Article

On Fast Jerk–, Acceleration– and Velocity–Restricted Motion Functions for Online Trajectory Generation

Burkhard Alpers

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

Abstract: Finding fast motion functions to get from an initial state (distance, velocity, acceleration) to a final one has been of interest for decades. For a solution to be practically relevant, restrictions on jerk, acceleration and velocity have to be taken into account. Such solutions use optimization algorithms or try to directly construct a motion function allowing online trajectory generation. In this contribution, we follow the latter strategy and present an approach which first deals with the situation where initial and final accelerations are 0, and then relates the general case as much as possible to this situation. This leads to a classification with just four major cases. A continuity argument guarantees full coverage of all situations which is not the case or is not clear for other available algorithms. We present several examples that show the variety of different situations and, thus, the complexity of the task. We also describe an implementation in MATLAB® and results from a huge number of test runs regarding accuracy and efficiency, thus demonstrating that the algorithm is suitable for online trajectory generation.

Keywords: online trajectory generation; fast motion functions; seven segment profile; jerk restriction

check for updates

Citation: Alpers, B. On Fast Jerk–, Acceleration– and Velocity–Restricted Motion Functions for Online Trajectory Generation. *Robotics* **2021**, *10*, 25. https://doi.org/10.3390/robotics10010025

Received: 7 January 2021
Accepted: 26 January 2021
Published: 1 February 2021

1. Introduction

For achieving high throughput in handling machines, one of the most basic tasks consists of quickly moving from one position to the next one, taking into account machine restrictions regarding velocity, acceleration and jerk. This task is generally known as "path and trajectory planning" for robots and other automatic machines [1,2], and beside short execution times, other goals such as low energy consumption or low jerk have been considered in [3]. In this contribution, we restrict ourselves to a one-dimensional case where a certain distance Δs is given and a motion function $s(t)$ is to be found, such that symmetric restrictions regarding velocity $v(t)$, acceleration $a(t)$ and jerk $j(t)$ apply, i.e., $-j_{max} \leq j(t) \leq j_{max}$, $-a_{max} \leq a(t) \leq a_{max}$ and $-v_{max} \leq v(t) \leq v_{max}$.

The limit to the jerk function implicitly requires the acceleration function to be continuous, whereas the jerk function is allowed to have discontinuities. This task can be subdivided into subtasks regarding the given initial and final states (velocity and acceleration). In the classification given in the respective guideline of the German Association of Engineers [4], 16 combinations are stated depending on whether the initial state $(v(t_0), a(t_0))$ is $(0,0)$ (called "rest": R), $(\neq 0, 0)$ (G), $(0, \neq 0)$ (U) or $(\neq 0, \neq 0)$ (B) (German abbreviations). The most well-known task is RR ("rest-to-rest"), which, in robotics, is usually called "point-to-point motion". Kröger and Wahl [5] provide a different classification that also includes the types of restrictions, but for our purposes, the VDI classification suffices.

The task of finding a fast (sometimes also called "optimal") motion function has been investigated for several decades. Roughly, one can distinguish between two kinds of approaches: In the first approach, the task is formulated as a restricted optimization problem and then well-known or self-developed algorithms from the fields of parameter optimization or optimal control are used [6–10]. The class of functions considered are often spline functions [6,7,10]. Lin et al. [8] use a specific class of piecewise-defined functions (“seven segments”) which are optimized. Such approaches have the advantage that optimization can take into account further aspects by modifying the objective (goal) function, e.g., energy consumption or deviation from a certain path, and they can also handle variable restrictions on velocity, acceleration and jerk. On the other hand, they are more time-consuming, such that they are mainly used for so-called off-line trajectory construction taking place in advance as opposed to on-line trajectory planning where the trajectory is changed “on the fly” during a controller cycle due to unforeseen events [11,12].

The second approach constructs the motion function directly using piecewise-defined functions [1,13–22]. We first consider the most basic task, RR. Choosing a piecewise constant jerk function leads to the so-called seven-segment profile (depicted in Figure 1) with continuous acceleration but discontinuous jerk, which has been fully specified in [1] (pp. 90–93) and can be found in industrial implementations [13] (Section 11.2). The approach has been extended to a 15-segment profile with piecewise constant snap in [1] (pp. 107–114), making the jerk function continuous. Considerable improvements have been achieved by using higher-order polynomials or the sigmoid function [14,15]. They provide fast motion functions with continuity in higher-order derivatives (or even all derivatives [15]) and are, hence, very suitable for vibration reduction. Smoothing can also be achieved by applying moving averages, as is shown in [16]. It is open whether and how the use of such advanced functions can be extended to motion tasks other than RR. When non-zero values are prescribed for initial and/or final velocities and/or acceleration, direct construction seems to be much harder. Nearly all contributions known to the author use the relatively simple seven-segment profile since this already leads to complex computations. The subtask GG (zero acceleration at initial and final states) has been dealt with for seven-segment profiles by [17] and [1], but both solutions are incomplete (for [17], cf. Kröger [10] (Section 4.1.2)) or can be improved (in [1], absolute values of the occurring minimal and maximal acceleration are assumed to be equal even if they are lower than the limit value, cf. Example 3.11, p. 84). Using, also, the seven-segment profile, Haschke et al. [18] developed an algorithm for the subtask BR (arbitrary initial state, rest), and Kröger and Wahl [5] and Kröger [12] provided a solution for the subtask BG, going through all possible branches in their decision tree. Their work shows the complexity of the task since they could provide only a small portion of their decision trees in their publications (cf. [5] (p. 106) for information on the number of nodes in these trees).

For the general task BB (arbitrary values for initial and final velocities and accelerations), Ezair et al. [19] developed a recursive algorithm which even enables them to handle a case where initial and final values as well as restrictions are given for an arbitrary number of derivatives and higher orders of continuity can be achieved, but only for order 2 (i.e., continuous acceleration); the algorithm is fast enough for online trajectory generation. In this case, it provides a seven-segment profile but not necessarily the fastest one. As we will show, there might be several seven-segment profiles solving the problem with quite different execution times. Broquère [20] and Broquère et al. [21] present a complex algorithm for the seven-segment profile but, as observed by Kröger [12] (Section 4.1.2), their work does not provide a solution in all cases, and we will give examples in Section 4 for configurations which are easily overseen. Recently, Sidobre and Desormeaux [22] modified the work in [20,21] and developed an algorithm where they claim to be able to reduce the general situation to just four simple sub-cases, but there is no proof, and the claim is not supported by extensive test runs with random values for all input data.

It is the goal of this contribution to provide an algorithm for constructing a seven-segment profile for the general BB task which provenly covers all possible situations, has a clear systematic structure with manageable complexity and is efficient enough for being suitable for online trajectory generation. Note that our results have some overlap with those of Sidobre and Desormeaux [22], but we apply a different approach by reducing the general BB task as much as possible to the GG subtask using intermediate velocities. We do not claim to provide an optimal solution (although this might be the case) since we do not have mathematical proof for this. Therefore, we just label our solution as “fast” since, at any time, at least one restriction is exactly fulfilled, but we also show that this does not imply global optimality.

The paper is structured as follows: In the next section, we present the seven-segment motion profile with modifications and recall the "acceleration–velocity plane" introduced by Broquère et al. [20] and Broquère [21]. Section 3 models the subtask GG where initial and final velocities are 0 and, based on this, Section 4 deals with the general task. Section 5 discusses implementation issues and describes the results of the validation in MATLAB®. Section 6 summarizes the results and discusses further work.

2. Basic Models, Quantities and Visualizations

The task we want to solve in this contribution consists of finding a fast motion function when the following eight values are given:

j_{max}, a_{max}, v_{max} (each > 0), such that $-j_{max} \le j(t) \le j_{max}$, $-a_{max} \le a(t) \le a_{max}$ and $-v_{max} \le v(t) \le v_{max}$; initial conditions: v_A, a_A; end conditions: v_E, a_E; distance: Δs (arbitrary real numbers).

Note that the distance is meant to be the difference between the final and the initial value of the path scale ($\Delta s = s_E - s_A$), not the "travelled distance", which might be much longer when the motion goes back first and then forward again.

As in [20–22], we work with a so-called 7-segment motion profile, shown in Figure 1. It is based on a piecewise constant jerk function together with initial conditions on velocity and acceleration. In this profile, the motion function has, at most, 4 segments where the jerk is $\pm j_{max}$ and, at most, 3 segments with zero jerk. In the classical RR situation ("rest-to-rest", see Figure 1a), the sign pattern of the jerk segments is $(+,-,-,+)$, but we allow arbitrary patterns such as $(+,-,+,-)$ to occur in order to have full coverage of all possible configurations as will be seen later. Formally stated, we have:

$$j(t) = \pm j_{max} \text{ in } [t_0, t_1],\ [t_2, t_3],\ [t_4, t_5] \text{ and } [t_6, t_7];\ j(t) = 0 \text{ in } [t_1, t_2],\ [t_3, t_4] \text{ and } [t_5, t_6].$$

Note that all of these intervals might have zero length. In the example displayed in Figure 1a, the maximal values for velocity and acceleration are reached, which need not always be the case. Figure 1b provides an example for the general situation where v_A, a_A, v_E and a_E are non-zero. In that example, only a_{max} is reached and two of the potentially seven segments have length zero.

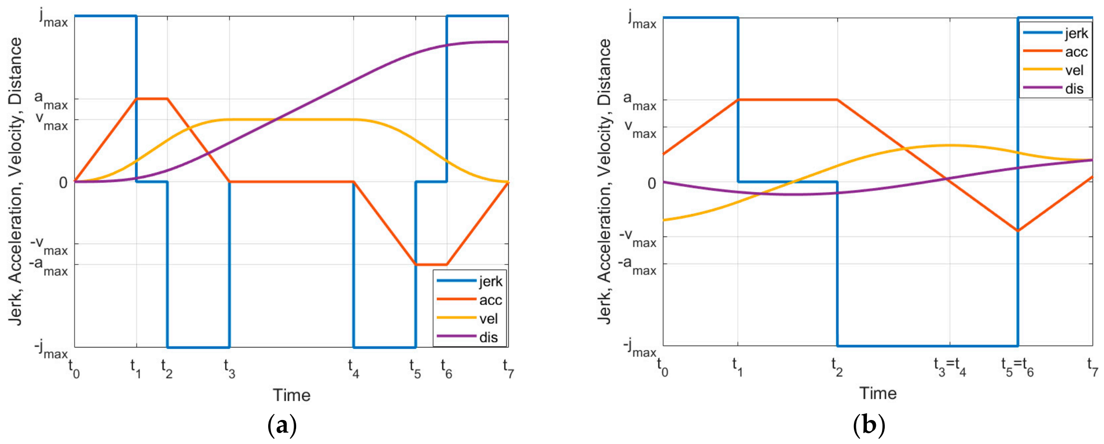
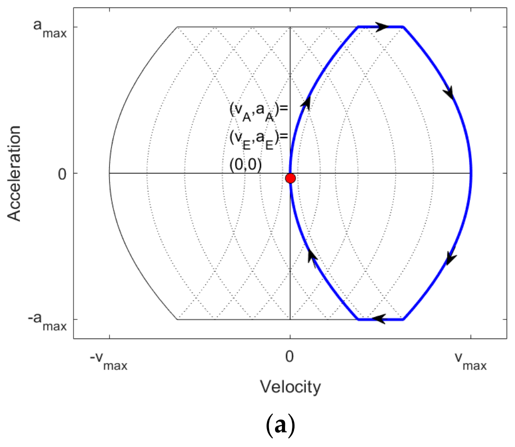


Figure 1. Seven-segment profile (**a**) for RR; (**b**) for BB—two segments with length 0.

In order to visualize the change of velocity and acceleration in such seven-segment motion functions, Broquère [20,21] introduced the velocity–acceleration plane, shown in Figure 2. In this plane, motion functions are displayed mathematically as parametric curves $(v(t), a(t))$, $t \in [t_0, t_7]$. When one assumes that $j(t) = \pm j_{max}$ or 0, then the curves "move" along square root curves or horizontally (e.g., from $a(t) = j_{max}t$ and $v(t) = \frac{1}{2} j_{max} t^2$ one obtains $a(v) = \sqrt{2j_{max}v}$). The square root curves can be of the kind $a = \pm\sqrt{c_1(v - c_2)}$ (open to the right) or $a = \pm\sqrt{c_1(c_2 - v)}$ (open to the left) with constants c_1, c_2. We assume that the horizontal motion only occurs when $a(t) = \pm a_{max}$ since otherwise, faster motion is possible. Motion along the square root curves is only possible clockwise since when the acceleration is positive (resp. negative), the velocity must increase (resp. decrease). The only points where a fast motion function might stay for a time of length >0 are the points ($\pm v_{max}$,0). This is the case when there is a segment with zero jerk and acceleration. Figure 2a shows a representation of the classical RR situation where v_{max} and a_{max} are reached. Note that only points within the area bounded by the left and right square root curves and the horizontal lines at a_{max} and $-a_{max}$ are admissible since only those points can be reached from (0,0) and one can go back from there to (0,0) without violating the restrictions (a larger opening of the square root curve corresponds to a higher value of j_{max}). The velocity–acceleration plane has one particular "pitfall" where misunderstandings can easily occur, so special care is called for: It does not show the distance Δs, and this distance heavily depends on where in the velocity–acceleration plane a curve is placed. The length of the curve has nothing to do with the distance Δs. The blue curve in Figure 2b depicts the motion function of Figure 1b (with positive Δs) in the velocity–acceleration plane. The green curve has the same initial and end conditions but a negative value for Δs.



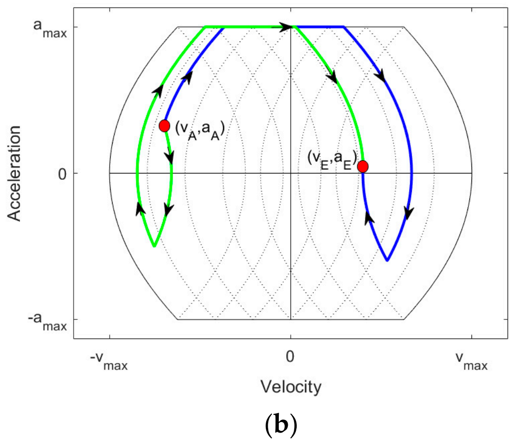


Figure 2. Velocity–acceleration curve for the motion function (**a**) in Figure 1a; (**b**) in Figure 1b.

