

# Time-Optimal Path Tracking for Jerk Controlled Robots

Alessandro Palleschi<sup>1,2</sup>, Manolo Garabini<sup>1,2</sup>, Danilo Caporale<sup>1,2</sup>, Lucia Pallottino<sup>1,2</sup>

**Abstract**—This paper presents a new approach to solve the Time-Optimal Path Tracking under limited joint range and bounds on velocity, acceleration and jerk. To obtain smooth and continuous accelerations, with beneficial effects for the load and wear on the actuators but a limited impact on performance, we state the minimum-time path tracking problem with the jerk as the control input. The main contribution of this paper is a formulation that includes the jerk constraints in the optimization problem and that, even if the resulting Non-Linear Programming (NLP) problem is non-convex, allows to perform an efficient and reliable convex relaxation using McCormick Envelopes. Simulations and experimental tests on two 7-DoF manipulators have been carried out to show the benefits of the proposed approach and to compare it to state-of-the-art techniques.

**Index Terms**—Motion and Path Planning; Optimization and Optimal Control; Motion Control

## I. INTRODUCTION

IN recent years, in a wide range of industries, a fast transition towards automation has been driven by the request of greater flexibility, efficiency and speed in industrial processes. This desire for high performance has to practically deal with the structural and operational limitations typical of a robotic system. Robot operational limits are defined by bounds on joint space variables, namely range, velocity, acceleration, torque and torque derivative. Additional constraints could be included to characterize limits on the operative region of some geometric part of the robot, dictated by the characteristics of its workspace, e.g., the presence of obstacles. Although motion capabilities are easily expressed as bounds on joint space variables, the task specifications are generally given in terms of a set of reference Cartesian poses for the end-effector. Solving the generalized time-optimal motion planning problem is complex since it is typically a constrained non-convex NLP problem. Due to the complexity of the problem, a common approach is to split it into two subproblems. Using a decoupled approach, a path planner is used to compute a geometric path accounting for high-level geometric aspects

Manuscript received: February, 24, 2019; Revised April, 27, 2019; Accepted June, 26, 2019.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments.

This work has received funding from the European Union's Horizon 2020 research and innovation program under agreement no. 732737 (ILIAD) and by the Italian Ministry of Education, and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence)

<sup>1</sup> Research Center Enrico Piaggio, University of Pisa, Largo Lucio Lazzarino 1, 56126 Pisa, Italy

<sup>2</sup>Dipartimento di Ingegneria dell'Informazione, University of Pisa, Largo Lucio Lazzarino 1, 56126 Pisa, Italy

alessandropalleschi94@gmail.com

Digital Object Identifier (DOI): see top of this page.



Fig. 1. KUKA LWR IV+ Manipulator used for the experimental tests executing the desired trajectory

and a dedicated kinematics inversion stage is used to map the obtained path into a reference joint space trajectory, accounting for the geometric constraints of the robot (see, e.g., [1], [2], [3]). The inclusion of velocities, accelerations and torques constraints directly at this stage could lead to the formulation of a constrained IK problem (see, e.g. [4] and [5]), which is complex and usually non-convex. A more attractive approach consists in placing a path tracking stage right after the inversion stage. It deals with the constraints mentioned above and finds the minimum time needed to track the given joint space path. (see, e.g., [6],[7]). If the accelerations or the torques are employed as the control input, this approach leads to the formulation of a convex optimization problem [6]. However, a significant drawback for this torque-based control is the high rate of change of the optimal solution. A discontinuous, or at least non-smooth, acceleration profile has a noticeable impact in terms of stress on the actuators and vibrations. An additional term can be included in the optimization problem to address this, as in [6]. Other than the lack of an evident physical meaning of the introduced term and problems on its weight in the cost function, the solution could still be discontinuous. The main contributions of this paper are: (1) to present a formulation that, using the jerk as constrained control input, allows obtaining smoother acceleration profiles, reducing the wear and load on the actuators [8], (2) to show that even if this formulation results in a non-convex NLP problem, it is possible to obtain the globally optimal solution by performing an efficient convex relaxation of the non-convex constraints. The outline of this paper is as follows. Section II discusses current state-of-the-art methods to include constraints in motion planning, while Section III introduces our jerk-based formulation for time-optimal path tracking. This

particular formulation and the numerical solution are discussed in Section IV. Eventually, simulations and experimental tests on a real 7-DoF manipulator are presented in Section V, while considerations about the inclusion of the dynamics of the robots are presented in the Appendix.

## II. RELATED WORKS

The task specifications for a robot are given as a set of reference poses for the end-effector, hence expressed in the Operational (Cartesian) space, and local inversion of differential kinematics (IK) is used to map the reference trajectory to the joint space [1], with the possibility of controlling the robot up to the torque-level. Recently, some methods have been presented to deal with the limited capabilities of redundant robots. These bounds are usually expressed in terms of limited joint ranges, velocities, accelerations and torques. Many of the currently used solutions suffer from the presence of singularities, both kinematic and algorithmic, which affect tasks execution. A novel IK solver based on a Reverse Priority approach (RP), largely unaffected by algorithmic singularities has been presented in [9], and later extended to include unilateral geometric constraints in a nice fashion in [2]. To the best of the authors' knowledge, RP-based approaches are yet to be extended to include dynamic constraints, such as velocities and accelerations. Several approaches (see, e.g., [6],[7]) transform the problem as an optimal tracking problem over a fixed path. These methods for the time-optimal path tracking under actuators constraints exploit the fact that the motion along a fixed path is described by a single path coordinate  $s$  and its time derivative  $\dot{s}$  (see, e.g., [10], [11]). As a result, the multi-dimensional state space of a robotic system can be reduced to a two-dimensional state space. This reduction has been widely described in [6] applied to robotic manipulators and has been extended for more general systems, see, e.g., [12] for quadrotors and [7] for a wide range of vehicles, such as space vehicles, car models and aircraft. Thanks to the use of a nonlinear change of variables already recognized in [10], it is possible to transform the problem into a convex optimal control problem [6]. A numerical solution can be obtained efficiently using a direct transcription method [13]. In these works jerk or torque rate limits are not considered. A formulation able to include limits on jerk and/or torque rate of change is of great interest, despite leading to increased time needed to complete the task, in order to reduce actuator loads for the demanded torque and robot wear potentially expanding its life-span [14]. Since constraints on the rate of torque change,  $\dot{\tau}(s)$ , make the problem non-convex [6], it is possible to constrain the variations of the control input, so a bound on  $\Delta u(s)$ , in order to reduce acceleration/torque jumps while maintaining convexity of the problem. This would still result in discontinuous, or at least non-smooth, acceleration commands, still piecewise constant and theoretically with an infinite jerk. Torque rate and jerk can also be limited by inclusion on the cost function (see, e.g., [15],[16],[17]), but they are not limited to a specified value. Inclusion of jerk or torque rate constraints is possible by extending the state space to three variables, as stated in [18], where the constructed

