Smooth Cubic Polynomial Trajectories for Human-Robot Interactions

Daniel Sidobre, Kevin Desormeaux

▶ **To cite this version:**

Daniel Sidobre, Kevin Desormeaux. Smooth Cubic Polynomial Trajectories for Human-Robot Interactions. Journal of Intelligent and Robotic Systems, 2019, 95 (3-4), pp.851-869. 10.1007/s10846-018-0936-z . hal-01672043v3

HAL Id: hal-01672043

https://laas.hal.science/hal-01672043v3

Submitted on 25 Jan 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NoDerivatives 4.0 International License

Smooth Cubic Polynomial Trajectories for Human-Robot Interactions

Daniel Sidobre Kevin Desormeaux
LAAS-CNRS, Université de Toulouse, CNRS, UPS
Toulouse, France
email {daniel.sidobre, kevin.desormeaux}@laas.fr

September 2018

Abstract

With the growing importance of Human-Robot Interaction (HRI), the movement of the robots requires more and more attention to address the issues related to safety, efficiency and ergonomics. Trajectories are excellent candidates in the making of desirable motions designed for collaborative robots, because they allow to simply and precisely describe the motions. Despite the large number of works available for Online Trajectory Generation (OTG), there was, to our knowledge, no complete solution capable to simultaneously meet all the requirements of these new applications. In this paper we present a complete trajectory generation algorithm that build trajectories from arbitrary initial and final conditions, subject to general asymmetric bounds on jerk, acceleration and velocity. A review of the state of the art exposes the limits of the previous OTG works and reveals the difficult problem of non-linearity related with short motions. We explain how these non-linearities introduce discontinuities and we propose a solution based on sequences of segment of third degree polynomial functions.

Keywords Online trajectory generation time-optimal trajectories human-robot interaction motion control and planning third degree polynomial trajectories

1 Introduction

Trajectory generation then Online Trajectory Generation (OTG) has generated a lot of work and results from the beginning of robotic and until now. A good trajectory allows improving, not only, the performances of the robots in terms of speed, accuracy or reliability, but also, the safety and comfort of humans working with them. By giving a better description of the robot motion, this model improves the design of planners and controllers as well as the communication between them, thus leading to better architectures. As a trajectory describes both the path and the time evolution along it, the associated mathematical function is complex. Even in the case of a straight-line point-to-point motion, a double-S time trajectory is necessary to describe a smooth motion. Smoothness is associated to the number of derivative of the function, generally to the functions that are at least two times differentiable.

With the new robotic application domains, beside the traditional constraints of collisions, accuracy, kinematic or dynamic, trajectories must comply with a growing number of requirements relative to safety, comfort, ergonomics, energy consumption, or flexibility. A large variety of mathematical functions have been proposed to address the multiple constraints that a motion has to satisfy. In this paper we thoroughly examine the sequences of third order polynomial functions, one of the simpler models to build smooth trajectories.

However, despite the many works in the field, trajectory generation still encounters serious difficulties especially for short and asymmetric moves. These harsh points have been encountered in several previous papers; we attempt to address them in this work. The time-optimal curve regarding the length of the motion

might be non-linear and present discontinuities, particularly for short motions or velocity shift.

We propose the first algorithm to generate smooth trajectories defined by a chain of cubic polynomial functions that joins two arbitrary conditions defined by position, velocity and acceleration in minimum time under asymmetric bounds on velocity, acceleration and jerk. In particular, this algorithm solves efficiently the complex and highly non-linear cases where the distance to travel is short and the initial and final velocities are non-null. These cases appear in particular for trajectory control and multi-axis synchronization. Also, although only the one-dimensional problem of time-optimal motion is considered in this paper, these results have direct consequences for control as well as for multi-axis synchronization. Solving this fundamental problem is necessary to solve more general problems in N-dimension and with several via-points.

The paper is organized as follows. In Section 1.1, a state of the art is presented. The time-optimal cubic polynomials trajectories are introduced in Section 2. The duration of the trajectories according to the length is presented in Section 3. The time-optimal trajectories are developed in Section 4. Sections 5 and 6 gives some details in the solving of the related equations. Discussions are presented in Section 7 and Section 8 concludes this paper.

1.1 From path planning to trajectory planning

Paths, which are pure geometric representation, have been the most widely used approach to describe movements. Among the numerous works that have covered this subject, the works of Latombe [38], LaValle [39] and Kavraki et al. [27] offer a good synthesis. However the time is not taken into account by these path-based approaches, which suggests considering trajectories. Trajectories are continuous and differentiable functions of time, defining the evolution of the position of the robot. They can be defined using either Cartesian or joint coordinates. A trajectory can also be seen as the combination of a path with a law of temporal evolution. A summary of trajectory generation in an industrial context is proposed by Biagiotti [4], while Khalil [28] and Dombre [12] offer the same for robotic manipulators.

Using trajectories instead of paths provides significant advantages [52]:

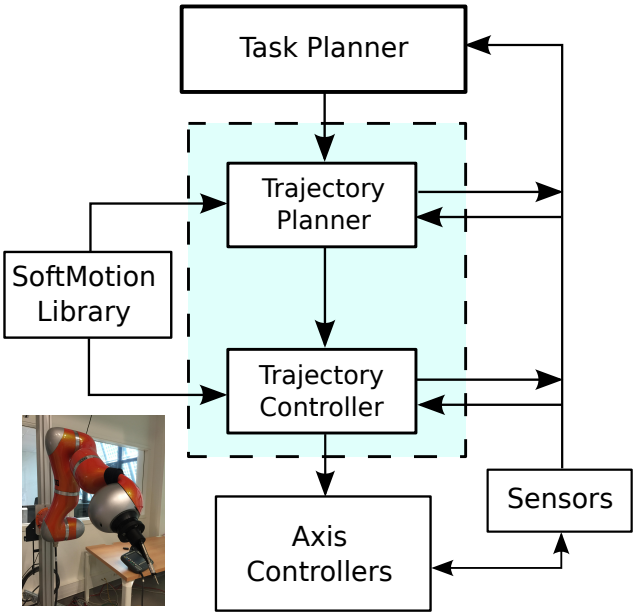


Figure 1: A possible architecture of a robot that uses trajectories for the control and the communication between the different modules.

- Not only the position can be planned and controlled, but also the velocity, the acceleration, the jerk and eventually higher derivative, allowing to define the smoothness properties.
- The travel time can be optimized taking into account kinematic constraints.
- The movement is more precisely described allowing a better control.

Moreover, the use of trajectories as the main support of information between a slow task planner and the fast robot controllers leads to a more refined architecture that simplifies the communications (See Fig. 1). A higher granularity in the architecture is desirable as it simplifies the input for each node that can run at different frequencies according to the nature of the task. The overall communication is improved, and in case of unforeseen event, a part of the task can be replanned at a lower level, thus faster. A similar architecture is proposed by Macfarlane et al. [41].

1.2 Trajectory planning in HRI context

With the emergence of new generations of robots that are always more sophisticated, one of the most chal-

lenging features of today's robotic, and even more in the future, is Physical Human-Robot Interaction (PHRI). These new robots will have to coexist and cooperate with humans in a variety of applications such as collaborative assembly, rehabilitation, domestic assistance, and many others. A survey on human-machine cooperation in assembly is proposed by Krüger et al. [32]. They study the forms of cooperation between a Human and a Robot that can be used in assembly processes as well as the organizational and economic aspects. They describe the advantages of combining the respective strength of a Human and a Robot, and advocate in favor of these new kinds of cooperation, even compared to fully automated systems. These robots will have to be adaptable, flexible, and reusable.

To be able to work in proximity of humans, collaborative robots must meet a number of requirements that often differ from those required in a classic industrial environment, where robots are confined in cages with prohibited access for humans. While safety remains the main criterion, others emerge such as the level of stress and discomfort the human can feel in the vicinity of the robot. In this context, a robot should not cause excessive stress and discomfort to the human for extended periods of time [2, 9, 19, 35–37]. To fulfill these requirements, some works focus on the behavioral and societal aspects [44, 53]. They attempt to understand the implicit rules and codes that define Humans interactions. Such an approach can be used for the robot to anticipate Humans actions as a Human will be more efficient and more satisfied of the interaction if the robot can anticipate his actions [24]. It can also be used for a robot to communicate its intents. A user will feel more comfortable knowing the goal of the robot early in its movements [13, 14]. Most of these works are integrated at the highest layers of planning, but they make sense only if the movement has been designed beforehand at a lower level. Motions have to be built from a model satisfying strict criteria. This model has to be simple enough for real-time constraints that are linked to security. It should also be capable to generate human-friendly movements to satisfy ergonomics constraints for a large variety of applications. Considering the previous bibliographic study, smooth trajectories appear as excellent candidates since they possesses the advantages necessary to ensure safety, ergonomics, efficiency and adaptability required in the making of collaborative motions.

1.3 Smooth trajectories

Smooth trajectories were first introduced by Hogan [26]. The first objective was to reduce wear on systems and improve path tracking [11, 33]. These qualities make smooth trajectories particularly interesting for machining. For instance a study realized by Rivera-Guillen et al. [50] shows that tool-life can be improved up to 60%. One can resume the advantages of working with smooth trajectories as follow:

- Improve accuracy, thus moves can be executed more rapidly and accurately.
- Extend the life span of the manipulators as vibrations are reduced thereby preventing actuators to be damaged and reducing wear of the robot joints.
- Reduce stress and discomfort of human co-worker.

Because of their qualities, smooth trajectories can be used in many contexts and can be employed in the making of more efficient and flexible robots. The smoothness of a trajectory can be defined by the number of derivative of the position and the extreme values of these derivatives. It is generally accepted that a smooth trajectory has at least continuous speed and acceleration, hence a bounded jerk. Considering a constant jerk during a period of time and its triple integral with respect to time, the obtained trajectory is defined by a cubic polynomial function of time. As these cubic functions are simple and their properties well known, they are easy to manipulate, and thus are widely used.

However higher order polynomials, especially quintics, are often used in the literature to obtain smooth trajectories. The main reason being the need to compute trajectories with the possibility to specify position, velocity and acceleration at both ends, so that the robot is able to react quickly to unforeseen events, an imperative in HRI context to ensure safety [29]. One quintic is enough to compute a trajectory that meet these criteria, hence justifying their use [11]. Unfortunately, quintics generate more computational burden than cubics and their behavior between the waypoints is more unpredictable and less faithful to the expected trajectory. This is explained by the tendency to oscillate of the quintics and generally higher order trajectories [41]. A solution is to use more than one quintic, but doing so they loose all interest, as simpler solutions exist. More recently, sequences of cubic polynomial functions have been used to define a smooth trajectory

joining arbitrary positions, velocities and accelerations [8, 57]. In the following paragraphs, we present the major results on smooth trajectories categorized in two main approaches.