3. The Subtask GG: Zero Acceleration Initially and Finally

In this section, we consider a motion with zero acceleration initially and finally. Let v_A resp. v_E be the initial resp. final velocity. We first determine the time $T(v_A, v_E)$ necessary to get from the initial to the final state by applying first maximal (minimal) jerk and then minimal (maximal) jerk where in between the jerk might be zero and the acceleration equal to $\pm a_{max}$. Examples are shown in Figure 3 in the velocity–acceleration plane. Type 1 depicts the situation where $|v_E - v_A|$ is so small that maximal/minimal acceleration is not reached, whereas in type 2, the latter is the case. When $\pm a_{max}$ is just reached, we have $|v_E - v_A| = \frac{a_{max}{}^2}{j_{max}}$, so type 1 occurs when $|v_E - v_A| < \frac{a_{max}{}^2}{j_{max}}$ and type 2 when $|v_E - v_A| \geq \frac{a_{max}{}^2}{j_{max}}$.

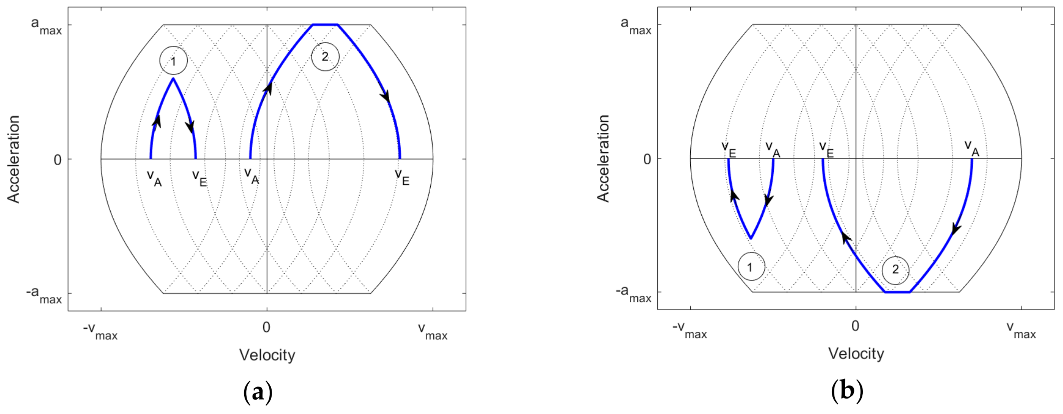


Figure 3. Switching between velocities: (**a**) positive acceleration; (**b**) negative acceleration.

In type 1, the maximally reached acceleration is $a_m = j_{max} \cdot \frac{T}{2}$, where T is the overall time. Hence, $|v_E - v_A| = a_m \cdot \frac{T}{2} = j_{max} \cdot \frac{T^2}{4}$, and thus, $T = 2\sqrt{\frac{|v_E - v_A|}{j_{max}}}$. The distance travelled in this case is $S(v_A, v_E) = \left(\frac{v_A + v_E}{2}\right) \cdot T(v_A, v_E) = (v_A + v_E) \cdot \sqrt{\frac{|v_E - v_A|}{j_{max}}}$. In type 2, we have $|v_E - v_A| = \left(T - \frac{a_{max}}{j_{max}}\right) \cdot a_{max}$, and hence, $T = \frac{|v_E - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}}$. The distance travelled in this case is again $S(v_A, v_E) = \left(\frac{v_A + v_E}{2}\right) \cdot T(v_A, v_E) = \left(\frac{v_A + v_E}{2}\right) \cdot \left(\frac{|v_E - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}}\right)$. We summarize the result as follows:

$$T(v_A, v_E) = \begin{cases} 2\sqrt{\frac{|v_E - v_A|}{j_{max}}} \; if \; |v_E - v_A| < \frac{a_{max}{}^2}{j_{max}} \\ \frac{|v_E - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}} \; if \; |v_E - v_A| \geq \frac{a_{max}{}^2}{j_{max}} \end{cases} \quad (1)$$

$$S(v_A, v_E) = \begin{cases} (v_A + v_E) \cdot \sqrt{\frac{|v_E - v_A|}{j_{max}}} \; if \; |v_E - v_A| < \frac{a_{max}{}^2}{j_{max}} \\ \left(\frac{v_A + v_E}{2}\right) \cdot \left(\frac{|v_E - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}}\right) \; if \; |v_E - v_A| \geq \frac{a_{max}{}^2}{j_{max}} \end{cases} \quad (2)$$

Figure 3 shows examples for the GG situation, each of which works just for one specific given distance. For other distances, the curve goes from v_A to an intermediate velocity v_m and then further on to the final velocity v_E, as is shown in Figure 4 for three situations ($v_{m1} > v_E$, $v_{m2} < v_A$, $v_A \leq v_{m3} \leq v_E$), where the third situation is not the fastest as we will see.

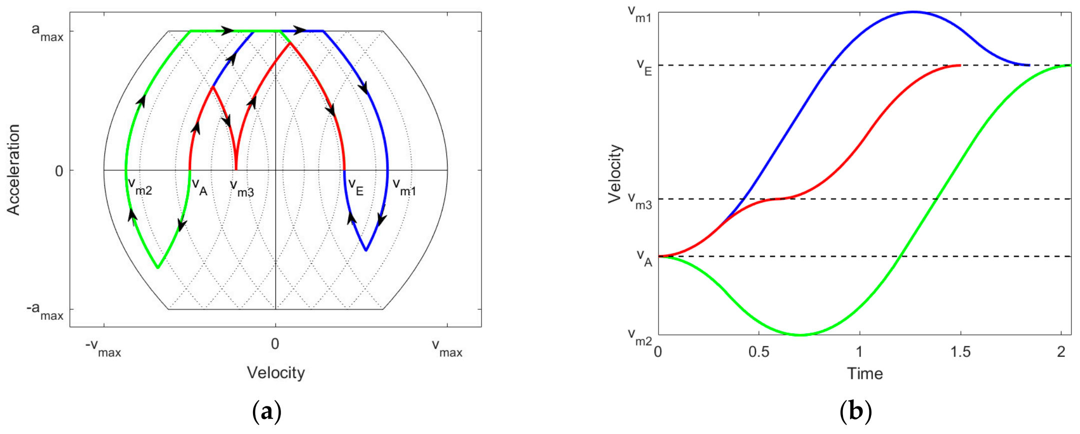


Figure 4. Examples for getting from v_A to v_E: (**a**) velocity–acceleration plane; (**b**) velocity over time.

From (1) and (2), we can easily compute the overall time and distance when going via an intermediate velocity v_m:

$$T(v_A, v_E, v_m) = T(v_A, v_m) + T(v_m, v_E)\ ,\ S(v_A, v_E, v_m) = S(v_A, v_m) + S(v_m, v_E) \quad (3)$$

In order to visualize how the distance changes when v_A and v_E are fixed and v_m runs from v_{min} to v_{max}, one can plot $S(v_A, v_E, v_m)$ over v_m, as shown in Figure 5a. For seeing in one plot how $T(v_A, v_E, v_m)$ varies simultaneously, a parametric curve $(S(v_A, v_E, v_m),$ $T(v_A, v_E, v_m), t \in [-v_{max}\ , v_{max}])$ is adequate, as shown in Figure 5b. Both parts provide interesting information which will also be of use for the general situation (Section 4). If $v_m = v_{max}$ (resp. $v_m = v_{min}$) and remains constant for a certain time interval, the distance increases (resp. decreases) linearly with time. Therefore, in Figure 5b, the graph could be extended linearly to the left and to the right. We further observe that the distance varies continuously with v_m, which follows mathematically from (2) and (3); therefore, for any given distance, a solution can be found.

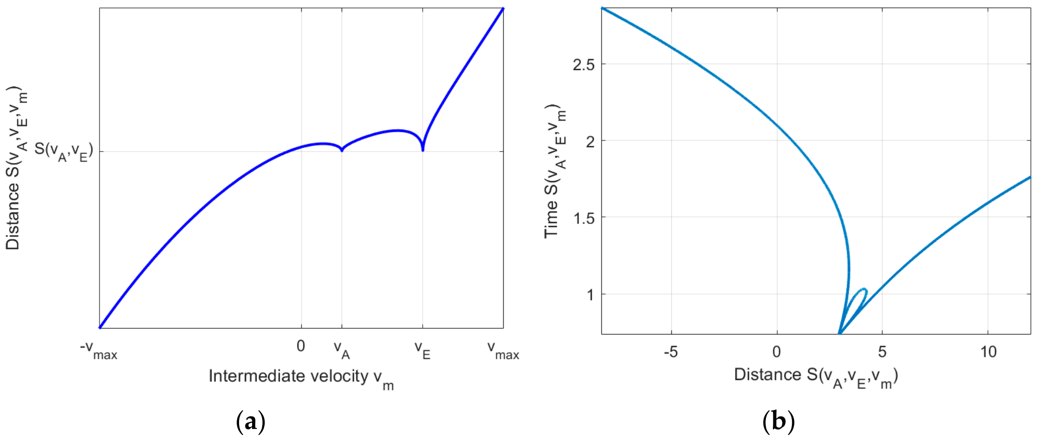


Figure 5. For $v_A > 0$ and $v_E > 0$: (**a**) S over v_m; (**b**) T over S.

The behavior shown in Figure 5a might be astonishing at first sight; the distance increases first when v_m is below v_A before decreasing monotonously. We explain this using Figure 4b. Here, there are two controversial effects: decreasing v_m makes the velocity curve lie "lower", leading to a smaller integral (less distance); on the other hand, the time needed to get to v_E becomes larger, leading to a longer integration interval (larger distance). Obviously, in the long run, the first effect becomes dominant, but in the short term, the second one is dominant, leading to an increase in distance. This gives, already, an impression of the complexity of the situation, which is easily overlooked, leading to false assumptions.

Figure 5a shows that depending on the given distance, there are up to five solutions to the motion problem using an intermediate velocity v_m. Five solutions occur when the given distance is slightly above $S(v_A, v_E)$. This also shows that the fact that, at any time, one of the restrictions is taken does not guarantee optimality. Figure 5b shows which solution is the fastest one; for a given distance, the lowest intersection of the vertical line at that distance with the curve provides the best solution. One can also observe that the best time is discontinuous over the given distance. In [22], Sidobre and Desormeaux also provide examples for this phenomenon. They also use a parametric curve of distance over time needed, but they use a different parameter (time).

For finding the best solution within the class of seven-segment functions, one has to solve the equation $S(v_A, v_E, v_m) = \Delta s$. We will explain how to do this efficiently in Section 5.

4. The General Task: Arbitrary Velocity and Acceleration, Initially and Finally

In this section, we consider the general case where initial and final velocities and accelerations can be arbitrary. We start with introducing some auxiliary functions that will be helpful in the later exposition. For an arbitrary admissible point (v_0, a_0) in the velocity–acceleration plane, $v_b(v_0, a_0)$ is defined to be the velocity for which the following holds: If you start at the point $(v_b(v_0, a_0), 0)$ and apply minimum (resp. maximum) jerk if $a_0 < 0$ (resp. $a_0 \geq 0$), then you reach (v_0, a_0). The index b here stands for "backward", i.e., going backward from (v_0, a_0). Figure 6 illustrates the definitions. Analogously, $v_f(v_0, a_0)$ is defined to be the velocity for which the following holds: If you start at the point (v_0, a_0) and apply maximum (resp. minimum) jerk if $a_0 < 0$ (resp. $a_0 \geq 0$), then you reach $(v_f(v_0, a_0), 0)$. The index f here stands for "forward". An easy computation yields:

$$v_b(v,a) = v - sgn(a)\cdot\frac{1}{2}\frac{a^2}{j_{max}}\ ,\ v_f(v,a) = v + sgn(a)\cdot\frac{1}{2}\frac{a^2}{j_{max}}. \tag{4}$$

Moreover, let $S_b(v_0, a_0)$ be the additional distance when one goes from $(v_b(v_0, a_0), 0)$ to (v_0, a_0), applying minimum (resp. maximum) jerk if $a_0 < 0$ (resp. $a_0 \geq 0$). Analogously, $S_f(v_0, a_0)$ is defined to be the distance when one goes from (v_0, a_0) to $(v_f(v_0, a_0), 0)$, applying maximum (resp. minimum) jerk if $a_0 < 0$ (resp. $a_0 \geq 0$). Again, an easy computation shows:

$$S_b(v,a) = v_b(v,a)\frac{|a|}{j_{max}} + \frac{1}{6}\cdot\frac{a^3}{j_{max}^2}\ ,\ S_f(v,a) = v_f(v,a)\frac{|a|}{j_{max}} - \frac{1}{6}\cdot\frac{a^3}{j_{max}^2} \tag{5}$$

Finally, let $T_a(a_0)$ be the time needed to go from a point (v, a_0) (v arbitrary) to $v_f(v, a_0)$ (or, equivalently, to go from $v_b(v, a_0)$ to (v, a_0)), applying maximum resp. minimum jerk. We have:

$$T_a(a) = \frac{|a|}{j_{max}} \tag{6}$$

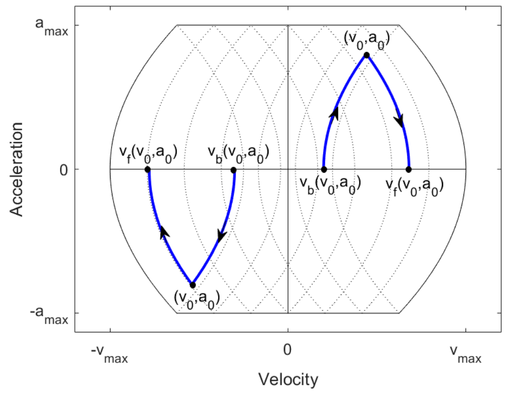


Figure 6. Illustration of auxiliary functions.