optimization problem is however still time-dependent. Given the appealing properties and the advantages of a smooth, jerk-limited solution, a method to rewrite the problem combining a nonlinear transformation of the optimization variable similar to [6] with a jerk-level control strategy has been investigated. Since the resulting NLP is non-convex, the complexity of this formulation, presented in the next section, is higher than the classical acceleration-bounded approaches and requires using a solver able to perform global optimization.

## III. PROBLEM FORMULATION

Unlike the torque-constrained model presented in [6], in this paper a kinematic approach has been investigated, where the dynamics of each joint is modelled as a triple integrator. In the appendix, the formulation is extended to include the dynamics of the robot. This section is organized as follows. First, a presentation of the transformation of the multi-dimensional state space of the robot in a three-dimensional state space is presented in III-A. Secondly, the jerk-level formulation of the problem, along with the exploited nonlinear change of variables, is presented and discussed in III-B.

### A. Path Coordinates Parametrization

Given a path in joint space coordinates,  $\mathbf{q}(t) \in \mathbb{R}^n$ , it can be expressed as a function of a scalar path coordinate  $s \in \mathbb{R}$ . This coordinate determines the geometry of the path, whereas the timing of the trajectory is expressed by the relation between the path coordinate and the time, i.e.,  $s(t)$ . Without any loss of generality, it is assumed to be  $s(0) = 0 \leq s(t) \leq 1 = s(T)$ , which means the trajectory starts at  $t = 0$  and ends at  $t = T$ . Moreover,  $\dot{s}(t) \geq 0$  everywhere and  $\dot{s}(t) > 0$  almost everywhere.

In this formulation, the joint velocities, accelerations and jerks are rewritten using the chain rule:

$$\begin{cases} \dot{\mathbf{q}}(s) = \mathbf{q}'(s)\dot{s} \\ \ddot{\mathbf{q}}(s) = \mathbf{q}'(s)\ddot{s} + \mathbf{q}''(s)\dot{s}^2 \\ \dddot{\mathbf{q}}(s) = \mathbf{q}'(s)\ddot{s} + 3\mathbf{q}''(s)\dot{s}\ddot{s} + \mathbf{q}'''(s)\dot{s}^3 \end{cases} \quad (1)$$

where  $\dot{s} = ds/dt$ ,  $\ddot{s} = d^2s/dt^2$ ,  $\ddot{s} = d^3s/dt^3$ ,  $\mathbf{q}' = \partial\mathbf{q}(s)/\partial s$ ,  $\mathbf{q}'' = \partial^2\mathbf{q}(s)/\partial s^2$  and  $\mathbf{q}''' = \partial^3\mathbf{q}(s)/\partial s^3$ .

### B. Minimum-Time Problem Formulation

Since the proposed formulation needs the jerk to be used as the control input of the system, i.e.,  $\mathbf{u}(t) = \ddot{\mathbf{q}}(t)$ , the time-optimal path tracking problem can be written as:

$$\begin{cases} \min_{T, s(\cdot), \mathbf{u}(\cdot)} T \\ \text{subject to } \mathbf{u}(s(t)) = \ddot{\mathbf{q}}(s(t)) \\ s(0) = 0 \text{ and } s(T) = 1 \\ \dot{s}(0) = \dot{s}_0 \text{ and } \dot{s}(T) = \dot{s}_T \\ \dot{s}(t) \geq 0 \\ \ddot{s}(0) = \ddot{s}_0 \text{ and } \ddot{s}(T) = \ddot{s}_T \\ \dot{\mathbf{q}}(s(t)) \leq \dot{\mathbf{q}}(s(t)) \leq \bar{\dot{\mathbf{q}}}(s(t)) \\ \ddot{\mathbf{q}}(s(t)) \leq \ddot{\mathbf{q}}(s(t)) \leq \bar{\ddot{\mathbf{q}}}(s(t)) \\ \mathbf{u}(s(t)) \leq \mathbf{u}(s(t)) \leq \bar{\mathbf{u}}(s(t)) \\ \text{for } t \in [0, T]. \end{cases} \quad (2)$$

Similar to [6], a nonlinear variable change is used in order to drop the time dependency and transform the optimization problem. First, the objective function is rewritten as:

$$T = \int_0^T 1 dt = \int_{s(0)}^{s(T)} \frac{1}{\dot{s}} ds = \int_0^1 \frac{1}{\dot{s}} ds. \quad (3)$$

