OPTIMAL CONTROL APPLICATIONS AND METHODS
Optim. Control Appl. Meth. (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/oca.2234

A convex relaxation for the time-optimal trajectory planning of robotic manipulators along predetermined geometric paths

P. Reynoso-Mora*,†, W. Chen‡ and M. Tomizuka

Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720, USA

SUMMARY

In this paper, we deal with the problem of time-optimal trajectory planning and feedforward controls for robotic manipulators along predetermined geometric paths. We propose a convex relaxation to generate time-optimal trajectories and feedforward controls that are dynamically feasible with respect to the complete nonlinear dynamic model, considering both Coulomb friction and viscous friction. Even though the effects of viscous friction for time-optimal motions become rather significant due to the required large speeds, in previous formulations, viscous friction was ignored. We present a strategic formulation that turns out non-convex because of the consideration of viscous friction, which nonetheless leads naturally to a convex relaxation of the referred non-convex problem. In order to numerically solve the proposed formulation, a discretization scheme is also developed. Importantly, for all the numerical instances presented in the paper, focusing on applying the algorithm results to a six-axis industrial manipulator, the proposed convex relaxation solves exactly the original non-convex problem. Through simulations and experimental studies on the resulting tracking errors, torque commands, and accelerometer readings for the six-axis manipulator, we emphasize the importance of penalizing a measure of total jerk and of imposing acceleration constraints at the initial and final transitions of the trajectory. Copyright © 2016 John Wiley & Sons, Ltd.

Received 3 June 2015; Revised 27 December 2015; Accepted 28 December 2015

KEY WORDS: robotic manipulators; time-optimal trajectories; nonlinear dynamic model; convex optimization

1. INTRODUCTION

In many industrial applications of robot manipulators, time-optimality is crucial for maximizing robot productivity. Typical applications, such as palletizing and pick-and-place, require an operator to specify a collision-free geometric path that the robot must follow in order to accomplish a particular task. This path specification is usually performed through a so-called teach-pendant or through a path-planning algorithm [1]. Once the geometric path has been specified, it is important to find out how to move the robot optimally along that path in the shortest time physically possible. Solving this optimal trajectory planning problem entails obtaining as a function of time the trajectories and feedforward controls, which guarantee motion along the path in minimum time while satisfying the manipulator's nonlinear dynamic model without exceeding the physical torque limits of the actuators. We shall refer to the property of satisfying the nonlinear robot model as dynamic feasibility of the trajectories and feedforward controls [2].

First successful approaches to address this problem date back to the 1980s. Pioneering work is presented in the classic papers [3–5], where the referred problem of time-optimal trajectory planning is addressed for manipulators with n degrees of freedom (DOF). In [3], it was suggested that time-

*Correspondence to: Pedro Reynoso-Mora, currently a Sr. Control Systems Engineer at Nikon Research Corporation of America, Belmont, CA 94002, USA.

†E-mail: preynoso@berkeley.edu

‡Currently a Sr. Research Engineer, Learning Robot Development Department, Robot Laboratory, FANUC Corporation, Yamanashi-ken, Japan.

optimal solutions are found by choosing the acceleration to make the velocity as large as possible at every point without violating the constraints. It was then shown that to minimize time, the acceleration always takes either its largest or its smallest possible value, the so-called bang–bang control. Therefore, search algorithms were proposed that find switching points in the so-called velocity limit curve. At these switching points, instant changes from maximum acceleration to maximum deceleration and vice versa must occur. In [4], additional properties of the velocity limit curve are rigourously analyzed, which allowed for simplification in the computation of the switching points. The main disadvantage of these algorithms is that pure time-optimal solutions produce accelerations and torques that require sudden changes, which are impossible to handle by real servo-amplifiers. Besides, these kind of bang–bang time-optimal solutions are bound to excite undesired vibration modes, which are inherent to the mechanical structure.

Optimal control methods have also been utilized to tackle the referred time-optimal trajectory planning problem, which in principle make it possible to incorporate additional criteria that generate more feasible solutions for implementation. In [5], additional criteria are added to trade off time-optimality against squared velocity and joint torques. To solve the optimal control problem, a dynamic programming approach was adopted. In [6], time-optimality is traded off against a term that represents control energy; the method to solve the referred problem is based on the necessary conditions given by Pontryagin's maximum principle, which then requires solving a 2-point boundary value problem using shooting methods.

