Advanced Robotics

ISSN: 0169-1864 (Print) 1568-5535 (Online) Journal homepage: https://www.tandfonline.com/loi/tadr20

Time-optimal parabolic interpolation with velocity, acceleration, and minimum-switch-time constraints

Puttichai Lertkultanon & Quang-Cuong Pham

To cite this article: Puttichai Lertkultanon & Quang-Cuong Pham (2016) Time-optimal parabolic interpolation with velocity, acceleration, and minimum-switch-time constraints, Advanced Robotics, 30:17-18, 1095-1110, DOI: 10.1080/01691864.2016.1204247

To link to this article: https://doi.org/10.1080/01691864.2016.1204247

ADVANCED ROBOTICS, 2016
VOL. 30, NOS. 17–18, 1095–1110
http://dx.doi.org/10.1080/01691864.2016.1204247

FULL PAPER

Time-optimal parabolic interpolation with velocity, acceleration, and minimum-switch-time constraints

Puttichai Lertkultanon and Quang-Cuong Pham

School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore, Singapore

ABSTRACT

Time-optimal trajectories with bounded velocities and accelerations are known to be parabolic, i.e. piecewise constant in acceleration. An important characteristic of this class of trajectories is the distribution of the switch points – the time instants when the acceleration of any robot joint changes. When integrating parabolic trajectory generation into a motion planning pipeline, especially one that involves a shortcutting procedure, resulting trajectories usually contain a large number of switch points with a dense distribution. This high frequency acceleration switching intensifies joint motor wear as well as hampers the robot performance. In this paper, we propose an algorithm for planning parabolic trajectories subject to both physical bounds, i.e. joint velocity and acceleration limits, and the minimum-switch-time constraint. The latter constraint ensures that the time duration between any two consecutive switch points is always greater than a given minimum value. Analytic derivations are given, as well as comparisons with other methods to demonstrate the efficiency of our approach.

ARTICLE HISTORY
Received 29 October 2015
Revised 3 March 2016
and 13 June 2016
Accepted 15 June 2016

KEYWORDS
Parabolic trajectories; shortcutting; switch time

1. Introduction

As the execution time of robot movements is a determining factor in industrial productivity, planning fast robot trajectories is an important topic in industrial robotics.[1] A large body of work has been devoted to the planning of time-optimal trajectories for robot manipulators, subject to various types of constraints, such as torque bounds,[2,3] gripper and payload constraint,[4] or velocity and acceleration bounds.[5–7]

Most trajectory generation methods for industrial robots make use of polynomials. The approaches mostly differ on boundary conditions and optimization objectives. Jerk-bounded trajectories have been shown to possess some desirable behaviors.[8,9] To achieve the jerk-boundedness, polynomial trajectories need to be of order three or more. In [8], the authors generated rest-to-rest minimum-jerk trajectories using cubic polynomials. This class of polynomials was also used in [10], where the authors proposed a method for generating time-optimal trajectories, given that the trajectories end at rest. The author of [11] presented a *seven-segment* method of generating a single-axis cubic trajectory. The trajectory was divided into seven segments and was limited to be one of several forms. The exact trajectory was calculated once the best form of the trajectory had been chosen. This method was also limited to zero boundary conditions. The method was later extended for multi-axis systems.[12] Quartic (fourth-order) polynomials were used in [13] subject to zero final acceleration. The authors of [14] also proposed a framework for quartic polynomial trajectory generation. The method was claimed to be computationally light. However, it did not take into account velocity bounds. Quintic (fifth-order) polynomials were used in [15] subject to zero terminal acceleration. Recently, there was also the work addressing generation of polynomial trajectories of any order [16] subject to general boundary conditions. Although high-order polynomials generate smooth trajectories, polynomials with order higher than three are not generally used due to their tendency to produce oscillation in joint positions.[15,17]

Splines are a particular class of polynomials that have also been used for industrial robots. B-splines were used in [18] to interpolate rest-to-rest trajectories (beginning and ending with zero velocity and acceleration). In [17], the authors used cubic splines to interpolate minimum-jerk trajectories. The traveling time was required as an input. Cubic splines were also used in [9], where the trajectory generation problem was formulated as an optimization problem. The objective function was set to be a weighted sum between traveling time and integral of squared jerk. In [19], the authors proposed a quintic spline interpolation method which gave continuous acceleration profiles. Apart from polynomial splines, the use of trigonometric splines were also investigated in [20] to generate smooth trajectories.

CONTACT Quang-Cuong Pham cuong.pham@normalesup.org

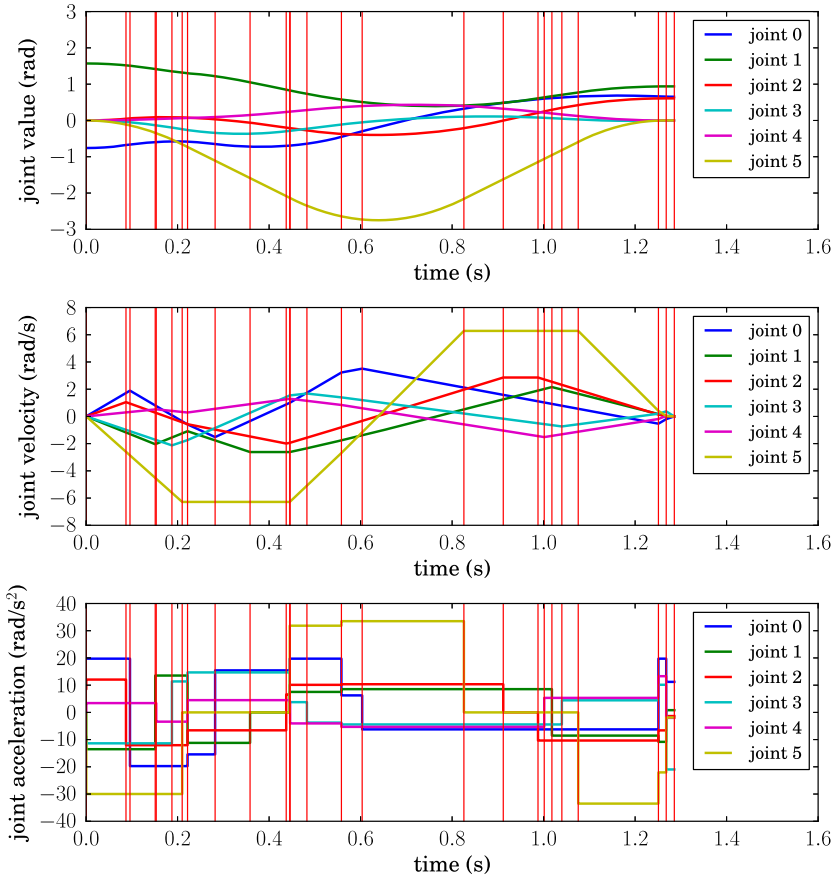


Figure 1. An example trajectory planned and shortcut using the algorithm in [6]. Red vertical lines indicate time instants when at least one DOF switches its acceleration (switch points). The minimum-switch-time of this trajectory is 0.49 ms, which is attained near $t = 0.1$ s.

Problems considering both time-optimality and jerk bounds are, however, too complex to be solved without resorting to more restricted boundary conditions [8] or computationally expensive optimization routines.[9] Although jerk-boundedness is a desirable property for manipulator trajectories, it is not a necessary condition for successful trajectory execution. Therefore, in many cases, it might be preferable to trade off jerk-boundedness for time-optimality of the trajectory *and* computational efficiency. This can be done using parabolic interpolation, which produces quadratic (second-order) trajectories, since the time-optimal parabolic interpolation method subject to velocity and acceleration bounds can be computed *analytically*. Another reason quadratic trajectories are acceptable is that a significant proportion of industrial robots are controlled in acceleration, i.e. their inputs are time-series of the joint accelerations.

Hauser and Ng-Thow-Hing [6] and Kröger et al. [21] investigated the problem of planning time-optimal piecewise parabolic trajectories subject to velocity and acceleration bounds (note that in this case, the parabolicity of the optimal trajectories is *a posteriori*). However, the trajectories computed by these authors tend to have a large number of *switch points*, i.e. time instants when the acceleration of any joint changes. Furthermore, these switch points are sometimes concentrated in a short time interval, see Figure 1, especially after a shortcutting procedure [6,22] has been applied.

Yet, switch points that are too close to each other might hamper the performance of the system. For example, one might not want to switch the robot joint accelerations faster than the rate at which a controller communicates with the robot. Trajectories with dense switch point distribution might also be poorly tracked, which in turn can result in undesired collisions with the environment or slower actual execution times. Furthermore, a dense distribution can also increase motor wear. In applications where human operators must implement the planned trajectory, time durations between two switches cannot be smaller than the cognitive processing time of the operator, which can be of the order of seconds. Thus, it is necessary to include the minimum-switch-time constraint into time-optimal parabolic interpolation problems with velocity and acceleration bounds. The new constraint ensures that any two consecutive switch points, either along the trajectory of the same joint or of different

joints, are separated by at least some given duration. This constraint also appears in attitude control problems with specific kinds of thrusters.[23,24]

One possibility to take into account the minimum-switch-time constraint could be to include this constraint, along with velocity and acceleration bounds, into an *optimization program*. However, this program turns out to be non-convex, and therefore cannot be efficiently solved (see Section 5.1).

In this paper, we propose an efficient algorithm for planning piecewise parabolic trajectories subject to velocity and acceleration bounds, and which also takes into account the minimum-switch-time constraint. Moreover, when the minimum-switch-time constraint is removed, our algorithm still outperforms existing algorithms in the literature. We first start with the algorithm of [6,21], which computes time-optimal interpolations subject to only velocity and acceleration bounds (recalled in Section 2). Then, we examine all the possibilities of violation of the minimum-switch-time constraint and propose a solution to address the violation for each case (Sections 3 and 4). A proof of time-optimality is provided, as well as an open-source implementation (see https://github.com/Puttichai/parabint). The new interpolation routine is integrated into a full-fledged motion planner and its efficiency is demonstrated through simulations (Section 5).