1.3.1 Minimum-jerk model

In order to generate suitable movements for a Human-Robot interaction, human motion appears naturally as a source of inspiration. Among the numerous works that have covered this field, the minimum jerk model described by Hogan is very popular. This mathematical model describes the organization of a class of voluntary arm movements. A major uncertainty of this model is that it lays on a subjective criterion: since movements tends to be smoother and more graceful with skill and practice, they suppose the ideal motion should be the smoothest [17]. Maximizing the smoothness is here obtained by a dynamic optimization of an objective function that integrates the square of the jerk over the duration of the movement. This model was first verified on monkeys [6, 25] and later on humans [17]. This model was built to reproduce the bell-shaped tangential velocity profiles observed for human motions. However this was not experimented on a large variety of movements and was mainly verified for intermediate velocity profiles. Different works confirmed the fact that velocity and acceleration curves are asymmetric for a large variety of motions [3, 45, 47], especially for skilled motions. Thus a complete trajectory planner should be able to handle asymmetric kinematics. This approach was built around the hypothesis that the human behavior could be derived from a single organizing principle [26], which provides a simple model, but also, as we will see, leads to a lack of flexibility and adaptability.

Since then, minimum-jerk model has been popular and used in order to obtain coordinated and natural human-like motion. Kyriakopoulos et al. use a minimax approach to minimize the maximum jerk [33]. This work is later extended by Piazzi et al. [48, 49] where interval analysis is used to globally minimize the absolute maximum value of the jerk along a trajectory. This global minimization avoids a flaw present in minimum-jerk works, generally subjects to get stuck in local minima. Amirabdollahian et al. use fifth order polynomials for minimum-jerk control under symmetric or asymmetric jerk bounds [1].

In the different approaches listed above kinematic constraints are not considered and the time has to be set a priori. Some studies have considered minimizing a mixed criterion [20, 56]. Using this approach, Gasparetto et al. minimize both the jerk and time to obtain a trade-off between a short execution time and smoothness of the motion [20]. Kinematic constraints are taken as inputs, avoiding setting the time a priori. These works constitute a good attempt against the lack of flexibility of the model. However, in the next paragraph, we consider the constrained-jerk model that is not affected by this drawback.

1.3.2 Constrained-jerk model

In this second approach, a suitable application dependent maximum jerk J_{max} is established through experiments or information provided by the manufacturer. This threshold is determined according to certain criteria related to the nature of the task. Once the maximum jerk is defined the problem left is that of a classic time's optimization. This is done by maximizing the jerk under the constraint $J < J_{max}$. It provides time-optimal trajectories under task dependent constraints defined by the user. This approach can also be extended to acceleration, velocity and other derivatives. The main limitation of this model is that kinematic constraints must be specified by the user. However, unlike the previous approach, it provides qualities that are essential to build the future generations of collaborative robots that we expect to be efficient, adaptable and reusable [32].

Liu proposes a real-time algorithm [40] to generate smooth trajectories from current velocity under constraints on jerk, acceleration and velocity. Optimal in most cases, this paper points the difficulty of managing non-null initials and finals conditions. In the calculation steps for the maximum reachable speed, if the motion is too short to reach the maximum speed or acceleration, it becomes very difficult to compute an analytical solution online. A suboptimal strategy is then adopted by keeping the initial speed for a certain period. Haschke et al. present a similar approach [22] to generate third order time-optimal trajectories. The main contribution concerns the ability to handle arbitrary initial conditions, while end conditions must stay at rest. Nonetheless this work encounters numerical problems and produces infinite jerk for short displacements. In that case, second order trajectories, hence not smooth since the jerk is not bounded, are employed

as a fallback solution. Quadratic trajectories are also present in the work of Kroger et al. [31]. A general Online Trajectory Generation algorithm and an instance of it using third order polynomials is introduced by Kroger et al. [29]. It brings a manipulator from an arbitrary initial state to an arbitrary final state except for the final acceleration, which is always null. This work as been extended such that time-variant kinematic motion constraints are considered [30].

Third order polynomial trajectories offer a simple solution to generate jerk bounded trajectories as they are easy to manipulate, keep the jerk bounded and can be generated online. They also avoid some major drawbacks of higher degree polynomials such as the tendency to oscillate, and are better for approximation [41]. Moreover it has been demonstrated that a concatenation of at most seven cubic is enough to represent any time-optimal trajectory [7]. Herrera-Aguilar et al. propose seven cubic equations to obtain "soft motions" for robot service applications [23]. This work is later extended by Broquere et al. [7] to compute online time-optimal trajectories given any initial and final conditions, under bounded jerk, acceleration and velocity, for any number of independently acting axis. A solution for time-imposed trajectories is presented by Broquere et al. [8], which leads to a simpler axis-synchronization for multi-axis systems. Yet, this solution has drawbacks, since it is not possible to both impose the time and keep the jerk bounded. An improvement is proposed by Zhao et al. [57], which allows to have an imposed time and bounded jerk for sufficient large motions. For a point to point movement in an N-dimensional space, the time optimum straight-line motion is obtained by projecting the constraints on the line [52].

Polynomial and trigonometric models have also been combined in order to design smooth trajectories [41, 46]. Macfarlane et al. introduce an online method to compute smooth trajectories for industrial robots [41]. Concatenations of fifth order polynomials are employed to join the waypoints approximating the desired trajectories. Oscillations due to the use of quintics are here corrected by sine-wave template for accelerations. The solution presented is not optimal, and the jerk continuity is not guaranteed either, despite the use of fifth order polynomials. Trigonometric and polynomial models are also combined to design s-curve motions of any order from rest to rest [46].

In more recent works [15, 16], Ezair et al. display a new algorithm to generate smooth trajectories of any order under kinematic constraints and for multi-axis systems. A key point is its ability to deal with any general initial and final state. The algorithm builds trajectories from an input speed, which is updated iteratively by binary search until a near-optimal cruise speed is found (or peak when the motion is too short). A recursive approach is used to build higher order trajectories from lower ones. An interesting addition of this paper is the introduction of asymmetric constraints. Unfortunately this work presents some limits other than its non-optimality regarding a time criterion. Binary search in non-linear systems can lead to difficulties, such as being trapped in local minima. Moreover, it does not guarantee a solution can be found.

To our knowledge (See Table. 1), there is no work proposing a complete answer to trajectory generation satisfying simultaneously all the following criteria to build safe, efficient, adaptable and human-friendly robots:

- Real-time capable.
- General initial and end conditions for both velocity and acceleration.
- General bounded jerk, acceleration and velocity.
- Asymmetric bounds.
- Time optimal.

The algorithm we present here is the first, to our knowledge, to fulfill simultaneously all those criteria. We also attempt to understand the harsh points associated to non-linear systems that only few papers address [5, 10, 21].

2 Time-optimal cubic polynomial trajectories

One key lesson to be drawn from this bibliography is that solutions are well known for large movements, but shorter movements exhibit more complex behavior. This problem has been approached in some works [22, 40, 41], but avoided by the majority. To better understand these short moves, we first develop a graphical approach based on the phase diagram and then plot the time-optimal solution as a function of the move length x_f. The phase diagram not only provides a good description of the sequence of trajectory segments and,

Table 1: Classification of main trajectory planning algorithms in HRI context

Reference	[40]	[41]	[34]	[31]	[46]	[7]	[22]	[29]	[15]	Ours
$J_{max} \in \mathbb{R}$	+	+	+		+	+	−	+	+	+
$V_I \in \mathbb{R}$	+	+		+		+	+	+	+	+
$A_I \in \mathbb{R}$				+		+	+	+	+	+
$V_F \in \mathbb{R}$		+		+		+		+	+	+
$A_F \in \mathbb{R}$				+		+			+	+
Asymmetric bounds									+	+
Online	+	+		+		+	+	+	+	+
Optimal	−			+		+	+	+		+

"—": the criterion is not satisfied for every scenario.

in particular, of the complex behavior of the optimal trajectory, but it supports also the computation of the characteristic points of the trajectories. From these results we then introduce the complete algorithm to compute the time-optimal motion.

2.1 Problem definition

The initial condition $C_i = (x_i, v_i, a_i)$ is defined by the position x_i, the velocity v_i and the acceleration a_i. The final condition is similarly defined by $C_f = (x_f, v_f, a_f)$. Without loss of generality, we choose $x_i = 0$, i.e. we define x_i as the origin. The trajectory must comply with the asymmetric bounds (J_{min}, J_{max}, A_{min}, A_{max}, V_{min}, V_{max}) for jerk, acceleration and velocity.

From these conditions, our objective is to find the time-optimal trajectory $\mathcal{T}(t)$, which is known to be a series of at most seven trajectory segments that each saturates one of the bounds [7, 40]. Each of these trajectory segments S_n is a polynomial cubic function of time t and it is characterized by a constant jerk $J_n \in \{J_{min}, J_{max}, 0\}$, an initial instant τ_n, a duration T_n and its initial condition $C_n = (x_n, v_n, a_n)$. For all t such that $\tau_n \leqslant t \leqslant (\tau_n + T_n)$:

$$\begin{aligned} S_n(t) &= \frac{1}{6}J_n(t-\tau_n)^3 + \frac{1}{2}a_n(t-\tau_n)^2 + v_n(t-\tau_n) + x_n \\ \dot{S}_n(t) &= \frac{1}{2}J_n(t-\tau_n)^2 + a_n(t-\tau_n) + v_n \\ \ddot{S}_n(t) &= J_n(t-\tau_n) + a_n \end{aligned} \tag{1}$$

2.2 The phase diagram

These trajectories are well described with a phase diagram (Fig. 2) where abscissa is velocity and ordinate is acceleration. In this diagram, constant jerk trajectories associated to third degree polynomial functions define horizontal parabolas, constant acceleration motions associated to second degree polynomial functions are horizontal lines and constant velocities motions are associated to points of the null acceleration axis. The equations of the parabolas are obtained by eliminating the time in the two last equation of (1):

$$\dot{S}_n = \frac{\ddot{S}_n{}^2}{2J_n} + v_n - \frac{a_n{}^2}{2J_n} \tag{2}$$

The area where acceleration and speed constraints are verified is plotted in green in the figures. The acceleration bounds A_{min} and A_{max} define two horizontal boundary lines while the left and right boundaries are parabolas respectively defined by (V_{min}, J_{max}) and (V_{max}, J_{min}). By continuously varying the length x_f from a large negative value to a large positive one, the trajectory in the phase diagram reaches successively different limit shapes. These shapes are defined by a sequence of trajectory segments. For example, the large negative values are associated to the classical seven segments trajectory plotted in red in Fig. 2 and the limit case of this trajectory sequence is reached when the minimum velocity V_{min} segment lasts zero seconds. Just after this limit, five segments only define the trajectories. We associate the type JAJVJAJ to these sequence of trajectories and the type JAJAJ to the