Then the following variables are introduced as optimization variables with differential constraints:

$$\begin{cases} b(s) = \dot{s}^2, & a(s) = \ddot{s}, & c(s) = \ddot{\ddot{s}}/\dot{s} \\ b'(s) = 2a(s), & b''(s) = 2c(s) \end{cases} \quad (4)$$

and the problem (2), along with (3) becomes:

$$\left\{ \begin{array}{l} \min_{a(\cdot), b(\cdot), c(\cdot), \mathbf{u}(\cdot)} \int_0^1 \frac{1}{\sqrt{b(s)}} ds \\ \text{subject to } \mathbf{u}(s) = \sqrt{b(s)}(\mathbf{q}'(s)\mathbf{c}(s) + 3\mathbf{q}''(s)\mathbf{a}(s) + \\ + \mathbf{q}'''(s)\mathbf{b}(s)) \quad (*) \\ b(0) = \dot{s}_0^2 \text{ and } b(T) = \dot{s}_T^2 \\ b(s) \geq 0 \\ b'(s) = 2a(s) \text{ and } b''(s) = 2c(s) \\ b'(0) = 2\dot{s}_0 \text{ and } b'(T) = 2\dot{s}_T \\ b(s) \leq \bar{b}(s) \quad (**) \\ \bar{\mathbf{q}}(s) \leq \mathbf{q}'(s)\mathbf{a}(s) + \mathbf{q}''(s)\mathbf{b}(s) \leq \bar{\bar{\mathbf{q}}}(s) \\ \underline{\mathbf{u}}(s) \leq \mathbf{u}(s) \leq \bar{\mathbf{u}}(s) \\ \text{for } s \in [0, 1] \end{array} \right. \quad (5)$$

where (\*\*) is used to include velocity constraints as in [6]. Clearly, the presented change of variables is not defined in  $\dot{s} = 0$ , hence some precautions have to be taken in order to avoid such singularity. The resulting problem is non-convex, due to (\*), and, at first glance, it is not possible to easily assert whether a nice convex relaxation for this kind of problem exists. From this perspective, if the change of variables  $\mathbf{v}(s) = \mathbf{u}(s)/\sqrt{b(s)}$  is used, it is possible to rewrite (\*) as:

$$\mathbf{v}(s) = \mathbf{q}'(s)\mathbf{c}(s) + 3\mathbf{q}''(s)\mathbf{a}(s) + \mathbf{q}'''(s)\mathbf{b}(s),$$

that is linear in the optimization parameters, and the jerk bounds  $\underline{\mathbf{u}}(s) \leq \mathbf{u}(s) \leq \bar{\mathbf{u}}(s)$  in (5) as:

$$\underline{\mathbf{u}}(s) \leq \mathbf{v}(s)D(s) \leq \bar{\mathbf{u}}(s),$$

where a new function  $D(s)$  is included with following constraints:

$$\mathbf{D}^2(s) = \mathbf{b}(s), \quad \mathbf{D}(s) > 0.$$

The advantages resulting from this formulation become clear once direct transcription is used to solve the problem numerically, as shown in the next section.

#### IV. NUMERICAL SOLUTION

If we use direct transcription, discretizing the path coordinate  $s$  on  $N + 1$  grid points  $s_0 = 0 \leq s_k \leq 1 = s_N$  and modeling the functions  $b(s)$ ,  $a(s)$ ,  $c(s)$ ,  $D(s)$  and  $v_i(s)$  by a finite number of variables  $b^k$ ,  $a^k$ ,  $c^k$ ,  $D^k$  and  $v_i^k$ , the optimal control problem can be discretized to obtain a large sparse optimization problem. Since  $c(s)$  can be treated as the control input for the problem, it is reasonable to model it as

piecewise constant. Then, functions  $b(s)$ ,  $a(s)$  are piecewise quadratic and piecewise linear, respectively. All the variables are assigned on the grid points. The piecewise quadratic function  $b(s)$  is computed using for each interval  $[s^k, s^{k+1}]$  a quadratic Lagrange polynomial to interpolate  $b^k$ ,  $b^{k+1}$  and  $b^{k+2}$ :

$$\begin{aligned} b(s) &= b^k \frac{(s - s^{k+1})(s - s^{k+2})}{(s^k - s^{k+1})(s^k - s^{k+2})} + \\ & b^{k+1} \frac{(s - s^k)(s - s^{k+2})}{(s^{k+1} - s^k)(s^{k+1} - s^{k+2})} + \\ & b^{k+2} \frac{(s - s^k)(s - s^{k+1})}{(s^{k+1} - s^k)(s^{k+2} - s^k)}, \end{aligned} \quad (6)$$

with  $k = 0, \dots, N - 1$ . Since for the last interval only two points are available, the Lagrange polynomial through  $b^{N-2}$ ,  $b^{N-1}$ ,  $b^N$  is used for  $[s^{N-1}, s^N]$ . The following large scale optimization problem is obtained:

$$\left\{ \begin{array}{l} \min_{a^k, b^k, c^k, D^k, v^k} \sum_{k=0}^{N-1} \int_{s^k}^{s^{k+1}} \frac{1}{\sqrt{b(s)}} ds \quad (*) \\ \text{subject to } \mathbf{v}^k = \mathbf{q}'(s^k)\mathbf{c}^k + 3\mathbf{q}''(s^k)\mathbf{a}^k + \mathbf{q}'''(s^k)\mathbf{b}^k \\ b^0 = \dot{s}_0^2 \text{ and } b^N = \dot{s}_T^2 \\ b^k > 0 \text{ and } b^N > 0 \quad (**) \\ (b^{k+1} - b^k) = 2a^k \Delta s^k \\ a^0 = \dot{s}_0 \text{ and } a^N = \dot{s}_T \\ a^{k+1} - a^k = c^k \Delta s^k \\ a^N - a^{N-1} = c^N \Delta s^{N-1} \\ b^k \leq \bar{b}(s^k) \\ (D^k)^2 = b^k \text{ and } (D^N)^2 = b^N \\ D^k > 0 \text{ and } D^N > 0 \\ \bar{\mathbf{q}}(s^k) \leq \mathbf{q}'(s^k)\mathbf{a}^k + \mathbf{q}''(s^k)\mathbf{b}^k \leq \bar{\bar{\mathbf{q}}}(s^k) \\ \underline{\mathbf{u}}(s^k) \leq \mathbf{v}^k D^k \leq \bar{\mathbf{u}}(s^k) \quad (+) \end{array} \right. \quad (7)$$

where constraint (\*\*) is used to avoid the aforementioned singular condition  $\dot{s} = \sqrt{b} = 0$  and  $\Delta s^k = s^{k+1} - s^k$  has been defined. The objective function (\*) is a sum of integrals over  $[s^k, s^{k+1}]$ , each of which can be computed analytically or using numerical approximations like Simpson's Rule.