If one changes the positive direction of the distance, then the original problem is transformed into a problem with data $-\Delta s$, $-v_A$, $-a_A$, $-v_E$ and $-a_E$ which has the same solution regarding time intervals, only the sign factors for j_{max} have to be changed (see Broquère, 2011). Therefore, we can restrict ourselves to two main situations: $a_A \leq 0$ and $a_E \geq 0$, or $a_A \geq 0$ and $a_E \geq 0$. The other combinations can be transformed into the former ones by changing the positive direction of distance. These two main cases are further split up into four sub-cases which depend additionally on $v_b(v_A, a_A)$, $v_b(v_E, a_E)$, $v_f(v_A, a_A)$ and $v_f(v_E, a_E)$; see Table 1.

Table 1. Basic cases in the BB situation.

Case	Conditions on a_A, a_E	Conditions on v_b,v_f
Case A	$a_A \leq 0$ and $a_E \geq 0$	$v_f(v_A, a_A) \leq v_b(v_E, a_E)$
Case B	$a_A \leq 0$ and $a_E \geq 0$	$v_f(v_A, a_A) > v_b(v_E, a_E)$
Case C	$a_A \geq 0$ and $a_E \geq 0$	$v_b(v_A, a_A) \leq v_b(v_E, a_E)$ and $v_f(v_A, a_A) \leq v_f(v_E, a_E)$
Case D	$a_A \geq 0$ and $a_E \geq 0$	$v_b(v_A, a_A) > v_b(v_E, a_E)$ or $v_f(v_A, a_A) > v_f(v_E, a_E)$

From Table 1, it is logically clear that this is a complete coverage of all cases. We will see that only in case C do we need a further division into two sub-cases. In the sequel, we use the following abbreviations:

- $A = (v_A, a_A)$, $E = (v_E, a_E)$;
- $v_{bA} = v_b(v_A, a_A)$, $v_{fA} = v_f(v_A, a_A)$, $v_{bE} = v_b(v_E, a_E)$, $v_{fE} = v_f(v_E, a_E)$;
- $S_{bA} = S_b(v_A, a_A)$, $S_{fA} = S_f(v_A, a_A)$, $S_{bE} = S_b(v_E, a_E)$, $S_{fE} = S_f(v_E, a_E)$;
- $T_A = T_a(a_A)$, $T_E = T_a(a_E)$.

Our general procedure in the following treatment of the cases is based on the principle that we try to relate the general case BB as much as possible to the special case GG by either embedding the general case in a GG situation or extending a GG situation to the general case BB. We discuss case A in detail, elaborating on the lines of argumentation, and will then treat cases B–D shorter.

Case A: $a_A \leq 0$ and $a_E \geq 0$, $v_{fA} \leq v_{bE}$

In this case, E lies "right of" A in the sense that E lies on a square root curve (open to the right) right of the square root curve (open to the right) on which the point A lies or the curves are identical.

For a certain subset of all distances Δs, we can embed the task in a GG task for going from $(v_{bA}, 0)$ to $(v_{fE}, 0)$ via a $v_m \leq v_{fA}$ as is shown in Figure 7a. Making v_m smaller and inserting a phase with constant velocity v_{min} enables us to realize arbitrarily small values for Δs (mathematically spoken: Δs goes to $-\infty$ when the length of the phase with constant v_{min} goes to ∞). We get the corresponding distances by computing:

$$S(v_m) = S(v_{bA}, v_{fE}, v_m) - S_{bA} - S_{fE} \text{ for } v_m \leq v_{fA} \text{ (Part I)} \tag{7}$$

Note that this holds for $v_{min} \leq v_m$, but we will disregard, first, this restriction on v_m and allow v_m to be arbitrarily small and correct that later.

We can similarly cover arbitrarily large distances by extending the GG task going from $(v_{fA}, 0)$ to $(v_{bE}, 0)$ via a $v_m \geq v_{bE}$, as is shown in Figure 7b. When v_m reaches v_{max}, one can insert a phase with constant velocity v_{max}. Again, we will disregard this restriction on v_m first and correct that later. We obtain the corresponding distances by computing:

$$S(v_m) = S(v_{fA}, v_{bE}, v_m) + S_{fA} + S_{bE} \text{ for } v_m \geq v_{bE} \text{ (Part III).} \tag{8}$$

As we will see in examples, there might be a gap between the interval of distances covered in Part I and that covered in Part III. Distances in this gap can be achieved in two ways.

We again extend the GG task going from $(v_{fA}, 0)$ to $(v_{bE}, 0)$ via a v_m with $v_{bE} < v_m < v_{fE}$, where we let v_m run from right to left (!), i.e., from v_{fE} to v_{bE}, and then "cut off" the part going from v_{bE} to v_m and back to v_{bE} (shown in Figure 7c as a dashed pink curve). This continuously connects the distance covered in Part I when using $v_m = v_{fA}$ with the distance covered in Part III when using $v_m = v_{bE}$. Therefore, we have a full coverage of all possible real distances. We obtain the corresponding distances by computing:

$$S(v_m) = S(v_{fA}, v_{bE}, v_m) + S_{fA} + S_{bE} - 2 \cdot S(v_{bE}, v_m) \text{ for } v_{bE} < v_m < v_{fE} \text{ (Part IIa).} \tag{9}$$

Another way to close the gap is a bit more complicated and is shown in Figure 7d. The first part up to point C is the same as in Part IIa, but then we have the sequence $-j_{max}$, j_{max}, $-j_{max}$ instead of j_{max}, $-j_{max}$, j_{max} in Part IIa. We can embed part of the curve in a GG task going from $(v_{bE}, 0)$ to $(v_{fE}, 0)$ via a v_m with $v_{fA} < v_m < v_{bE}$, as is shown in Figure 7d. Actually, v_{fA} can be replaced by $\max\left\{v_{fA}, v_{bE} - \frac{a_{max}^2}{j_{max}}\right\}$ since at $v_{bE} - \frac{a_{max}^2}{j_{max}}$, the minimal acceleration $-a_{max}$ is reached and going on to the left with v_m does not further change the distance Δs since the parts added are cut off again later, as can be seen by drawing a figure which is omitted here. We can compute the corresponding distances in the following way:

$$S(v_m) = S(v_{bE}, v_{fE}, v_m) + S(v_{fA}, v_{bE}) + S_{fA} - S_{fE} - 2 \cdot S(v_{bE}, v_m) \text{ for}$$
$$\max\left\{v_{fA}, v_{bE} - \frac{a_{max}^2}{j_{max}}\right\} < v_m < v_{bE} \text{ (Part IIb).} \tag{10}$$

In our experiments (see the examples below), we have encountered a situation where a solution using Part IIb provides a shorter time than a solution using Part IIa, but then the distances were already covered by Part I or III, which provided even shorter times. However, since we have no mathematical proof that this is always the case, we simply include both possibilities. As we will see in the implementation section, this adds only the effort of finding the roots of three additional polynomials. Note that in case of $v_{fA} = v_{bE}$ or $a_E = 0$, parts IIa/b do not occur.

For each of the parts, the times needed can be easily computed:

$$T(v_m) = T(v_{bA}, v_{fE}, v_m) - T_A - T_E \text{ for } v_m \le v_{fA} \text{ (Part I).} \tag{11}$$

$$T(v_m) = T(v_{fA}, v_{bE}, v_m) + T_A + T_E \text{ for } v_m \ge v_{bE} \text{ (Part III).} \tag{12}$$

$$T(v_m) = T(v_{fA}, v_{bE}, v_m) + T_A + T_E - 2 \cdot T(v_{bE}, v_m) \text{ for } v_{bE} < v_m < v_{fE} \text{ (Part IIa).} \tag{13}$$

$$T(v_m) = T(v_{bE}, v_{fE}, v_m) + T(v_{fA}, v_{bE}) + T_A - T_E - 2 \cdot T(v_{bE}, v_m) \text{ for}$$
$$\max\left\{v_{fA}, v_{bE} - \frac{a_{max}^2}{j_{max}}\right\} < v_m < v_{bE} \text{ (Part IIb)} \tag{14}$$

As in the GG task, we can illustrate different situations by plotting $(S(v_m), T(v_m))$ as a parametric curve for parts I–III. We present a few examples showing interesting situations and behavior. In the examples, all distances have the unit m, velocities m/s, acceleration m/s^2 and jerk m/s^3. Units are omitted for the sake of brevity. As restrictions, we have chosen values similar as the ones used in [1]: $j_{max} = 30$, $a_{max} = 10$, $v_{max} = 20$.

Example 1: No gap between Parts I and III, but Parts IIa or IIb are better.

Initial and final conditions: $(v_A, a_A) = (10, -5)$, $(v_E, a_E) = (18, 9)$. Note that the colors used in the following figures have nothing to do with the colors used in Figure 7. The corresponding curve is shown in Figure 8a, where the green part (Part I) goes on to the left, covering arbitrarily small (negative) distances, and the black part (Part III) goes on to the right, covering arbitrarily large distances. One can see that Parts I and III cover all distances but Parts IIa and IIb provide solutions needing less time in the interval of distances where they are valid. There is no huge difference between Parts IIa and IIb, but when zooming in (Figure 8b), one can see that, in this example, the red part (Part IIa) is better. Figure 8 also shows that for a given distance, there might be several solutions in different parts or in the same part, as we have already recognized in the GG situation in Section 3. For example, for the distance $\Delta s = 18$, there are two solutions in Part I and one solution each in Parts IIa and IIb.

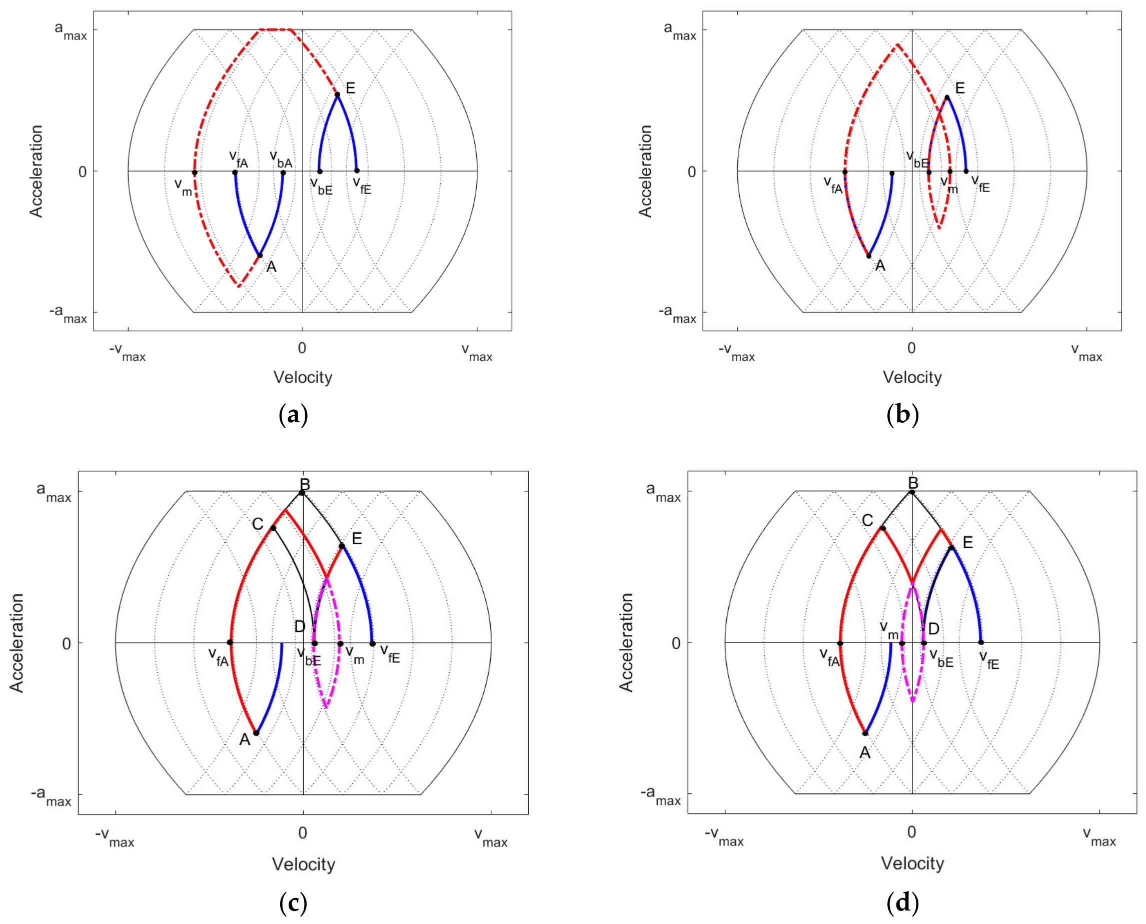


Figure 7. Case A. (**a**) Part I; (**b**) Part III; (**c**) Part IIa; (**d**) Part IIb.

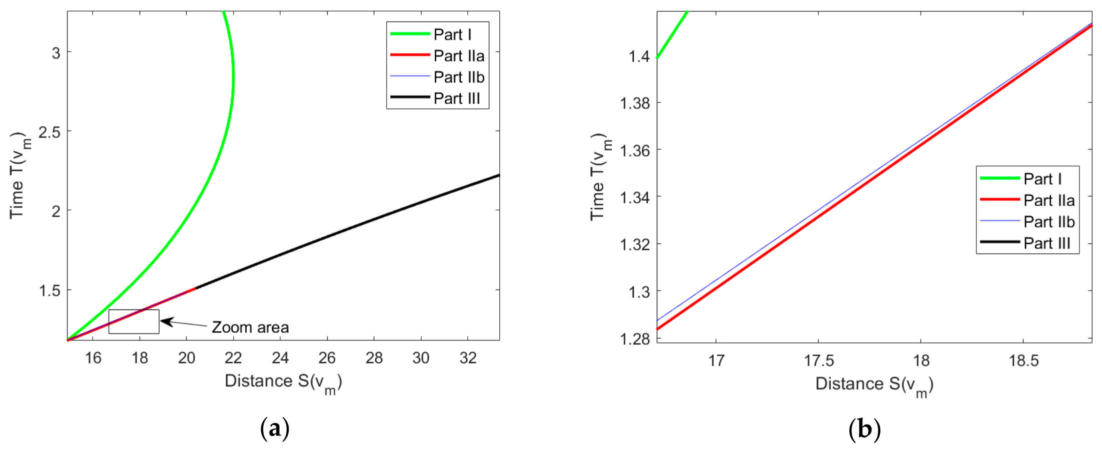


Figure 8. (**a**) No gap between Parts I and III, but Parts IIa or IIb are better; (**b**) zoom into area.