Over the past decades, a prominent approach to address numerical optimal control problems has gained popularity, mainly because it makes it possible to obtain numerical solutions for seemingly complex problems [7]. In this approach, the optimal control problem is directly discretized to obtain a large-scale nonlinear optimization problem, allowing the incorporation of more realistic equality and inequality constraints that are easily handled by nonlinear optimization methods. In the specific context of time-optimal trajectory planning along predefined paths, in [8], the problem is formulated as a large-scale nonlinear optimization problem. Attention is paid to the adverse effects on performance induced by pure time-optimal solutions. The formulation in [8] is a general nonlinear optimization, and therefore, there are no guarantees for finding the global minimum.

More recently, motivated by the widespread reputation of convex optimization to efficiently solve engineering and science problems, a modern formulation of the referred problem of time-optimal trajectory planning was proposed in [9], where theory and tools from convex optimization are utilized. The advantage of formulating a problem as a convex optimization problem is twofold: (i) theoretical and conceptual advantages, for example, any local minimum is a global minimum of the problem and (ii) the problem can be solved reliably and efficiently using mature interior-point methods or other specialized methods for convex optimization [10].

A main shortcoming of the approach presented in [9] is that viscous friction is neglected in order to have a convex formulation. This drawback has already been pointed out in [11], although no solution is proposed other than assuming instead that the full dynamic model is linear, which is clearly not true at all for robotic manipulators. It is also well known that time-optimal trajectories represent the fastest motions achievable by the physical system, which require the manipulator to move at really high speeds. This means that the effects of viscous friction, which are proportional to velocity, will be rather significant (even more significant than Coulomb friction). Therefore, in order to truly obtain the fastest solutions dynamically feasible for the real robotic system, viscous friction should not be neglected in the problem formulation.

In this paper, we present contributions to the referred problem of time-optimal trajectory planning along predetermined geometric paths. We attempt to incorporate the complete nonlinear dynamic model, considering the effects of both Coulomb friction as well as viscous friction. Motivated by theoretical properties and efficient algorithms to solve convex optimization problems, we explicitly pursue a convex formulation, which improves and expands the scope of the formulation presented in [9]. We construct the initial formulation in a specific manner that results in a non-convex optimization problem, which nevertheless leads to intuitively propose a convex relaxation that seems likely to find the optimal solution to the original non-convex problem. In order to numerically solve the proposed convex relaxation, a discretization scheme is developed using methods from numerical optimal control. Importantly, for all the numerical instances presented in this paper, the proposed convex relaxation solves exactly the original non-convex problem. We then demonstrate that large nonzero accelerations at the beginning and end of the trajectory, which are required by pure time-optimal solutions, would seriously degrade the system performance near those time instants. Therefore, acceleration constraints are incorporated that guarantee smooth transitions from 0 and to 0 at the beginning and at the end of the trajectory, respectively. Likewise, it is shown how to incorporate a term that penalizes a measure of total jerk to trade off time-optimality, which generates near time-optimal trajectories with no sudden changes at the intermediate points.

The final formulation with acceleration constraints and penalization of total jerk is also a convex problem, which still represents a convex relaxation to the original non-convex problem. Mercifully, for all the numerical examples presented in this paper and the many more tried out by the authors, this convex relaxation turns out to solve exactly the original non-convex problem, entailing the generation of near time-optimal trajectories that are dynamically feasible with respect to the full nonlinear dynamic model including viscous friction. This suggests our formulation as a powerful heuristic that shall often find the optimal solution to the referred non-convex problem. The near time-optimal trajectories and feedforward torques are feasible for implementation on the real robotic manipulator without seriously degrading the system performance, at the expense of a modest increase in traveling time. Experimental results on a six-axis industrial manipulator are finally presented to verify the concrete benefits of our formulation.

2. DYNAMIC MODEL OF ROBOTIC MANIPULATORS

The dynamic model of a robot manipulator can be systematically derived with a Lagrangian formalism. For a manipulator with n joints, a set of coordinates q_i, $i = 1, \ldots, n$, known as the generalized coordinates, is chosen to effectively describe the link positions of this n DOF manipulator [12]. For a serial-chain industrial manipulator, q_i represents the relative angle of link i with respect to link $i-1$. Likewise, the generalized forces Q_i are given by the nonconservative forces, that is, the joint torques generated by the actuators as well as the joint torques due to friction [13]. By considering viscous and Coulomb friction, therefore, $Q_i = \tau_i - d_{vi}\dot{q}_i - f_{ci}\,\mathrm{sign}\,(\dot{q}_i)$, where d_{vi} and f_{ci} are the coefficients of viscous and Coulomb friction and τ_i represents the joint torque generated by the i-th actuator.