#### A. Convex Relaxation

Spatial Branch-and-Bound (sBB) is commonly used for the solution of NLPs to global optimality. This requires the computation of a lower bound of the solution, usually obtained by solving a convex relaxation of the problem [19]. An upper bound can be obtained by solving the original non-convex problem using values obtained from the relaxed problem and then checking for feasibility. Even if it is generally possible to form a convex relaxation of any NLP, tight convex underestimators have been studied for particular types of constraints [20]. Problem (7) has a bilinear constraint, (+), which is non-convex, but can be relaxed using McCormick Envelopes [21]. A new variable replaces the bilinear term and linear inequality constraints representing the convex envelope of this term are added to the problem. McCormick Envelopes provide an envelope that retains convexity while minimizing the size of the new feasible region. The lower bound solution obtained

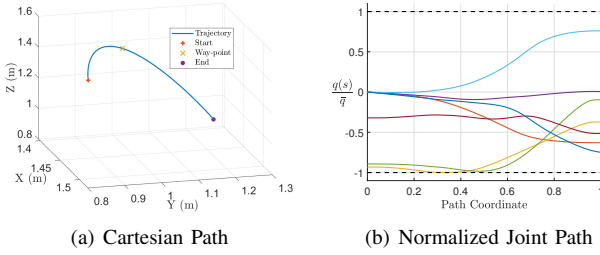


Fig. 2. Cartesian and Joint Path. All the joints respect the limits.

solving the relaxed convex problem (see, e.g., [22] as a general reference on convex optimization) is closer to the true solution than if other convex relaxations were used.

## V. EXPERIMENTAL RESULTS

In order to verify our formulation, simulations and tests on a KUKA LWR robot, a 7-DoF manipulator, have been carried out. This section is organized as follows. First, a numerical solution of the proposed formulation for a given joint space path is presented. This solution is then compared with the solutions of three different strategies for the same path, namely a velocity control with constrained speed (**A - Bounded Velocity**), an acceleration control with constrained speed and acceleration (**B - Bounded Acceleration**) and an acceleration control with additional constraints on control input variations  $\Delta u(s)$  (**C - Bounded Acceleration Variation**). Note, that strategy (**B**) uses the same formulation presented in [6], neglecting the dynamics. These results are presented in V-B, while a discussion about the generation of the joint space path and the robot kinematics and dynamics bound is presented in V-A.

### A. Robot Constraints and Path Generation

The bounds on the 7-DoF robot motion capabilities, later used during optimization, are assumed to be symmetric ( $\bar{q} = -\underline{q}$ ,  $\bar{\dot{q}} = -\underline{\dot{q}}$ ,  $\bar{\ddot{q}} = -\underline{\ddot{q}}$ ,  $\bar{\ddot{\ddot{q}}} = -\underline{\ddot{\ddot{q}}}$ ) and have been set as:

$$\begin{cases} \bar{q} = [170^\circ, 120^\circ, 170^\circ, 120^\circ, 170^\circ, 120^\circ, 170^\circ] \\ \bar{\dot{q}} = [110^\circ/\text{s}, 110^\circ/\text{s}, 128^\circ/\text{s}, 128^\circ/\text{s}, \\ \quad 204^\circ/\text{s}, 184^\circ/\text{s}, 184^\circ/\text{s}] \\ \bar{\ddot{q}}_i = 2860^\circ/\text{s}^2 \quad i = 1, \dots, 7 \\ \bar{\ddot{\ddot{q}}}_i = 17200^\circ/\text{s}^3 \quad i = 1, \dots, 7 \end{cases} \quad (8)$$

The task to be executed has been first designed in the Cartesian space, defining a path for the 7<sup>th</sup> link of the robot. A 6<sup>th</sup> order polynomial to interpolate from the initial pose to the final pose through an intermediate way-point is used to design the positions to follow, Fig. 2(a), while the orientation part of the path is designed using quaternion SLERP [1] from initial to the final configuration. The corresponding joint path, shown in Fig. 2(b) normalized to the upper and lower bounds of each joint, is obtained using the RP algorithm [9] where the desired position and orientation of the end-effector are considered together as a single task, while joint limits are introduced as higher priority tasks, like in [2].

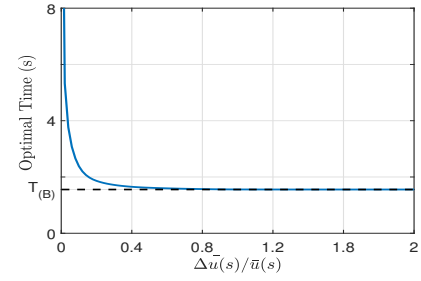


Fig. 3. Optimal Time for different bound on  $\Delta u(s)$ . The value  $T_{(B)} = 1.57$  s represents the optimal time obtained using strategy (B).