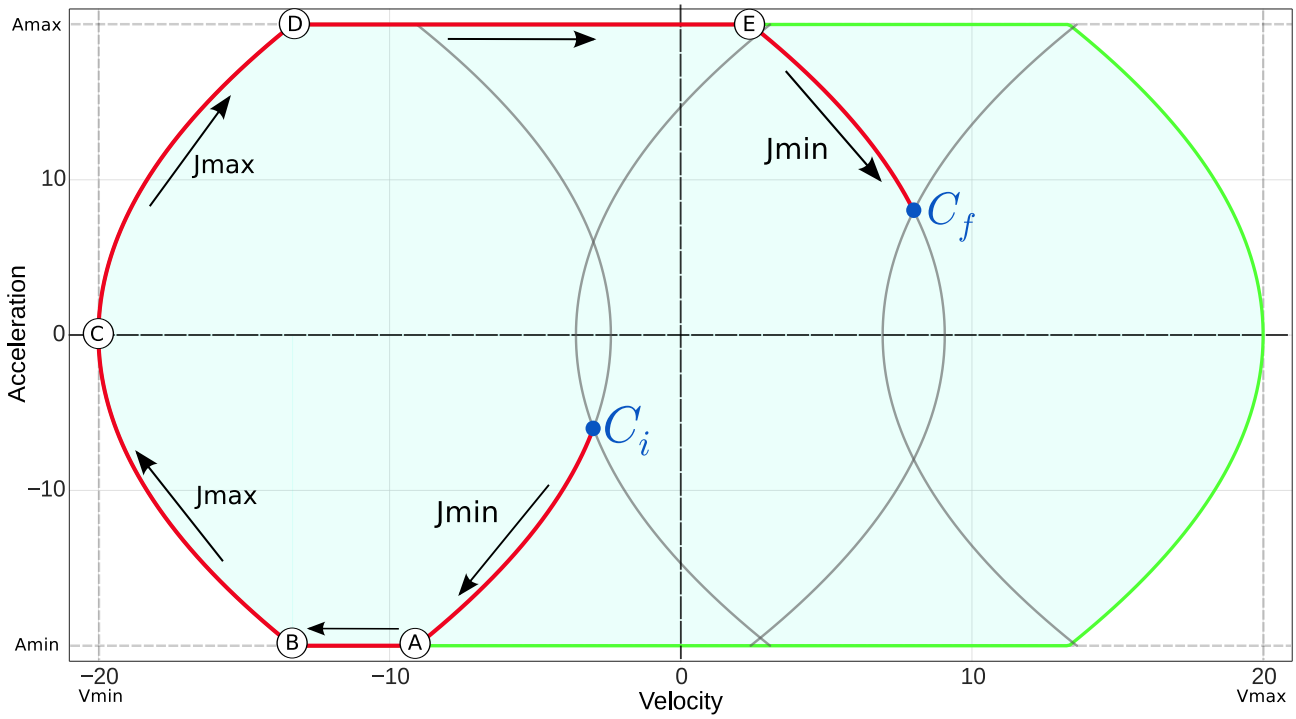


Figure 2: Phase diagram of a classical seven segments trajectory for a large negative motion. The green area of admissible conditions is limited by the acceleration bounds (A_{min}, A_{max}) and the parabolas associated to velocity bounds (V_{min}, V_{max}). The jerks J_{min} and J_{max} define four condition parabolas passing throw the initial condition C_i and the final condition C_f. The red curve joins C_f from C_i with the following 7 segments: $J_{min} \to A_{min} \to J_{max} \to V_{min} \to J_{max} \to A_{max} \to J_{min}$. The segment with saturated speed V_{min} holds on point C.

one without the V_{min}/V_{max} segment where J stands for jerk segment and A for acceleration segment.

Introducing the notion of type allows decomposing the problem in three sub-problems, the last one being trivial:

1. Find the possible types of the solution.
2. For each possible type, compute the associated trajectory.
3. Select the faster trajectory.

In the following, we firstly develop a general method to compute the border trajectories between two types.

2.3 Local parabolas

From the initial condition C_i defined by $(0, v_i, a_i)$, the two jerk bounds allows only two possible optimal motions, which define two condition parabolas in the phase diagram (See Fig. 2). Similarly, only two motions i.e. two condition parabolas are possible to reach the final condition C_f defined by v_f and a_f. As these four condition parabolas are symmetric with respect to the horizontal axis of zero acceleration, they only define two relative configurations depending on whether only two parabolas intersect (Fig. 2) or the two interior parabolas also intersect (Fig. 4). A trajectory joining C_i to C_f and composed by two jerk bounds segments is called a direct trajectory. In the phase diagram, these direct trajectories are associated to arcs of condition parabolas. When C_i and C_f are on the same parabola, the direct trajectory is reduced to one arc. If a part of these two arcs of parabolas is outside the admissible area, it is replaced by an horizontal segment associated to the acceleration A_{min} or A_{max}.

Each pair of initial and final conditions defines at

least one direct trajectory, for example the orange trajectory of the Fig. 3a. When the internal parabolas also intersect defining shortcut trajectories, up to three direct trajectories can exists (See Fig. 4).

2.4 Varying the trajectory length

Now, we propose to explore the possibility of varying the trajectory length x_f in the neighborhood of a direct trajectory of length x_d by adding an optimum motion, i.e. a J_{min} or J_{max} parabolic arc A^p. For example, the red trajectory of the Fig. 3b is the result of adding a small arc, associated to a negative jerk segment, at the beginning of the motion that translate the negative parabola toward the left. The length of the red trajectory is $x_f < x_d$. Similarly, adding a small positive motion at the end of the trajectory increases the length of the motion, see for example the blue trajectory in Fig. 3b.

The modification of the length is less intuitive in the case of the Fig. 5a where direct trajectory doesn't reach the zero acceleration line. In this case, adding a short J_{min} jerk trajectory at the beginning of the motion, translates the parabola with J_{max} jerk to the right. The extension of this added segment brings the end of the first segment to progressively reach the zero acceleration axis until a cusp appears (Fig. 5b). Continuing to extend this added segment create a loop (Fig. 5c) that can be extended to the V_{min} limit. From the cusp, the second segment translates back to the left. Generally, during this extension the length x_f is not monotonically decreasing as we will see bellow.

This complex behavior appears each time one parabola of the direct trajectory doesn't cross the zero acceleration line to reach the condition C_i or C_f. In the particular case of the Fig. 5, it appears on both sides.

2.5 Internal parabolas also intersect

In the case of the Fig. 4 where the internal parabolas also intersect, three direct trajectories are possible: (C_i, D_3, C_f), (C_i, D_2, C_f) and (C_i, D_1, C_f). The last trajectory provides an evolution similar to the case of the Fig. 3 and can be extended to both V_{min} and V_{max}. The two first trajectories introduce alternative trajectories for a range of values for the length x_f. Two samples, $(C_i, C_{ip}, C_{fp}, C_f)$ and $(C_i, C_{in}, C_{fn}, C_f)$, illustrate these shortcut trajectories in the Fig. 4.

The trajectory (C_i, D_1, C_f) is endlessly expandable, but $(C_i, C_{ip}, C_{fp}, C_f)$ and $(C_i, C_{in}, C_{fn}, C_f)$ have shorter curves in the phase diagram where they generally define faster trajectories for a small interval of values.

To summarize, in all cases the direct trajectory (Fig. 5a) can be respectively extended to long negative motions and to long positive motions. The behavior of these trajectories is more complex for short motions, i.e. around the direct trajectories, where shortcut trajectories can exist (Fig. 4). This point is developed further and constitutes a major contribution of our work.

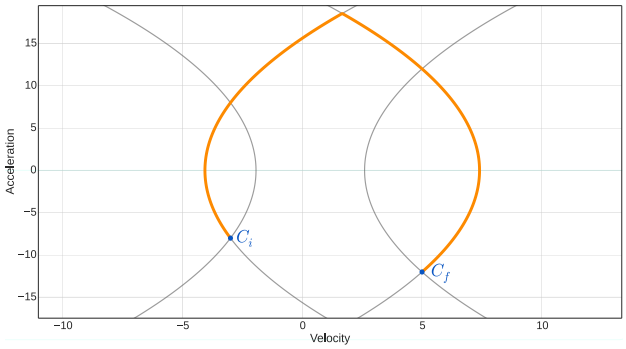
3 Duration of the trajectories according to the length

As expected, trajectories show a more complex behavior around the direct trajectory. We are going to plot the optimal time of these trajectories versus their length x_f to obtain the curve $\mathcal{C}_{opt}$. In a first stage, we propose to plot the duration of the trajectory according to its length as a parameterized curve $\mathcal{C}(x_f, t_f)$. We choose the time length of the added arc A^p (see 2.4) as parameter, thereby defining a parameter t_n for left negative side and a parameter t_p for right positive.

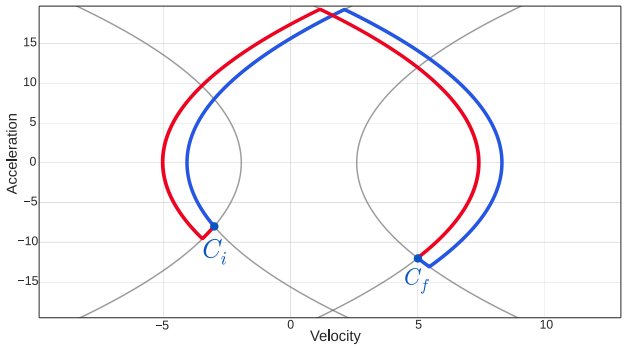
3.1 A simple case

In order to simplify the presentation, we firstly plot the curves $\mathcal{C}(t_n)$ and $\mathcal{C}(t_f)$ (see Fig. 6) for the particular case of the Fig. 5 where internal parabolas doesn't intersect and without considering the bounds. Using the conditions of the Fig. 5, we compute the times of the trajectory segments and then apply the equations (1). The times t_n and t_p are null for the direct trajectory (Fig. 3a). A time $t_n > 0$ defines a parabolic arc A^p between the initial acceleration a_0 and the acceleration $a_n = a_0 + J_{min} t_n$. Respectively, $a_p = a_0 + J_{max} t_p$ for a positive parabolic arc. For example, in the case of the Fig. 5c, the first purple negative arc starts from C_i with acceleration a_i and ends at $-a_i$. Its duration is $t_n = -2a_i / J_{min}$.

Using (1), we obtain the condition at the end of the first segment associated to the arc A^p. This condition defines the second parabolic arc, enabling the computation of the intersection of the two last parabolic arcs. From this intersection we determine the durations t_2 and t_3 of the two last segments and then the length



(a) The direct trajectory $x_f = x_d$.



(b) Trajectories with $x_f < x_d$ in red and $x_f > x_d$ in blue.

Figure 3: Evolution of the phase of the trajectory with the length x_f around the direct trajectory.

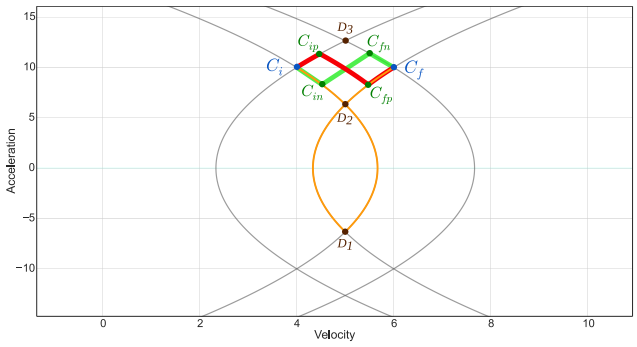


Figure 4: Limits trajectories in the case where the two parabolas defined by C_i intersect the two parabolas defined by C_f. The main direct trajectory (C_i, D_1, C_f) is in orange, the two other are (C_i, D_3, C_f) and (C_i, D_2, C_f). The red and green trajectories are two shortcuts trajectories. (J_{min}=$-30, J_{max}$=30, A_{min}=$-20, A_{max}$=20, V_{min}=$-20, V_{max}$=20, a_i=10, v_i=4, a_f=10, v_f=6).

x_f using (1). More details are given in appendix A. It is worth noting that these geometric constructions are simple, but depend on the relative position of C_i and C_f, in particular, the arc A^p of parabola can be the first or the last segment.