2. Background: time-optimal parabolic interpolation without minimum-switch-time constraint

In the sequel, bold lower-case letters will denote vectors, normal lower-case letters will denote scalars, and bold upper-case letters will denote matrices.

2.1. Motion planning pipeline

Before recalling the time-optimal parabolic interpolation algorithm of [6,21], this section presents the global motion planning pipeline that is based on this algorithm. Although there exists a large number of motion planning methods, this *plan-and-shortcut* pipeline is one of the most robust [25] and widely used in industrial robotics; it is the default pipeline in the robot programming environment OpenRAVE,[26] which in turn is used by a large number of research groups and robotics companies worldwide.

A trajectory planning problem consists in finding a fast trajectory connecting two robot configurations $\boldsymbol{x}_{\text{init}}$ and $\boldsymbol{x}_{\text{goal}}$ in the robot joint space $\mathcal{C} \subset \mathbf{R}^n$, where n is the number of degrees of freedom (DOF) of the robot, subject to velocity and acceleration bounds of each joint, as well as collision avoidance. This pipeline is based on the path-velocity decomposition approach [27] which decompose the problem into planning a *path* and a *velocity profile*.

In the first stage, a path planner such as a Rapidly-Exploring Random Tree (RRT) planner [28] is used to search for a collision-free *path* connecting $\boldsymbol{x}_{\text{init}}$ and $\boldsymbol{x}_{\text{goal}}$. The solution path is piecewise linear since it is formed by concatenating a linear path segment together. Each linear path segment P connecting $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ can be represented by a parameterization $\boldsymbol{x}(s) = \boldsymbol{x}_a + s(\boldsymbol{x}_b - \boldsymbol{x}_a)$, where $s \in [0, 1]$.

The second stage, called time-parameterization, assigns to a path a *velocity profile*, which is the time-derivative of the path parameterization function $s : [0, T] \rightarrow [0, 1]$, where T is the total duration of the velocity profile and thus of the trajectory. The problem can be alternatively viewed as a **velocity profile interpolation problem**. To respect the path geometry (avoiding thereby new collisions at this stage), one must time-parameterize paths of different joints *simultaneously* through the path parameterization function s. Furthermore, to avoid discontinuities in the velocity vector at the junctions of the linear segments, one needs to ensure that the velocities at the beginning and the end of each segment are zero. In the end, one will have a trajectory $\boldsymbol{x} : [0, T] \rightarrow \mathcal{C}$, where $\boldsymbol{x}(0) = \boldsymbol{x}_{\text{init}}$ and $\boldsymbol{x}(T) = \boldsymbol{x}_{\text{goal}}$.

In the third stage, a randomized shortcutting procedure is applied to improve the execution time of the trajectory, which is initially high because of the nature of a randomized path planner[1] and of the start–stop behavior at the junctions between linear segments. In each shortcutting iteration, two random time instants t_a and t_b are selected. Then a time-optimal trajectory is interpolated between $(\boldsymbol{x}(t_a), \dot{\boldsymbol{x}}(t_a))$ and $(\boldsymbol{x}(t_b), \dot{\boldsymbol{x}}(t_b))$. To effectively *shortcut* the trajectory, different DOFs may be interpolated *independently*, provided that all the interpolants have the same duration. If the shortcut has shorter duration than $t_b - t_a$ and is collision-free, then one replaces the original trajectory segment by the shortcut.

In the aforementioned procedure, two interpolation primitives are required:

(1) simultaneous interpolation with zero terminal velocities (in the planning stage)
(2) independent interpolation with arbitrary terminal velocities subject to the condition that the interpolants have the same duration (in the shortcutting stage).

The following sections recall the algorithms proposed in [6] for each case.

2.2. Useful formulae for linear paths with constant accelerations

Consider a linear path covering a displacement d. Let u be the initial velocity, v the final velocity, a the acceleration, and t the duration. The following relations hold:

$$d = ut + \frac{1}{2}at^2, \tag{1}$$

$$v^2 = u^2 + 2ad, \tag{2}$$

$$v = u + at, \tag{3}$$

$$d = \frac{1}{2}(u + v)t. \tag{4}$$

2.3. Simultaneous interpolation with zero terminal velocities

Definition 1: A ramp is a constant-acceleration velocity profile. A ramp with non-zero acceleration is called a parabolic ramp or P-ramp. A ramp with zero acceleration is called a linear ramp or L-ramp.

Definition 2: A P-ramp with maximal (respectively minimal) acceleration is noted P^+ (respectively P^-). An L-ramp with maximal (respectively minimal) velocity is noted L^+ (respectively L^-).

Consider a linear path segment P connecting $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ which is parameterized by a path parameterization function $s : [0, T] \to [0, 1]$, i.e. $\boldsymbol{x}(t) = \boldsymbol{x}_a + s(t)(\boldsymbol{x}_b - \boldsymbol{x}_a)$. A valid velocity profile must be subjected to boundary conditions $\boldsymbol{v}(0) = \boldsymbol{v}(T) = \boldsymbol{0}$, where $\boldsymbol{v}(t) = \dot{\boldsymbol{x}}(t)$. Let $\boldsymbol{v}_m$ and $\boldsymbol{a}_m$ be the vectors of velocity and acceleration bounds, respectively. Since the velocity $\boldsymbol{v}(t)$ is described as $\boldsymbol{v}(t) = \dot{s}(t)(\boldsymbol{x}_b - \boldsymbol{x}_a)$, the path velocity $\dot{s}$ is then bounded by $\dot{s}_m = \min_i (\boldsymbol{v}_{m,i}/|\boldsymbol{x}_{b,i} - \boldsymbol{x}_{a,i}|)$. Similarly, since the acceleration $\boldsymbol{a}$ is written as $\boldsymbol{a}(t) = \dot{\boldsymbol{v}}(t) = \ddot{s}(\boldsymbol{x}_b - \boldsymbol{x}_a)$, the path acceleration $\ddot{s}$ is then bounded by $\ddot{s}_m = \min_i (\boldsymbol{a}_{m,i}/|\boldsymbol{x}_{b,i} - \boldsymbol{x}_{a,i}|)$.

The n-DOF path-parameterization problem of the segment has been transformed to a single-DOF problem in the variable s subject to the velocity bound $\dot{s}_m$ and the acceleration bound $\ddot{s}_m$. The time-optimal velocity profile can be shown to be symmetric and is either $\mathrm{P}^+\mathrm{P}^-$ or $\mathrm{P}^+\mathrm{L}^+\mathrm{P}^-$. In the former case, the switch point, which can be computed from Equation (1), is given by $t_s = 1/\sqrt{\ddot{s}_m}$. In the latter case, the first switch point is given by $t_0 = \dot{s}_m/\ddot{s}_m$ and the switch time between the two switch points is $t_1 = 1/\dot{s}_m - 1/\ddot{s}_m$.

2.4. Independent interpolation with arbitrary terminal velocities

2.4.1. Single DOF

Given a straight-line path from x_0 to x_1, the time-optimal velocity profile with the initial and final velocities v_0 and v_1, subject to velocity and acceleration bounds v_m and a_m can be shown to have only four possible types : $\mathrm{P}^+\mathrm{P}^-$, $\mathrm{P}^-\mathrm{P}^+$, $\mathrm{P}^+\mathrm{L}^+\mathrm{P}^-$, and $\mathrm{P}^-\mathrm{L}^-\mathrm{P}^+$.[2] For given boundary conditions and bounds, we can compute (or *interpolate*) the time-optimal velocity profile by exploring each of the four cases above. The detailed calculations of the switch points in each case can be found, for example, in [6,7,21].[3]

2.4.2. Multiple DOFs

Here, the authors of [6] first interpolate (the velocity profile of) each joint independently. Suppose the kth joint has the longest duration, T. Then they *re-interpolate* the velocity profile of the remaining joints such that all new velocity profiles have the same duration T. This re-interpolation is, however, not always possible, and determining whether the re-interpolation is possible for a given boundary conditions and bounds is actually a difficult problem.[7,21] In [6], the authors suggested to explore four possibilities similar to $\mathrm{P}^+\mathrm{P}^-$, $\mathrm{P}^-\mathrm{P}^+$, $\mathrm{P}^+\mathrm{L}^+\mathrm{P}^-$, and $\mathrm{P}^-\mathrm{L}^-\mathrm{P}^+$. In each case, they constrained both P-ramps to have the same magnitude of acceleration $|a|$. Doing so allowed them to subsequently solve for $|a|$. Nevertheless, this extra constraint considerably decreased the performance of the shortcutting method. A new re-interpolation method – with or without considering the minimum-switch-time constraint – which achieves a higher success rate is proposed in this paper, see Section 4.2.

3. Simultaneous interpolation with zero terminal velocities subject to the minimum-switch-time constraint

Let the minimum allowed switch time be $\delta \geq 0$. To re-interpolate a constraint-violating velocity profile, we enumerate all the possible modifications and then choose the shortest (re-interpolated) velocity profile as a solution.

Consider first the case when the velocity profile $\dot{s}(t)$ computed in Section 2.3 is $\mathrm{P}^+\mathrm{P}^-$. Suppose that the duration of each ramp (both ramps have equal duration by construction) is $t_s < \delta$. To address this constraint violation, we construct a new velocity profile, $\dot{s}'(t)$. To make the velocity profile time-optimal while not violating the constraint, both ramps must have their duration being δ, hence $t'_s = \delta$. Thus, the new peak velocity, computed using Equation (1), becomes $\dot{s}'_p = 1/\delta$ and the new acceleration of each ramp has the magnitude $|\ddot{s}'| = \dot{s}'_p/\delta$. It can be easily verified from direct calculations that $\dot{s}'_p/ < \dot{s}_m$ and $|\ddot{s}'| < \ddot{s}_m$.

Next, consider the case when the velocity profile $\dot{s}(t)$ computed in Section 2.3 is $\mathrm{P}^+\mathrm{L}^+\mathrm{P}^-$ and violates the minimum-switch-time constraint. There are two possible

subcases depending on the duration of P-ramps, $t_0 = \dot{s}_m/\ddot{s}_m$.