Example 2: No gap between Parts I and III, Parts IIa and IIb are not needed.

Initial and final conditions: $(v_A, a_A) = (-18, -5)$, $(v_E, a_E) = (-10, 9)$. We use the same accelerations as in Example 1 but shift the velocities to the left by 28. In this way, variations are also produced in [22].

The corresponding curve is shown in Figure 9a. Again, the green part (Part I) goes on to the left, covering arbitrarily small (negative) distances, and the black part (Part III) goes on to the right, covering arbitrarily large distances. Parts I and III cover all distances and one of them provides the best solution, so Parts IIa and IIb are not needed. There is no huge difference between Parts IIa and IIb, but when zooming in (Figure 9b), one can see that, in this example, the blue part (Part IIb) is better. Part IIb provides a solution where the jerk changes sign three times (see Figure 7d), whereas in the solution of Part IIa, the jerk changes sign only two times. Therefore, the conjecture that solutions with less changes are better, which might look plausible at first sight, is wrong. What might be true is that the best solution (here from Part I) always has the lowest number of sign changes, but we do not have mathematical proof for this conjecture.

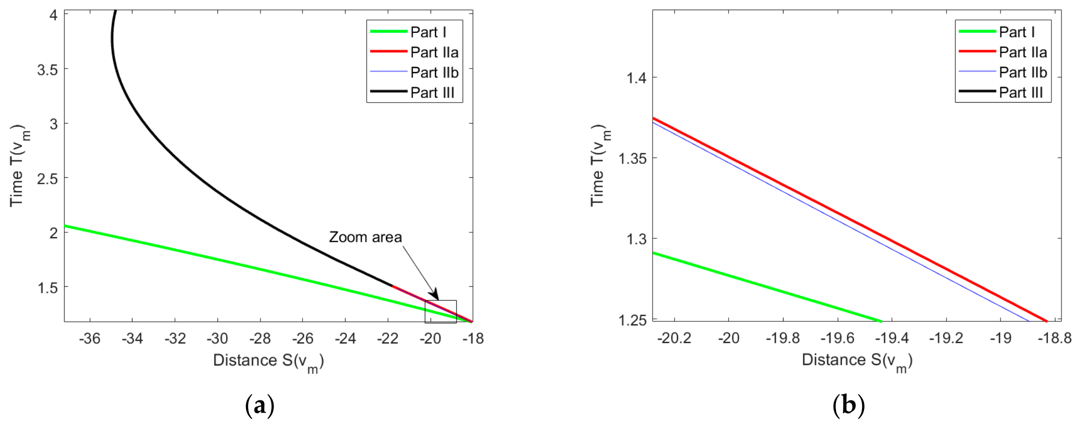


Figure 9. (a) No gap between Parts I and III, Parts IIa or IIb are not needed; (**b**) zoom into area.

Example 3: Gap between Parts I and III, Parts IIa or IIb are needed.

Initial and final conditions: $(v_A, a_A) = (-1.5, -5)$, $(v_E, a_E) = (6.5, 9)$. We use the same accelerations as in Example 1 but shift the velocities by -11.5. For Parts I and III, the remarks given in Examples 1 and 2 are still valid; see Figure 10. However, here, in both parts, the distance changes monotonously. This example shows that without using the constructions of Parts IIa or IIb, a certain range of distances (here, roughly from 1.4 to 3) might not be covered. In the example, Part IIa provides the better solution and we have not encountered an example of case A where Part IIb is best, but we do not have proof that this is always so.

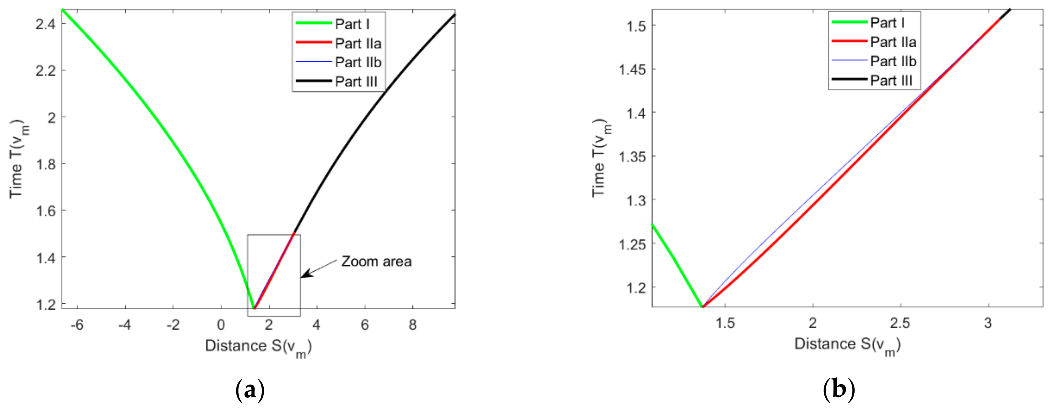


Figure 10. (**a**) Gap between Parts I and III, Parts IIa or IIb are needed; (**b**) zoom into area.

Example 4: Best solution is in Part IIa where the curve is "bulging".

Initial and final conditions: $(v_A, a_A) = (-7.2, -5)$, $(v_E, a_E) = (0.8, 9)$. We use the same accelerations as in Example 1 but shift the velocities to the left by -17.2.

In this example, zooming shows that the red Part IIa curve "bulges" to the right (Figure 11b) (cf. similar examples in [22]). Therefore, there is a small interval of distances to the right by about -5.34 where Part IIa provides the best solution, ending at the S-value of the utmost right point of the red curve. This example demonstrates that for finding the best solution, it is, in general, not sufficient to just consider the interval of distances between the "end points" of the parts. If one considers the interval of S-values between the end points of the red curve, then on that interval, the Part I curve (green) is always better. Therefore, an approach working with "critical distances" as in [20] has its limits.

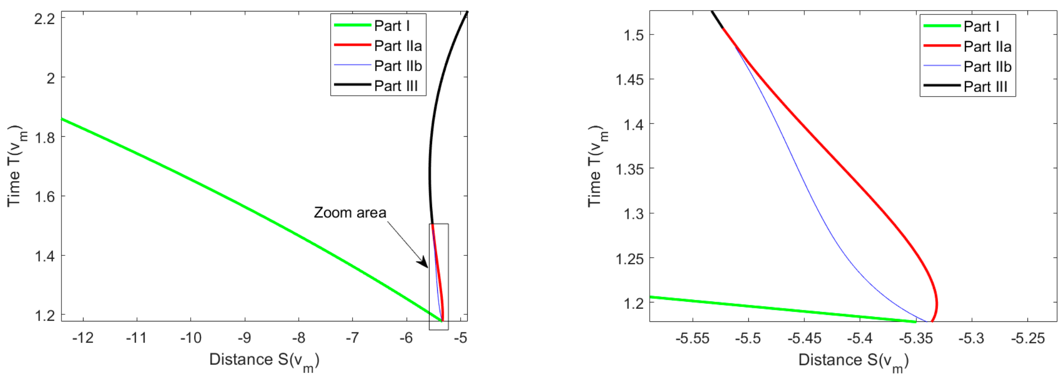


Figure 11. (**a**) Best solution is in Part IIa where the curve is "bulging"; (**b**) zoom into area.

The examples given above show that a visualization using a parametric (S, T) curve allows to easily create a variety of examples for finding counter-examples to conjectures one might have in mind. This has also been presented in [22] using a different parameterization. Using v_m allows us to easily recognize in which part the best solution occurs. Moreover, the parameter v_m has a direct meaning in the velocity–acceleration plane. One might be inclined to look for further patterns to see in advance which part provides the best solution, but even this small set of examples shows the variability. We will make no further distinctions but just compute the solution(s) provided by each part and take the best one (this approach is also used in [22]). We consider this as a sound compromise between complexity (comprehensibility) on the one hand and computational effort on the other hand. We find the best solution within the solutions provided by Parts I–III by inserting the given distance in Equations (7)–(10) and computing v_m. As we will see in the next section, this requires finding the roots of polynomials. For each solution candidate, it is checked whether v_m is admissible (real and within the allowed range). For the admissible ones, one computes the times and chooses the solution with lowest time. If $v_m < v_{min}$ or $v_m > v_{max}$, one sets $v_m = v_{min}$ resp. $v_m = v_{max}$ and inserts a phase with constant (minimum or maximum) velocity.

Case B: $a_A \leq 0$ and $a_E \geq 0$, $v_{fA} > v_{bE}$

In this case, E lies "left of" A in the sense that E lies on a square root curve (open to the right) left of the square root curve (open to the right) that the point A lies on. We can proceed in the same way as we did in case A and have, again, four parts, as shown in Figure 12.

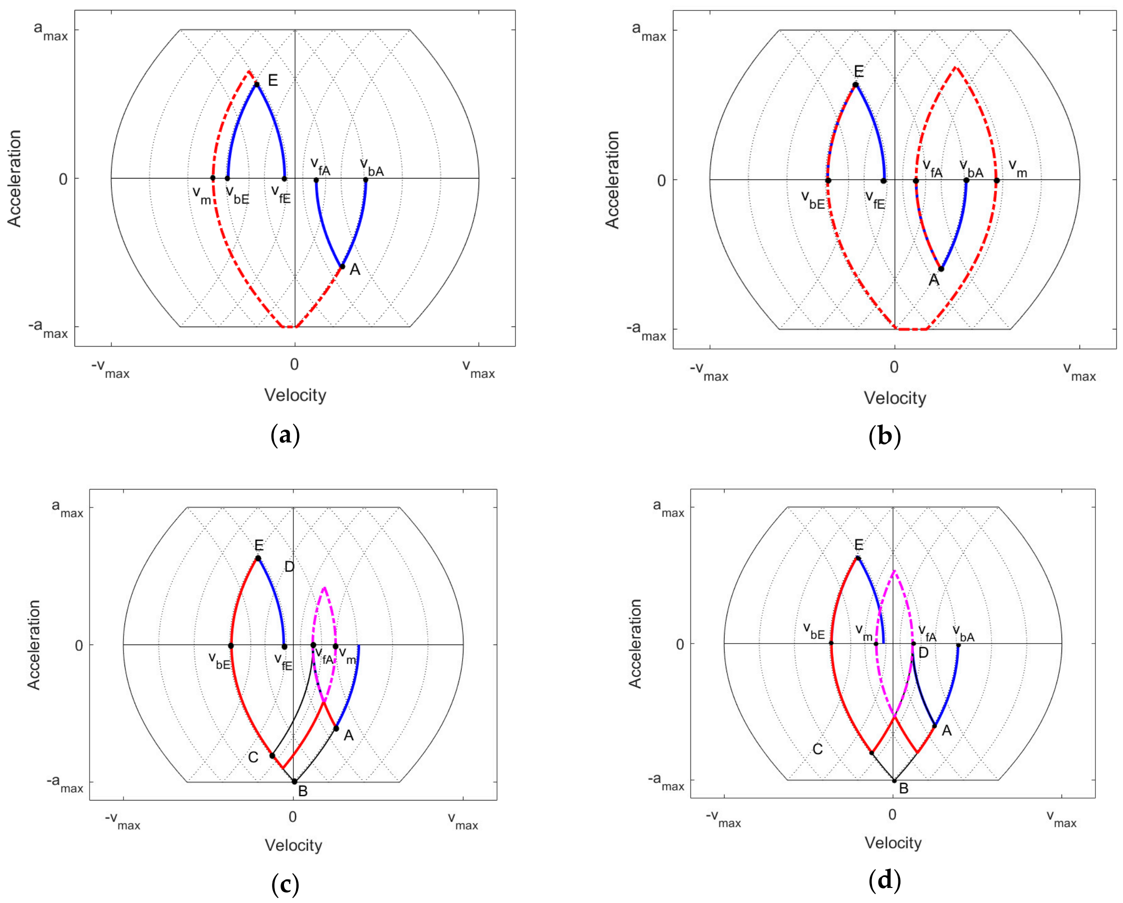


Figure 12. Case B. (**a**) Part I; (**b**) Part III; (**c**) Part IIa; (**d**) Part IIb.

Again, with Part I resp. III, we can realize arbitrarily small resp. large distances going via intermediate velocities v_m. Arguing as in case A, we obtain for Parts I and III the same formulae as in (7) and (8) resp. (11) and (12), but the ranges for v_m are different for Part I $v_m \le v_{bE}$ and for Part III $v_m \ge v_{fA}$. The formulae for Parts IIa and IIb are different and provided below.

$$S(v_m) = S(v_{fA}, v_{bE}, v_m) + S_{fA} + S_{bE} - 2 \cdot S\left(v_{fA}, v_m\right) \text{ for } v_{fA} < v_m < v_{bA} \text{ (Part IIa)}. \tag{15}$$

$$S(v_m) = S(v_{fA}, v_{bE}, v_m) + S_{fA} + S_{bE} - 2 \cdot S\left(v_{fA}, v_m\right) \text{ for } v_{fA} < v_m < v_{bA} \text{ (Part IIa) for}$$
$$\max\left\{v_{bE}, v_{fA} - \frac{a_{max}^2}{j_{max}}\right\} < v_m < v_{fA} \text{ (Part IIb)}. \tag{16}$$

$$T(v_m) = T(v_{fA}, v_{bE}, v_m) + T_A + T_E - 2 \cdot T\left(v_{fA}, v_m\right) \text{ for } v_{fA} < v_m < v_{bA} \text{ (Part IIa)}. \tag{17}$$

$$T(v_m) = T(v_{bA}, v_{fA}, v_m) + T(v_{fA}, v_{bE}) + T_E - T_A - 2 \cdot T\left(v_{fA}, v_m\right) \text{ for}$$
$$\max\left\{v_{bE}, v_{fA} - \frac{a_{max}^2}{j_{max}}\right\} < v_m < v_{fA} \text{ (Part IIb)}. \tag{18}$$

We omit examples for this case since they provide no new insights. The procedure for computing the best solution follows the same lines as the one described in case A.