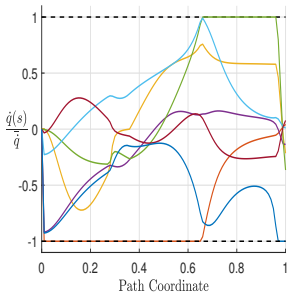
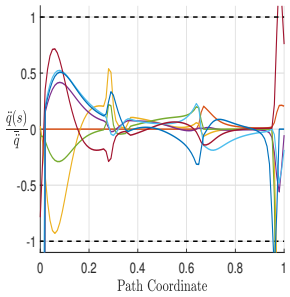
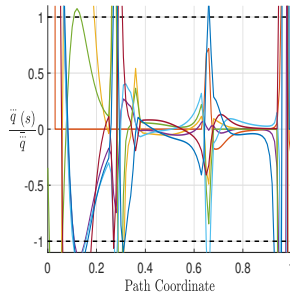
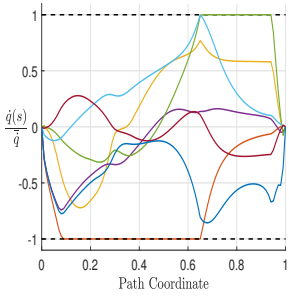
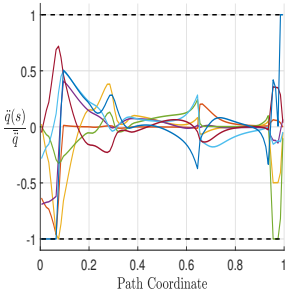
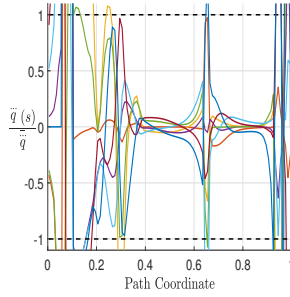
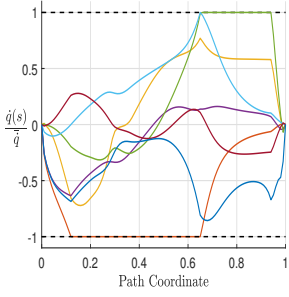
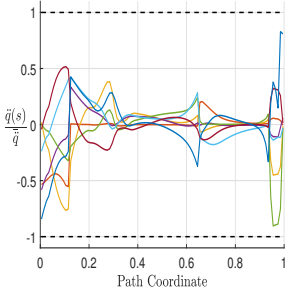
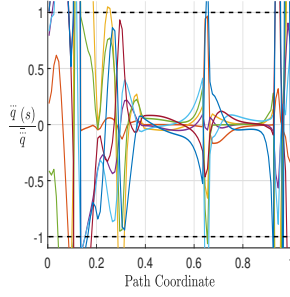
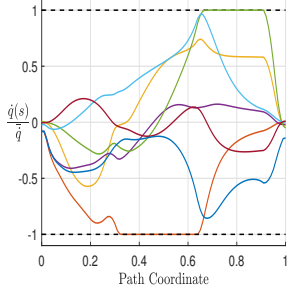
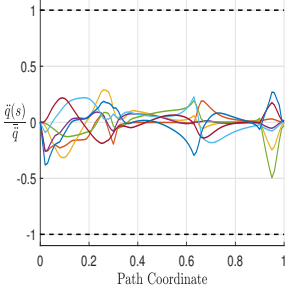
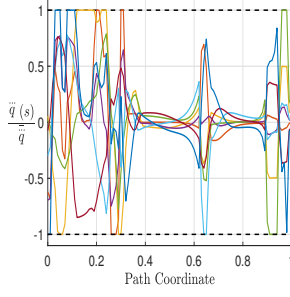
### B. Numerical Results

Here are presented numerical results for the given path for the three strategies defined beforehand (A, B and C) and our jerk-level solution with constrained zero initial and final acceleration along the path (**D**). All the optimization problems have been implemented in MATLAB using YALMIP [23] and solved using SCIP solver [24] provided by Opti Toolbox [25]. The path coordinate is discretized using  $N = 100$ . The results and the worst-case performance with respect to the given constraints for each strategy are summarized in the table below:

	Time (s)	$\max_i  \dot{q}_i /\bar{\dot{q}}_i$	$\max_i  \ddot{q}_i /\bar{\ddot{q}}_i$	$\max_i  \ddot{\ddot{q}}_i /\bar{\ddot{\ddot{q}}}_i$
<b>A</b>	1.4866	1.000	<b>21.57</b>	<b>656.9</b>
<b>B</b>	1.5547	1.000	1.000	<b>29.96</b>
<b>C</b>	1.5700	1.000	0.900	<b>2.810</b>
<b>D</b>	1.7476	1.000	1.000	1.000

As expected, with strategies A, B and D there is at least one joint that saturates one among velocity, acceleration or jerk and only D respects the bound on the jerk. Using C, the acceleration profile does not reach saturation since bounds on the acceleration variations have been included. The resulting profiles for each strategy are displayed in Table I, where all the quantities have been normalized to their upper and lower bounds and displayed with variable  $s$  on the x-axis. For strategy D, it can be seen that a smoother acceleration profile is obtained and the jerk is correctly within the bounds. The total time needed to complete the task is larger for strategy D, as expected. Some considerations have to be made for strategy C and the imposed bounds on the acceleration variations. It is not straightforward to understand how to impose this bound in order to limit acceleration jumps while not affecting the performance considerably. To quantify how much these bounds can affect the performance, the optimization problem has been solved for different values of the bounds on the acceleration variation, i.e.,  $\Delta \bar{u}(s) = -\Delta \underline{u}(s) \in [0.1, 2\bar{\ddot{q}}(s)]$  and the results are reported in Fig. 3. As expected, too small values of the upper bound degrade the performance considerably, while for high values the performance become closer to the ones obtained with B. Moreover, using strategy C, we are able to limit the jerk but we can not guarantee that is limited to a specified value.

TABLE I  
NUMERICAL RESULTS FOR DIFFERENT STRATEGIES APPLIED TO THE SAME PATH

Strategy and Time	Joint Velocities	Joint Accelerations	Joint Jerks
Bounded Velocity - (A) Time = 1.4866 s			
Bounded Acceleration - (B) Time = 1.5547 s			
Bounded Acceleration Variation - (C) Time = 1.5700 s			
Bounded Jerk - (D) Time = 1.7476 s			

