Online Trajectory Generation: Reactive Control With Return Inside an Admissible Kinematic Domain

Kevin Desormeaux, Daniel Sidobre

Online Trajectory Generation: Reactive Control With Return Inside an Admissible Kinematic Domain

Kevin Desormeaux[1] and Daniel Sidobre[1]

***Abstract*— As humans and robots work more and more closely, robots must quickly react to unforeseen human behavior. Online Trajectory Generation (OTG), based on simple trajectory models like series of polynomial cubic functions, has demonstrated its efficiency to plan and control reactive motions of robots. However to ensure the safety and comfort of humans, fast trajectory adaptation algorithms are necessary to bring back the robot inside an acceptable domain that is defined by a set of kinematic constraints.**

The algorithm presented herein extends a time-optimal OTG to cope with non admissible robot's state. This feature enables time-variant kinematic constraints. With the possibility to specify velocity and acceleration at both ends under short computation times, it makes the robot able to react quickly to unforeseen events. Short computation times can lead to more refined architectures where sensors can be integrated to a low control level and make the system more reactive.

I. INTRODUCTION

Robotics has already transformed our industry, and its acceptance by our society is under way. With the constant progress made in this field, leading to better generations of robots, the next major issue will be physical Human-Robot Interaction (pHRI). Robots will have to coexist and cooperate with humans to accomplish a variety of tasks. In [10], Krüger et al. present a survey studying the forms of cooperation between a human and a robot that can be used in assembly processes as well as organizational and economic aspects. They emphasize the advantages of these new kinds of cooperation, even compared to fully automated systems.

However, collaborative robots have behaviours that differ from those expected of robots working in cages. It is essential to find efficient models of motions that ensure both safety and comfort of the human co-worker [1, 5, 11–13].

Smooth trajectories are well known to provide safety and good ergonomics. They are mostly used as a solution to mimic human gesture [7]. The smoothness of a trajectory can be defined by the number of derivatives 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. Such trajectories can be defined by series of cubic polynomial functions that comply with kinematic constraints [16].

To ensure the safety and comfort of a human moving near the robot, the robot must continuously adapt its speed and other kinematic limits according to the human's presence and behavior. We propose to use the real-time ability of OTG to continuously do this adaptation during the trajectory generation. The robot can switch between different behaviours depending on the human presence and optimize its efficiency.

In this context where the motions are defined by a large set of constraints, the use of standard reactive control is generally not very efficient and it is preferable to continuously plan a trajectory in a receding-horizon (model predictive) control fashion. Such an approach needs for a trajectory generator that comply with the following requirements:

- General initial and end conditions for both velocity and acceleration.
- General time varying bounds on jerk, acceleration and velocity.
- Time-optimal with respect to the kinematic bounds defined.
- Real-time.

These trajectories and constraint models also define precise means of communication with higher software levels that plan and monitor the robot's actions. This planning and the associated monitoring can take into account the task as a whole, and define the motion and the associated constraints to generate the trajectories [17].

In this paper we extend the previous OTG algorithm[16] to take into account time-varying kinematic constraints. The main contribution of the paper is a solution to bring the system back into the new kinematic domain as quickly as possible after a reducing of the kinematic constraints of the system. From a detailed study of the acceptable domain, we propose a new and more precise approach to generate the return trajectories.

The paper is organized as follows. In Section II we present the state of the art. The main elements of our previous OTG works are presented in Section III. An extension of this algorithm dealing with non admissible initial condition is detailed in Section IV. Discussion and results are presented in Section V and Section VI concludes this paper.

II. RELATED WORKS

In [14], Liu proposes a real-time algorithm to generate smooth trajectories from current velocity under symmetric jerk, acceleration and velocity bounds. Optimal in most cases, this paper points the difficulty of managing non-null initials and finals conditions. A similar approach is proposed by Haschke et al. to generate third order time-optimal trajectories[6]. The main contribution concerns the ability to handle arbitrary initial conditions, while end conditions must stay at rest. The jerk is not guaranteed to be bounded, and the kinematic bounds can only be symmetrical. Moreover this

[1]Daniel Sidobre and Kevin Desormeaux are with CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France; and Univ de Toulouse, UPS, F-31400 Toulouse, France; `kevin.desormeaux@laas.fr`, `daniel.sidobre@laas.fr`