Case C: $a_A \ge 0$ and $a_E \ge 0$, $v_{bA} \le v_{bE}$ and $v_{fA} \le v_{fE}$

In this case, E lies "right of" A in the sense that E lies in the area right of or on the square root curve through A open to the right, and right of or on the square root curve through A open to the left (Figure 13).

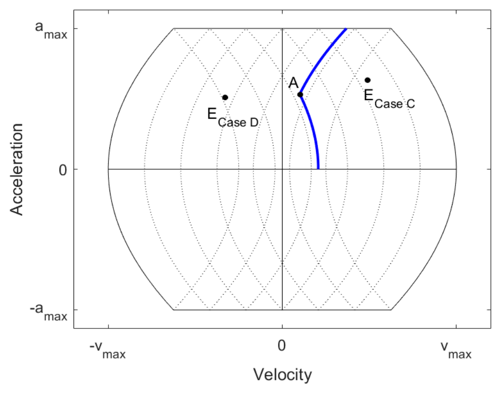


Figure 13. Positioning of A and E in cases C and D.

We split this case up into two sub-cases ($v_{bE} \leq v_{fA}$ and $v_{bE} > v_{fA}$) that need partially different treatments. The sub-cases are shown in Figure 14. In the first sub-case (left), we have a closed curved quadrilateral ABED, whereas in the second one (right), we have a curved pentagon, ABCED. Note that in the first sub-case, there might also occur a curved pentagon when a_{max} is reached, but this does not require a further distinction.

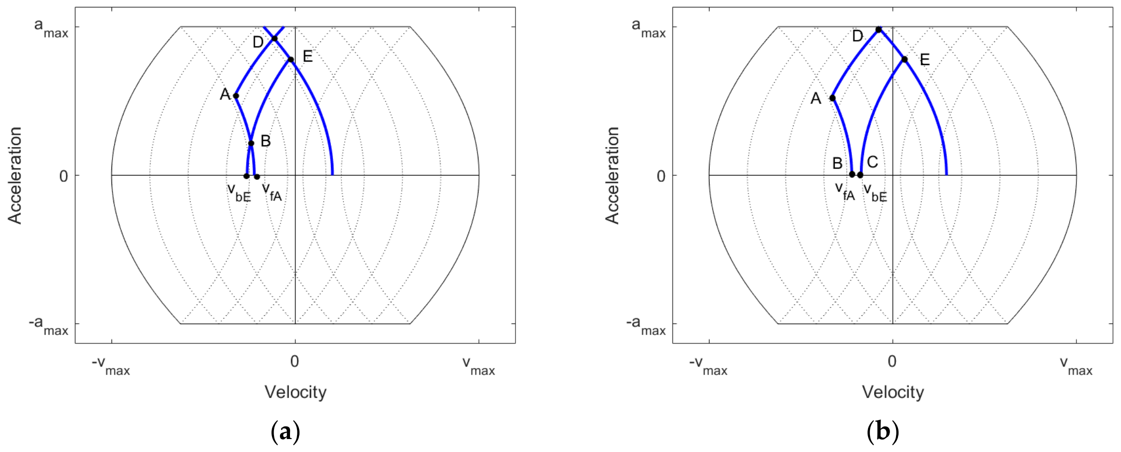


Figure 14. (**a**) Sub-case Ca; (**b**) sub-case Cb.

Sub-Case Ca: $v_{bE} \leq v_{fA}$

As in case A, we split up the construction of solutions for all distances into different parts, as shown in Figure 15. With Part I resp. III, we can realize arbitrarily small resp. large distances going via intermediate velocities $v_m \leq v_{bE}$ resp. $v_m \geq v_{fA}$. If equality holds, then Part I and Part III provide the same point and, hence, the same distance Δs. Therefore, Parts I and III already cover all distances but do not necessarily provide the shortest time. Some distances can be realized better by going directly within the curved quadrilateral ABED using the sequence j_{max}, $-j_{max}$, j_{max} in Part IIa or the sequence $-j_{max}$, j_{max}, $-j_{max}$ in Part IIb (see Figure 15c,d). We will give an example for this situation below.

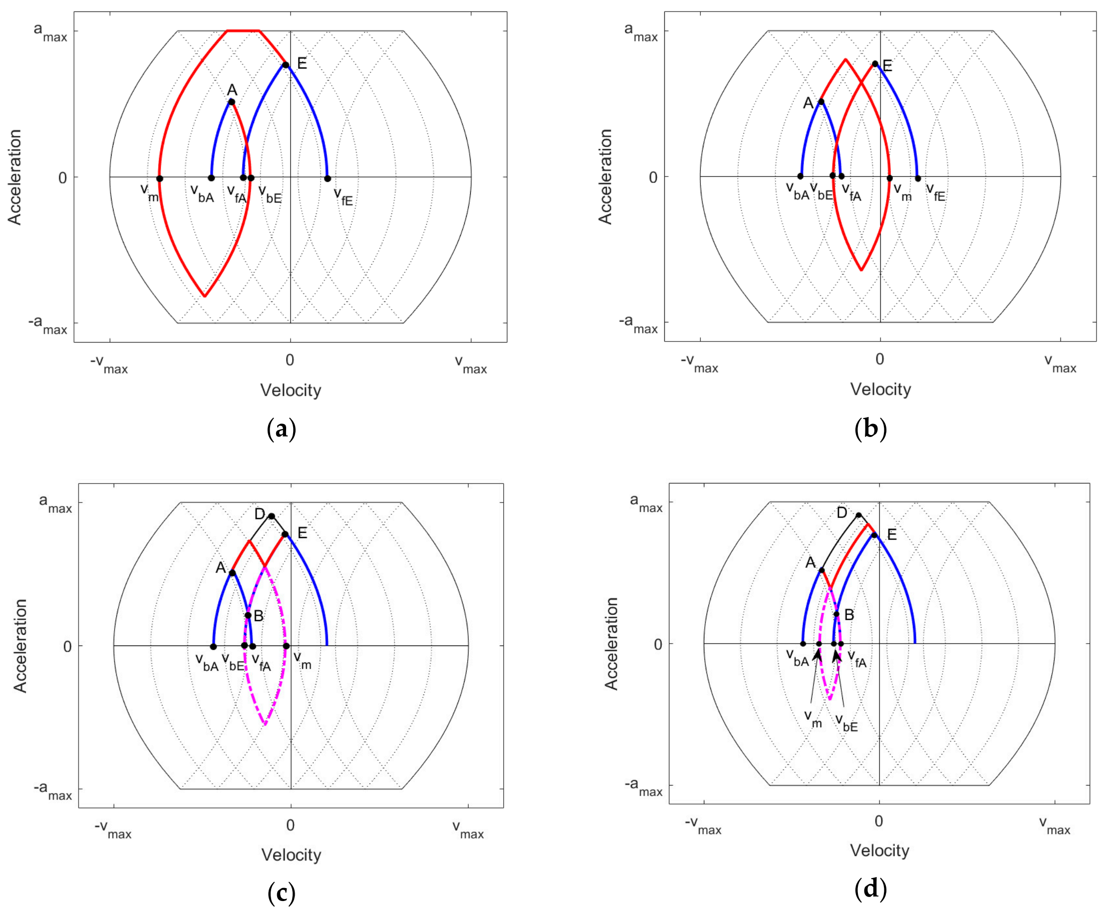


Figure 15. Case Ca. (**a**) Part I; (**b**) Part III; (**c**) Part IIa; (**d**) Part IIb.

Arguing as in case A, we obtain the following formulae for distances and times:

$$S(v_m) = S(v_{fA}, v_{fE}, v_m) + S_{fA} - S_{fE} \text{ for } v_m \le v_{bE} \text{ (Part I)}. \tag{19}$$

$$S(v_m) = S(v_{bA}, v_{bE}, v_m) - S_{bA} + S_{bE} \text{ for } v_m \ge v_{fA} \text{ (Part III)}. \tag{20}$$

$$S(v_m) = S(v_{bA}, v_{bE}, v_m) - S_{bA} + S_{bE} - 2 \cdot S(v_{bE}, v_m) \text{ for } v_{fA} < v_m < v_{fE} \text{ (Part IIa)}. \tag{21}$$

$$S(v_m) = S(v_{fA}, v_{fE}, v_m) + S_{fA} - S_{fE} - 2 \cdot S\Big(v_{fA}, v_m\Big) \text{ for } v_{bA} < v_m < v_{bE} \text{ (Part IIb)}. \tag{22}$$

$$T(v_m) = T(v_{fA}, v_{fE}, v_m) + T_A - T_E \text{ for } v_m \le v_{bE} \text{ (Part I)}. \tag{23}$$

$$T(v_m) = T(v_{bA}, v_{bE}, v_m) - T_A + T_E \text{ for } v_m \ge v_{fA} \text{ (Part III)}. \tag{24}$$

$$T(v_m) = T(v_{bA}, v_{bE}, v_m) - T_A + T_E - 2 \cdot T(v_{bE}, v_m) \text{ for } v_{fA} < v_m < v_{fE} \text{ (Part IIa)} \tag{25}$$

$$T(v_m) = T(v_{fA}, v_{fE}, v_m) + T_A - T_E - 2 \cdot T\Big(v_{fA}, v_m\Big) \text{ for } v_{bA} < v_m < v_{bE} \text{ (Part IIb)}. \tag{26}$$

We provide some examples that give an insight into the variability of situations. We use the same values for the restrictions as in the examples given for Case A, i.e., $j_{max} = 30$, $a_{max} = 10$, $v_{max} = 20$. We have, again, one set of initial and final conditions and shift this in the v-direction. The values used for producing Figure 16 are: $(v_A, a_A) = (-3.3, 8)$, $(v_E, a_E) = (-0.3, 11.8)$.

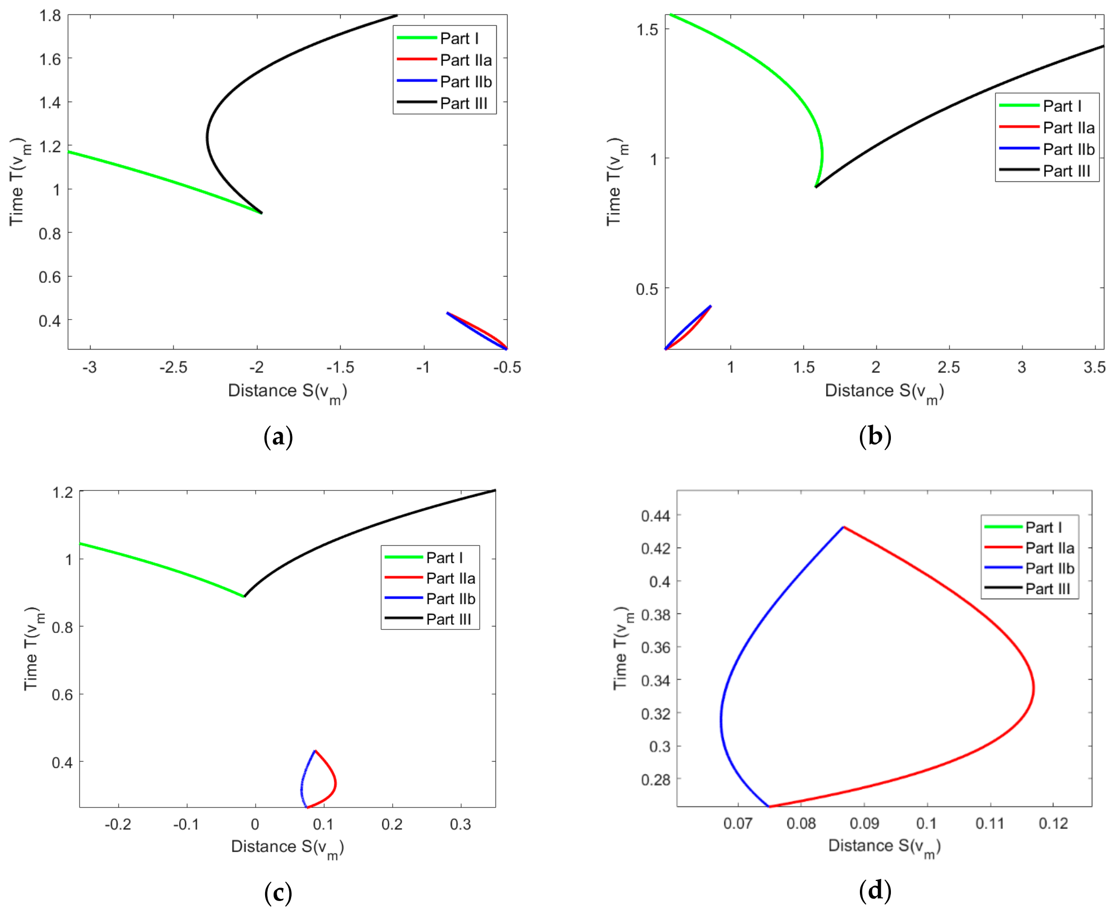


Figure 16. (**a**) Example 1; (**b**) example 2; (**c**): example 3; (**d**) example 3 with zoom.

Example 1: Shift = 0, i.e., we use exactly the data given above.

Example 2: Shift = 4, i.e., we use: $(v_A, a_A) = (0.7, 8)$, $(v_E, a_E) = (3.7, 11.8)$.

We recognize that the parts are no longer connected, but there are two "connectivity components", Parts I/III and IIa/b. It can also be seen that Parts I and III provide solutions for all distances, but for the distances for which Parts IIa and IIb are applicable, they provide shorter times. In example 1, Part IIb is better than Part IIa, and in example 2, it is the other way around.