This procedure defines the duration $t_f(t_n) = t_n + t_2 + t_3$ and the length $x_f(t_n)$ of the trajectory, namely a point of the parametric curve $\mathcal{C}(t_n)$. The repetition of this procedure allows to plot the curve $\mathcal{C}(t_n)$ defining the duration of the trajectory versus its length (Fig. 6). A similar procedure is used to plot $\mathcal{C}(t_p)$.

3.2 Effects of the bounds

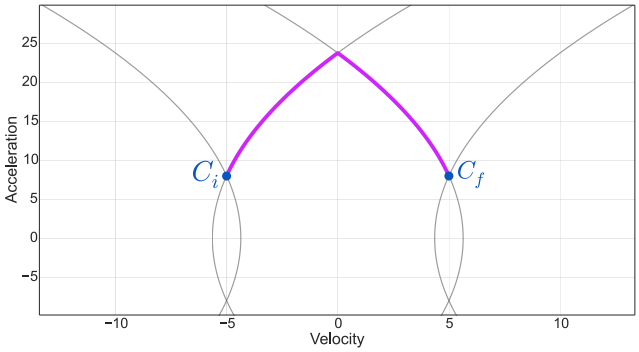
The acceleration bounds A_{min} and A_{max} limit the increase of the time parameters t_n and t_p and the arc A^p to some time t_j defining the intersecting point of the jerk parabolic trajectory with the boundary line. From this point, we follow the same process, but by defining the intermediate parabola from the end of the acceleration segment of time t_s. To keep the same parameter t_n (resp. t_p) we define t_s by $t_s + t_j = t_n$ (resp. $t_s + t_j = t_p$).

Similarly, when the intermediate parabola reaches the velocity bound V_{min} or V_{max}, we define the duration of the constant velocity segment t_v by $t_v + t_a + t_j = t_n$ (resp. t_p) where t_a is the duration of the constant acceleration segment reaching the limit velocity parabola. In some cases, the parabola defined by t_j can reach the minimal or maximal velocity parabolas before the acceleration bounds: the geometric constructions are similar, but t_a is null.

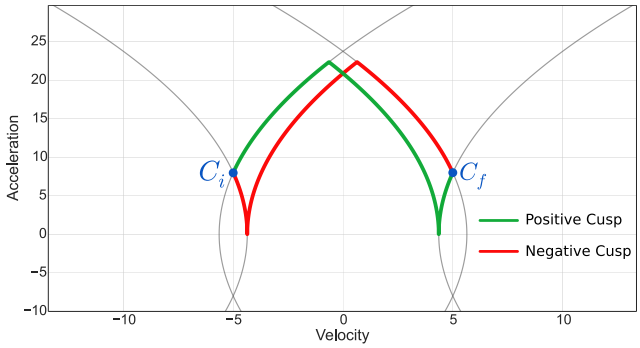
These bounds introduce changes in the nature of the portions of the curves $\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$ (See Fig. 6), in particular the large values of t_n (resp. t_p) correspond to straight line segments associated to constant (minimum or maximum) velocity motions. The second parabola can also reach the second acceleration bound, resulting in a change in the trajectory sequence without affecting the parameterization of the curve.

4 The time-optimal trajectories

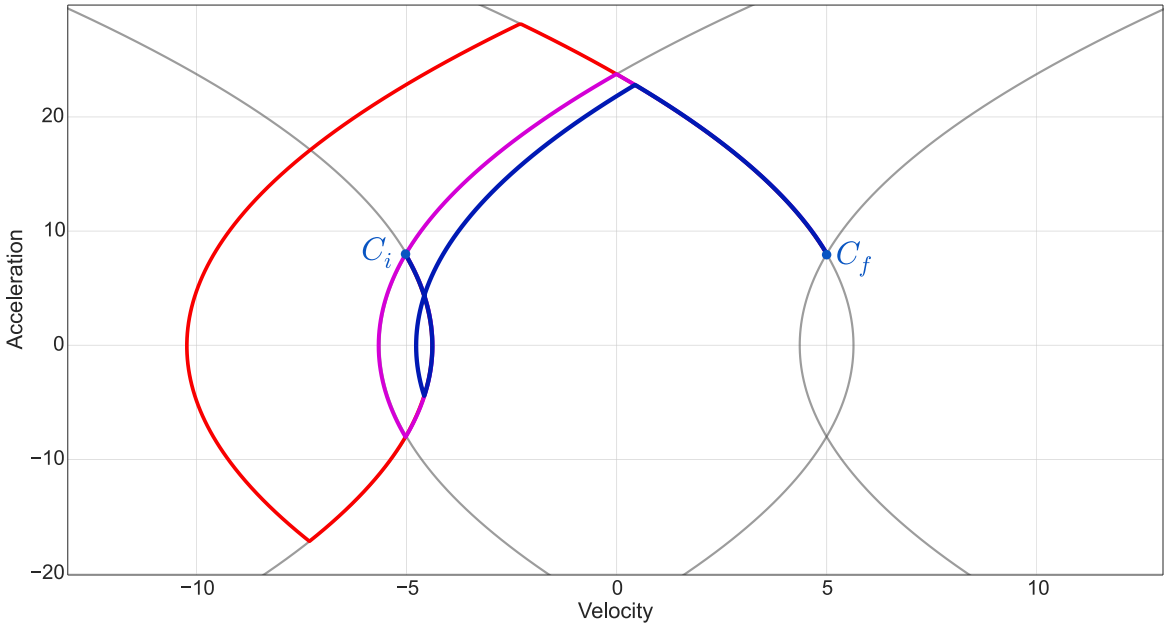
In the previous case (Fig. 6), the parametric curves $\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$ give directly the time-optimal function



(a) Direct trajectory



(b) Negative and positive Cusps



(c) Negative loop expansion

Figure 5: The direct trajectory displayed in (a) can be extended by adding a positive or a negative arc of parabola. In (b) two limit cases are displayed, which separate the trajectories with a cusp from the trajectories with a loop. Different trajectories with a loop are plotted in (c): a small loop in blue, the loop reaching the symmetric condition of C_i and a larger loop in red. ($J_{min}=-50$, $J_{max}=50$, $A_{min}=-30$, $A_{max}=30$, $V_{min}=-30$, $V_{max}=30$, $a_i=8$, $v_i=-5$, $a_f=8$, $v_f=5$).

$\mathcal{C}_{opt}(x_f)$ that associates the optimal time to the length x_f. In general, the non-linearities in the definition of these functions generate far more complex curves. We will now consider two types of non-linearities: the influence of the velocity that distorts the curve and the presence of shortcuts that split the curve in two.

4.1 Influence of the velocity

Having now defined a tool to plot the time-optimal function versus the length of the motion, we can study the influence of the different parameters. Considering the case of the Fig. 5, we shift the initial and final velocities by 15 (from (-5, 5) to (10, 20)). This shift just translate the phase diagram to the right, but the para-

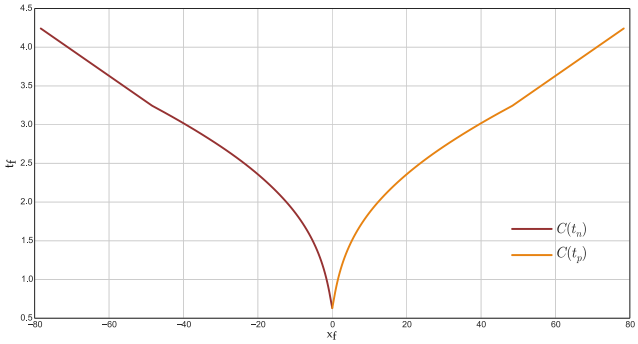


Figure 6: The optimal time solutions (t_f) versus the trajectory length (x_f). A trajectory length of zero means the start and end positions are the same. For example, the trajectory with $x_f = 0$ length is here associated to the direct trajectory of Fig. 5a, and the null length is explained by the presence of symmetries. ($J_{min}=-50$, $J_{max}=50$, $A_{min}=-30$, $A_{max}=30$, $V_{min}=-30$, $V_{max}=30$, $a_i=8$, $v_i=-5$, $a_f=8$, $v_f=5$).

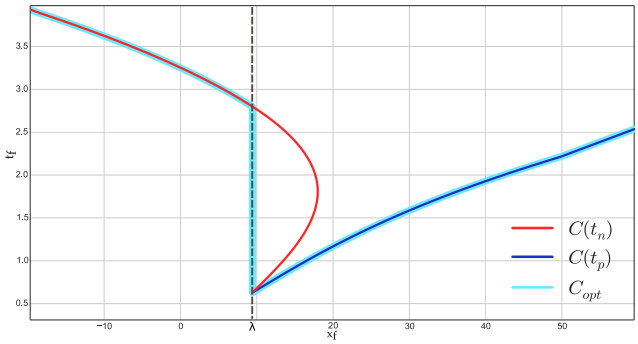


Figure 7: A shift of the initial and final velocities distort the curves $\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$. It can be noted that the curve $\mathcal{C}_{opt}(x_f)$ has a discontinuity for $x_f = \lambda$. ($J_{min}=-50$, $J_{max}=50$, $A_{min}=-30$, $A_{max}=30$, $V_{min}=-30$, $V_{max}=30$, $a_i=8$, $v_i=10$, $a_f=8$, $v_f=20$).

metric curves $\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$ plotted in the Fig. 7 have now a more complex shapes and, for a range of values of x_f, there are multiple associated trajectories with different time t_f. Therefore the time-optimal function $\mathcal{C}_{opt}(x_f)$ is no more directly defined by the union of the two curves, but by the minimum time for each value of x_f. The resulting curve may exhibit discontinuities, which can impact the computation of the optimal trajectory.

For example such discontinuity is present in the case of Fig. 7 at $x_f = \lambda$. For $x_f < \lambda$ the optimal solution begin with a jerk negative segment, but for $x_f >= \lambda$ the optimum trajectory begins with a positive jerk. The corresponding trajectories and their derivatives are plotted in the Fig. 8.

