K.D.; Ng, T.; Chen, I. On Algorithms for Planning S-curve Motion Profiles. *Int. J. Adv. Robot. Syst.* **2008**, *3*, 99–106. [[CrossRef](#)]
28. Fan, W.; Gao, X.; Yan, W.; Yuan, C. Interpolation of parametric CNC machining path under confined jounce. *Int. J. Adv. Manuf. Technol.* **2012**, *62*, 719–739. [[CrossRef](#)]
29. Lee, A.Y.; Choi, Y. Smooth trajectory planning methods using physical limits. *J. Mech. Eng. Sci.* **2015**, *229*, 2127–2143. [[CrossRef](#)]
30. Fang, Y.; Hu, J.; Liu, W.; Shao, Q.; Qi, J.; Peng, Y. Smooth and time-optimal S-curve trajectory planning for automated robots and machines. *Mech. Mach. Theory* **2019**, *137*, 127–153. [[CrossRef](#)]
31. Wang, H.; Wang, H.; Huang, J.; Zhao, B.; Quan, L. Smooth point-to-point trajectory planning for industrial robots with kinematical constraints based on high-order polynomial curve. *Mech. Mach. Theory* **2019**, *139*, 284–293. [[CrossRef](#)]
32. Wu, Z.; Chen, J.; Bao, T.; Wang, J.; Zhang, L.; Xu, F. A Novel Point-to-Point Trajectory Planning Algorithm for Industrial Robots Based on a Locally Asymmetrical Jerk Motion Profile. *Processes* **2022**, *10*, 728. [[CrossRef](#)]
33. Biagiotti, L.; Melchiorri, C.; Moriello, L. Damped Harmonic Smoother for Trajectory Planning and Vibration Suppression. *IEEE Trans. Contr. Syst. Technol.* **2020**, *28*, 626–634. [[CrossRef](#)]
34. Ezair, B.; Tassa, T.; Shiller, Z. Planning high order trajectories with general initial and final conditions and asymmetric bounds. *Int. J. Robot. Res.* **2014**, *33*, 898–916. [[CrossRef](#)]
35. Zhang, Y.; Ye, P.; Wu, J.; Zhang, H. An optimal curvature-smooth transition algorithm with axis jerk limitations along linear segments. *Int. J. Adv. Manuf. Technol.* **2018**, *95*, 875–888. [[CrossRef](#)]
36. Zhang, L.; Du, J. Acceleration smoothing algorithm based on jounce limited for corner motion in high-speed machining. *Int. J. Adv. Manuf. Technol.* **2018**, *95*, 1487–1504. [[CrossRef](#)]
37. VDI. *Guideline 2143: Motion Rules for Cam Mechanisms: Theoretical Fundamentals*; Beuth Verlag: Berlin, Germany, 1980. (In German)
38. Broquère, X.; Sidobre, D.; Herrera-Aguilar, I. Soft motion trajectory planner for service manipulator robot. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 2808–2813.