3.1. $t_0 \geq \delta$

Here, only the L-ramp violates the constraint. We have two possible ways to address this violation: (a) we stretch the duration of the L-ramp to δ. The acceleration of the two P-ramps remain unchanged but their durations are shortened so as to maintain the total displacement of 1; (b) we re-interpolate the velocity profile such that it has two P-ramps of equal duration $t_s' = 1/\dot{s}_m$. The peak velocity still saturates the velocity bound while the magnitude of acceleration of both ramps is reduced to $|\ddot{s}_m'| = \dot{s}_m^2$. By comparing the two velocity profiles above, the shorter and valid velocity profile is then chosen as a solution.

3.2. $t_0 < \delta$

Here both P-ramps violate the constraint. We can re-interpolate the original velocity profile into four possible velocity profiles. The conditions for choosing each case follows from direct calculations: (a) $\dot{s}_m\delta \leq 1/2$. In this case, a two-ramp velocity profile, each ramp having the duration of δ, is possible. We re-interpolate the profile accordingly. (b) $\dot{s}_m\delta \in [2/3, 1)$. In this case, we re-interpolate the velocity profile to have two ramps. The new peak velocity is $\dot{s}_m$. Each ramp will have its duration longer than δ. (c) $\dot{s}_m\delta \in [1/2, 2/3)$. In this case, we re-interpolate the velocity profile to have three ramps while each ramp has its duration of δ. The new peak velocity can be computed by $\dot{s}_m' = 1/(2\delta)$. (d) $\dot{s}_m\delta < 1/2$. In this case, the new velocity profile will have three ramps. The first and the last ramps have the duration δ while their accelerations have magnitude $|\ddot{s}'| = \dot{s}_m/\delta$. The middle ramp (L-ramp) will cruise at the maximum velocity $\dot{s}_m$ for a duration of $1/\dot{s}_m - \delta$.

4. Independent interpolation with arbitrary terminal velocities subject to the minimum-switch-time constraint

4.1. Single DOF

The task here is to interpolate a velocity profile for each DOF subject to boundary conditions $(x(0), v(0)) = (x_0, v_0)$ and $(x(T), v(T)) = (x_1, v_1)$ while respecting the velocity and acceleration bounds v_m and a_m, and the minimum-switch-time constraint δ. Note that the total duration T is not known *a priori*.

We use a two-step method. In the first step, we compute a time-optimal velocity profile subject to only velocity and acceleration constraints. Then in case the minimum-switch-time constraint is violated, we *re-interpolate* the profile by also taking into account the minimum-switch-time constraint. Let t_0, t_1, and t_2 be the durations of the first ramp, the second ramp and the third ramp (for PLP) of the original velocity profile. Since the original velocity profile can be either PP or PLP, there are 10 exhaustive cases of possible minimum-switch-time-constraint violation which are summarized in Table 1.

Table 1. All possibilities of minimum-switch-time-constraint violation given that the minimum allowed switch time is δ.

	t_0	t_1	t_2
PP1	$< \delta$	$\geq \delta$	–
PP2	$\geq \delta$	$< \delta$	–
PP3	$< \delta$	$< \delta$	–
PLP1	$< \delta$	$\geq \delta$	$\geq \delta$
PLP2	$\geq \delta$	$\geq \delta$	$< \delta$
PLP3	$\geq \delta$	$< \delta$	$\geq \delta$
PLP4	$< \delta$	$< \delta$	$\geq \delta$
PLP5	$\geq \delta$	$< \delta$	$< \delta$
PLP6	$< \delta$	$\geq \delta$	$< \delta$
PLP7	$< \delta$	$< \delta$	$< \delta$

4.1.1. PP1

First of all, we stretch the duration of the first ramp to δ. We constrain the second ramp to remain at the same acceleration, hence shortened. The new peak velocity at the new switch point $t_0' = \delta$ can be computed as

$$v_p' = \frac{1}{2}\left(k_1 - \mathrm{sgn}(k_1)\sqrt{k_1^2 + 4k_1v_0 + 4v_1^2 - 8a_1d}\right), \tag{5}$$

where $d = x_1 - x_0$ is the total displacement of the trajectory, $k_1 = a_1\delta$, and a_1 is the acceleration of the last ramp. The new acceleration of the first ramp, a_0', can be computed accordingly from $a_0' = (v_p' - v_0)/\delta$. Figure 2 shows an example of our re-interpolation according to the case PP1. A proof in Appendix 1 shows that the terms inside the square root in Equation (5) is always non-negative.

As we have remarked in Section 2.4.1 that there are some special cases when the velocity at one end of the velocity profile has maximum magnitude. For PP1, this is the case when $|v_1| = v_m$, hence $a_1 = 0$. In this case, Equation (5) is not valid since its derivation involves division by a_1. However, we can still proceed in the same way, i.e. we stretch the duration of the first ramp to δ while the last ramp remains at the same acceleration.

After following the first step, if the new duration of the last ramp, $t_1' = (v_1 - v - p')/a_1$, is shorter than δ, we need to re-interpolate the profile again. This can be done in two ways: (a) the new velocity profile has two ramps of equal duration δ; (b) the new velocity profile has one single ramp. In both cases, we can compute explicitly the total durations as well as the accelerations and peak

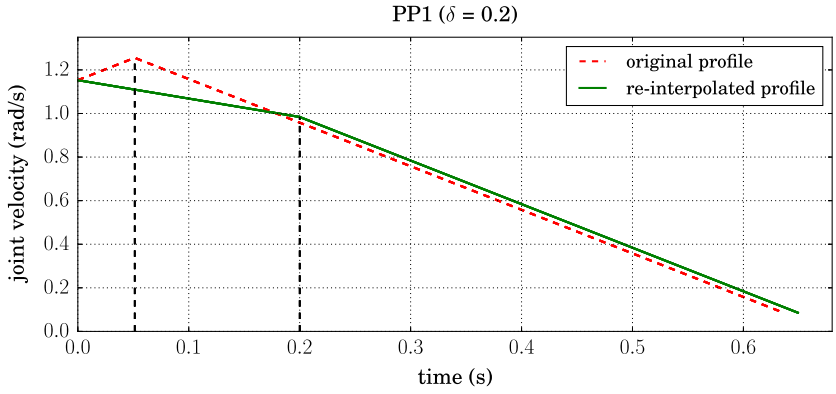


Figure 2. This figure illustrates our re-interpolation procedure for the case PP1. The minimum allowed switch time in this case is $\delta = 0.2$ s. The original velocity profile is shown in red. The re-interpolated velocity profile is shown in green. The first ramp of the new velocity profile is stretched such that it has the duration of δ. The last ramp is constrained to be at the same acceleration.

velocities. Finally, we choose the time-optimal one (note that at least (b) is guaranteed to be valid; see Appendix 1).

4.1.2. PP2

This case is *similar*[4] to the case PP1 and therefore the procedure can be directly adapted from that of PP1.

4.1.3. PP3

There are two possibilities of re-interpolation: (a) the new velocity profile has two ramps of equal duration δ; (b) the new velocity profile has one single ramp. However, unlike the previous two cases where the one-ramp velocity profile is guaranteed to be feasible, i.e. not shorter than δ, it is possible that the one-ramp profile is still shorter than δ. To handle such cases, we shall *flip* the original velocity profile. If the original profile is $\mathrm{P}^+\mathrm{P}^-$, then the new profile will be $\mathrm{P}^-\mathrm{P}^+$ and so on. The idea of flipping a velocity profile is illustrated in Figure 4. Both ramps of the new velocity profile still saturate the acceleration bounds. The new peak velocity can then be computed from

$$v_p'^2 = (v_0^2 + v_1^2)/2 + a_0'(x_1 - x_0), \tag{6}$$

where a_0' is the acceleration of the new first ramp. Note that v_p' can be computed from the above equation provided that the right-hand term is non-negative. Of the two solutions, we shall select the one which makes the new velocity profile valid and shortest possible.

In case the right-hand side of Equation (6) is negative, or it is positive but both of the resulting velocity profiles are not feasible (e.g. the minimum-switch-time constraint is still violated after the modification), this means that both ramps of the new velocity profile cannot saturate the acceleration bound at the same time. This implies that the minimum-switch-time constraint is to be saturated, instead of the acceleration bound. Therefore, we modify the velocity profile following the idea for PP1 and PP2. For example, consider the case when $t_0 > t_1$, i.e. the first ramp of the original velocity profile is longer than the last ramp. The *flipped* velocity profile will have its last ramp longer than its first. To obtain the flipped velocity profile, we follow the routine for PP1 to compute the new peak velocity. The first ramp of the new velocity profile will have a duration of δ, i.e. it saturates the minimum-switch-time constraint, while the last ramp saturates the acceleration bound.

4.1.4. PLP1

Here our re-interpolation routine explores two possibilities: (A) the re-interpolated profile has three ramps; (B) the re-interpolated profile has two ramps. Then, we choose the case which gives the shorter duration.

In PLP1A, we stretch the duration of the first ramp of the original profile to δ while its final velocity, v_p, remains at the velocity bound. The acceleration of the new first ramp is then $a_0' = (v_p - v_0)/\delta$. The velocity profile of the remaining portion of the trajectory is re-interpolated using the routine for PP1. In PLP1B, the first two ramps of the original velocity profile are merged into a single ramp by retaining the boundary conditions. The third ramp of the original velocity profile remains unchanged. Finally, we choose the case which gives the shorter duration. Note that at least the case PLP1B is valid since the duration of merged ramp will always be longer than δ. Here, the case where the re-interpolated profile has one ramp is not explored since it cannot be shorter than the velocity profile from PLP1B.

4.1.5. PLP2

This case is *similar* to the case PLP1 and therefore the procedure can be directly adapted from that of PLP1.