4.2 Duration in the presence of shortcut

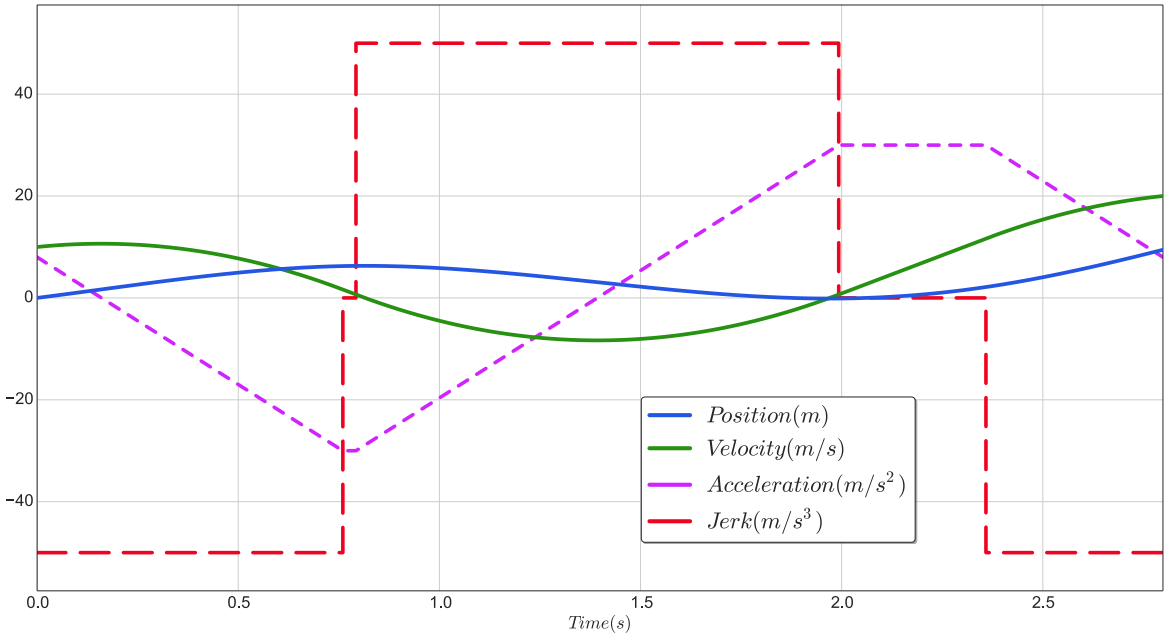
The shortcut solutions associated with intersecting parabolas (See 2.5) also introduce discontinuities in the optimal function $\mathcal{C}_{opt}(x_f)$. The curve of the Fig. 9, plotted from a similar case of the Fig. 4, shows a small lens shaped curve just below the cusp point of the parametric curves. Beside the main curve ($\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$) obtained by the previous procedure, the two parts ($\mathcal{C}(tl_n)$ and $\mathcal{C}(tl_p)$) of the lens shaped curve are similarly plotted from the two shortcut trajectories $(C_i, C_{in}, C_{fn}, C_f)$ and $(C_i, C_{ip}, C_{fp}, C_f)$, where the parameters tl_n and tl_p define respectively the duration of the first segments (C_i, C_{in}) and (C_i, C_{ip}).

In the neighborhood of the singular case where the interior parabolas become tangential, the lens shaped curve joins the main curve and disappears to extend the main curve on both sides (Fig. 10). It can be noted that for some values of x_f, the curve exhibit up to five solutions (Fig. 10).

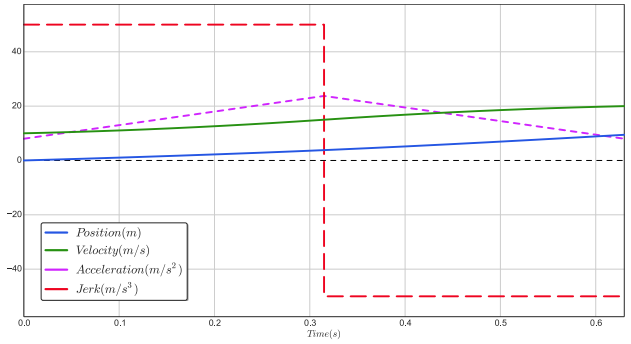
We have seen the influence of the initial and final velocity and acceleration on the optimal curve $\mathcal{C}_{opt}(x_f)$. The jerks J_{min} and J_{max} deform also the curve, but do not introduce new particular case. The values of the bounds influence the shape of the area of the admissible conditions, which change the nature of the function defining the curve $\mathcal{C}_{opt}(x_f)$, but do not introduces new types of discontinuities.

Considering only the jerk bounds, it is possible to compute analytically the parametric curve $\mathcal{C}(x_f(t_n), t_f(t_n))$ (See Appendix A). The expression obtained is large and complicated and generates an even more complicated derivative. Unfortunately, we could not manage to analytically compute the zero of the derivative of $\mathcal{C}_{opt}(x_f)$ with respect to x_f associated to its points of discontinuity.

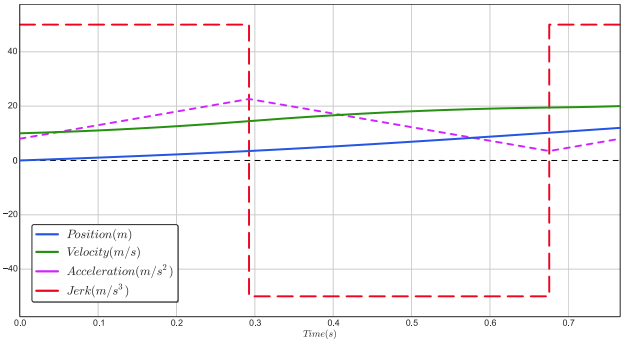
Therefore, the direct calculation of the optimum time from the zero of the derivative of $\mathcal{C}(t_n)$ and $\mathcal{C}(t_p)$ relatively to x_f appears incompatible with real-time constraints. Also, in the following we propose to compute all the solutions (up to 5) and then select the optimal one.



(a) Solution in 5 segments for $x_f = 9.44920 < \lambda$



(b) Direct trajectory for $x_f = \lambda = 9.4492105$



(c) Solution in 3 segments $x_f \gg \lambda$

Figure 8: Major effects of discontinuities on the time representation of the optimal trajectory and its derivative.

4.3 The time-optimal algorithm

From the previous elements, several strategies are possible to compute the time optimum trajectory $T_{opt}(x_f)$. However, as our main motivation is real-time control, we propose now the fast algorithm 1.

The first three stages of this algorithm were previously detailed and the last one is trivial, therefore the following sections will detail how to compute a sequence of trajectory segments from the length x_f. We begin by showing how to transform the associated systems of equations in a quartic polynomial equation and then we will detail how to solve these quartic equations.

The seven segment trajectory labelled as JAJVJAJ doesn't generally include all the seven segments. When the V segment exists, it is the only one for which the duration is varying accordingly with x_f. If the V segment is not reached, the sequence comprises at most five elements JAJAJ. This problem can always be reduced further to a sequence of three segments: Each time an acceleration segment is reached, the first or the last jerk segment duration is fixed and defined by $t_j = (A_b - a_i)/J$ where $A_b \in \{A_{min}, A_{max}\}$ and $J \in \{J_{min}, J_{max}\}$. Therefore after simplification, the four problems left to solve are: JJJ when no acceler-

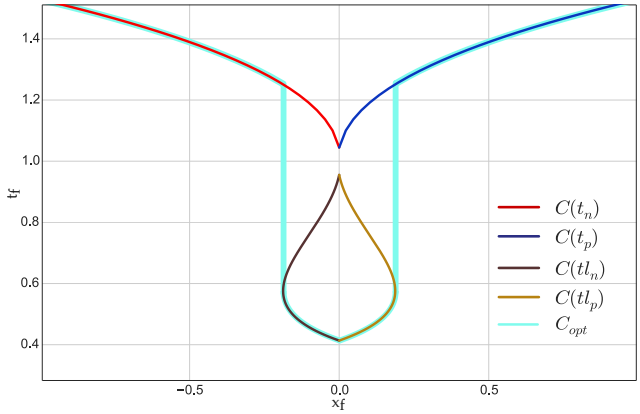


Figure 9: Shortcut trajectories introduces discontinuities in the time-optimal curve $\mathcal{C}_{opt}$. (J_{min}=−40, J_{max}=40, A_{min}=−30, A_{max}=30, V_{min}=−30, V_{max}=30, a_i=20, v_i=−4.99, a_f=20, v_f=4.99).

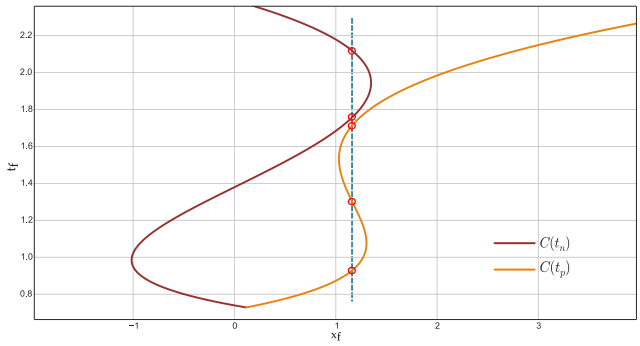


Figure 10: Curve exhibiting up to 5 solutions when initial and final velocities are shifted. (J_{min}=−40, J_{max}=50, A_{min}=−55, A_{max}=50, V_{min}=−40, V_{max}=70, a_i=−39.0, v_i=17.205, a_f=−39.0, v_f=−17.105).

ation bounds are reached, AJJ or JJA when only one acceleration bound is reached and AJA when both the acceleration bounds are reached.

The next paragraph details the JJJ case, the three others being similar and introduced in appendix B and the V case is trivial because only one segment is varying.

1 Compute the local parabolas;
2 Compute the singular limit trajectories;
3 From x_f, determine what are the possible sequences of trajectories;
4 For each possible sequence compute the time optimum trajectory;
5 Select the faster trajectory.