### C. Tests on a real manipulator

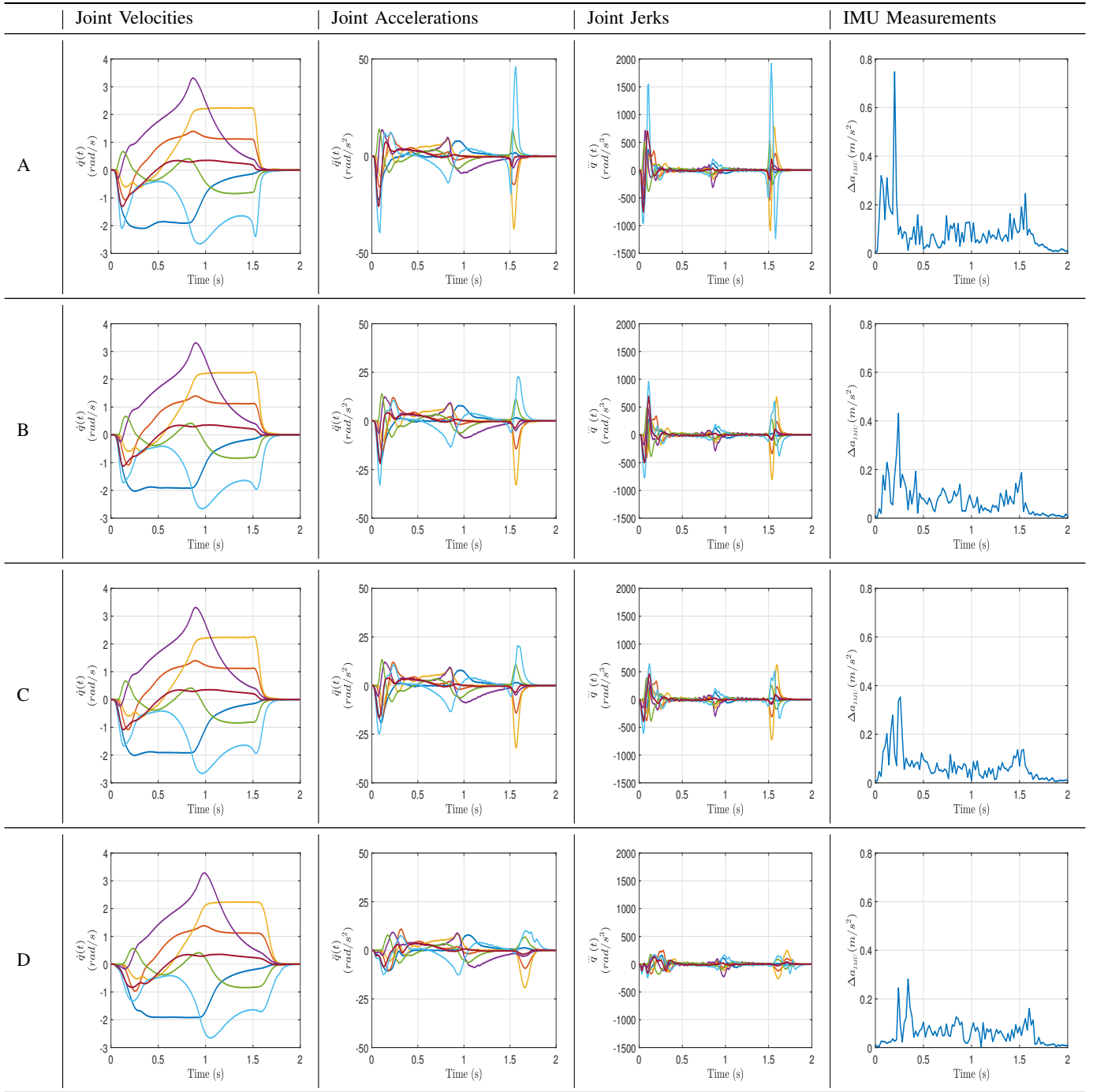
The proposed formulation has been tested on a KUKA LWR IV+ robot, a 7-DoF manipulator shown in Fig. 1. The optimization is used to plan the motion of the robot, i.e., generating the optimal jerk-limited trajectory  $\mathbf{q}(t)$  which is then sent to the robot. The manipulator has been controlled using its ROS-based interface [26]. To show the quality of our solution, our approach has been compared to the three strategies presented beforehand. For each strategy the maximum jerk is reported. Moreover, further analyses have been carried out to compare

the smoothness of the acceleration profiles. To do so, we have defined a function to evaluate the jerk profile of all the seven joints. To do so, we introduce the function  $S: R^{7 \times N} \rightarrow R^+$ , with  $N$  number of samples:

$$S(\ddot{\mathbf{q}}) = \frac{1}{t_f} \cdot \sqrt{\sum_{i=1}^7 |\ddot{q}_i|^2} \quad (9)$$

where  $t_f$  is the total trajectory time and  $|\ddot{q}_i|$  is the  $L^1$ -norm of the jerk profile of the joint  $i$ . The results are reported in Table

TABLE II  
EXPERIMENTAL RESULTS FOR DIFFERENT STRATEGIES APPLIED TO THE SAME PATH



III. The velocity, acceleration and jerk profiles are reported in Table II, where it can be seen how the jerk is considerably lower for strategy D, while it can reach considerably high values for strategies A and B. Moreover, in order to show how jerk-limited trajectories can contribute to reducing the vibrations of the overall system, we used an IMU placed on the same structure where the manipulator is mounted, a cube of  $2m \times 2m \times 2m$  with Misumi aluminum beams, as highlighted in figure 4(a), to measure the accelerations due to the motion of the manipulator. The data provided by the IMU reported in

Table II highlights the stated reduction of structural stresses if a jerk-limited trajectory is used, clearly at the expense of the task accomplishment time. Statistics about these measurements are reported in the Table III, where we have a 62% reduction on the peak value between our approach and strategy A and overall improvements with respect to the other strategies.



TABLE III  
NUMERICAL RESULTS FOR EACH STRATEGY

	$\max_i  \ddot{q}_i $	$S(\ddot{q})$	$\max \Delta a_{IMU}$	$\overline{\Delta a}_{IMU}$	$\sigma(\Delta a_{IMU})$
<b>A</b>	1918.1	20788	0.7475	0.0864	0.0926
<b>B</b>	959.80	15610	0.4309	0.0703	0.0654
<b>C</b>	641.19	14275	0.3530	0.0658	0.0622
<b>D</b>	252.45	9201	0.2816	0.0577	0.0477

The results show that using our approach, it is possible to reduce the maximum jerk and obtain lower value for the introduced function  $S$ . Values for the maximum value, mean and standard deviation of the IMU measurements are also reported.