4.1.6. PLP3

For PLP3, there are two possibilities for the re-interpolated profile: (A) the re-interpolated profile has

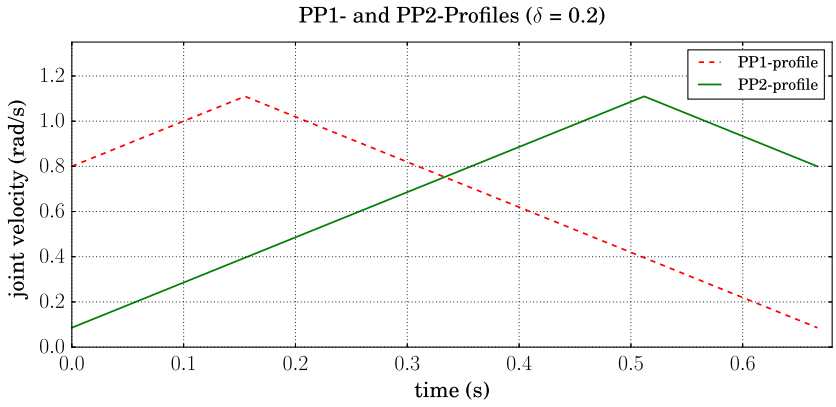


Figure 3. This figure shows similarity between a PP2-profile and its corresponding PP1-profile. One can see that swapping the initial and final velocities of a PP-2 profile results in the corresponding PP1-profile and vice versa. From this similarity, the re-interpolation procedure for the case PP2 can therefore be directly adapted from the case PP1.

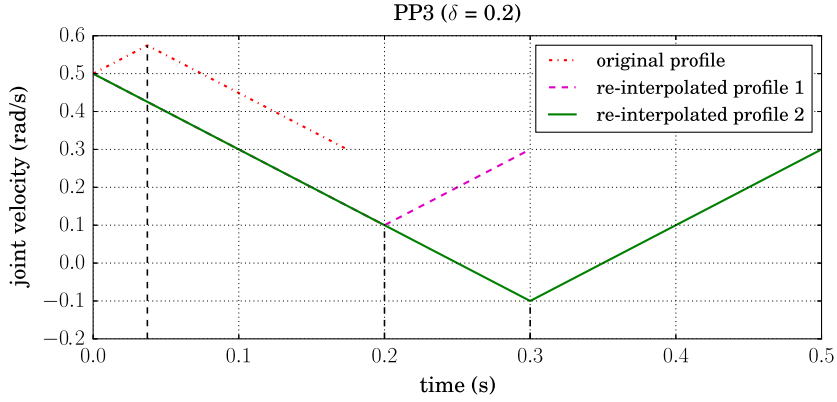


Figure 4. The original velocity profile is show in red. The minimum-switch-time constraint δ is set to be 0.2. By using v_p' computed from Equation (6), two velocity profiles can be constructed (shown in dashed magenta and solid green). In this case both profiles are valid.

three ramps; (B) the re-interpolated profile has two ramps. Then, we choose the case which gives the shorter duration.

In PLP3A, we proceed by stretching the duration of the middle ramp to δ while the accelerations of the other two ramps remain the same (maximum magnitude). Note that the new acceleration of the middle ramp might not be zero. Let t_0' and t_2' be the new durations of the first and the last ramps after modification. Assume, without loss of generality, that the acceleration of the first ramp is positive. We can derive the relationship between the two as

$$\left(t_0' + \left(\frac{\delta}{2} + \frac{v_0}{a_m}\right)\right)^2 + \left(t_2' + \left(\frac{\delta}{2} + \frac{v_1}{a_m}\right)\right)^2 = \frac{1}{a_m^2}\left(v_0^2 + v_1^2 + 2a_m d + \frac{1}{2}a_m^2\delta^2\right) \tag{7}$$

which is actually a circle centered at $(-\delta/2 - v_0/a_m, -\delta/2 - v_1/a_m)$. We can find the pair (t_0, t_2') which minimizes $t_0' + t_2'$ as follows. By considering the minimum-switch-time constraint and the velocity bounds, we have that $t_0' \in [\delta, (v_m - v_0)/a_m]$ and $t_2' \in [\delta, (v_m - v_1)/a_m]$. And by considering the acceleration bounds, we have the relation

$$\frac{v_0 - v_1}{a_m} - \delta \leq t_2' - t_0' \leq \frac{v_0 - v_1}{a_m} + \delta. \tag{8}$$

Finally, we can find a segment of the circle whose points satisfy all the constraints above. Note also that the segment is in the first quadrant in $t_0' - t_2'$ plane. Since we wish to minimize $t_0' + t_2'$, from the graph we can see that the minimizer must be at either end of the segment. We simply test which point gives the shorter trajectory duration and select it as the solution to t_0' and t_2'.

In some cases, there might be no intersection between the circle and the feasible region of t_0' and t_2'. This is because the acceleration constraint cannot be saturated at both the first and the last ramps. Therefore, the only constraint that can be saturated is the minimum-switch-time constraint. From direct calculations, we found that in order for the velocity profile to be time-optimal, the minimum-switch-time constraint has to be saturated at

all ramps. From that condition, we can derive the relationship between two peak velocities v'_{p0} and v'_{p1} as $v'_{p0} + v'_{p1} = d/\delta - (v_0 + v_1)/2$. By enforcing the acceleration constraint at all ramps, we can select the values v'_{p0} and v'_{p1} which satisfy all the constraints. Then, we re-interpolate the velocity profile accordingly.

In PLP3B, we merge two consecutive ramps together to into a single ramp. We choose either the first two ramps or the last two ramps depending on the resulting trajectory duration.

4.1.7. PLP4

This case can be divided into two subcases: (A) the re-interpolated profile has three ramps; (B) the re-interpolated profile has two ramps.

For PLP4A, we proceed in the same way as in PLP1A. For PLP4B, we proceed by merging the first two ramps together to get one single ramp. However, unlike PLP1 where PLP1B (merging the first two ramps together) will always give a valid velocity profile, the merged ramp can sometimes be shorter than δ. In that case, we re-interpolate the velocity profile again using the routines for PP1, i.e. the duration of the new first ramp is stretched to δ; the new peak velocity is calculated from Equation (5); the acceleration of the remaining ramp remains unchanged. Finally, we choose the case which gives the shorter duration.

4.1.8. PLP5

This case is *similar* to the case PLP4 and therefore the procedure can be directly adapted from that of PLP4.

4.1.9. PLP6

Here, we examine four exhaustive ways to modify the original velocity profile: (A) the durations of the first and the last ramps are stretched to δ. The new middle ramp stays at the velocity bound, hence shortened; (B) we stretch only the duration of the first ramp to δ. There are two subcases depending on the final velocity of the first ramp, v'_{p0}. In the first subcase, v'_{p0} is set to be at the velocity bound. The remaining portion of the velocity profile is re-interpolated as a two-ramp velocity profile using the routines for PP. In the second subcase, v'_{p0} is calculated from Equation (5) and the remaining portion is re-interpolated accordingly; (C) we stretch only the duration of the last ramp to δ. There are two subcases similar to (B); (D) we re-interpolate the velocity profile such that it has only one ramp.

Finally, we choose the case which gives the shorter duration. Note that at least the case (D) is valid since the total duration after the re-interpolation procedure is always greater than the duration of the original middle ramp, which is greater than δ.

4.1.10. PLP7

We first examine three cases of re-interpolated profiles: (A) the new velocity profile has three ramps. Since all ramps of the original velocity profile are shorter than δ, the duration of each ramp of the new velocity profile must be δ. To re-interpolate a velocity profile to be three-ramp with specified durations for all ramps, we use the procedure described in Appendix 3; (B) the new velocity profile has two ramps; and (C) the new velocity profile has one ramp.

To re-interpolate a velocity profile into two- or one-ramp (PLP7B or PLP7C), we can use the routine for the case PP3. After exploring all three cases, we choose a valid velocity profile (if any) which gives the shortest duration.

However, the case which all PLP7A, PLP7B, and PLP7C do not give any valid velocity profile can occur when the velocity limit is very low and one of the terminal velocities is zero. In this case, we proceed as follows. Let v_p be the peak velocity of the original trajectory. If $|v_p - v_0| > |v_p - v_1|$, we stretch the duration of the new first ramp to δ while the final velocity remains the same. Then, we re-interpolate the remaining portion to be one ramp. On the other hand, if $|v_p - v_0| < |v_p - v_1|$, we stretch the duration of the last ramp to δ while the initial velocity of the last ramp remains the same. Then, we re-interpolate the remaining portion to be one ramp.

4.2. Multiple DOFs

After independently interpolation velocity profiles for different joints for a shortcut path, each joint may have different time duration. However, in order to be a valid shortcut, velocity profiles of all the joints for the shortcut path must also have the same duration. Let the mth joint, $m \in \{1, 2, \ldots, n\}$, have the slowest velocity profile with the duration T. We have that the shortcut cannot have its duration less than T. The purpose here is to re-interpolate all the remaining $n - 1$ joints to have the same duration T.

Given a trajectory and a fixed time T, a velocity profile re-interpolation problem has either infinitely many solutions or no solution.[7] Moreover, even when the minimum-switch-time constraint is not taken into account deciding whether the problem has a solution is difficult.[21] Therefore, to simplify the problem so that we can solve it analytically, the class of the re-interpolated velocity profile (two- or three-ramp) and some free variables might be specified beforehand. The authors of [6] suggested to explore both classes and to constrain the accelerations of the first and the last ramps to have equal magnitude. However, constraining the acceleration magnitude is restrictive and therefore results in low

re-interpolation success rate, as can be seen in Section 5.3.

We propose here a routine which also explores both classes of possible velocity profiles but instead of constraining the accelerations, it makes some initial guesses on the duration of each ramp of the re-interpolated velocity profile. In addition, our routine can also take into account the minimum-switch-time constraint for multiple DOFs by requiring that

$$\min_{i,k,j,l} |t^{sw}(i,j) - t^{sw}(k,l)| \geq \delta, \tag{9}$$

where $t^{sw}(i,j)$ denotes the j^{th} switch point of the ith DOF. This implies that any pair of switch points will be separated by a duration of at least δ.