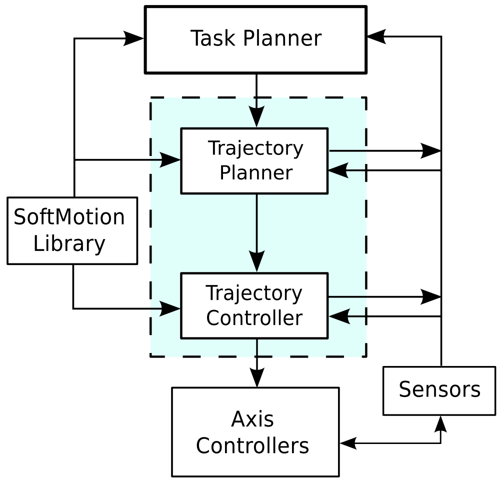


Fig. 1: A possible architecture of a robot that uses trajectories for the control and the communication between the different modules.

work encounters numerical problems and produces infinite jerk for short displacements. A well known OTG algorithm is the one of Kröger et al. presented in [9]. End-point acceleration must remain null and only symmetric bounds are considered. In [3, 4], general kinematics bounds can be specified. The presented algorithm generates smooth trajectories of any order under kinematic constraints with general initial and final conditions. However the algorithm cannot always return a solution, and if one is found it is not guaranteed to be time-optimal. In [16], Sidobre et al. introduce a complete algorithm based on sequences of third order polynomial functions. The algorithm generates in real-time time-optimal trajectories subject to general initial and final conditions under asymmetric bounds on jerk, acceleration and velocity. The main contribution of this paper is to explore a difficulty identified by OTG works, that is the non-linearity of the time-optimal solution and the appearance of discontinuities mainly for short motions.

Very little work has been done on time-varying kinematic constraints. Kröger extended its OTG to build robot controllers capable to react to unforeseen events by changing the kinematic constraints[8, 9]. The strategy implemented consists of reducing acceleration before speed using a decision tree. More recently, the same approach is adopted to synchronize multi-DOF trajectories[18].

In this paper, we propose a more precise definition of the problem that allows us to define the optimal solution according to the optimization criteria.

III. TIME-OPTIMAL CUBIC POLYNOMIAL TRAJECTORIES

A. Problem description

This section introduces our previous work that is extended in this paper. The algorithm[1] presented in [16] computes the time optimal trajectory, let's call it $\mathcal{T}_{opt}$, which is known to be a series of at most seven trajectory segments that each saturates one of the bounds [2, 14].

Each of these trajectory segments S_n is a polynomial cubic function of time t and is characterized by a constant jerk $\mathcal{J}_n \in \{\mathcal{J}_{min}, \mathcal{J}_{max}, 0\}$, 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 $\tau_n \leqslant t \leqslant (\tau_n + T_n)$:

$$S_n(t) = \frac{1}{6}J_n(t-\tau_n)^3+\frac{1}{2}a_n(t-\tau_n)^2+v_n(t-\tau_n)+x_n \quad (1)$$

$$\dot{S}_n(t) = \frac{1}{2}J_n(t-\tau_n)^2+a_n(t-\tau_n)+v_n \quad (2)$$

$$\ddot{S}_n(t) = J_n(t-\tau_n)+a_n \quad (3)$$

The trajectory must comply with the asymmetric bounds for jerk, acceleration and velocity:

$$\mathcal{V}_{min} < \dot{S}_n(t) < \mathcal{V}_{max} \quad (4)$$

$$\mathcal{A}_{min} < \ddot{S}_n(t) < \mathcal{A}_{max} \quad (5)$$

$$\mathcal{J}_{min} < \dddot{S}_n(t) < \mathcal{J}_{max} \quad (6)$$

A trajectory's limits are denoted as this: $\mathcal{L}_{\mathcal{T}} = (\mathcal{V}_{min}, \mathcal{V}_{max}, \mathcal{A}_{min}, \mathcal{A}_{max}, \mathcal{J}_{min}, \mathcal{J}_{max})$. They define the admissible kinematic domain $\mathcal{D}$.

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. C_i and C_f must be confined inside $\mathcal{D}$ (see Fig. 2).

The addition of this paper is to extend our previous work so that time-varying kinematic constraints can be considered. If enlarging the bounds is not a problem, the reaction to a decrease is more complex, because the system can be outside $\mathcal{D}$, i.e. C_i can be outside of the green area (Fig. 2).