Example 3: Shift = 2.2, i.e., we use: $(v_A, a_A) = (-1.1, 8)$, $(v_E, a_E) = (1.9, 11.8)$.

In this example, we also recognize the disconnectedness. In addition, we see that both the red curve (Part IIa) and the blue curve (Part IIb) "bulge". Part IIb provides the shortest solution within a range of distances from about 0.068 to about 0.074, whereas for Part IIa, this holds for a range from about 0.074 to about 0.116. This again shows that an approach considering different cases only between the distances reached at the boundary points of the parts cannot always provide the best solution.

Sub-Case Cb: $v_{bE} > v_{fA}$

This case is depicted in Figure 14b. Regarding Part I resp. III, the same holds as was stated for sub-case Ca, i.e., we can realize arbitrarily small resp. large distances going via intermediate velocities, but here, $v_m \leq v_{fA}$ resp. $v_m \geq v_{bE}$ (see Figure 17a,b). Moreover, if equality holds, then Part I and Part III do not provide the same point and, hence, the same distance Δs.

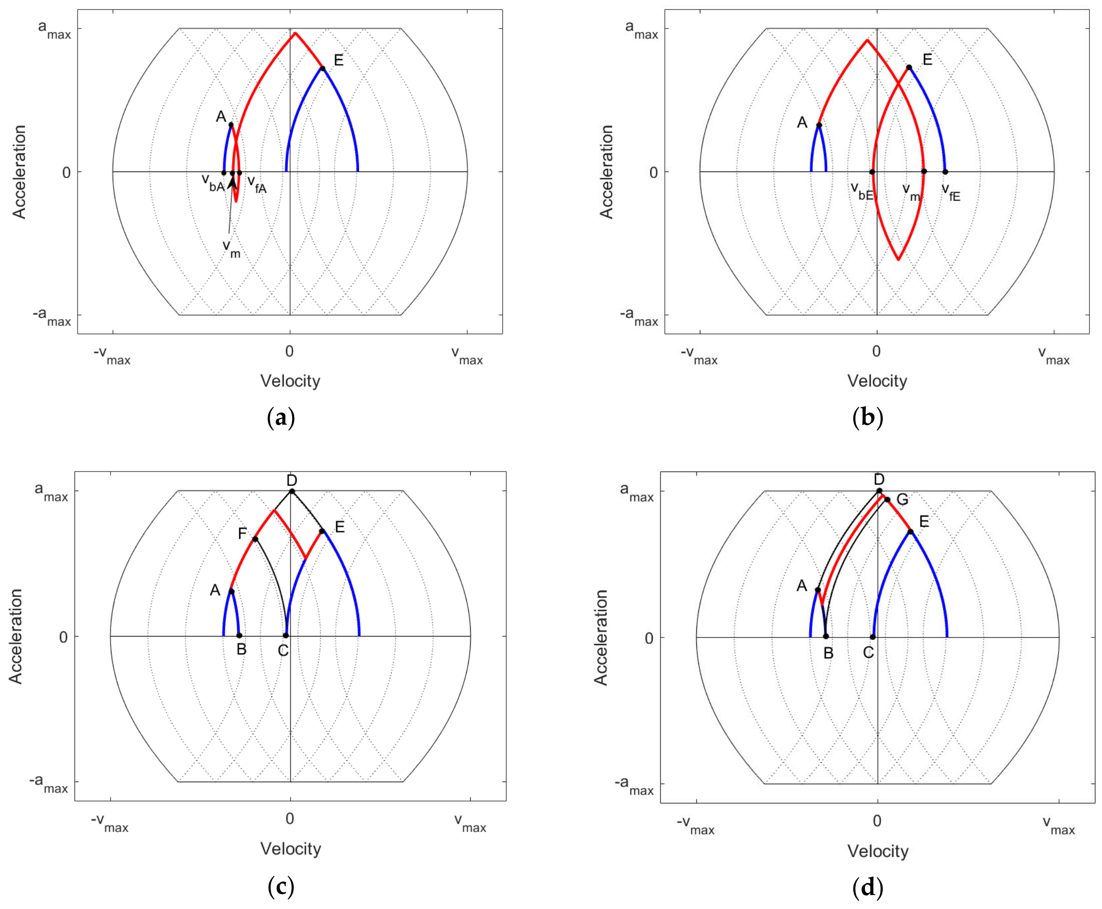


Figure 17. Case Cb. (**a**) Part I; (**b**) Part III; (**c**) Part IIa; (**d**) Part IIb.

In order to guarantee the continuity of covered distances, we again include two other parts called Part IIa and IIb, using the sequence $j_{max}, -j_{max}, j_{max}$ in Part IIa and the sequence $-j_{max}, j_{max}, -j_{max}$ in Part IIb (see Figure 17c,d). In Part IIa, the motion curve starting at A moves up in the direction of D beyond the point F, then moves down to a point on the curve CE and then up to E (see Figure 17c). This is much the same as shown in Figure 15c for the previous sub-case Ca, but note that now, the motion curve must go beyond F (otherwise one would need a sequence with three changes in jerk instead of two). In Part IIb, the motion curve starting at A moves down in the direction of B, then moves up to a point on the curve DE (which lies above the point G) and then down to E (see Figure 17d). We checked that all possible distances are covered by using continuity: If in Part I we choose $v_m = v_{fA}$, then we obtain the motion curve ABGE. This is also the motion curve one obtains in Part IIb when one goes down to the point B. At the other "end" of Part IIb, one obtains the motion curve ADE (i.e., going down by 0). This is also the motion curve one obtains in Part IIa when one goes up from A to D. At the other "end" of Part IIa, one obtains the motion curve AFCE. We also obtains this motion curve when we choose $v_m = v_{bE}$ in Part III. Therefore, Parts IIa and b fulfill the purpose to continuously connect Parts I and III. As we will see in the examples, in this sub-case, the special situation might occur that we need both Parts IIa and IIb to have a full coverage of all distances, whereas in the other cases considered so far, only one of the parts was needed for this purpose.

The formulae for distances and time are the same as in the previous sub-case (i.e., Equations (19)–(26)) but the ranges for v_m are different. Therefore, we just provide these ranges:

$$v_m \leq v_{fA} \text{ (Part I)},\ v_m \geq v_{bE} \text{ (Part III)}\ ,\ v_{bE} < v_m < v_{fE} \text{ (Part IIa)},\ v_{bA} < v_m < v_{fA} \text{ (Part IIb)}. \quad (27)$$

We provide some examples that give an insight into the variability of situations and the specifics of this sub-case. We use the same values for the restrictions as in the examples given for case A, i.e., $j_{max} = 30$, $a_{max} = 10$, $v_{max} = 20$. We have, again, one set of initial and final conditions and shift this in the v-direction: $(v_A, a_A) = (0.5, 4.9)$, $(v_E, a_E) = (2.76, 7.9)$.

Example 1: Shift = 0, i.e., we use exactly the data given above (Figure 18a).

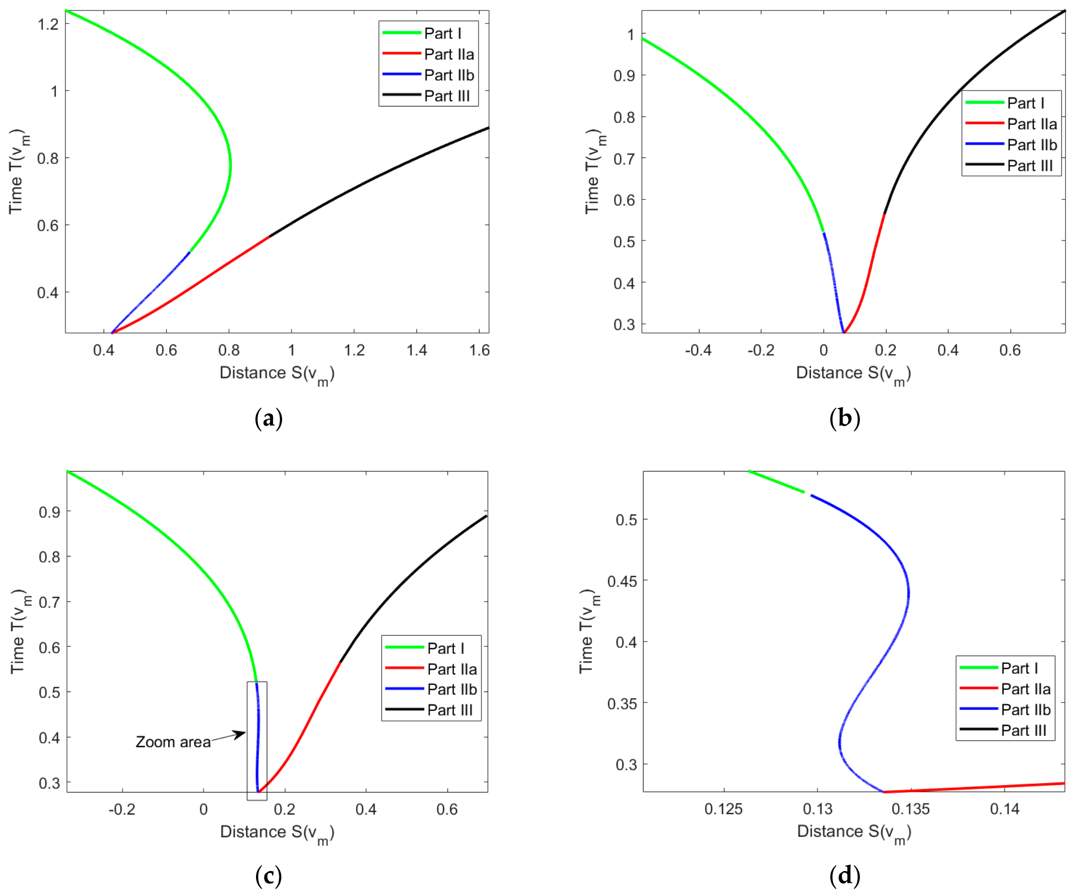


Figure 18. (**a**) Example 1; (**b**) example 2; (**c**) example 3; (**d**) example 3 with zoom.

In this example, there is a gap between Parts I and III. Only Part IIa is needed to fill the gap. It also provides the shortest time where it is valid. One can see how one goes continuously from Part I via IIb and IIa to III as described above.

Example 2: Shift = -1.3, i.e., we use: $(v_A, a_A) = (-0.8, 4.9)$, $(v_E, a_E) = (1.46, 7.9)$ (Figure 18b).

Here, both parts IIa and IIb are needed to cover all possible distances. For a certain distance, there is only one part that covers that distance and is, hence, automatically the best one.

Example 3: Shift =-1.05, i.e., we use: $(v_A, a_A) = (-0.55, 4.9)$, $(v_E, a_E) = (1.71, 7.9)$ (Figure 18c).

This example looks, at first, similar to example 2, but when one zooms into the interesting area, one can recognize that for some distances covered by Part IIb (blue), there is also a coverage by Part IIa with a shorter time. Moreover, for some distances (e.g., 0.132), there are even three solutions within Part IIb, one of them having the shortest time. This again shows the variability of situations.

Case D: $a_A \geq 0$ and $a_E \geq 0$, $v_{bA} > v_{bE}$ or $v_{fA} > v_{fE}$

This case is illustrated in Figure 19. Case D is the easiest one since it can be completely reduced to the GG situation by a combination of embedding and extension and we just need Parts I and III. Figure 19 shows situations when both of the two velocity conditions are fulfilled, but it is easy to check that nothing changes when only one of them holds. Consider the GG task for going from $(v_{fA}, 0)$ to $(v_{fE}, 0)$ via a $v_{m_I} \leq v_{bE}$. If one extends this by going from A to $(v_{fA}, 0)$ and cuts off the part going from E to $(v_{fE}, 0)$, one obtains a motion curve from A to E with a certain distance. Making v_{m_I} smaller and inserting a phase with velocity v_{min} enables us to realize arbitrarily small values for Δs. Similarly, the red motion curve can be constructed from the GG task going from $(v_{bA}, 0)$ to $(v_{bE}, 0)$ via a $v_{m_{II}} \geq v_{fA}$. Again, making $v_{m_{II}}$ larger and inserting a phase with velocity v_{max} enables us to realize arbitrarily large values for Δs. Since for $v_{m_I} = v_{bE}$ and $v_{m_{II}} = v_{fA}$ we obtain the same motion curve, we continuously cover all distances.

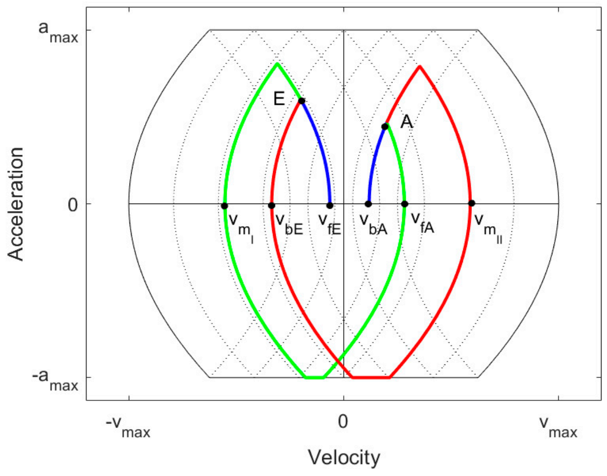


Figure 19. Case D.

We obtain the same formulae for distances and times as for Parts I and III in Case Ca, i.e., Equations (19), (20), (23), (24) also hold in this case. Since there are no remarkable special situations occurring in this case, we give no examples.

5. Implementation and Validation with MATLAB®

In both the GG and the BB situations, beside splitting into several cases, which is easy to implement, the main task consists of solving an equation. In GG and in Parts I and III of the BB cases, the equation is of the type:

$$w = S(v_A, v_E, v_m) = S(v_A, v_m) + S(v_m, v_E) \tag{28}$$

where v_m is unknown and the distance w and the velocities v_A, v_E are given. The equations for Parts IIa and IIb in the BB situation (9), (10), (15), (16), (21) and (22) can be rewritten in the form:

$$w = S(v_A, v_m) - S(v_m, v_E). \tag{29}$$

We abbreviate the difference on the right hand side as $S_{neg}(v_A, v_E, v_m)$.