Suppose velocity profiles are $\gamma_i, i = \{1, 2, \ldots, n\}$ and γ_m is the slowest with the duration T. We need to re-interpolate all other velocity profiles, $\gamma_i, i = \{1, 2, \ldots, n\}$, $i \neq m$, such that they all have their new durations equal to T. We start by diving each ramp of γ_m into equally long segments whose durations are the smallest possible but still longer than δ. (in our implementation, the total number of such segments is capped to 50 to avoid long computation time when δ is small or null, i.e. no minimum-switch-time constraint). For example, suppose the first ramp of γ_m has the duration of t. We divide this ramp into N segments, where $t/(N+1) < \delta \leq t/N$. For this first ramp, we obtain a set of point $\{t/N, 2t/N, \ldots, t\}$. Then we draw vertical lines (on a $v-t$ plane), each line passing through each point (on the horizontal axis) in the set. We then repeat this procedure for all the remaining ramps. We call this set of all vertical lines a *grid*. An example of a constructed grid is shown in Figure 5. After constructing a grid, we explore the two classes of velocity profiles as follows.

4.2.1. Two-ramp velocity profile

We try to locate the unique switch point at each of grid lines. The acceleration of each ramp can be analytically determined given the location of the switch point (see Appendix 2).

4.2.2. Three-ramp velocity profile

There are $\binom{N}{2}$ possible ways to choose two locations of switch points out of N grid lines. To reduce computational cost, instead of trying all possibilities, we make further guess on the locations of the switch points. In particular, let t_0', t_1', and t_2' be durations of the first, the middle, and the last ramps of the new velocity profile. Further guesses are made on the ratios $t_0'/T, t_1'/T$, and t_2'/T. Choices of these three ratios are *arbitrary*. However, with some trial-and-error testings, we suggest the values $t_0'/T = 1/4$, $t_1'/T = 1/2$, and $t_2'/T = 1/4$ which give fairly high success rate of re-interpolation. Then, we examine four ways of snapping those switch points to their nearest grid lines. Given the duration of each ramp, we can formulate the problem of finding accelerations, a_0', a_1', and a_2', for all ramps as a feasibility problem which can be solved analytically (see Appendix 3).

If neither the two- nor the three-ramp attempt succeeds, i.e. solutions of the accelerations do not respect the acceleration bounds or there is no solution available, then the re-interpolation is considered as failed and the corresponding shortcut path is discarded.

5. Implementation and use in motion planning

5.1. Implementation and comparison with an optimization-based method

We implemented the algorithm in Python and provided it as open-source at https://github.com/Puttichai/parabint. Note that the running times reported in the sequel will be largely improved once our implementation is transcribed into C++.

Finding the time-optimal parabolic interpolation subject to velocity, acceleration, and minimum-switch-time constraints can also be formulated as an optimization problem. The optimization variables are $\boldsymbol{y} = [t_0', t_1', v_p']^T$ for two-ramp profiles, where v_p' is the new peak velocity and $\boldsymbol{z} = [t_0', t_1', t_2', v_{p0}', v_{p1}']^T$ for three-ramp profiles, where v_{p0}' is the initial velocity of the new second ramp and v_{p1}' is the initial velocity of the new third ramp.

Let the boundary conditions be (x_0, v_0) and (x_1, v_1), and $d = x_1 - x_0$. For two-ramp velocity profiles, the problem can be formulated as

$$\begin{aligned} \text{minimize}_{\boldsymbol{y}} \quad & \boldsymbol{c}_0^T \boldsymbol{y} \\ \text{subject to} \quad & \boldsymbol{G}_0 \boldsymbol{y} \preceq \boldsymbol{h}_0 \\ & \frac{1}{2} \boldsymbol{y}^T \boldsymbol{A}_0 \boldsymbol{y} + \boldsymbol{b}_0^T \boldsymbol{y} - 2d = 0, \end{aligned} \tag{10}$$

where $\boldsymbol{A}_0 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$, $\boldsymbol{b}_0 = \begin{bmatrix} v_0 \\ v_1 \\ 0 \end{bmatrix}$, $\boldsymbol{c}_0 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$,

$$\boldsymbol{G}_0 = \begin{bmatrix} -a_m & 0 & 1 \\ -a_m & 0 & -1 \\ 0 & -a_m & 1 \\ 0 & -a_m & -1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \text{ and } \boldsymbol{h}_0 = \begin{bmatrix} v_0 \\ -v_0 \\ v_1 \\ -v_1 \\ -\delta \\ -\delta \\ v_m \\ v_m \end{bmatrix}.$$

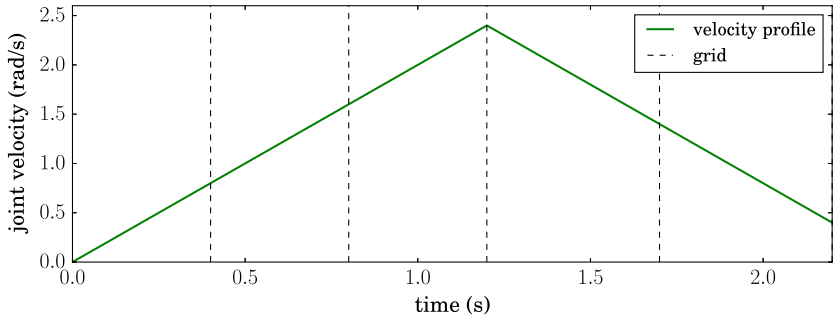


Figure 5. An example of a grid constructed for a velocity profile (shown in green). Here δ is set to 0.4. Therefore, each ramp must be divided into equally long segments of the smallest duration while still longer than 0.4 s. The first ramp of the velocity profile, which has the duration of 1.2 s., is then divided into three segments. The second ramp, with the duration of 1.0 s., is then divided into only two segments.

Similarly, an optimization problem for the three-ramp case can be formulated as

$$\begin{aligned} &\text{minimize}_{\boldsymbol{y}} && \boldsymbol{c}_1^T \boldsymbol{z} \\ &\text{subject to} && \boldsymbol{G}_1 \boldsymbol{z} \preceq \boldsymbol{h}_1 \\ & && \frac{1}{2}\boldsymbol{z}^T \boldsymbol{A}_1 \boldsymbol{z} + \boldsymbol{b}_1^T \boldsymbol{z} - 2d = 0, \end{aligned} \tag{11}$$

where $\boldsymbol{A}_1 = \begin{bmatrix} 0&0&0&1&0 \\ 0&0&0&1&1 \\ 0&0&0&0&1 \\ 1&1&0&0&0 \\ 0&1&1&0&0 \end{bmatrix}$, $\boldsymbol{b}_1 = \begin{bmatrix} v_0 \\ 0 \\ v_1 \\ 0 \\ 0 \end{bmatrix}$, $\boldsymbol{c}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$,

$$\boldsymbol{G}_1 = \begin{bmatrix} -a_m & 0 & 0 & 1 & 0 \\ -a_m & 0 & 0 & -1 & 0 \\ 0 & -a_m & 0 & 1 & -1 \\ 0 & -a_m & 0 & -1 & 1 \\ 0 & 0 & -a_m & 0 & 1 \\ 0 & 0 & -a_m & 0 & -1 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}, \text{ and } \boldsymbol{h}_1 = \begin{bmatrix} v_0 \\ -v_0 \\ 0 \\ 0 \\ v_1 \\ -v_1 \\ -\delta \\ -\delta \\ -\delta \\ v_m \\ v_m \\ v_m \\ v_m \end{bmatrix}.$$

Inequality constraints represent constraints on velocity, acceleration, and minimum-switch-time while equality constraints represent constraints on the total displacement of trajectories. From the above formulations, we can obviously see that both problems are non-convex since the matrices $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$ are not positive-semidefinite, i.e. some of their eigenvalues are negative. The non-convexity of the problems makes them difficult and more computationally expensive to solve. In addition, a solver can also be stuck in local minima.

For comparison, we created 10,000 random problem instances for each case of minimum-switch-time constraint violations (as enumerated in Table 1). Then, we used both proposed re-interpolation routines (Section 4) written in Python and PyIpopt[5] to solve the same problems. For PyIpopt solver, we set the tolerance to $1e-8$ and the maximum number of iterations to 2,000. Average computation time over all runs, the number of times the optimizer was stuck in the local minimum (the duration of the re-interpolated trajectory was longer than that returned by our proposed method), and the number of times the optimizer failed were recorded. The statistics are reported in Table 2. As it can be seen from Table 2, our proposed method took much less time to calculate the same solutions compared to solving via optimization. This emphasizes advantages of solving the problem analytically via the proposed method.

Table 2. A comparison between an optimization-based and the proposed methods.

	PP		PLP	
Algorithms	Optim.	Exact	Optim.	Exact
Comp. time (ms.)	6.51	0.0587	14.3	0.257
% stuck	8.5%	–	36.8%	–
% failed	6.8%	–	0.95%	–

Remark: An optimization-based method for interpolating cubic spline trajectories such as that presented in [9] required solving for a large number of optimization variables. The number of variables also grows with the number of via-points added between the initial and final robot configurations. However, one can see that even finding time-optimal parabolic interpolation subject to velocity, acceleration, and minimum-switch-time constraints via an optimization-based method, which requires solving for only a few unknown variables, took significantly more time than using our proposed method. This suggests that our method provides a reasonable trade-off between jerk-boundedness and computational efficiency.