B. The phase diagram

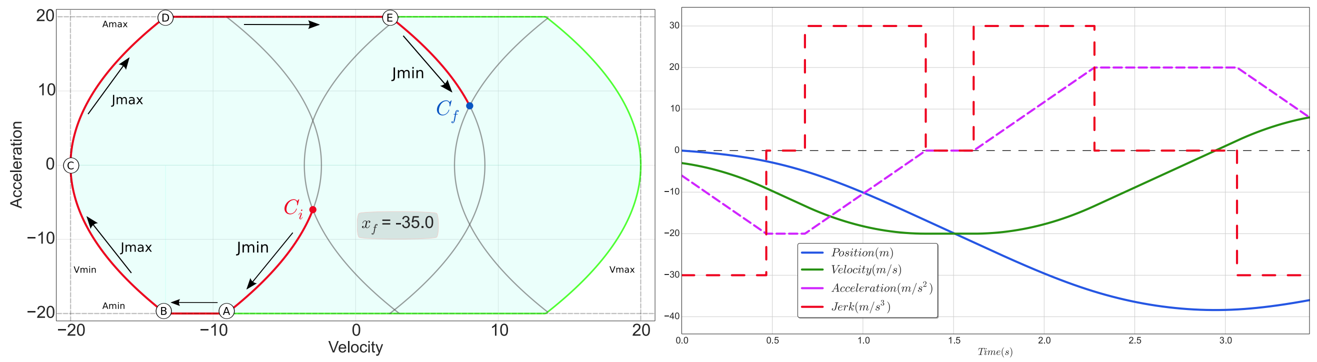
These trajectories are well described with a phase diagram (Fig. 2) where abscissa is velocity and ordinate is acceleration. In this diagram, constant jerk trajectories associated to third degree polynomial functions define horizontal parabolas, constant acceleration motions are horizontal lines and constant velocities motions are points of the null acceleration axis. The area where acceleration and speed constraints are verified is plotted in green in the figures. The acceleration bounds $\mathcal{A}_{min}$ and $\mathcal{A}_{max}$ define two horizontal boundary lines while the left and right boundaries are parabolas respectively defined by ($\mathcal{V}_{min}$, $\mathcal{J}_{max}$) and ($\mathcal{V}_{max}$, $\mathcal{J}_{min}$).

From the initial condition C_i defined by v_i and a_i, the two jerk bounds allow only two optimal motions, which define two condition parabolas in the phase diagram. 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.

The equations of the parabolas are obtained by eliminating the time in the equations (2, 3):

$$\dot{S}_n = \frac{\ddot{S}_n^{\ 2}}{2J_n} + v_n - \frac{a_n^{\ 2}}{2J_n} \quad (7)$$

[1]The documentation, the softMotion library and examples are available at `https://git.openrobots.org/projects/softmotion/wiki`.



(a) (b)

Fig. 2: Left: Phase diagram, the green area of the admissible kinematic domain $\mathcal{D}$ is limited by the acceleration bounds ($\mathcal{A}_{min}, \mathcal{A}_{max}$) and the parabolas associated to velocity bounds ($\mathcal{V}_{min}, \mathcal{V}_{max}$). The jerks $\mathcal{J}_{min}$ and $\mathcal{J}_{max}$ define four condition parabolas passing through the initial conditions C_i and the final condition C_f. The red curve joins C_f from C_i with the following 7 segments: $\mathcal{J}_{min} \rightarrow \mathcal{A}_{min} \rightarrow \mathcal{J}_{max} \rightarrow \mathcal{V}_{min} \rightarrow \mathcal{J}_{max} \rightarrow \mathcal{A}_{max} \rightarrow \mathcal{J}_{min}$. The segment with saturated speed $\mathcal{V}_{min}$ holds on point C. Right: The corresponding trajectory.

IV. TIME VARIANT KINEMATIC CONSTRAINTS

A. Notations

We will use the following notations for the rest of the paper: $\mathcal{T}_r$ is the trajectory reaching $\mathcal{D}$ from C_i. $\mathcal{T}_r$ is composed of at most two trajectory segments with the characteristics described in III-A. $C_{i'}$ denotes the end condition of $\mathcal{T}_r$, which is located on the limits of the extended phase diagram (see Fig. 3). $\mathcal{T}_{opt}$ is the classical time-optimal trajectory introduced in our previous paper. In this situation $\mathcal{T}_{opt}$ is computed from $C_{i'}$ to C_f. Finally $\mathcal{T}_{ext}$ names the overall trajectory, that is the concatenation of $\mathcal{T}_r$ and $\mathcal{T}_{opt}$.