Throughout this paper, the following vector differential equation, which is given as the standard dynamic model in most robotics textbooks [12, 14], shall be used as the nonlinear dynamic model for an n DOF robotic manipulator:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{D}_{\mathrm{v}}\dot{\boldsymbol{q}} + \boldsymbol{F}_{\mathrm{C}}\,\mathrm{sign}\,(\dot{\boldsymbol{q}}) = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$ are the joint-space positions, velocities, and accelerations; $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the positive-definite inertia matrix; and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal forces matrix; $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ represents the vector of torques due to gravity; diagonal matrices $\boldsymbol{D}_{\mathrm{v}} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{F}_{\mathrm{C}} \in \mathbb{R}^{n\times n}$ represent the coefficients of viscous and Coulomb friction, respectively.

On the other hand, it is a standard exercise in Lagrangian dynamics to show that the equations of motion (1) are equivalently written in scalar form [12, 13]:

$$\sum_{j=1}^{n} m_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j + g_i(\boldsymbol{q}) = \tau_i - d_{vi}\dot{q}_i - f_{ci}\,\mathrm{sign}\,(\dot{q}_i), \quad i = 1, \ldots, n, \tag{2}$$

where $m_{ij}(\boldsymbol{q})$ is the (i, j)-th element of $\boldsymbol{M}(\boldsymbol{q})$, $c_{ijk}(\boldsymbol{q})$ are known as the Christoffel symbols of the first kind, and $g_i(\boldsymbol{q})$ is the i-th element of the gravity vector. The equations of motion in scalar form (2) shall prove useful later, when showing a certain result of the time-optimal trajectory planning problem, which would otherwise seem rather obscure.

3. PROBLEM STATEMENT AND FORMULATION

In order to generate trajectories and feedforward controls that are dynamically feasible, we consider the full nonlinear dynamic model in vector form (1), or equivalently, in scalar form (2). Assume that the joint-space geometric path has already been determined and parameterized by $\boldsymbol{h}(s) \in \mathbb{R}^n$, where s is a normalized and monotonically increasing parameter, that is, $s(t) \in [0, 1]$ and $\dot{s} > 0$. It should be required that $\boldsymbol{h}(s)$ be twice continuously differentiable in s. The goal is to figure out how to move the robot along $\boldsymbol{h}(s)$ in the shortest time physically possible. Equivalently, obtain the time histories of desired position, velocity, and acceleration trajectories $(\boldsymbol{q}_{\mathrm{d}}(t), \dot{\boldsymbol{q}}_{\mathrm{d}}(t), \ddot{\boldsymbol{q}}_{\mathrm{d}}(t))$, and the corresponding feedforward controls $\boldsymbol{\tau}_{\mathrm{d}}(t)$, which are dynamically feasible with respect to the complete robot dynamics (1), achieving motion along $\boldsymbol{h}(s)$ in the shortest time physically possible while satisfying the torque limit constraints: $\boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}_{\mathrm{d}}(t) \leqslant \boldsymbol{\tau}_{\max}$.

3.1. Formulation as a mathematical optimization

From the equality constraint $\boldsymbol{q}_{\mathrm{d}} = \boldsymbol{h}(s)$, it is readily shown that

$$\dot{\boldsymbol{q}}_{\mathrm{d}} = \boldsymbol{h}'(s)\dot{s}, \quad \ddot{\boldsymbol{q}}_{\mathrm{d}} = \boldsymbol{h}'(s)\dot{s}^2 + \boldsymbol{h}'(s)\ddot{s}, \tag{3}$$

where $\boldsymbol{h}'(s) := d\boldsymbol{h}/ds$, $\boldsymbol{h}''(s) := d^2\boldsymbol{h}/ds^2$. Notice that $\dot{s} = ds/dt$ and $\ddot{s} = d^2s/dt^2$ could be thought of as the pseudo-speed and pseudo-acceleration along the path, respectively. Because a solution $(\boldsymbol{q}_{\mathrm{d}}, \dot{\boldsymbol{q}}_{\mathrm{d}}, \ddot{\boldsymbol{q}}_{\mathrm{d}}, \text{and} \boldsymbol{\tau}_{\mathrm{d}})$ must be dynamically feasible with respect to (1), we must have