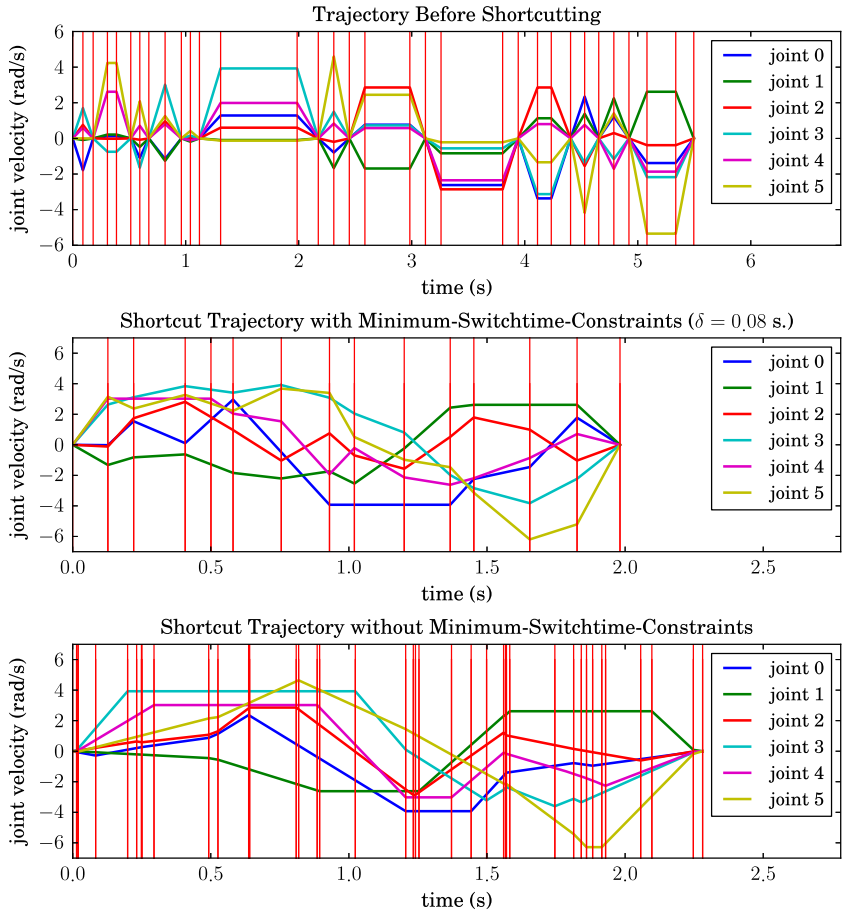


Figure 6. Velocity profiles of the original trajectory before shortcutting and of trajectories after shortcutting the original trajectory with and without considering minimum-switch-time constraints. The minimum-switch-time constraint is set to be $\delta = 0.08$ s. Red vertical lines indicate time instants when at least one DOF switches its acceleration (switch points).

5.2. *Trajectory planning and shortcutting with minimum-switch-time constraint*

Here, we include the minimum-switch-time constraint into the motion planning pipeline described in Section 2.1. In the trajectory planning stage, we use the routines described in Section 3 to time-parameterize the path taking into account the minimum-switch-time constraint. The parameterization is done through the path parameter s and therefore does not affect the path geometry.

In the shortcutting stage, the condition (9) also needs to be satisfied. In each shortcutting iteration, we begin by randomly selecting two time instants t_a and t_b. Let t_a^{sw} be the nearest switch point preceding t_a and t_b^{sw} the nearest consecutive switch point of t_b. If $t_a - t_a^{sw} < \delta$, then we re-assign t_a to be t_a^{sw}. Similarly, if $t_b^{sw} - t_b < \delta$, then we re-assign t_b to be t_b^{sw}. Next, we continue in a usual manner by first interpolating a velocity profile for each DOF independently and determining the duration T of the slowest DOF. In case $T > t_b - t_a$ the shortcut is discarded right away. Otherwise, we use the routines described in Section 4 to re-interpolate all the remaining velocity profiles again. And if the resulting trajectory is collision-free, then we replace the original pagination trajectory segment with the shortcut. Figure 6 shows velocity profiles of the original trajectory before shortcutting, shortcut trajectory with the minimum-switch-time constraint, and shortcut trajectory without the constraint. The original trajectory has duration 5.495 s. which is relatively long since it stops at every waypoint. The shortcut trajectory with the minimum-switch-time constraint (in this case $\delta = 0.08$ s.) has duration 1.981 s. while the shortcut trajectory (from the same original trajectory) without the minimum-switch-time constraint has duration 2.280 s. The shortcut trajectory with the minimum-switch-time constraint is shorter and its distribution of switch points is much more sparse. (Red vertical lines in each subfigure indicate time instants where at least one joint changes its acceleration.) In terms of the number of switch points, when not considering the minimum-switch-time constraint, the resulting trajectory has 40 switch points while our method produced only 14 switch points.

Table 3. Independent interpolation with arbitrary terminal velocities (6 DOFs): success rate and average computation time.

	$(\boldsymbol{x}_0, \dot{\boldsymbol{x}}_0)$ and $(\boldsymbol{x}_1, \dot{\boldsymbol{x}}_1)$ generated randomly		$(\boldsymbol{x}_0, \dot{\boldsymbol{x}}_0)$ and $(\boldsymbol{x}_1, \dot{\boldsymbol{x}}_1)$ randomly picked from trajectories			
			Trajectories connecting $\boldsymbol{x}_{\text{init},0}$ and $\boldsymbol{x}_{\text{goal},0}$		Trajectories connecting $\boldsymbol{x}_{\text{init},1}$ and $\boldsymbol{x}_{\text{goal},1}$	
	Success rate	Avg. comp. time (ms.)	Success rate	Avg. comp. time (ms.)	Success rate	Avg. comp. time (ms.)
From [6]	36.5%	0.364	61.0%	0.324	51.8%	0.381
$\delta = 0.0$	78.1%	1.394	90.8%	1.512	85.8%	1.606
$\delta = 0.008$	77.0%	1.431	89.8%	1.581	85.2%	1.666
$\delta = 0.1$	75.7%	1.420	75.4%	2.806	74.0%	2.859

Table 4. Both proposed method and the one from [6] were given 1 s. to shortcut given trajectories. The experiment was repeated 1000 times for each trajectory.

	A trajectory connecting $\boldsymbol{x}_{\text{init},0}$ and $\boldsymbol{x}_{\text{goal},0}$			A trajectory connecting $\boldsymbol{x}_{\text{init},1}$ and $\boldsymbol{x}_{\text{goal},1}$		
	Initial dur. (s.)	Avg. final dur. (s.)	Avg.# iterations	Initial dur. (s.)	Avg. final dur. (s.)	Avg. # iterations
From [6]	2.831	1.802	413.1	7.000	3.002	237.3
$\delta = 0.0$	2.831	1.360	171.71	7.000	2.275	131.6
$\delta = 0.008$	2.831	1.372	178.4	7.000	2.272	127.8
$\delta = 0.1$	2.903	1.710	228.7	7.088	2.235	153.1

5.3. Simulation results and comparisons with an existing method

For all the simulations, we used the joint velocity and acceleration bounds of Denso VS-060 which is a 6-DOF industrial manipulator.

First, we made a comparison between a re-interpolation routine proposed in [6] and the routine proposed in Section 4 with three minimum-switch-time constraints: $\delta = 0$ (no minimum-switch-time constraint), $\delta = 0.008$ s., and $\delta = 0.1$ s., in terms of success rate of independent interpolation. Comparison was made with two cases of generated pairs of boundary conditions $\{(\boldsymbol{x}_0, \dot{\boldsymbol{x}}_0), (\boldsymbol{x}_1, \dot{\boldsymbol{x}}_1)\}$. In the first case, we randomly generated 1000 pairs of boundary conditions while in the other case, 1000 pairs of boundary conditions were randomly picked from existing trajectories. For the latter case, we started with two (random) pairs of $\boldsymbol{x}_{\text{init}}$ and $\boldsymbol{x}_{\text{goal}}$. Then, we planned 50 trajectories connecting each pair using the aforementioned planning procedure (without shortcutting). Then for each of 100 trajectories, we randomly picked 10 pairs of boundary conditions. The statistics in Table 3 show that the re-interpolation routines proposed in [6] give relatively low success rate due to restrictive constraints on ramp accelerations.

For the next comparison, a piecewise linear path was generated for each of two pairs of the initial and goal configurations. Then for each of path segments, we simultaneously interpolated velocity profiles with zero terminal velocities, both with and without the minimum-switch-time constraint. Finally, we gave 1 s. to both proposed method and the one proposed in [6] to shortcut each trajectory. This process was repeated 1000 times. Initial trajectory durations, average final durations, and numbers of shortcutting iterations executed from each method are reported in Table 4.

Despite the fact that the proposed method is more computationally expensive as can be seen from Table 3, when the same amount of time are given to both methods, the proposed method still performs better in terms of resulting trajectory duration after shortcutting.

Remark: From the results presented in Table 3, the proposed method is fast enough to be used in a real-time controller to interpolate a trajectory between two given waypoints. Suppose a robot has six DOFs. The longest average computation time for independent interpolation with arbitrary terminal velocities is less than 3 ms. Therefore, the proposed interpolation method can be used in a real-time controller with frequency of at least 300 Hz.

Remark: Applicability of parabolic trajectories to real robots has also been verified through our recent experiment. We employed our proposed method to plan and shortcut parabolic trajectories for a pick-and-place manipulation task.[30] The video of our robot executing the motion can be found at https://youtu.be/tLouwj0wITQ. One can see that the robot motion is sufficiently smooth to complete the task even without the jerk-boundedness constraint.

6. Conclusion

In this paper, we have studied the minimum-switch-time constraint, which is a constraint on time intervals between consecutive switch points. Integrating this constraint into interpolation routines prevents high-frequency switching. We presented an algorithm for planning piecewise parabolic trajectories subject to velocity, acceleration, and minimum-switch-time constraints. Simulation results show that our method is more computationally efficient than an optimization-based

method and also achieves a higher interpolation success rate and produces higher quality trajectories as compared to existing methods.

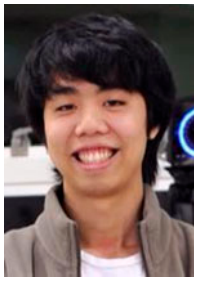
Notes

1. Some planners, such as RRT* [29] optimize path quality during the planning itself, as opposed to the post-processing approach advocated here. However, such planners have large execution overheads and are seldom used in practical industrial settings.
2. There are cases where the magnitude of velocity is maximum at one end. In these cases, the velocity profiles have two ramps, one ramp having zero acceleration. In this paper, we consider these cases as special cases of P^+P^- and P^-P^+.
3. The calculations in those papers are actually similar. They differ from one another in some minor details and applications. In the sequel, when we discuss velocity profile interpolation without the minimum-switch-time constraint, we will primarily address the work in [6] since the authors also integrated this interpolation method into a trajectory smoothing (shortcutting) process.
4. The term similarity here refers to similarity in the geometric sense. In particular, a PP2-velocity profile can be seen as a mirror image along a horizontal axis of the corresponding PP1-profile. Figure 3 shows a PP2-profile and its corresponding PP1-profile. Therefore, the procedure for the case PP2 is essentially the same as the procedure for the case PP1.
5. PyIpopt is a freely available Python optimization module; see https://github.com/xuy/pyipopt.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors



Puttichai Lertkultanon was born in Bangkok, Thailand. He received the BE degree (First-Class Honours) in Electrical Engineering from Chulalongkorn University, Bangkok, Thailand, in 2013. He is currently working towards the PhD degree in the School of Mechanical and Aerospace Engineering at Nanyang Technological University, Singapore. His research interest includes robotic motion planning, kinodynamic motion planning, and manipulation planning.