Algorithm 1: Time optimum trajectory

5 Solving the JJJ problem

After reducing the problem, the three jerk trajectories problem JJJ is defined by seven parameters: the initial condition $(0, v_i, a_i)$, the final condition (x_f, v_f, a_f) and two jerks J_a and J_b. The unknowns are the durations of the three segments (t_1, t_2, t_3). Two conditions (a_1, v_1, x_1) and (a_2, v_2, x_2) are associated to the transitions between the segments and defined by:

$$\begin{aligned} a_1 &= J_a * t_1 & +a_i & & \\ v_1 &= J_a * {t_1}^2/2 & +a_i * t_1 & +v_i & \\ x_1 &= J_a * {t_1}^3/6 & +a_i * {t_1}^2/2 & +v_i * t_1 & \end{aligned} \tag{3}$$

$$\begin{aligned} a_2 &= J_b * t_2 & +a_1 & & \\ v_2 &= J_b * {t_2}^2/2 & +a_1 * t_2 & +v_1 & \\ x_2 &= J_b * {t_2}^3/6 & +a_1 * {t_2}^2/2 & +v_1 * t_2 & +x_1 \end{aligned} \tag{4}$$

Then the system of polynomial equations to solve can be written as:

$$\begin{aligned} a_f &= J_a * t_3 & +a_2 & & \\ v_f &= J_a * {t_3}^2/2 & +a_2 * t_3 & +v_2 & \\ x_f &= J_a * {t_3}^3/6 & +a_2 * {t_3}^2/2 & +v_2 * t_3 & +x_2 \end{aligned} \tag{5}$$

The solution of this system of equation can be expressed from the roots of a quartic equation. Maple [42] and Maxima [43] softwares were used to perform the manipulations of the algebraic equations. Firstly we define a set of intermediate variables (k_1 to k_4) and the coefficients of the polynomial equation:

$$
\begin{aligned}
k_1 &= 2 * J_b - J_a \\
k_2 &= {J_b}^2 + J_a * (J_a - 2 * J_b) \\
k_3 &= J_a - J_b \\
k_4 &= 2 * J_a * v_f \\
c_4 &= -J_b * ({J_b}^3 + J_a * (J_a * (5 * J_b - 2 * J_a) - 4 * {J_b}^2)) \\
c_3 &= 0 \\
c_2 &= -6 * (2 * J_a * k_2 * v_i - {J_b}^2 * {a_i}^2 \\
&\quad + J_a * (k_1 * {a_i}^2 + 2 * k_2 * v_f) + {a_f}^2 * (J_a * k_1 - {J_b}^2)) \\
c_1 &= -8 * (a_i * (3 * J_a * J_b * v_i - (3 * {J_a}^2 * v_i + J_b * {a_i}^2)) \\
&\quad + J_a * {a_i}^3 - 3 * J_a * k_3 * x_f) \\
&\quad + a_f * (3 * J_a * k_3 * v_f - {a_f}^2 * k_3)) \\
c_0 &= 3 * (4 * J_a * v_i * (J_a * v_i - ({a_i}^2 + k_4 - {a_f}^2)) + {a_i}^4 \\
&\quad + 2 * ((k_4 - {a_f}^2) * {a_i}^2 + 2 * {J_a}^2 * v_f * v_f) \\
&\quad + {a_f}^2 * ({a_f}^2 - 4 * J_a * v_f))
\end{aligned}
$$

The quartic polynomial equation is then defined by: $c_4 * x^4 + c_2 * x^2 + c_1 * x + c_0 = 0$. If r_i is one of its roots, the solution can be written as:

$$
\begin{aligned}
t_1 &= -(k_6 * (k_5 - {a_i}^2 + 2 * (J_a * (r_i * a_i - v_f) \\
&\quad - Jb * r_i * a_i) + k_3 * J_b * {r_i}^2 + {a_f}^2)) * 0.5 \\
t_2 &= r_i \\
t_3 &= (k_6 * (k_5 - (a_i{}^2 + k_4) - J_b * k_3 * {r_i}^2 \\
&\quad + a_f * (2 * k_3 * r_i + a_f))) * 0.5
\end{aligned}
$$

with:

$$
\begin{aligned}
k_5 &= 2 * J_a * v_i \\
k_6 &= 1/(J_a * k_3 * r_i)
\end{aligned}
$$

As the times t_1, t_2 and t_3 must be positive, only the positive solutions define a valid trajectory. This system can have up to four solutions, but we never find a particular case with more than three admissible triplets. It must be noted that the trajectories can begin with one of the two jerk bounds, defining two different problems, which defines up to three solutions each. So, in some cases like the one of the Fig. 10, five different trajectories composed of a sequence of potentially optimum segments can be computed.

The last step is to compute the time-optimal solution, which is the one that minimize $t_1 + t_2 + t_3$.

The approach to compute the polynomial equation in the case of the sequences AJJ, JJA or AJA is really similar and presented in appendix B.

Table 2: Comparison of the mean computation times for different trajectory lengths. Results obtained for 10^8 runs.

Case	General	Near the direct trajectory	With cruising velocity phase
Times (μs)	1.049	2.535	0.845

6 Solving the quartic polynomial equation

It is well known that solving a quartic polynomial equation is difficult. The analytical solutions have been known since the 16th century, but this approach is time consuming and can fail for some particular equations. Numerical algorithms like Newton-Raphson based algorithms are efficient, but require initial information about the root, precisely the information we do not have in our case. Recent works have proposed to associate the two approaches: the analytical results are used as inputs for a numerical solver [54]. This has generated a new class of faster and more accurate algorithms [18, 55].

We used a solver derived from the one of Schwarze [51] to compute a first approximation of the solution. To improve the accuracy, we directly applied a three dimensional Newton method to the durations of the three segments. The analytical expression of the derivative of the polynomial functions (1) is given in appendix C.

7 Discussion

The characteristics of our method are summarized in the classification table 1, which compare the possibilities of the online trajectory generators. The implementation[1] of this algorithm on a system equipped with an

[1]The documentation, the softMotion library and examples including scripts for plotting figures are available at `https://git.openrobots.org/projects/softmotion/wiki`. The library is wroten in C++. A python (`https://www.python.org/`) interface that uses swig (`http://www.swig.org/`) allows to use the graphic libraries jupyter (`http://jupyter.org/`) and matplotlib

Intel Core i7 processor running at 2.2 GHz gives a mean time of 1.049 μs with a standard deviation of 0.857 μs observed for 10^8 random tests, allowing to use it in real time and for planning (See table 2). These performances are comparable or better than the previous algorithms that do not always give the optimal solution. More precisely, the comparison with the more general algorithm of Ezair et al. [15], which obtain a time of 375 μs for 3 axes that is 75 μs for one axis using their formula, must take into account their objective of a method capable to solve for series of polynomial function of any order. Similarly the method of Kroger et al. [29] that consider only null final acceleration, has an average execution time of 135 μs for six axes and 540 μs in the worst-case. As these times include the synchronization time, a precise comparison cannot be done.

The longest times we get are relative to particular case where the Newton-Raphson method have difficulty to improve the accuracy of the solution. The figure 11 gives an example of such a configuration where the analytical solution is not precise enough and the numerical one struggle to converge. In this case, one duration is considerably smaller than the other two, which disturbs the calculation algorithm. Fortunately these cases are hardly relevant and the system always returns a solution for such configurations, eventually a sub-optimal solution.

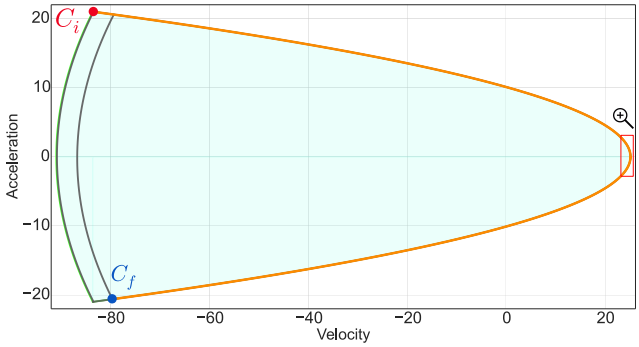
Solving the optimal trajectory problem in the vicinity of the direct trajectory opens the way to an intensive use of these simple trajectories for control and planning. Concerning the trajectory control, where the objective is to compute in real time a trajectory to bring back smoothly the mobile to the target trajectory from the current state, we can notice that the connection trajectories are short and consequently close to the direct trajectories. In this case, the proposed trajectory generator provides a good solution.

Similarly for multi-axes trajectory generation, one classical solution is to compute the time optimal trajectory for each axis, select the slowest one and synchronize the other axis with the selected one. The proposed trajectory generator can improve the calculation of the time optimal trajectories.

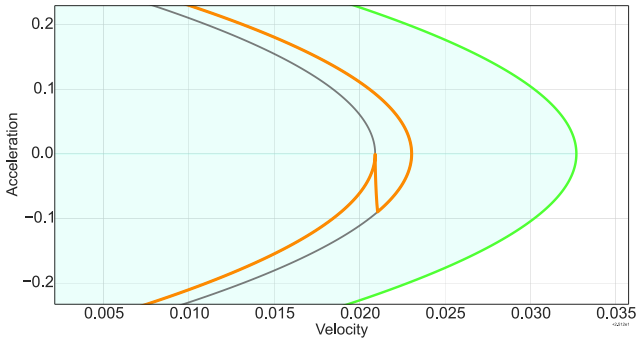
Sampling-based motion planners are really efficient to find a polygonal path, even in the case of cluttered and high-dimensional space, but planning efficient and smooth trajectories is more difficult. Here also the proposed trajectory generator could improve the smoothing of an initial trajectory built from the polygonal path.

The generator can be used in joint or operational spaces. In the first case kinematic bounds can be directly deduced from joint characteristics, whereas Cartesian space is suitable to incorporate the constraints related to safety and ergonomics. The non-symmetrical bounds can be employed to enhance the ergonomics properties of our model as they allow designing more natural human-like motions. They can also be very useful in the making of vertical motions under gravity, or motions in the presence of a human, for which an approaching move is more scaring than a withdrawal move.

(`https://matplotlib.org/`) for testing and plotting.



(a) In this case parabolas are almost identical. The green area of admissible conditions is only limited by the two parabolas (V_{min}, J_{max}) and (V_{max}, J_{min}).



(b) The zoomed area where the positive cusp produces a very short positive jerk segment.

Figure 11: The phase diagram illustrating a general and difficult case where the Newton algorithm has difficulty to converge. (V_{min}= − 90.9696, V_{max}=25.1527, J_{min}= − 2.02754, J_{max}=29.7968, a_i=20.9815, v_i= − 83.4179, a_f= − 20.6076, v_f= − 79.5853).

As this work explains the discontinuities of the time-optimal curve $\mathcal{C}_{opt}$ and solves the optimum time problem, it will contribute to the development of trajectory based robotic architectures. In these architectures, trajectories will be used as the main support of communication and facilitate the link between planning and control, leading towards an improvement of robots motions.

By explaining the complex behavior of the jerk bounded trajectories, this work defines also a step in the solving of the snap bounded optimal trajectory problems, where the snap is the derivative of the jerk.

8 Conclusion

With the emergence of HRI, the problem of the generation of safe, efficient and human-friendly movements has to be addressed. From the review of the state of the art it appears that no complete solution for the making of collaborative motions exists yet. To the best of our knowledge, the algorithm presented herein that uses trajectory of class $\mathcal{C}^2$ defined by a chain of cubic polynomial functions is the first one that:

- joins two arbitrary conditions defined by position, velocity and acceleration,
- in minimum time under general and asymmetric bounds on velocity, acceleration and jerk.

By explaining graphically the behavior of the optimal trajectories, this work allows to explain and solve the difficulties highlighted by the previous works. The proposed trajectory generator completes the existing tools for planning and controlling multi-axes cubic polynomial trajectories, which open the way for more flexible and friendly robots. This applies particularly to the HRI domain where the underlying constrained jerk model approach makes easier the consideration of the different types of constraints related to safety and ergonomics. More specifically, close cooperation between humans and industrial robots needs more flexibility, adaptability and reusability, which can be improved by the models and tools developed in this work.

Even if the model of the chain of cubic polynomial trajectories is efficient, the questions relative to higher degrees polynomial still remain. These models are necessary to solve specific problems. The underactuated vehicles, for example, need one more derivative to control the motion obtained by integration. It is the case, for example, to obtain a jerk bounded horizontal move with a quadrotor. The same problem appears also for double and, more generally, multiple pendulum. Given the difficulties encountered to solve the cubic trajectories, the higher degree appears as really challenging.