B. Problem description

In the situations where C_i is outside of $\mathcal{D}$, our previous algorithm cannot compute the time-optimal trajectory $\mathcal{T}_{opt}$ joining C_i to C_f. Moreover being time-optimal is not sufficient and the previous heuristics must be adapted.

The objective is to find the trajectory $\mathcal{T}_r$ that restore the fastest the robot's velocity and acceleration inside their limits. Jerk can be switched instantly. $\mathcal{T}_r$ end-point is then a valid initial condition from which the general algorithm can compute the time-optimal trajectory $\mathcal{T}_{opt}$ joining C_f.

Similarly to the general problem of finding $\mathcal{T}_{opt}$ between two conditions, there is an infinite number of trajectories that can restore the integrity of the system. Furthermore it is important to note that for the computation of $\mathcal{T}_r$ the end-point is not specified, expanding the range of solutions. To simplify the construction of $\mathcal{T}_r$ the objectives must be clearly specified. We enumerate the following conditions the solution must verify:

C.1 Valid constraints cannot be intentionally transgressed.
C.2 Invalid constraints cannot be further violated.
C.3 Time-optimality of $\mathcal{T}_r$ in accord with C.1 and C.2.

In C.1 definition, the word "intentionally" is used to distinguish from situations where initial acceleration and velocity are under their respective limits but only temporarily. In Fig. 4 T_7 has both velocity and acceleration under their maxima, however velocity must be transgressed due to the acceleration value. For this reason $C_i \notin \mathcal{D}$.

C.3 purpose is to restore the system into the admissible domain $\mathcal{D}$ as fast as possible. This choice is mainly motivated by safety and efficiency reasons. However to lower the total time the system stays outside $\mathcal{D}$, C.3 alone justify the transgression of initially valid constraints or to further transgress the invalids. To prevent these conditions C.1 and C.2 are added. This addition is motivated by the study of the cause of the deregulation. Little transgression come usually at control level due to tracking error or excessive vibrations. In this case using a higher jerk will only accentuate the problem. When bounds are largely exceeded it is because they have been redefined at higher level possibly in response to some events. The new bounds correspond to a policy defined by an entity that has the authority and that can be trusted since higher in the architecture hierarchy (Fig. 1).

The use of these conditions together provide our algorithm (Alg. 1) with an heuristic that ensure the uniqueness of the solution for every C_i. The construction of $\mathcal{T}_r$ is now simple and straightforward.

C. The extended phase diagram

A first step is to extend the phase diagram (Fig. 2a). The extension of $\mathcal{D}$ represented by the dashed green line in (Fig. 3) is available for C_i only as C_f must have been defined accordingly to the kinematic constraints. The other two symmetrical corners are not available since the robot cannot stay in this area without breaking the speed limit.

The construction of $\mathcal{T}_r$ depends only on the velocity and acceleration value of the starting point. The distance x_f only matters in special cases that will be discussed in (V). The choice of $\mathcal{T}_r$ can then be made from the location of C_i in the phase diagram.

The phase diagram is a useful tool that provides a global view of all the possible trajectories available to construct $\mathcal{T}_r$ according to C_i location. We can now distinct a total of

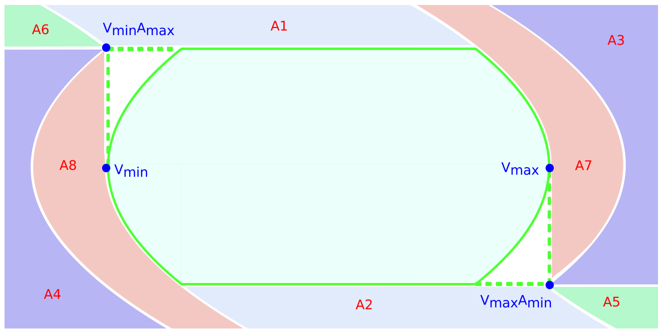


Fig. 3: The extended phase diagram is built from the initial C_i configurations that define areas associated with a trajectory type. Some areas are similar due to symmetries, and have been sorted by colors.