We split up Equation (28) into sub-cases depending on whether the maximum acceleration is reached going from v_A to v_m resp. from v_m to v_E (cf. Equations (1) and (2)):

$$w = (v_A + v_m) \cdot \sqrt{\frac{|v_m - v_A|}{j_{max}}} + (v_m + v_E) \cdot \sqrt{\frac{|v_E - v_m|}{j_{max}}} \tag{30}$$

$$w = (v_A + v_m) \cdot \sqrt{\frac{|v_m - v_A|}{j_{max}}} + \left(\frac{v_m + v_E}{2}\right) \cdot \left(\frac{|v_E - v_m|}{a_{max}} + \frac{a_{max}}{j_{max}}\right) \tag{31}$$

$$w = \left(\frac{v_A + v_m}{2}\right) \cdot \left(\frac{|v_m - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}}\right) + (v_m + v_E) \cdot \sqrt{\frac{|v_E - v_m|}{j_{max}}} \tag{32}$$

$$w = \left(\frac{v_A + v_m}{2}\right) \cdot \left(\frac{|v_m - v_A|}{a_{max}} + \frac{a_{max}}{j_{max}}\right) + \left(\frac{v_m + v_E}{2}\right) \cdot \left(\frac{|v_E - v_m|}{a_{max}} + \frac{a_{max}}{j_{max}}\right) \tag{33}$$

An analogous set of equations can be produced for (29) using minus instead of plus between the two parts. Moreover, a closer investigation of the interval where v_m lies in Parts IIa and IIb shows that sub-case (33) does not occur, so when considering Equation (29), we can restrict ourselves to (30)–(32). Available numerical equation solvers for (28) and (29) normally find, at most, one solution and then stop. Therefore, we transform the equations symbolically into problems where all zeros of polynomials are to be found. For doing this, we first remove the absolute values by making a further distinction between the cases (α) $v_m \leq v_A, v_E$ and (β) $v_m \geq v_A, v_E$ (the overlap at equality does not matter). For example, in case (α), we write $v_A - v_m$ for $|v_m - v_A|$. Using a Computer Algebra System (here, Maple®), one can then reduce the task of solving the equations (30)–(32) to find the roots of polynomials of degree 4 and one obtains even a symbolic solution for (33).

We first consider the equation $w = S(v_A, v_E, v_m)$. The polynomials for Equations (30)–(32) are given in Table A1 (Appendix A) in form of lists of coefficients (starting from the largest power) and the solutions for Equation (33) are given directly. If we consider instead the equation $w = S_{neg}(v_A, v_E, v_m)$, we obtain only polynomials for the modified Equations (30)–(32), as shown in Table A2 (Equation (33) does not occur as stated above). Once all zeros z of the polynomials for the (modified) Equations (30)–(32) have been found using a polynomial root solver (here: MATLAB®), the solutions for Equations (30)–(32) for both equations $w = S(v_A, v_E, v_m)$ and $w = S_{neg}(v_A, v_E, v_m)$ can be found according to Table A3 (provided by the Computer Algebra System).

All solution candidates are then further filtered:

- Only the real solutions (not the complex ones) are taken.
- It is tested whether the solutions fulfill the case conditions in (α) resp. (β).
- It is tested whether the solutions fulfill the requirements of Equations (30)–(33) regarding reaching the maximum/minimum acceleration.
- Not all zeros of the polynomial might solve the original equation (only the implication that all solutions of the equation can be found using zeros of the polynomials and the computations given in Table A3 holds in general; there is no equivalence relationship). Therefore, all zeros are tested (by insertion) and only those ones fulfilling the equation are kept.

Only those solutions that pass the tests are then used for further processing. This algorithm is described semi-formally in pseudo-code in Algorithm 1.

Algorithm 1. Solving $w = S(v_A, v_E, v_m)$ (resp. $w = S_{neg}(v_A, v_E, v_m)$).

Input: w, v_A v_E, a_{max}, j_{max}, *case* (**α** *or* **β**)

Output: Candidates for v_m

```
1.  If case = α Then
2.    Compute the zeros of polynomial (30) in Table A1 (resp. A2) listed under (α)
3.  From the zeros compute candidates for $v_m$ according to Table A3 listed under (α)
4.  For each candidate $v_m$ do
5.  If $v_m$ is not real Then omit $v_m$ End If
6.  If not $(v_m \leq v_A, v_E)$ Then omit $v_m$ End If % Not case α
7.  If not $\left(|v_m - v_A| < \frac{a_{max}^2}{j_{max}}\right.$ and $|v_m - v_E| < \frac{a_{max}^2}{j_{max}}$) Then omit $v_m$ End If % necessary condition for (30)
8.  If not $(w = S(v_A, v_E, v_m))$ (resp. not $(w = S_{neg}(v_A, v_E, v_m))$ Then omit $v_m$ End If % equation not solved
9.  End For
10.  Do the same (lines 2–9) for polynomials (31) and (32) adapting the inequalities in line 7
11.  Compute candidates for $v_m$ directly from (33) in Table A1 (not applicable for $w = S_{neg}(v_A, v_E, v_m)$)
12.  If applicable Then Do the same as in lines 4–9, adapting the inequalities in line 7 End if
13.  else if case = β Then
14.  Do the same as in case α now using the polynomials/expressions listed under case β in Table A1 (resp. A2)
15.  End If
16.  Return the remaining candidates for $v_m$ % might be the empty set
```

If we have the BB task, we first determine which of the cases, A to D, is applicable (or rewrite the problem by changing the direction of distance). In the respective case, we determine the solutions for an equation of type $w = S(v_A, v_E, v_m)$ for Parts I and III and the solutions for an equation of type $w = S_{neg}(v_A, v_E, v_m)$ for Parts IIa and IIb as described above. Note that even for Parts I and III where we can relate the BB task to a GG task by embedding or extension, we cannot simply use an algorithm for the GG task since we have to take into account the restrictions on the interval where v_m must lie. We have to determine the zeros of, at most, 12 polynomials of degree 4 (four parts with three polynomials each at most). Then, we additionally check whether the solutions for v_m are within the given range for the respective part (the ranges are given in the equations above) and drop those who are not. If $v_m < v_{min}$ or $v_m > v_{max}$, we insert a phase with constant velocity. We then compute the lengths for the different time intervals in the seven-segment profile and sum these up. The candidate with the lowest sum is then the best one within the set of all candidates considered. The algorithm is described semi-formally in Algorithm 2.

Algorithm 2. Computing the time intervals and signs for the 7-segment profile.

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

Output: Time vector $[t_1, t_2, t_3, t_4, t_5, t_6, t_7]$, sign vector for j_{max} in the intervals where $j = \pm j_{max}$

```
1: **If not** case $\in \{A, B, Ca, Cb, D\}$ **Then**
2: Set $\Delta s - \Delta s,\ v_A = -v_A,\ a_A = -a_A,\ v_E = -v_E,\ a_E = -a_E$
3: **End If**
4: **If** case = A Then
5: **Call** Algorithm 1 according to Equation (7)
6: **For each** $v_m$ returned
7: **If** condition on $v_m$ in Equation (7) is not fulfilled **Then**
8: omit $v_m$
9: **Else**
10: **If** $v_m < -v_{max}$ **Then** Set $v_m = -v_{max}$ **End If** % Take into account restriction on v
11: Compute time vector and sign vector
12: **End if**
13: **End For**
14: **Call** Algorithm 1 according to Equation (8) and perform lines 6–13
15: **Call** Algorithm 1 with $w = S_{neg}(v_A, v_E, v_m)$ according to Equation (9) and perform lines 6–13
16: **Call** Algorithm 1 with $w = S_{neg}(v_A, v_E, v_m)$ according to Equation (10) and perform lines 6–13
17: **Else If** case = B
18: Execute lines 5–16 using the equations of case B
19: **Else If** case = Ca
20: Execute lines 5–16 using the equations of case Ca
21: **Else If** case = Cb
22: Execute lines 5–16 using the equations of case Cb
23: **Else If** case = D
24: Execute lines 5–16 using the equations of case D
25: **End If**
26: **Return** Time vector $[t_1, t_2, t_3, t_4, t_5, t_6, t_7]$ and sign vector for $j_{max}$ where $t_7 = min$
```

If we have a GG task, we can determine in advance which one of the Equations (30)–(33) is applicable and determine whether we have case (α) or (β) (we omit the details). Therefore, we would have to find the roots of at most one polynomial.

Regarding validation, we have to state first what we intend to validate. Since the presented algorithm logically covers all possible situations by the continuity argument we explained in the previous section, theoretically, there is no necessity for validation otherwise than checking for a mistake in the argumentation that might come up during test runs. What certainly needs validation is the implementation of the algorithm, particularly regarding numerical issues such as cancellation effects (accuracy). Moreover, for checking the algorithm's suitability for online trajectory planning, it is also important to investigate efficiency, i.e., the time necessary for finding the solution. Our accuracy and efficiency check was performed by having a huge number of test runs with randomly generated data covering all the cases and parts described in Section 4. We used the following scheme:

- We first prescribe upper boundary values for j_{max}, a_{max}, v_{max}, called j_{maxb}, a_{maxb}, v_{maxb}, and the largest distance Δs_{max} is considered. We choose 100 for each (units: m/s^3, m/s^2, m/s, m) for our test runs to allow for a larger range but this is a bit arbitrary.
- Within the interval $[0.0001 \cdot j_{maxb}, j_{maxb}]$, we generate a random value for j_{max} and proceed analogously for a_{max}, v_{max}. Moreover, for $\Delta s \in [-\Delta s_{max}, \Delta s_{max}]$, we do so randomly.
- Within the area in the velocity–acceleration plane that is bounded by the horizontal lines at $\pm a_{max}$ and the left and right square root curves shown in Figure 3, we choose values for (v_A, a_A) and (v_E, a_E) randomly.
- Once the configuration $[j_{max}, a_{max}, v_{max}, \Delta s, v_A, a_A, v_E, a_E]$ has been generated randomly this way, a test run for computing the best function is performed and it is checked whether the solution really gives the prescribed final velocity and acceleration and the distance Δs within a certain margin of error. If the error is larger than 10–7, the configuration producing it is recorded for further investigation.

Given this scheme, we made several runs with 10^7 random configurations which took about 40 min each (depending on whether the profiler in MATLAB was turned on or off). During the first test runs, we encountered, a few times, the situation that no solution was found, which is in contradiction with the logical argumentation in Section 4. The reasons were, as already assumed, of numerical origin and of two different kinds. In the first type of situation, the solutions were at border points between different parts. Consider, for example, case B: If there is a solution very close to the border for v_m between Part IIa and Part IIb (i.e., $v_m = v_{fA}$), then it might not be accepted in Part IIa because it is required there that $v_m > v_{fA}$. By simply allowing equality here (which in theory means one has an overlap between Parts IIa and III), one can avoid this problem. One could also allow a small overlapping ε-interval here. The second type of situation where no solution was found had its origin in the error margin for the distance in the computation of candidates (roots of the polynomials). By enlarging the error margin, this type of problem could be avoided in the subsequent runs. Since, in the final solution, the distance is again checked, this does not lead to the acceptance of solutions with errors above a specified value. Actually, since distances are computed using Equations (30)–(33), stated above in this section, we might have typical cancellation effects when v_A and v_m or v_E and v_m have nearly the same absolute value.

We performed a test run with 10^8 configurations, which provided a solution for all configurations, so the numerical remedies reported above seem to work. Table 2 gives information on errors. The maximum error in Δs was 6.57×10^{-6} and there were only 15 configurations with errors in Δs of more than 1×10^{-7}. Therefore, we conclude that the implementation provides acceptable results for realistic situations regarding the values for j_{max}, a_{max} and v_{max}.

Table 2. Largest errors in a test run with 10^8 configurations.

	Run with Lower Bounds
Max. error Δs	6.57×10^{-6} (m)
Max. error vE	4.67×10^{-12} (m/s)
Max. error aE	7.11×10^{-14} (m/s^2)
Number of Errors in $\Delta s >= 10^{-7}$	15

Table 3 provides information on the case and part coverage. It shows that all cases and parts were sufficiently covered except for Part IIb in cases A and B (in case D, Parts IIa and IIb do not occur; see Section 4). In earlier test runs with less configurations, it was nearly always the case that there was no example for Part IIb in cases A and B, and we conjectured already in Section 4 that Part IIb might be unnecessary. A closer look at the configurations that led to the 12 entries for Part IIb in cases A and B revealed that, again, this is a numerical effect. As is shown in Figure 10, if the solution point is very close to the "meeting point" of the curves for Parts IIa, IIb and III, it might happen that because of round-off errors, IIb gives a lower time value than IIa, although zooming in shows that Part IIb is above Part IIa. Practically, this does not really matter since the solutions provide nearly the same time. Therefore, we still stick to our conjecture that in cases A and B, Part IIb never contains the best solution.

Table 3. Coverage of cases and parts.

	Case A	Case B	Case Ca	Case Cb	Case D
Part I	10,561,715	10,563,034	1,085,981	9,209,731	13,996,250
Part IIa	727,205	728316	17,555	691,269	0
Part IIb	3	9	17,756	692,692	0
Part III	13,710,379	13,713,522	1,087,589	9,206,664	13,990,330
Sum	24,999,302	25,004,881	2,208,881	19,800,356	27,986,580

Regarding efficiency, we wanted to investigate whether the computation of the best function can be performed within a controller cycle which might be 1 ms (cf. [12]) or less, and for very fast controllers, even 100 µs. In our computations, we used a desktop PC with a Xeon E5-1630v4 processor with 3.7 GHz.