Appendices

A Analytic expression of the parametric curve $\mathcal{C}$

We consider here the particular case where the trajectory $\mathcal{C}(x_f(t_n), t_f(t_n))$ is defined by:

- Three non null jerk segments associated to parabolas in the phase diagram.
- The first and the last segments are associated to $J_{min} < 0$..
- The third segment begin with a positive acceleration a_2.

The first trajectory segment is entirely defined by the initial conditions a_i and v_i and the duration t_1 using the equations (3). The parabola associated to the second segment cross the abscissa axis at v'_i and the one associated to the third segment at v'_f with:

$$v'_i = v_i - \frac{{a_i}^2}{2J_{max}} \qquad v'_f = v_f - \frac{{a_f}^2}{2J_{min}}$$

And the acceleration at the intersection point of the two parabolas is a_2 defined by:

$$a_2 = \sqrt{\frac{2(v'_i - v'_f) \times J_{max} \times J_{min}}{J_{max} - J_{min}}}$$

We choose the positive solution as defined in the hypothesis and compute the durations t_2 and t_3 of the two last segments.

$$t_2 = \frac{(a_2 - a_1)}{J_{max}} \qquad t_3 = \frac{(a_f - a_2)}{J_{min}}$$

From the equations (4) and (5) and using an algebraic calculator, the values of x_f and $t_f = t_1 + t_2 + t_3$ can be easily computed. Then the zeros of the derivative of the x_f function with respect to t_1 would define the points of discontinuity of the optimal curve C_{opt}. Unfortunately, we cannot manage to obtain these points in a usable form.

B Solving the three segments problems

Solving the AJA problem

In this case the jerk J_a of the first and last segments is null in the equations (3) and (5) that respectively become (6) and (7):

$$\begin{aligned} a_1 &= a_i \\ v_1 &= a_i \times t_1 + v_i \\ x_1 &= a_i \times {t_1}^2/2 + v_i \times t_1 \end{aligned} \quad (6)$$

$$\begin{aligned} a_f &= a_2 \\ v_f &= a_2 \times t_3 + v_2 \\ x_f &= a_2 \times {t_3}^2/2 + v_2 \times t_3 + x_2 \end{aligned} \quad (7)$$

Using an algebra system, we obtain the results:

$$\begin{aligned} k_1 &= a_i - a_f \\ k_2 &= {k_1}^{-1} \\ k_3 &= a_i \times ((-J_a \times v_i) + {a_f}^2/2 + a_i \times (-a_f + a_i/2)) \\ k_4 &= a_f \times J_a \times v_i \\ k_5 &= 12 \times {J_a}^2 \times {v_i}^2 \\ k_6 &= a_f - a_i \\ k_7 &= (3 \times (12 \times a_i \times {J_a}^2 \times (2 \times a_f \times k_6 \times x_f + k_1 \times {v_f}^2) \\ &\quad + a_f \times (a_i \times (a_i \times (4 \times {a_f}^3 + a_i \times (a_i \times (4 \times a_f - a_i) \\ &\quad - 6 \times {a_f}^2)) - (k_5 + a_f^4)) + 12 \times a_f \times {J_a}^2 \times {v_i}^2)))^{0.5} \\ k_8 &= k_6 \times {J_a}^{-1} \\ k_9 &= k_1 \times a_f \\ k_{10} &= 2 \times J_a \times v_f \\ k_{11} &= 1/\sqrt{3} \\ k_{12} &= (k_6 \times (12 \times a_i \times {J_a}^2 \times (2 \times a_f \times x_f - {v_f}^2) \\ &\quad + a_f \times (k_5 + a_i \times (a_i \times (3 \times {a_f}^2 + a_i \times (a_i - 3 \times a_f)) \\ &\quad - {a_f}^3))))^{0.5} \end{aligned}$$

And finally the system has two solutions (t_1, t_2, t_3) and (t'_1, t_2, t'_3) with:

$$\begin{aligned} t_1 &= {a_i}^{-1} \times k_2 \times {J_a}^{-1} \times ((-k_7/6) + k_4 + k_3) \\ t_2 &= k_8 \\ t_3 &= ({a_f}^{-1} \times {J_a}^{-1} \times (k_{11} \times k_2 \times k_{12} + k_{10} + k_9))/2 \\ t'_1 &= {a_i}^{-1} \times k_2 \times {J_a}^{-1} \times (k_7/6 + k_4 + k_3) \\ t'_3 &= {a_f}^{-1} \times {J_a}^{-1} \times ((k_{10} + k_9)/2 - (k_{11} \times k_2 \times k_{12})/2) \end{aligned}$$

Solving the JJA problem

The system of polynomial equations to solve is defined by (6), (4) and (5). Using an algebra system, we obtain the following result where r_i is one solution of the quartic equation defined by the coefficients c_i with $0 \leq i \leq 4$:

$$\begin{aligned} k_1 &= 1 - 2 \times a_f \\ k_2 &= 2 \times a_f \\ k_3 &= {a_f}^2 \times J_b + k_1 \times J_a \\ c_4 &= 3 \times {a_f}^4 \times {J_b}^4 \\ &\quad + J_a \times (2 \times k_1 \times {a_f}^2 \times J_b + (4 \times (a_f - 1) \times a_f + 1) \times J_a)) \\ c_3 &= 4 \times a_f \times J_b \times ({a_f}^3 \times {J_b}^2 \\ &\quad + J_a \times ((3 \times a_f - 1) \times J_a - 3 \times {a_f}^2 \times J_b)) \\ c_2 &= 6 \times J_b \times (2 \times J_a \times k_3 \times v_i + ((k_2 - 1) \times J_a - {a_f}^2 \times J_b) \times {a_i}^2) \\ c_1 &= 12 \times a_f \times (2 \times J_a \times (a_f \times J_b - J_a) \times v_i + (J_a - a_f \times J_b) \times {a_i}^2) \\ c_0 &= 3 \times (4 \times {J_a}^2 \times {v_i}^2 + a_i \times (4 \times J_a \times (k_2 - a_i) \times v_i + {a_i}^3)) \\ &\quad + 4 \times (3 \times {J_a}^2 \times (2 \times a_f \times x_f - {v_f}^2) - 2 \times a_f \times {a_i}^3) \\ t1 &= -{J_a}^{-1} \times (a_i + a_f \times J_b \times r_i) \\ t2 &= r_i \\ t3 &= -\frac{Ja^{-1} \times (2 \times Ja \times v_i - ({a_i}^2 + 2 \times Ja \times v_f) + J_b \times k_3 \times {r_i}^2)}{2 \times a_f} \end{aligned}$$

Solving the AJJ problem

An AJJ system can be solved similarly as a JJA one or using a symmetry with respect to the acceleration axis to build an equivalent JJA system.

C Derivative of the cubic polynomial functions

The derivatives of the functions x_f, v_f and a_f relatively to the three times t_1, t_2 and t_3 can be grouped in a matrix. The newton method uses the inverse of this matrix to compute the times increments that, at the first oder, compensate the errors of the functions.

$$\begin{pmatrix} \frac{\partial x_f}{\partial t_1} & \frac{\partial x_f}{\partial t_2} & \frac{\partial x_f}{\partial t_3} \\ \frac{\partial v_f}{\partial t_1} & \frac{\partial v_f}{\partial t_2} & \frac{\partial v_f}{\partial t_3} \\ \frac{\partial a_f}{\partial t_1} & \frac{\partial a_f}{\partial t_2} & \frac{\partial a_f}{\partial t_3} \end{pmatrix} = \begin{pmatrix} lx_1 & lv_1 & la_1 \\ lx_2 & lv_2 & la_2 \\ lx_3 & lv_3 & la_3 \end{pmatrix}^{-1}$$

In the JJJ case defined by the equations (3), (4) (5), using an algebraic calculator we obtain:

$$
\begin{aligned}
k_1 &= J_a \times t_1 \\
k_2 &= J_b \times t_2 \\
k_3 &= J_a \times t_3 \\
k_4 &= a_i + k_3 + k_2 + k_1 \\
k_5 &= a_i + J_b \times (t_3 + t_2) + k_1 \\
k_6 &= J_a \times k_5 - J_b \times k_4 \\
k_7 &= {t_1}^2 \\
k_8 &= {t_2}^2 \\
k_9 &= {t_3}^2 \\
k_{10} &= (J_a \times (k_9 + k_8 + k_7))/2 \\
k_{11} &= t_1 \times a_i \\
k_{12} &= t_2 \times (a_i + k_1) \\
k_{13} &= t_3 \times (a_i + J_a \times (t_2 + t_1)) \\
k_{14} &= v_i + k_{13} + k_{12} + k_{11} + k_{10} \\
k_{15} &= a_i + J_a \times (t_3 + t_2 + t_1) \\
k_{16} &= J_b \times k_{15} - J_a \times k_5 \\
k_{17} &= J_a \times k_7 \\
k_{18} &= J_a \times k_9 + J_b \times k_8 + k_{17} \\
k_{19} &= t_3 \times (a_i + k_2 + k_1) \\
k_{20} &= v_i + k_{19} + k_{12} + k_{11} + k_{18}/2 \\
k_{21} &= (-k_{15}) + a_i + k_3 + k_2 + k_1 \\
k_{22} &= v_i + k_{19} + k_{12} + k_{11} + (J_b \times (k_9 + k_8) + k_{17})/2 \\
k_{23} &= 1/(J_a \times k_{21} \times k_{22} + k_{16} \times k_{20} + k_6 \times k_{14}) \\
lx_1 &= k_6 \times k_{23} \\
lv_1 &= (J_b \times k_{20} - J_a \times k_{22}) \times k_{23} \\
la_1 &= (k_4 \times k_{22} - k_5 \times k_{20}) \times k_{23}] \\
lx_2 &= J_a \times k_{21} \times k_{23} \\
lv_2 &= J_a \times (-(k_{19} + k_{12} + k_{11} - (-k_{18}/2)) \\
&\quad + k_{13} + k_{12} + k_{11} + k_{10}) \times k_{23} \\
la_2 &= (k_{15} \times k_{20} - k_4 \times k_{14}) \times k_{23}] \\
lx_3 &= k_{16} \times k_{23} \\
lv_3 &= (J_a \times k_{22} - J_b \times k_{14}) \times k_{23} \\
la_3 &= (k_5 \times k_{14} - k_{15} \times k_{22}) \times k_{23}]))
\end{aligned}
$$

The computation is done similarly in the case of AJA, AJJ and JJA sequences.

References

[1] F. Amirabdollahian, R. Loureiro, and W. Harwin. "Minimum jerk trajectory control for rehabilitation and haptic applications". In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292). Vol. 4. 2002, 3380–3385 vol.4.

[2] T. Arai, R. Kato, and M. Fujita. "Assessment of operator stress induced by robot collaboration in assembly". In: *CIRP Annals - Manufacturing Technology* 59.1 (2010), pp. 5–8.

[3] W. D. A. Beggs and C. I. Howarth. "The movement of the hand towards a target". In: *The Quarterly Journal of Experimental Psychology* 24.4 (Nov. 1972), pp. 448–453.