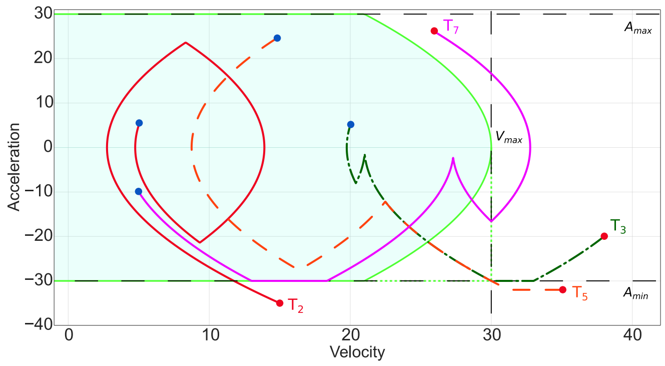


Fig. 4: Example of trajectories illustrating the algorithm (Alg. 1) for different C_i. They are numbered accordingly to areas described in Fig. 3.

eight areas and the belonging of C_i to one of those will be responsible for the shape of $\mathcal{T}_r$ (Fig. 3). For a specific area there is a unique trajectory $\mathcal{T}_r$ that restore the system with respect to the imposed conditions (see IV-B).

D. The extended algorithm

We define a set A of areas defining a partition of the phase diagram (Fig. 3). Let's denote $\mathcal{P}^-_{\mathcal{V}_{min}}$ and $\mathcal{P}^+_{\mathcal{V}_{min}}$ the parabolas defined by $\mathcal{V}_{min}$ and respectively $\mathcal{J}_{min}$ and $\mathcal{J}_{max}$. Similarly, $\mathcal{P}^{-/+}_{\mathcal{V}_{max}}, \mathcal{P}^{-/+}_{\mathcal{V}_{max}\mathcal{A}_{min}}$ and $\mathcal{P}^{-/+}_{\mathcal{V}_{min}\mathcal{A}_{max}}$ are the parabolas that pass through the points $\mathcal{V}_{max}, \mathcal{V}_{max}\mathcal{A}_{min}$ and $\mathcal{V}_{min}\mathcal{A}_{max}$. In the same way, the maximum acceleration linear segments are $\mathcal{A}_{min}$, $\mathcal{A}_{max}$ and the vertical maximum velocity segments are $\mathcal{V}_{min}$ and $\mathcal{V}_{max}$.

The area $A6$ is, for example, defined by $\mathcal{P}^-_{\mathcal{V}_{min}\mathcal{A}_{max}}$ and $\mathcal{A}_{max}$. Its symmetric counterpart $A5$ is defined by $\mathcal{P}^+_{\mathcal{V}_{max}\mathcal{A}_{min}}$ and $\mathcal{A}_{min}$.

In (Alg. 1) $computeT_{opt}$ denotes the general algorithm presented in [16] for $C_i \in \mathcal{D}$. $isValidCondition(C_i, \mathcal{L}_\mathcal{T})$ check if $C_i \in \mathcal{D}$, while $findConditionArea(C_i, \mathcal{L}_\mathcal{T})$ return the area C_i belongs to.

Algorithm 1 The extended algorithm.

```
$\mathcal{T}_r, \mathcal{T}_{opt}, \mathcal{T}_{ext} \leftarrow [\,]$
**if** $isValidCondition(C_i, \mathcal{L}_\mathcal{T})$ **then**
    $\mathcal{T}_{opt} \leftarrow computeT_{opt}(C_i, C_f, \mathcal{L}_\mathcal{T})$
**else**
    $area \leftarrow findConditionArea(C_i, \mathcal{L}_\mathcal{T})$
    **switch** $(area)$
    **case** $A1$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{min}$ jerk segment reaching $\mathcal{A}_{max}$
    **case** $A2$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{max}$ jerk segment reaching $\mathcal{A}_{min}$
    **case** $A3$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{min}$ jerk segment reaching $\mathcal{A}_{min}$
        $\mathcal{T}_r \leftarrow \mathcal{A}_{min}$ constant linear acceleration segment reaching $\mathcal{V}_{max}\mathcal{A}_{min}$
    **case** $A4$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{max}$ jerk segment reaching $\mathcal{A}_{max}$
        $\mathcal{T}_r \leftarrow \mathcal{A}_{max}$ constant linear acceleration segment reaching $\mathcal{V}_{min}\mathcal{A}_{max}$
    **case** $A5$:
        $\mathcal{T}_r \leftarrow \mathcal{A}_{min}$ constant linear acceleration segment reaching $\mathcal{P}^+_{\mathcal{V}_{max}\mathcal{A}_{min}}$
        $\mathcal{T}_r \leftarrow \mathcal{J}_{max}$ jerk segment reaching $\mathcal{V}_{max}\mathcal{A}_{min}$
    **case** $A6$:
        $\mathcal{T}_r \leftarrow \mathcal{A}_{max}$ constant linear acceleration segment reaching $\mathcal{P}^-_{\mathcal{V}_{min}\mathcal{A}_{max}}$
        $\mathcal{T}_r \leftarrow \mathcal{J}_{min}$ jerk segment reaching $\mathcal{V}_{min}\mathcal{A}_{max}$
    **case** $A7$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{min}$ jerk segment reaching $\mathcal{V}_{max}$
    **case** $A8$:
        $\mathcal{T}_r \leftarrow \mathcal{J}_{max}$ jerk segment reaching $\mathcal{V}_{min}$
    **end switch**
    $\mathcal{T}_{opt} \leftarrow computeT_{opt}(C_{i'}, C_f, \mathcal{L}_\mathcal{T})$
**end if**
$\mathcal{T}_{ext} \leftarrow \mathcal{T}_r + \mathcal{T}_{opt}$
```