Quang-Cuong Pham was born in Hanoi, Vietnam. He graduated from the Department of Computer Science and the Department of Cognitive Sciences, École Normale Supérieure, Paris, France, in 2007. He received the PhD degree in neuroscience from Université Paris VI and Collège de France in 2009. In 2010, he was a Visiting Researcher with University of São Paulo, Brazil. From 2011 to 2013, he was a Researcher with University of Tokyo, supported by a fellowship from the Japan Society for the Promotion of Science. He joined the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore, as an Assistant Professor in 2013.

References

[1] Pham QC. Trajectory planning. Handbook of manufacturing engineering and technology. London: Springer; 2015. p. 1873–1887.

[2] Bobrow JE, Dubowsky S, Gibson JS. Time-optimal control of robotic manipulators along specified paths. Int. J. Rob. Res. 1985;4:3–17.

[3] Pham QC. A general, fast, and robust implementation of the time-optimal path parameterization algorithm. IEEE Trans. Rob. 2014;30:1533–1540.

[4] Shiller Z, Dubowsky S. Robot path planning with obstacles, actuator, gripper, and payload constraints. Int. J. Rob. Res. 1989;8:3–18.

[5] Donald B, Xavier P, Canny J, et al. Kinodynamic motion planning. J. ACM. 1993;40:1048–1066.

[6] Hauser K, Ng-Thow-Hing V. Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA '10), 2010; Anchorage (AK); 2010. p. 2493–2498.

[7] Kunz T, Stilman M. Probabilisitically complete kinodynamic planning for robot manipulators with acceleration limits. IEEE/RSJ international conference on intelligent robots and systems (IROS), 2014; Chicago (IL); 2014. p. 3713–3719.

[8] Kyriakopoulos KJ, Saridis GN. Minimum jerk path generation. In: Proceedings of IEEE international conference on robotics and automation, 1988; Vol. 1, Philadelphia (PA); 1988. p. 364–369.

[9] Gasparetto A, Zanotto V. A technique for time-jerk optimal planning of robot trajectories. Rob. Comput. Integr. Manuf. 2008;24:415–426.

[10] Haschke R, Weitnauer E, Ritter H. On-line planning of time-optimal, jerk-limited trajectories. IEEE/RSJ international conference on intelligent robots and systems (Iros), 2008; Nice, France; 2008. p. 3248–3253.

[11] Liu S. An on-line reference-trajectory generator for smooth motion of impulse-controlled industrial manipulators. 7th international workshop on advanced motion control 2002; Maribor, Slovenia; 2002. p. 365–370.

[12] Broquére X, Sidobre D, Herrera-Aguilar I. Soft motion trajectory planner for service manipulator robot. IEEE/RSJ international conference on intelligent robots and systems (IROS); Nice, France; 2008. p. 2808–2813.

[13] Kröger T, Wahl FM. Online trajectory generation: Basic concepts for instantaneous reactions to unforseen events. IEEE Trans. Rob. 2010;26:94–111.

[14] Ahn K, Chung WK, Youm Y. Arbitrary states polynomial-like trajectory (ASPOT) generation. 30th annual conference of IEEE Industrial electronics society (IECON), 2004; Vol. 1, Busan, South Korea; 2004. p. 123–128.

[15] Macfarlane S, Croft EA. Jerk-bounded manipulator trajectory planning: design for real-time applications. IEEE Trans. Rob. Autom. 2003;19:42–52.

[16] Ezair B, Tassa T, Shiller Z. Planning high order trajectories with general initial and final conditions and asymmetric bounds. Int. J. Rob. Res. 2014;33:898–916.
[17] Piazzi A, Visioli A. Global minimum-jerk trajectory planning for robot manipulators. IEEE Trans. Ind. Electron. 2000;47:140–149.
[18] Thompson SE, Patel RV. Formulation of joint trajectories for industrial robots using B-splines. IEEE Trans. Ind. Electron. 1987;IE-34:192–199.
[19] Erkorkmaz K, Altintas Y. High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation. Int. J. Mach. Tools Manuf. 2001;41:1323–1345.
[20] Simon D, Isik C. A trigonometric trajectory generator for robotic arms. Int. J. Control. 1993;57:505–517.
[21] Kröger T, Tomiczek A, Wahl FM. Towards on-line trajectory computation. IEEE/RSJ international conference on intelligent robots and systems (IROS); Beijing, China; 2006. p. 736–741.
[22] Geraerts R, Overmars M. Creating high-quality paths for motion planning. Int. J. Rob. Res. 2007;26:845–863.
[23] Kienitz KH. Attitude stabilization with actuators subject to switching restrictions: an approach via exact relay control methods. IEEE Trans. Aerosp. Electron. Syst. 2006;42:1485–1492.
[24] Vieira MS, Galvão KH, Kienitz KH. Attitude stabilization with actuators subject to switching-time constraints using explicit MPC. Aerospace conference 2011. IEEE; Big Sky (MT); 2011. p. 1–8.
[25] Luo J, Hauser K. An empirical study of optimal motion planning. IEEE/RSJ international conference on intelligent robots and systems (IROS 2014). Chicago (IL); 2014. p. 1761–1768.
[26] Diankov R. Automated construction of robotic manipulation programs [PhD thesis]. Carnegie Mellon University, Robotics Institute. August 2010. Available from: http://www.programmingvision.com/rosen_diankov_thesis.pdf.
[27] Kant K, Zucker SW. Toward efficient trajectory planning: the path-velocity decomposition. Int. J. Rob. Res. 1986;5:72–89.
[28] Lavalle SM. Rapidly-exploring random trees: a new tool for path planning. Technical Report. Ames (IA): Iowa State University; 1998.
[29] Karaman S, Frazzoli E. Sampling-based algorithms for optimal motion planning. Int. J. Rob. Res. 2011;30:846–894.
[30] Lertkultanon P, Pham QC. A single-query manipulation planner. IEEE Rob. Autom. Lett. 2015;1:198–205.

Appendix 1. Proof of time-optimality

We present here detailed proofs for the cases PP1 and PLP1. Proofs for the other cases are not given: proofs for the cases PP2 and PLP2 can be easily adapted from PP1 and PLP1, respectively; for the remaining cases, we have already explored all possibilities of re-interpolation.

Consider first the case PP1. Let T_{PP1} be the total duration of the newly re-interpolated velocity profile and t_0' the duration of the new first ramp. Thus, $T_{\text{PP1}} = t_0' + (v_1 - v_p')/a_1$, where v_1 and a_1 are the final velocity and the acceleration of the second ramp, and v_p' is the new peak velocity. The following proposition holds.

Proposition A: *T_{PP1} is an increasing function in t_0'.*

Proof: We first consider the case P^+P^-. Substituting the expression of v_p' from (5) and $a_1 = -a_m$ into the expression of T_{PP1} yields

$$T_{\text{PP1}} = t_0' + \frac{1}{2a_m}\left(k_1 + \sqrt{k_1^2 + 4k_1v_0 + 4v_1^2 + 8a_md}\right) - \frac{v_1}{a_m}$$
$$T_{\text{PP1}} = \frac{1}{2}\left(t_0' - \frac{2v_1}{a_m}\right) + \frac{1}{2}\sqrt{\Delta},$$

where $\Delta = \left(t_0' - 2v_0/a_m\right)^2 + 4\left(v_1^2 - v_0^2 + 2a_md\right)/a_m^2$. The first derivative of T_{PP1} with respect to t_0' is then given by

$$\frac{\mathrm{d}}{\mathrm{d}t_0'}T_{\text{PP1}} = \frac{1}{2} + \frac{1}{2}\frac{t_0' - \frac{2v_0}{a_m}}{\sqrt{\left(t_0' - \frac{2v_0}{a_m}\right)^2 + \frac{4}{a_m^2}\left(v_1^2 - v_0^2 + 2a_md\right)}}. \tag{A1}$$

If $t_0' > 2v_0/a_m$, it follows directly that $\mathrm{d}T_{\text{PP1}}/\mathrm{d}t_0' > 0$. Otherwise, we examine the second term in the square root.

Before proceeding further, notice that for all P^+P^-1-velocity profile, we always have that $v_0 > v_1$. Assume, for contradiction, that a velocity profile falls in the case P^+P^-1 with $v_0 < v_1$. Since $|a_0| = |a_1| = a_m$, we then have that $(v_p - v_0)/a_m > (v_p - v_1)/a_m$ which results in $t_0 > t_1$. This contradicts with the assumption that the trajectory is in the case P^+P^-1.

Now consider when $v_0 > v_1$. The time-optimal velocity profile connecting v_0 and v_1 subject to velocity and acceleration bounds has one ramp if and only if the total displacement is $d^* = (v_0^2 - v_1^2)/(2a_m)$. Therefore, for any P^+P^--velocity profile, the total displacement must be greater than d^*, i.e.

$$v_1^2 - v_0^2 + 2a_md > 0 \tag{A2}$$

By applying (A2) to (A1), we can see that $\mathrm{d}T_{\text{PP1}}/\mathrm{d}t_0'$ is always positive. This completes the proof for the case P^+P^-. For the case P^-P^+, the same argument which leads to $\mathrm{d}T_{\text{PP1}}/\mathrm{d}t_0' > 0$ can be made. This completes the proof for the case PP1. □

Therefore, following Proposition A, to time-optimally re-interpolate a velocity profile subject to the minimum-switch-time constraint δ, the resulting duration of the first ramp must be the shortest possible, $t_0' = \delta$.