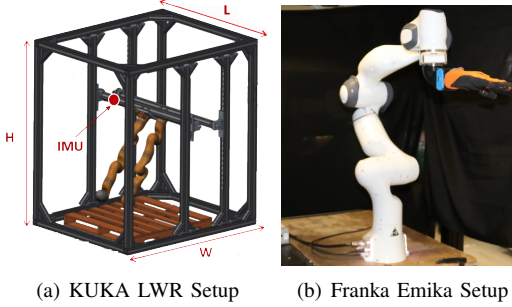


Fig. 4. Placement of the IMU on the structure using KUKA LWR robot (a) and experimental setup using Franka Emika Panda Robot (b)

#### D. Experimental Results on a Franka Emika Panda 7-DoF Manipulator

Further experiments have been conducted using a Franka Emika Panda 7-DoF Manipulator. We tested four different trajectories using strategies B, C and our approach D. We set the velocity and acceleration limits as 60% of their maximum values, while for the jerk we set a constraint equal to 0.6% of its maximum allowed. In these the tests we wanted to compare the average performance in terms of reduced jerk and smoothness for strategy C and D. The numerical results obtained are reported in Table IV, where we compared the two strategies to strategy B, which does not include any type of jerk limitation/reduction. We can see how our approach has better performance in terms of peak jerk and lower values of function  $S$ , at the expense of a larger time needed to complete the tasks. Results for Strategy A are not reported since the robot would stop due to violations of the acceleration limits. The accompanying video shows the execution of all presented trajectories.

## VI. CONCLUSIONS

In this paper, we presented a motion planning techniques to generate optimal jerk limited trajectories. Based on the formulation of a minimum-time path tracking problem with the jerk as control input, it has been stated that the resulting non-convex NLP problem can be transformed in a way to allow an efficient and reliable convex relaxation using McCormick Envelopes. The simulations and experimental tests carried out on two real 7-DoF manipulators demonstrate the improvements of the stated approach compared to other state-of-the-art methods. The tests showed how, using SCIP as solver, the time needed to compute the solution depends on the number of points used and on how tight the constraint on the jerk is, with times that vary from a minimum of 1.3 seconds to

TABLE IV  
RELATIVE CHANGES FOR A SET OF TRAJECTORIES USING STRATEGY C AND D TO REDUCE JERK COMPARED TO STRATEGY B

	Peak Jerk	$S(\ddot{q})$	Trajectory Duration
<b>C</b>	-13.2%	-7.42%	+7.23%
<b>D</b>	-44.6%	-33.9%	+22.3%

Using our approach, the peak Jerk is considerably reduced, at the expense of an increased time needed to complete the task.

a maximum of 5467.6 seconds using a Laptop PC equipped with Intel Core i7 Processor (4x2.80 GHz) and 32 GB DDR4 RAM. Experiments with different and powerful solvers are deferred to future works. Further investigation will be carried out, in order to apply this approach to Soft-Robots, extending the formulation for the case where jerk derivatives are used as control inputs in the time-optimal path tracking problem.

## APPENDIX

The formulation stated in Section III neglects the robot dynamic model. In this section, a way to model the problem including robot dynamics is carried out, discussing a formulation of the non-convex NLP that allows an appealing convex relaxation, similar to the one proposed in this paper.

### A. Including the Robot Dynamic Model

The equation of motion of a  $n$ -DoF robot with configuration vector  $\mathbf{q} \in \mathbb{R}^n$  can be written as [1]:

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}), \quad (10)$$

where  $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n \times n}$ ,  $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n \times n}$  and  $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^n$  are the positive definite mass matrix, the Coriolis matrix and the gravity vector, respectively. Following the same procedure reported in Section III, we use the torque rate,  $\dot{\boldsymbol{\tau}}$ , as limited control input, since we want to obtain a continuous and smooth torque profile. This rate is expressed as:

$$\begin{aligned} \dot{\boldsymbol{\tau}} = & \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \left( \sum_{i=1}^n \frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i} \dot{\mathbf{q}}_i \right) \ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \\ & \dot{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\dot{\mathbf{q}} + \dot{\mathbf{G}}(\mathbf{q}, \dot{\mathbf{q}}). \end{aligned} \quad (11)$$

Using (1) to highlight the dependency from  $s$  and knowing that:

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \left( \sum_{i=1}^n \frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i} \dot{\mathbf{q}}_i \right) \dot{\mathbf{q}} - \frac{1}{2} \dot{\mathbf{q}}^T \frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}} \quad (12)$$

after some algebraic manipulation, (11) can be written as:

$$\begin{aligned} \dot{\boldsymbol{\tau}}(s) = & K_1(s)\ddot{s} + K_2(s)\dot{s}\ddot{s} + K_3(s)\ddot{s} + \\ & K_4(s)\dot{s}^3 + K_5(s)\dot{s}^2 + K_6(s)\dot{s}, \end{aligned} \quad (13)$$

where  $K_i(s)$  are functions of the path coordinate  $s$  and the path  $q(s)$  and its derivatives with respects to  $s$ ; their analytic forms are not reported for the sake of brevity. Using the same nonlinear change of variables stated in (4) we can obtain:

$$\begin{aligned} \dot{\boldsymbol{\tau}}(s) = & K_1(s)c(s)\sqrt{b(s)} + K_2(s)a(s)\sqrt{b(s)} + \\ & K_3(s)a(s) + K_4(s)b(s)\sqrt{b(s)} + \\ & K_5(s)b(s) + K_6(s)\sqrt{b(s)}. \end{aligned} \quad (14)$$

which is clearly nonlinear. Introducing three auxiliary functions and the associated constraints:

$$D(s) = \sqrt{b(s)}, \quad v(s) = \frac{\dot{\tau}(s)}{D(s)}, \quad M(s) = \frac{a(s)}{D(s)}, \quad D(s) > 0,$$

the new control input  $v(s)$  is now affine in the expanded optimization variables set. Using direct transcription to solve the resulting NLP problem numerically we obtain:

$$\left\{ \begin{array}{l} \min_{a^k, b^k, c^k, v^k, D^k, M^k} \sum_{k=0}^{N-1} \int_{s^k}^{s^{k+1}} \frac{1}{\sqrt{b(s)}} ds \\ \text{subject to} \\ \mathbf{v}^k = K_1(s^k)c^k + K_2(s^k)a^k + K_3(s^k)M^k + K_4(s^k)b^k + \\ \quad K_5(s^k)D^k + K_6(s^k) \\ b^0 = \dot{s}_0^2 \text{ and } b^N = \dot{s}_T^2 \\ b^k > 0 \text{ and } b^N > 0 \\ D^k > 0 \text{ and } D^N > 0 \\ (b^{k+1} - b^k) = 2a^k \Delta s^k \\ a^0 = \dot{s}_0 \text{ and } a^N = \dot{s}_T \\ a^{k+1} - a^k = c^k \Delta s^k \text{ and } a^N - a^{N-1} = c^N \Delta s^{N-1} \\ b^k \leq \bar{b}(s^k) \\ (D^k)^2 = b^k \text{ and } (D^N)^2 = b^N \\ M^k D^k = a^k \text{ and } M^N D^N = a^N \quad (*) \\ \boldsymbol{\tau}(s^k) \leq \tilde{m}(s^k)\mathbf{a}^k + \tilde{c}(s^k)\mathbf{b}^k + \tilde{g}(s^k) \leq \bar{\boldsymbol{\tau}}(s^k) \quad (*+) \\ \mathbf{u}(s^k) \leq \mathbf{v}^k D^k \leq \bar{\mathbf{u}}(s^k) \quad (+) \end{array} \right.$$

where in  $(*+)$  the formulation presented in [6] for  $\tau(s)$  has been used. It can be noted that the non-convex constraints are the bilinear ones,  $(+)$  and  $(*)$ , as for the problem stated in Section IV. The considerations made about convex relaxation for these types of constraints still hold.

Using this formulation is, therefore, possible to write a non-convex NLP problem with torque rate as bounded control input. A convex relaxation of the nonlinear constraints can be efficiently performed, and the globally optimal solution can be computed.

Applying the same considerations made in [7], this analysis can be extended to systems with dynamics:

$$\mathbf{R}(\mathbf{q})\mathbf{u} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}),$$

where  $\mathbf{R}(\mathbf{q}) \in \mathbb{R}^{p \times n}$  is the control matrix of the system and  $p$  is the dimension of the control vector.

## REFERENCES

- [1] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.
- [2] F. Flacco and A. De Luca. Unilateral constraints in the reverse priority redundancy resolution method. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2564–2571, Sep. 2015.
- [3] F. Flacco, A. De Luca, and O. Khatib. Motion control of redundant robots under joint constraints: Saturation in the null space. In *2012 IEEE International Conference on Robotics and Automation*, pages 285–292, May 2012.
- [4] Fabrizio Flacco and Alessandro De Luca. Discrete-time redundancy resolution at the velocity level with acceleration/torque optimization properties. *Robotics and Autonomous Systems*, 70, 03 2015.
- [5] Enrico Mingo Hoffman, Alessio Rocchi, Nikos Tsagarakis, and Darwin G. Caldwell. *Robot Dynamics Constraint for Inverse Kinematics*, pages 275–283. Springer International Publishing, 01 2018.
- [6] D. Verschuere, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl. Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control*, 54(10):2318–2327, Oct 2009.
- [7] Thomas Lipp and Stephen P. Boyd. Minimum-time speed optimisation over a fixed path. *Int. J. Control*, 87:1297–1311, 2014.
- [8] S. Macfarlane and E. A. Croft. Jerk-bounded manipulator trajectory planning: design for real-time applications. *IEEE Transactions on Robotics and Automation*, 19(1):42–52, Feb 2003.
- [9] F. Flacco and A. De Luca. A reverse priority approach to multi-task control of redundant robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2421–2427, Sep. 2014.
- [10] F. Pfeiffer and R. Johanni. A concept for manipulator trajectory planning. *IEEE Journal on Robotics and Automation*, 3(2):115–123, April 1987.
- [11] J.E. Bobrow, S. Dubowsky, and J.S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotics Research*, 4(3):3–17, 1985 .
- [12] W. Van Loock, G. Pipeleers, and J. Swevers. Time-optimal quadrotor flight. In *2013 European Control Conference (ECC)*, pages 1788–1792, July 2013 .
- [13] Lorenz T. Biegler. Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation. *Computers & Chemical Engineering*, 8(3):243 – 247, 1984 .
- [14] John J Craig. *Introduction to robotics: mechanics and control*, 3/E. Pearson Education India, 2009.
- [15] Werther Serralheiro, Newton Maruyama, and Fabrício Saggin. Self-tuning time-energy optimization for the trajectory planning of a wheeled mobile robot. *Journal of Intelligent & Robotic Systems*, Sep 2018.
- [16] Yu Zhang, Huiyan Chen, Steven L. Waslander, Tian Yang, Sheng Zhang, Guangming Xiong, and Kai Liu. Toward a more complete, flexible, and safer speed planning for autonomous driving via convex optimization. *Sensors*, 18(7), 2018.
- [17] Pham Tien Hung and Quang-Cuong Pham. On the structure of the time-optimal path parameterization problem with third-order constraints. *CoRR*, abs/1609.05307, 2016 .
- [18] D. Constantinescu and E. Croft. Smooth and time-optimal trajectory planning for industrial manipulators along specified path. *Journal of Robotic Systems - J ROBOTIC SYST*, 17:233–249, 05 2000.
- [19] L. Liberti and C. C. Pantelides. Convex envelopes of monomials of odd degree. *Journal of Global Optimization*, 25(2):157–168, Feb 2003.
- [20] Ruth Misener and Christodoulos A. Floudas. Antigone: Algorithms for continuous / integer global optimization of nonlinear equations. *Journal of Global Optimization*, 59(2):503–526, Jul 2014.
- [21] Garth P. McCormick. Computability of global solutions to factorable nonconvex programs: Part i — convex underestimating problems. *Mathematical Programming*, 10(1):147–175, Dec 1976.
- [22] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.
- [23] J. Lofberg. Yalmip : a toolbox for modeling and optimization in matlab. In *2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508)*, pages 284–289, Sep. 2004.
- [24] Tobias Achterberg. Scip: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, Jul 2009.
- [25] J. Currie and D. I. Wilson. OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User. In Nick Sahinidis and Jose Pinto, editors, *Foundations of Computer-Aided Process Operations*, Savannah, Georgia, USA, 8–11 January 2012.
- [26] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.