V. Discussion

A. Heuristics

The heuristic responsible for the construction of $\mathcal{T}_r$ is defined by the conditions listed in IV-B. In this paragraph we discuss the alternatives.

1) Comparison with related works: To our knowledge there is no work proposing a thorough analysis on time variant kinematic constraints. In [9, 18] the adopted solution is to first lower acceleration then velocity. It is motivated in [18] to ensure the time-optimality of $\mathcal{T}_r$. But it is true only when the initial acceleration and velocity have the same sign.

We illustrate our point by comparing the different solutions (see Fig. 6). C_i must belong to either $A5$ or $A6$. Using equations 2 and 3 we can compute the duration of the two trajectories. The duration of $\mathcal{T}_r$ with our method is $0.277s$ while it takes $0.295s$ using the algorithms described in [9, 18]. We could argue that it is not the only advantage, as lowering velocity faster is safer, see next paragraph.

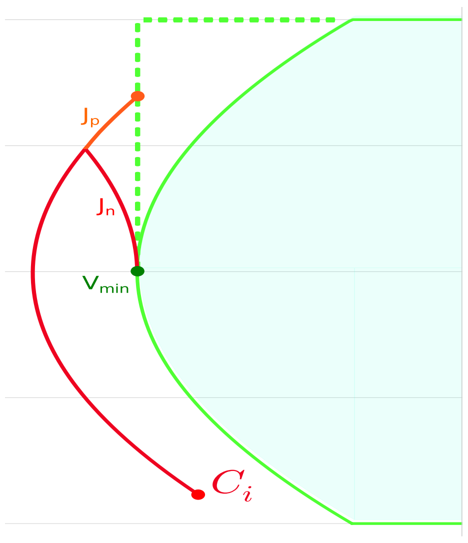


Fig. 5: Comparison of two different strategies when $C_i \in A8$. $\mathcal{T}_r$ is in orange and the trajectory joining $\mathcal{V}_{min}$ in red.

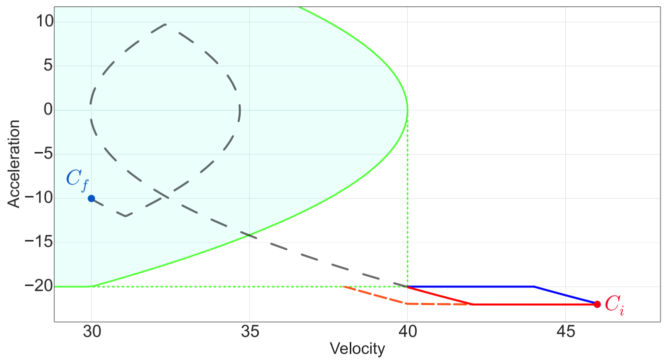


Fig. 6: Our solution to reach $\mathcal{D}$ here in red, and the one presented in [9, 18] in blue (V-A.1). The orange dashed solution privilege velocity regulation over acceleration (V-A.2). We choose the following configuration : $C_i = (0, -20, -22)$ and $V_{max} = 40, A_{min} = -20$ so C_i belongs to $A5$.

2) Velocity before acceleration: Another possibility is to give priority to velocity regulation over acceleration when both exceeds their limits. This choice is motivated by the expression of kinetic energy as it depends on velocity, and so safety: $E = \frac{1}{2}mv^2$. This solution is illustrated in Fig. 6. Obviously the time-optimality of $\mathcal{T}_r$ is no more guaranteed.