Next, consider the case PLP1. Let t_0' be the duration of the first ramp of the re-interpolated velocity profile. We claim that in order for the re-interpolated velocity profile to be time-optimal as well as satisfying all the constraints, t_0' must be only in $[t_0^A, t_0^B]$, where t_0^A is the duration of the new first ramp in case the velocity profile is re-interpolated according to the case PLP1A, i.e. $t_0^A = \delta$, and t_0^B is the duration of the new first ramp in case the velocity profile is re-interpolated according to the case PLP1B. Figure A1 shows velocity profiles of the original PLP1-velocity profile as well as resulting velocity profiles from the cases PLP1A and PLP1B.

The re-interpolation procedure of a PLP1-velocity profile can be thought of as first re-interpolating the first ramp with a duration $t_0' \in [t_0^A, t_0^B]$ and then re-interpolating the rest of the profile using routines for PP1. Let $d_0' = (v_p' - v_0)t_0'/2$ be the displacement covered by the first ramp of the newly re-interpolated velocity profile. The displacement covered by the remaining part of the velocity profile is then $d_{\text{rem}} = d - d_0'$. Let the total duration of the new velocity profile be T_{PLP1}. The following proposition holds.

Proposition B: *T_{PLP1}, as a function of t_0', attains its minimum only when $t_0' = t_0^A$ or $t_0' = t_0^B$.*

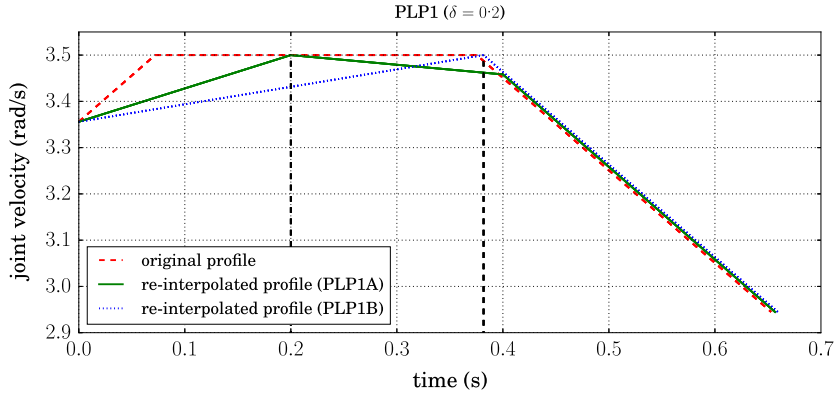


Figure A1. The original PLP1-velocity profile which violates the minimum-switch-time constraint $\delta = 0.2$ is shown in red. The green and blue velocity profiles result from re-interpolating according to PLP1A and PLP1B respectively. The time instants t_0^A and t_0^B are marked with a dashed green line and a dotted blue line, respectively. In this particular example, re-interpolation with PLP1A gives a shorter velocity profile.

Proof: We first consider the case when $v_p' = v_m$. The total duration T_{PLP1} can be expressed as

$$T_{\text{PLP1}} = t_0' + T_{\text{PP1}}$$
$$T_{\text{PLP1}} = t_0' + \frac{1}{2}\left(\delta - \frac{2v_1}{a_m}\right) + \frac{1}{2}\sqrt{\left(\delta - \frac{2v_p'}{a_m}\right)^2 + \frac{4}{a_m^2}\left(v_1^2 - {v_p'}^2 + 2a_m d_{rem}\right)}.$$

From the expression of T_{PLP1}, we can compute $d^2 T_{\text{PLP1}}/dt_0'^2$ which is always negative. This implies that the minimum of T_{PLP1} can only occur when either $t_0' = t_0^A$ or $t_0' = t_0^B$.

For the case when $v_p' = -v_m$, the similar argument which leads to $d^2 T_{\text{PLP1}}/\mathrm{d}t_0'^2$ can be made. This completes the proof for the case PLP1. □

Therefore, following Proposition B, we can conclude that the time-optimal velocity profile subject to the minimum-switch-time constraint δ must be obtained from either PLP1A or PLP1B.

Appendix 2. Velocity profile re-interpolation to be two-ramp

Let T be the given total duration and τ_0 the given switch point. Since the total displacement must remain the same after re-interpolation, we have

$$d = \left(v_0\tau_0 + \frac{1}{2}a_0'{\tau_0}^2\right) + \left((v_0 + a_0'\tau_0)\tau_1 + \frac{1}{2}a_1'{\tau_1}^2\right), \quad \text{(B1)}$$

where a_0' and a_1' are accelerations of the first and the last ramp of the new velocity profile respectively, and $\tau_1 = T - \tau_0$. We also have a relationship between v_0 and v_1 as

$$v_1 = v_0 + a_0'\tau_0 + a_1'\tau_1. \quad \text{(B2)}$$

We can rewrite Equations (B1) and (B2) as a system of linear equations in a_0' and a_1' as

$$\begin{bmatrix} \frac{1}{2}\tau_0^2 + \tau_0\tau_1 & \frac{1}{2}\tau_1^2 \\ \tau_0 & \tau_1 \end{bmatrix} \begin{bmatrix} a_0' \\ a_1' \end{bmatrix} = \begin{bmatrix} d - v_0 T \\ v_1 - v_0 \end{bmatrix}. \quad \text{(B3)}$$

We can then solve for a_0' and a_1'. Since the determinant of the square matrix is $\tau_0\tau_1(\tau_0 + \tau_1)/2 \neq 0$, the system (B3) is always solvable.

Appendix 3. Velocity profile re-interpolation to be three-ramp

Let T be the given total duration; τ_0, τ_1, τ_2 the desired durations of the first, the middle and the last ramps of the new velocity profile, respectively. Since the total displacement must remain the same after re-interpolation, we have

$$d = \left(v_0\tau_0 + \frac{1}{2}a_0'{\tau_0}^2\right) + \left((v_0 + a_0'\tau_0)\tau_1 + \frac{1}{2}a_1'{\tau_1}^2\right) + \left((v_0 + a_0'\tau_0 + a_1'\tau_1)\tau_2 + \frac{1}{2}a_2'{\tau_2}^2\right), \quad \text{(C1)}$$

where v_0 is the initial velocity; v_{p0}' and v_{p1}' are the first and the second peak velocities, respectively; a_0', a_1', and a_2' are accelerations of the first, middle, and last ramps of the new velocity profiles, respectively. Let $\alpha = \tau_0^2/2 + \tau_0\tau_1 + \tau_0\tau_2$, $\beta = \tau_1^2/2 + \tau_1\tau_2$, and $\sigma = \tau_2^2/2$. Then, Equation (C1) can be more compactly rewritten as

$$\alpha a_0' + \beta a_1' + \sigma a_2' = d - v_0 T. \quad \text{(C2)}$$

Furthermore, we can derive a relationship between v_0 and v_1, the final velocity, as

$$v_1 = v_0 + \tau_0 a_0' + \tau_1 a_1' + \tau_2 a_2'. \quad \text{(C3)}$$

We can write Equations (C2) and (C3) as a system of linear equations in $\boldsymbol{a} = [a_0', a_1', a_2']^T$ as

$$\boldsymbol{A}\boldsymbol{a} = \boldsymbol{b}, \quad \text{(C4)}$$

where $A = \begin{bmatrix} \alpha & \beta & \sigma \\ \tau_0 & \tau_1 & \tau_2 \end{bmatrix}$ and $\boldsymbol{b} = \begin{bmatrix} d - v_0 T \\ v_1 - v_0 \end{bmatrix}$.

The system (C4) is underdetermined. Therefore, a general solution, $\boldsymbol{a}_g$, to the system can be expressed as $\boldsymbol{a}_g = \boldsymbol{a}_p + \boldsymbol{a}_h$, where $\boldsymbol{a}_p$ is a particular solution and $\boldsymbol{a}_h$ is a non-trivial element of the null space of $\boldsymbol{A}$, i.e. satisfying $\boldsymbol{A}\boldsymbol{a}_h = 0$. Here we take the minimum-norm solution to (C4) as a particular solution, i.e. $\boldsymbol{a}_p = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}\boldsymbol{b}$.

For a homogeneous solution, we perform Gaussian elimination on $\boldsymbol{A}$ to obtain

$$\boldsymbol{a}_h = k \begin{bmatrix} (\beta\tau_2 - \sigma\tau_1)/(\alpha\tau_1 - \beta\tau_0) \\ (\alpha\tau_2 - \sigma\tau_0)/(\alpha\tau_1 - \beta\tau_0) \\ 1 \end{bmatrix},$$

where $k \in \mathbf{R}$.

Next, consider the velocity bound v_m. The peak velocities of the re-interpolated velocity profiles are $v'_{p0} = v_0 + a'_0\tau_0$ and $v'_{p1} = v_1 - a'_2\tau_2$. Therefore, we get two sets of constraints on the peak velocities, in terms of the accelerations, as

$$\begin{aligned} (-v_m - v_0)/\tau_0 \le a'_0 \le (v_m - v_0)/\tau_0 \text{ and} \\ (-v_m + v_1)/\tau_0 \le a'_2 \le (v_m + v_1)/\tau_0. \end{aligned}$$

Combining velocity and acceleration bounds together yields constraints on accelerations as

$$\boldsymbol{l} \preceq \boldsymbol{a}_p + k\boldsymbol{a}_h \preceq \boldsymbol{u}, \tag{C5}$$

where

$$\boldsymbol{l} = \begin{bmatrix} \min\left(\dfrac{v_m + v_0}{\tau_0}, a_m\right) \\ -a_m \\ \min\left(\dfrac{v_m - v_1}{\tau_0}, a_m\right) \end{bmatrix} \text{ and } \quad \boldsymbol{u} = \begin{bmatrix} \min\left(\dfrac{v_m - v_0}{\tau_0}, a_m\right) \\ a_m \\ \min\left(\dfrac{v_m + v_1}{\tau_0}, a_m\right) \end{bmatrix}.$$

Finally, we can solve for the feasible interval k from (C5). If there is no such k satisfying (C5), then no feasible solution for acceleration exists. Otherwise, any k in the interval can be used to construct a feasible solution for accelerations.