It is not straightforward to generate valid efficiency data when running MATLAB functions. As stated in Mathwork's own performance white paper [23], there is much "noise" to be taken into account, which might lead to considerable variation of times. This noise includes scheduling of processes and threads by the operating system and use of different memory caches. Moreover, the just-in-time compiler of MATLAB compiles a function once and uses then the compiled version. The variation can be made visible by running the best function with a fixed configuration several times, such that the computational work is identical. This gave values between a few milliseconds and a few hundred microseconds. We took this into account by repeating the computation for the same configuration several times and then taking the lowest value. We measured the times for the best function finder for 100,000 configurations, each repeated 1000 times, which provides a good coverage of all cases and parts. The maximum time was 303 µs. The result already shows that a run of the best function finder can be performed with a cycle time of 1 ms without any problem. However, there is even considerable potential for improvement. One of the main tasks in the best function finders is the computation of roots of the 12 polynomials of degree 4. In a test run with 1,000,000 configurations with 20 repetitions each, we measured an average time of 156 µs for computing the roots of the 12 polynomials. MATLAB's roots function is a general root-finding function for polynomials of an arbitrary degree. For polynomials of degree 4 (also called "quartics"), there are many more efficient special solvers available. For solving a quartic accurately and efficiently, Orellana and De Michele [24] needed less than 0.5 µs on a dual-core i7 processor with 3.3 GHz. This means that by using such a specific solver, we can reduce the effort for root finding to about 6 µs (12 polynomials). That would cut the "worst case" time for computing the best function of about 303 µs by half. Moreover, our current MATLAB code contains overhead for providing additional information, and the code is split up into several small functions for making it better readable. Modern compilers contain optimizers that try to reduce the overhead when actually producing the executable code, e.g., by so-called "inlining", where calls to "small functions" are substituted by inserting the code of the functions into the calling function. Orellana and De Michele [24] showed that by using a certain optimizer option, they could reduce the time needed by about half. This indicates that the time needed for computing the best function could even be brought well below 100 µs. This is comparable with the time reported in [12] for the worst case (540 µs for six axes) and in [19] as average (74 µs for one axis). Sidobre and Desormeaux [22] even achieved 2.5 µs, but the three-variable

Newton method they applied after first obtaining an approximation did not always provide the optimal solution. The most important criterion for online trajectory generation is not the absolute time as long as the computation can be performed within one controller cycle.

6. Conclusions

In this contribution, we investigate how a fast motion function can be found when restrictions for jerk, acceleration and velocity, initial and final values for velocity and acceleration and the distance are given. Our approach applies a seven-segment motion profile with piecewise constant jerk, which was already considered by other authors. We developed an algorithm for computing such a profile which considers only four major cases and, hence, has a manageable complexity. We investigate these cases systematically in the velocity–acceleration plane, which provides an easily understandable visual explanation. The investigation showed that there are often several seven-segment profiles fulfilling the condition differing considerably in execution time. We take the best one, whereas the algorithm by Ezair et al. [19] leads to just one of them. Since (like Sidobre/Desormeaux [22]) we do not have mathematical proof that there are no other seven-segment profiles with shorter times, we only claim to have a "fast" solution, not necessarily an "optimal" one. Moreover, a continuity argument proved that we cover all possible distances, whereas in the work by Sidobre and Desormeaux [22] treating the same problem, only a brief explanation but no proof is given.

We implemented our algorithm in MATLAB, where we reduced the computation essentially to the finding of roots of up to 12 polynomials of degree 4, which can easily be re-implemented. A large test run with 10^8 configurations showed that a solution was always found and all cases as well as all parts were covered satisfactorily. This also supports our claim of full coverage of all possible situations. Moreover, our timing experiments showed that our algorithm is suitable for online trajectory generation. Although finding the zeros of up to 12 polynomials makes our algorithm a bit slower than algorithms for special situations (such as [12]), the additional cost is small since very fast solvers for quartics are available.

There are also clear limitations to the approach using a seven-segment profile. Since the jerk function is piecewise constant, the discontinuity might lead to unwanted vibrations. As stated in the literature review, for special situations such as RR ("rest-to-rest"), smoother functions have been designed. This deficit of the seven-segment approach might be reduced by working with a 15-segment profile with continuous jerk. A feasible way for finding such a profile in the general BB task might be to generalize the approach developed in this contribution. A further limitation of the seven-segment approach is that it does not handle variable constraints but only constant ones, and the constraints are assumed to be symmetric, although it would be possible to modify our approach in order to include asymmetric constraints.

We have not tested the applicability of seven-segment profiles experimentally since this has already been performed by other authors (e.g., [12,22]), on whom we rely. In real applications such as robotics, the situation we investigated in this article is only one problem among many. Further considerations are necessary when there are several drives that need coordination or when restrictions change and one has to turn the initial state into an admissible one first. One can find strategies for this in [12,19,20,25].

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

Appendix A

Table A1. Polynomials and solutions for $w = S(v_A, v_E, v_m)$.

	List of Coefficients for (30)–(32) Resp. Solutions for (33)
Case (α)	
(30)	$\left[j_{max}^2 v_A - j_{max}^2 v_E,\ 2j_{max}^2 w,\ j_{max} v_A^2 - 2j_{max} v_A v_E + j_{max} v_E^2, -4j_{max} w \cdot v_E,\ j_{max} w^2 - v_A^3 - v_A^2 v_E + v_A v_E^2 + v_E^3\right]$
(31)	$\left[j_{max}^3,\ 2a_{max} j_{max}^2,\ a_{max}^2 j_{max} - 2j_{max}^2 v_A, -4a_{max} j_{max} v_A, -a_{max}^2 v_A - a_{max}^2 v_E + 2a_{max} j_{max} w + j_{max} v_A^2 - j_{max} v_E^2\right]$
(32)	$\left[j_{max}^3,\ 2a_{max} j_{max}^2,\ a_{max}^2 j_{max} - 2j_{max}^2 v_E, -4a_{max} j_{max} v_E, -a_{max}^2 v_A - a_{max}^2 v_E + 2a_{max} j_{max} w - j_{max} v_A^2 + j_{max} v_E^2\right]$
(33)	$\frac{a_{max}^2 \pm \sqrt{a_{max}^4 + 2a_{max}^2 j_{max} v_A + 2a_{max}^2 j_{max} v_E - 4a_{max} j_{max}^2 w + 2j_{max}^2 v_A^2 + 2j_{max}^2 v_E^2}}{2j_{max}}$
Case (β)	
(30)	$\left[j_{max}^2 v_A - j_{max}^2 v_E,\ 2j_{max}^2 w, -j_{max} v_A^2 + 2j_{max} v_A v_E - j_{max} v_E^2,\ 4j_{max} w \cdot v_E, -j_{max} w^2 - v_A^3 - v_A^2 v_E + v_A v_E^2 + v_E^3\right]$
(31)	$\left[j_{max}^3,\ 2a_{max} j_{max}^2,\ a_{max}^2 j_{max} + 2j_{max}^2 v_A,\ 4a_{max} j_{max} v_A,\ a_{max}^2 v_A + a_{max}^2 v_E - 2a_{max} j_{max} w + j_{max} v_A^2 - j_{max} v_E^2\right]$
(32)	$\left[j_{max}^3,\ 2a_{max} j_{max}^2,\ a_{max}^2 j_{max} + 2j_{max}^2 v_E,\ 4a_{max} j_{max} v_E,\ a_{max}^2 v_A + a_{max}^2 v_E - 2a_{max} j_{max} w - j_{max} v_A^2 + j_{max} v_E^2\right]$
(33)	$\frac{-a_{max}^2 \pm \sqrt{a_{max}^4 - 2a_{max}^2 j_{max} v_A - 2a_{max}^2 j_{max} v_E + 4a_{max} j_{max}^2 w + 2j_{max}^2 v_A^2 + 2j_{max}^2 v_E^2}}{2j_{max}}$

Table A2. Polynomials and solutions for $w = S_{neg}(v_A, v_E, v_m)$.

	List of Coefficients for Modified (30)–(32)
Case (α)	
(30)	$\left[j_{max}^2 v_A - j_{max}^2 v_E, -2j_{max}^2 w,\ j_{max} v_A^2 - 2j_{max} v_A v_E + j_{max} v_E^2,\ 4j_{max} w \cdot v_E,\ j_{max} w^2 - v_A^3 - v_A^2 v_E + v_A v_E^2 + v_E^3\right]$
(31)	$\left[j_{max}^3, -2a_{max} j_{max}^2,\ a_{max}^2 j_{max} - 2j_{max}^2 v_A,\ 4a_{max} j_{max} v_A, -a_{max}^2 v_A - a_{max}^2 v_E - 2a_{max} j_{max} w + j_{max} v_A^2 - j_{max} v_E^2\right]$
(32)	$\left[j_{max}^3, -2a_{max} j_{max}^2,\ a_{max}^2 j_{max} - 2j_{max}^2 v_E,\ 4a_{max} j_{max} v_E, -a_{max}^2 v_A - a_{max}^2 v_E + 2a_{max} j_{max} w - j_{max} v_A^2 + j_{max} v_E^2\right]$
Case (β)	
(30)	$\left[j_{max}^2 v_A - j_{max}^2 v_E, -2j_{max}^2 w, -j_{max} v_A^2 + 2j_{max} v_A v_E - j_{max} v_E^2, -4j_{max} w \cdot v_E, -j_{max} w^2 - v_A^3 - v_A^2 v_E + v_A v_E^2 + v_E^3\right]$
(31)	$\left[j_{max}^3, -2a_{max} j_{max}^2,\ a_{max}^2 j_{max} + 2j_{max}^2 v_A, -4a_{max} j_{max} v_A,\ a_{max}^2 v_A + a_{max}^2 v_E + 2a_{max} j_{max} w + j_{max} v_A^2 - j_{max} v_E^2\right]$
(32)	$\left[j_{max}^3, -2a_{max} j_{max}^2,\ a_{max}^2 j_{max} + 2j_{max}^2 v_E, -4a_{max} j_{max} v_E,\ a_{max}^2 v_A + a_{max}^2 v_E - 2a_{max} j_{max} w - j_{max} v_A^2 + j_{max} v_E^2\right]$

Table A3. Solution candidates for v_m.

Case (α)	
(30)	$-z^2 j_{max} + v_E$
(31)	$-z^2 j_{max} + v_A$
(32)	$-z^2 j_{max} + v_E$
Case (β)	
(30)	$z^2 j_{max} + v_E$
(31)	$z^2 j_{max} + v_A$
(32)	$z^2 j_{max} + v_E$

References

1. Biagiotti, L.; Melchiorri, C. *Trajectory Planning for Automatic Machines and Robots*; Springer: Berlin/Heidelberg, Germany, 2008.
2. Carbone, G.; Gomez-Bravo, F. (Eds.) *Motion and Operation Planning of Robotic Systems*; Mechanisms and Machine Science 29; Springer: Berlin/Heidelberg, Germany, 2015.
3. 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.

4. VDI (Ed.) *Guideline 2143: Motion Rules for Cam Mechanisms; Theoretical Fundamentals (German)*; Beuth Verlag: Berlin, Germany, 1980.
5. Kröger, T.; Wahl, F.M. Online Trajectory Generation: Basic Concepts for Instantaneous Reactions to Unforseen Events. *IEEE Trans. Robot.* **2010**, *26*, 94–111.
6. Macfarlane, S.; Croft, E.A. Jerk-bounded manipulator trajectory planning: Design for real-time applications. *IEEE Trans. Robot. Autom.* **2003**, *19*, 42–52. [CrossRef]
7. Liu, H.; Liu, 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]
8. 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.
9. Shen, P.; Zhang, X.; Fang, Y. Complete and time-optimal path-constrained trajectory planning with torque and velocity constraints: Theory and applications. *IEEE/ASME Trans. Mechatron.* **2018**, *23*, 735–746. [CrossRef]
10. Penin, B.; Giordana, P.R.; Chaumette, F. Minimum-time trajectory planning under intermittent measurements. *IEEE Robot. Autom. Lett.* **2019**, *4*, 153–160. [CrossRef]
11. 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.
12. Kröger, T. *On-Line Trajectory Generation in Robotic Systems*; Springer: Berlin/Heidelberg, Germany, 2010.
13. SEW Eurodrive. *Manual Program. Module MultiMotion*; SEW: Bruchsal, Germany, 2010.
14. 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]
15. 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]
16. Trigatti, G.; Boscariol, P.; Scalera, L.; Pillan, D.; Gasparetto, A. A look-ahead trajectory planning algorithm for spray painting robots with non-spherical wrists. In Proceedings of the 4th IFToMM Symposium on Mechanism Design for Robotics, Udine, Italy, 11–13 September 2018; pp. 235–242.
17. Liu, S. An On-line Reference-Trajectory Generator for Smooth Motion of Impulse-Controlled Industrial Manipulators. In Proceedings of the 7th International Workshop on Advanced Motion Control, Maribor, Slovenia, 3–5 July 2002; pp. 365–370.
18. Haschke, R.; Weitnauer, E.; Ritter, H. On-line planning of time-optimal, jerk-limited trajectories. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 27–31 October 2008; pp. 3248–3253.
19. 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]
20. Broquère, X. Planification de Trajectoire Pour la Manipulation D'objets et L'interaction Homme-Robot. Ph.D. Thesis, Université Paul Sabatier Toulouse III, Toulouse, France, 2011.
21. 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.
22. Sidobre, D.; Desormeaux, K. Smooth cubic polynomial trajectories for human-robot interactions. *J. Intell. Robot. Syst.* **2019**, *95*, 851–869. [CrossRef]
23. McKeeman, B. *MATLAB Performance Measurement*; Mathworks: Natick, MA, USA, 2020.
24. Orellana, A.G.; De Michele, C. Algorithm 1010: Boosting Efficiency in Solving Quartic Equations with No Compromise in Accuracy. *ACM Trans. Math. Softw.* **2020**, *46*, 20:1–20:28. [CrossRef]
25. Desormeaux, K.; Sidobre, D. Online Trajectory Generation: Reactive Control with Return Inside an Admissible Kinematic Domain. In Proceedings of the International Conference on Intelligent Robots and Systems (IROS 2019), Macau, China, 4–8 November 2019.