3) $\mathcal{T}_{ext}$ optimization: This scenario is described in (Fig. 5) in which $C_i \in A8$. The orange positive jerk segment reaching $\mathcal{V}_{min}$ defines the fastest trajectory to enter $\mathcal{D}$. The trajectory plotted in red is the fastest to reach the constant velocity segment $\mathcal{V}_{min}$.

The time gained on $\mathcal{T}_r$ with the first solution can be negligible, especially when this scenario happens in areas $A7$ and $A8$ that are close to $\mathcal{D}$. By contrast, the time gained on $\mathcal{T}_{ext}$ by choosing the red trajectory is significant for large negative motions that reach $\mathcal{V}_{min}$.

Moreover small violations are more susceptible to be linked to control rather to a decrease of the bounds. In this case it is less interesting to prioritize $\mathcal{T}_r$ optimization over $\mathcal{T}_{ext}$ as safety is not engaged and velocity is close to its limit. However kinematic restrictions coming from higher layers might be due to exterior events (IV-B). In that instance safety must be prioritized and so the time optimality of $\mathcal{T}_r$.

B. Experimental results

The work presented in this paper has been experimented on a Kuka-LWR4 robotic arm (Fig. 7). The trajectory controller runs at $10ms$ while the trajectory planner has a sampling time of $50ms$ (Fig. 1). We illustrate our algorithm for the different areas, one area of each symmetric pair. The trajectory planner sends a first trajectory to the trajectory controller. The purple vertical dashed line indicates the moment when the trajectory's limits $\mathcal{L}_\mathcal{T}$ are decreased and the current configuration of the arm is outside of $\mathcal{D}$. The trajectory planner must compute a new trajectory to restore the system into $\mathcal{D}$. Following Alg. 1 the trajectory $\mathcal{T}_{ext}$ joining C_f according to the updated $\mathcal{L}_\mathcal{T}$ is computed and sent to the trajectory controller. The brown vertical dashed line indicates the moment the system is brought back into $\mathcal{D}$, that is the end of $\mathcal{T}_r$ and the beginning of $\mathcal{T}_{opt}$.

Since only position measurements are available, velocity and acceleration are estimated using a Kalman filter. To smooth the trajectory when the switch occurs, the controller computes a junction trajectory using the three jerk method [15]. An associated video will present experimental results on the Kuka-LWR4 for multi-axes trajectories.

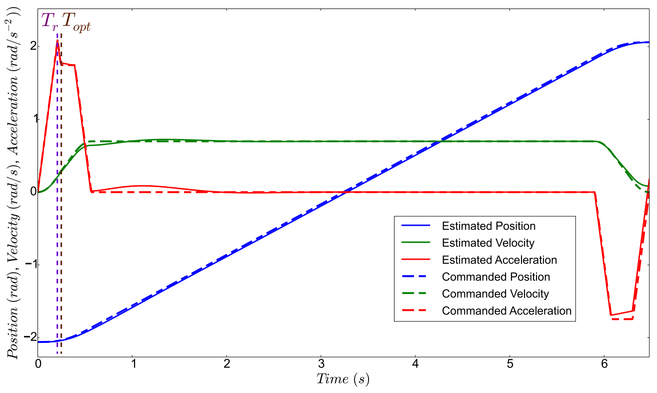
C. Performances

The implementation of this extended algorithm (computation of $\mathcal{T}_{ext}$, Alg. 1) on a system equipped with an Intel Core i7 processor running at 2.2 GHz gives a mean time of 1.16 μs with a standard deviation of 0.39 μs observed for 10^8 random tests, it certainly allows online usage. In [8] no computation times are specified. Since the paper is an extension of [9] that claims an average execution time of 135 μs for a 6DOF system it can be assumed that it is higher. In [18] the average execution time for a 6DOF system is 1.1 ms. It appears that our solution provides better execution times than the previous works.