[4] Luigi Biagiotti. *Trajectory Planning for Automatic Machines and Robots*. Springer, 2008.

[5] C. Guarino Lo Bianco and O. Gerelli. "Online Trajectory Scaling for Manipulators Subject to High-Order Kinematic and Dynamic Constraints". In: *IEEE Transactions on Robotics* 27.6 (Dec. 2011), pp. 1144–1152.

[6] E. Bizzi et al. "Posture control and trajectory formation during arm movement". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 4.11 (Nov. 1984), pp. 2738–2744.

[7] Xavier Broquere, Daniel Sidobre, and Ignacio Herrera-Aguilar. "Soft motion trajectory planner for service manipulator robot". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2808–2813.

[8] Xavier Broquere, Daniel Sidobre, and Khoi Nguyen. "From motion planning to trajectory control with bounded jerk for service manipulator robots". In: *Robotics and Automation (ICRA)*. IEEE, 2010, pp. 4505–4510.

[9] John Travis Butler and Arvin Agah. "Psychological effects of behavior patterns of a mobile personal robot". In: *Autonomous Robots* 10.2 (2001), pp. 185–202.

[10] D. Costantinescu and E. A. Croft. "Smooth and time-optimal trajectory planning for industrial manipulators along specified paths". In: *Journal of robotic systems* 17.5 (2000), pp. 233–249.

[11] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Pearson/Prentice Hall, 1986. 416 pp.

[12] Etienne Dombre and Wisama Khalil. *Robot manipulators: modeling, performance analysis and control*. John Wiley & Sons, 2013.

[13] Anca Dragan and Siddhartha Srinivasa. "Generating legible motion". In: Robotics: Science and Systems. Pittsburgh, PA, 2013.

[14] Anca D. Dragan, Kenton CT Lee, and Siddhartha S. Srinivasa. "Legibility and predictability of robot motion". In: *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*. IEEE, 2013, pp. 301–308.

[15] B. Ezair, T. Tassa, and Z. Shiller. "Planning high order trajectories with general initial and final conditions and asymmetric bounds". In: *The International Journal of Robotics Research* 33.6 (May 2014), pp. 898–916.

[16] Ben Ezair, Tamir Tassa, and Zvi Shiller. "Multi-axis High-order Trajectory Planning". In: *Workshop on Robot Motion Planning: Online, Reactive, and Real-time*. 2012.

[17] T. Flash and N. Hogan. "The coordination of arm movements: an experimentally confirmed mathematical model". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 5.7 (July 1985), pp. 1688–1703.

[18] N. Flocke. "Algorithm 954: An Accurate and Efficient Cubic and Quartic Equation Solver for Physical Applications". In: *ACM Transactions on Mathematical Software* 41.4 (Oct. 12, 2015), pp. 1–24.

[19] Marina Fujita, Ryu Kato, and Arai Tamio. "Assessment of operators' mental strain induced by hand-over motion of industrial robot manipulator". In: RO-MAN. IEEE, 2010, pp. 361–366.

[20] Alessandro Gasparetto and Vanni Zanotto. "A technique for time-jerk optimal planning of robot trajectories". In: *Robotics and Computer-Integrated Manufacturing* 24.3 (June 2008), pp. 415–426.

[21] O. Gerelli and C. G. Lo Bianco. "Nonlinear Variable Structure Filter for the Online Trajectory Scaling". In: *IEEE Transactions on Industrial Electronics* 56.10 (Oct. 2009), pp. 3921–3930.

[22] Robert Haschke, Erik Weitnauer, and Helge Ritter. "On-line planning of time-optimal, jerk-limited trajectories". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3248–3253.

[23] Ignacio Herrera-Aguilar and Daniel Sidobre. "Soft motion trajectory planning and control for service manipulator robot". In: *Workshop on Physical Human-Robot Interaction in Anthropic Domains at IROS*. 2006, pp. 13–22.

[24] Guy Hoffman and Cynthia Breazeal. "Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team". In: *Proceedings of the ACM/IEEE international conference on Human-robot interaction*. ACM, 2007, pp. 1–8.

[25] N. Hogan. "Control and Coordination of Voluntary Arm Movements". In: 1982 American Control Conference. June 1982, pp. 522–528.

[26] N. Hogan. "An organizing principle for a class of voluntary movements". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 4.11 (Nov. 1984), pp. 2745–2754.

[27] Lydia E. Kavraki and Steven M. LaValle. "Motion Planning". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano Prof and Oussama Khatib Prof. Springer Berlin Heidelberg, 2008, pp. 109–131.

[28] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*. Butterworth-Heinemann, July 1, 2004. 503 pp.

[29] T. Kroger and F.M. Wahl. "Online Trajectory Generation: Basic Concepts for Instantaneous Reactions to Unforeseen Events". In: *IEEE Transactions on Robotics* 26.1 (Feb. 2010), pp. 94–111.

[30] Torsten Kroger. "On-line trajectory generation: Nonconstant motion constraints". In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2048–2054.

[31] Torsten Kroger, Adam Tomiczek, and Friedrich M. Wahl. "Towards on-line trajectory computation". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 736–741.

[32] J. Krüger, T.K. Lien, and A. Verl. "Cooperation of human and machines in assembly lines". In: *CIRP Annals* 58.2 (2009), pp. 628–646.

[33] K. J. Kyriakopoulos and G. N. Saridis. "Minimum jerk path generation". In: 1988 IEEE International Conference on Robotics and Automation Proceedings. Apr. 1988, 364–369 vol.1.

[34] Paul Lambrechts, Matthijs Boerlage, and Maarten Steinbuch. "Trajectory planning and feedforward design for electromechanical motion systems". In: *Control Engineering Practice* 13.2 (Feb. 2005), pp. 145–157.

[35] P. A. Lasota, G. F. Rossano, and J. A. Shah. "Toward safe close-proximity human-robot interaction with standard industrial robots". In: 2014 IEEE International Conference on Automation Science and Engineering (CASE). Aug. 2014, pp. 339–344.

[36] Przemyslaw A. Lasota and Julie A. Shah. "Analyzing the Effects of Human-Aware Motion Planning on Close-Proximity Human–Robot Collaboration". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 57.1 (Feb. 2015), pp. 21–33.

[37] Przemyslaw A. Lasota, Terrence Fong, and Julie A. Shah. "A Survey of Methods for Safe Human-Robot Interaction". In: *Foundations and Trends in Robotics* 5.3 (2017), pp. 261–349.

[38] Jean-Claude Latombe. *Robot Motion Planning / Jean-Claude Latombe / Springer*. 1990.

[39] Steven M. LaValle. *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[40] Steven Liu. "An on-line reference-trajectory generator for smooth motion of impulse-controlled industrial manipulators". In: *Advanced Motion Control, 2002. 7th International Workshop on*. IEEE, 2002, pp. 365–370.

[41] S. Macfarlane and E.A. Croft. "Jerk-bounded manipulator trajectory planning: design for real-time applications". In: *IEEE Transactions on Robotics and Automation* 19.1 (Feb. 2003), pp. 42–52.

[42] Maple. *Maplesoft, a division of Waterloo Maple Inc., Waterloo, Ontario*. 2016. URL: `http://www.maplesoft.com/`.

[43] Maxima. *A Computer Algebra System. Version 5.39.0*. 2017. URL: `http://maxima.sourceforge.net/`.

[44] Eric Meisner, Volkan Isler, and Jeff Trinkle. "Controller design for human-robot interaction". In: *Autonomous Robots* 24.2 (Feb. 2008), pp. 123–134.

[45] H. Nagasaki. "Asymmetric velocity and acceleration profiles of human arm movements". In: *Experimental brain research* 74.2 (1989), pp. 319–326.

[46] Kim Doang Nguyen, Teck-Chew Ng, and I.-Ming Chen. "On algorithms for planning S-curve motion profiles". In: *International Journal of Advanced Robotic Systems* 5.1 (2008), pp. 99–106.

[47] David J. Ostry, James D. Cooke, and Kevin G. Munhall. "Velocity curves of human arm and speech movements". In: *Experimental Brain Research* 68.1 (1987), pp. 37–46.

[48] A. Piazzi and A. Visioli. "An interval algorithm for minimum-jerk trajectory planning of robot manipulators". In: Proceedings of the 36th IEEE Conference on Decision and Control. Vol. 2. Dec. 1997, 1924–1927 vol.2.

[49] Aurelio Piazzi and Antonio Visioli. "Global minimum-jerk trajectory planning of robot manipulators". In: *IEEE transactions on industrial electronics* 47.1 (2000), pp. 140–149.

[50] J. R. Rivera-Guillen et al. "Extending tool-life through jerk-limited motion dynamics in machining processes: An experimental study". In: *JSIR Vol.69(12)* (Dec. 2010).

[51] Jochen Schwarze. "Graphics Gems". In: ed. by Andrew S. Glassner. San Diego, CA, USA: Academic Press Professional, Inc., 1990. Chap. Cubic and Quartic Roots, pp. 404–407.

[52] Daniel Sidobre and Wuwei He. "Online task space trajectory generation". In: *Workshop on Robot Motion Planning Online, Reactive, and in Real-time*. 2012.

[53] Emrah Akin Sisbot et al. "A Human Aware Mobile Robot Motion Planner". In: *IEEE Transactions on Robotics* 23.5 (Oct. 2007), pp. 874–883.

[54] Peter Strobach. "The fast quartic solver". In: *Journal of Computational and Applied Mathematics* 234.10 (Sept. 2010), pp. 3007–3024.

[55] PETER Strobach. “The Low-Rank LDLT Quartic Solver”. In: *AST-Consulting Technical Report* 10.2 (2015), pp. 3955–7440.

[56] Vanni Zanotto et al. “Experimental Validation of Minimum Time-jerk Algorithms for Industrial Robots”. In: *Journal of Intelligent & Robotic Systems* 64.2 (Nov. 2011). 00033, pp. 197–219.

[57] Ran Zhao, Daniel Sidobre, and Wuwei He. “Online via-points trajectory generation for reactive manipulations”. In: *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, 2014, pp. 1243–1248.

Author biography

Kevin Desormeaux was born on June 27th, 1990 in the south region of France. He received a Master in Sciences and Technologies, Specialty Computer Sciences, Track in Artificial Intelligence and Pattern Recognition from University of Toulouse in 2014. On his last year he spent six months at CNES to work on path planning problematics as part of Exomars, the ESA mission. In 2016 he began a Ph.D. in robotics at LAAS-CNRS and is still working on it. His interests are mainly focused in A.I. and robotics.

Daniel Sidobre has a background in mechanics from Pierre et Marie Curie University and in control from University of Toulouse. He obtained a Ph.D. in robotics in 1990 and the HDR degree from University of Toulouse in 2009. He spent one sabbatical year at Mc Gill University, Canada to work on contact friction and three months at Università degli Studi di Napoli Federico II, Italie, to work on manipulation. He is associate professor at University of Toulouse where he teaches mechanical and production engineering. He participated to numerous European and national ANR research projects. He worked mainly on manipulation, grasp planning, trajectory generation and control, and human-robot interaction.