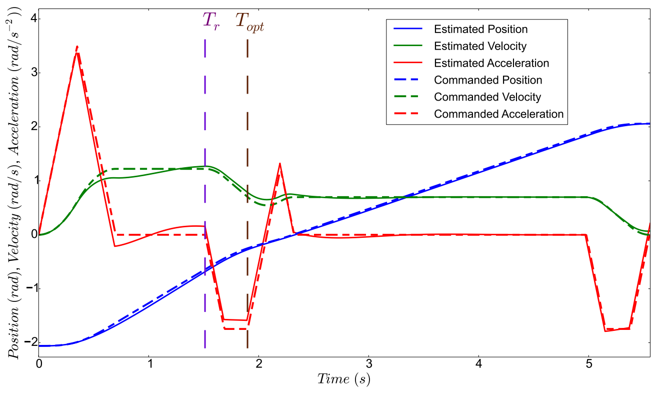
VI. CONCLUSIONS

We have presented an extension of our online trajectory generator in order to react outside of a kinematic domain and bring back the current state of the robot inside the admissible domain. This enable the trajectory generator to accept time-variant motion constraints. Based on an in-depth analysis of the phase diagram, the proposed approach is complete. Moreover the solution is fast and simple. For small violation of the acceptable kinematic domain, an alternative solution is proposed to prioritize the optimization of the global time.

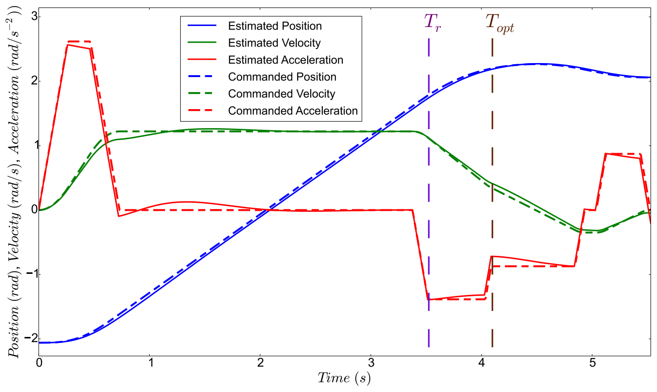
These results are important to control robots that collaborate with humans, because kinematic is related to human safety and ergonomics. Although addressing only the one-dimensional case, these results impact the planning and control of multi-axis system and in particular the synchronization of such systems, which will be the subject of a future work.



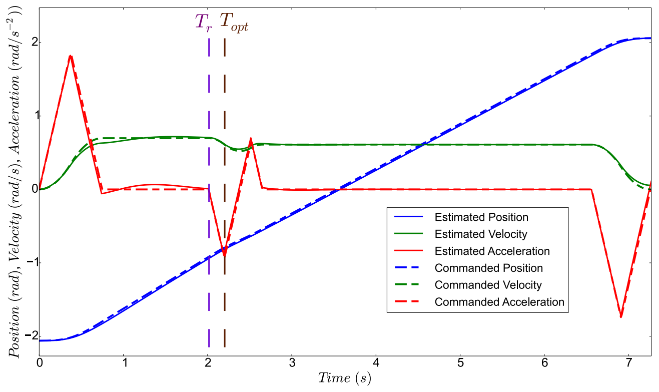
(a) $C_{i'} \in A1$.



(b) $C_{i'} \in A3$.



(c) $C_{i'} \in A5$.



(d) $C_{i'} \in A7$.

Fig. 7: Illustration of motion constraints abrupt switch on a KUKA-LWR4 joint axis.

References

[1] 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.

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

[3] B. Ezair, T. Tassa, and Z. Shiller. "Multi-axis High-order Trajectory Planning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2012.

[4] 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.

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

[6] R. Haschke, E. Weitnauer, and H. 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.

[7] 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 (1984), pp. 2745–2754.

[8] T. Kröger. "On-line trajectory generation: Nonconstant motion constraints". In: *Robotics and Automation (ICRA)*. IEEE, 2012, pp. 2048–2054.

[9] T. Kröger 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.

[10] 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.

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

[12] P. A. Lasota and J. 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.

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

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

[15] D. Sidobre and W. He. "Online task space trajectory generation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2012.

[16] D. Sidobre and K. Desormeaux. "Smooth Cubic Polynomial Trajectories for Human-Robot Interactions". In: *Journal of Intelligent & Robotic Systems* (Oct. 2018), pp. 1–19.

[17] E.A. Sisbot et al. "Synthesizing Robot Motions Adapted to Human Presence". In: *International Journal of Social Robotics* 2.3 (Sept. 1, 2010), pp. 329–343.

[18] R. Zhao and S. Ratchev. "On-line trajectory planning with time-variant motion constraints for industrial robot manipulators". In: 2017 IEEE International Conference on Robotics and Automation (ICRA). Singapore: IEEE, May 2017, pp. 3748–3753.