$$\begin{aligned}\boldsymbol{\tau}_{\mathrm{d}} = {} & \boldsymbol{M}(\boldsymbol{h}(s))\left[\boldsymbol{h}''(s)\dot{s}^2 + \boldsymbol{h}'(s)\ddot{s}\right] + \boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\dot{s}\right)\boldsymbol{h}'(s)\dot{s} + \boldsymbol{g}(\boldsymbol{h}(s)) \\ & + \boldsymbol{D}_{\mathrm{v}}\boldsymbol{h}'(s)\dot{s} + \boldsymbol{F}_{\mathrm{C}}\,\mathrm{sign}\,(\boldsymbol{h}'(s)\dot{s}).\end{aligned} \tag{4}$$

Because dynamic models (1) and (2) are equivalent, the i-th element of $\boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\dot{s}\right)\boldsymbol{h}'(s)\dot{s} \in \mathbb{R}^n$ is written as follows:

$$\begin{aligned}\left[\boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\dot{s}\right)\boldsymbol{h}'(s)\dot{s}\right]_i &= \sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}(\boldsymbol{h})h'_k\dot{s}h'_j\dot{s} \\ &= \left(\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}(\boldsymbol{h})h'_k h'_j\right)\dot{s}^2, \quad i = 1, \dots, n,\end{aligned}$$

from which it should be clear that $\boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\dot{s}\right)\boldsymbol{h}'(s)\dot{s} = \boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\right)\boldsymbol{h}'(s)\dot{s}^2$. On the other hand, because $\dot{s} > 0$, then $\boldsymbol{F}_{\mathrm{C}}\,\mathrm{sign}\,(\boldsymbol{h}'(s)\dot{s}) = \boldsymbol{F}_{\mathrm{C}}\,\mathrm{sign}\,(\boldsymbol{h}'(s))$. Therefore, (4) is written as follows:

$$\boldsymbol{\tau}_{\mathrm{d}} = \boldsymbol{a}_1(s)\ddot{s} + \boldsymbol{a}_2(s)\dot{s}^2 + \boldsymbol{a}_3(s)\dot{s} + \boldsymbol{a}_4(s), \tag{5}$$

where $\boldsymbol{a}_i(s) \in \mathbb{R}^n$, $i = 1, \dots, 4$, are defined as follows:

$$\begin{aligned}&\boldsymbol{a}_1(s) := \boldsymbol{M}\left(\boldsymbol{h}(s)\right)\boldsymbol{h}'(s), && \boldsymbol{a}_2(s) := \boldsymbol{M}\left(\boldsymbol{h}(s)\right)\boldsymbol{h}''(s) + \boldsymbol{C}\left(\boldsymbol{h}(s), \boldsymbol{h}'(s)\right)\boldsymbol{h}'(s) \\ &\boldsymbol{a}_3(s) := \boldsymbol{D}_{\mathrm{v}}\boldsymbol{h}'(s), && \boldsymbol{a}_4(s) := \boldsymbol{F}_{\mathrm{C}}\,\mathrm{sign}\,\left(\boldsymbol{h}'(s)\right) + \boldsymbol{g}\left(\boldsymbol{h}(s)\right).\end{aligned} \tag{6}$$

Because $\boldsymbol{h}(s)$, $\boldsymbol{h}'(s)$, and $\boldsymbol{h}''(s)$ are already known, then $\boldsymbol{a}_i(s)$, $i = 1, \dots, 4$ can be entirely pre-computed. The unknowns in parametrization (5) are $\ddot{s}$, $\dot{s}^2$, $\dot{s}$, and $\boldsymbol{\tau}_{\mathrm{d}}$, which means we can optimize over these functions so as to minimize the total traveling time along $\boldsymbol{h}(s)$. Consider therefore defining $a(s) := \ddot{s}$, $b(s) := \dot{s}^2$, $c(s) := \dot{s}$, and $\boldsymbol{\tau}_{\mathrm{d}}(s)$, as optimization functions to be determined as functions of s. Because a one-to-one relationship between s and t shall be enforced (i.e., $\dot{s} > 0$), finding the unknown functions of s implies that they can be unambiguously recovered as functions of t. In this manner, $\boldsymbol{\tau}_{\mathrm{d}}(s)$ has a simple affine parametrization in $a(s)$, $b(s)$, and $c(s)$:

$$\boldsymbol{\tau}_{\mathrm{d}}(s) = \boldsymbol{a}_1(s)a(s) + \boldsymbol{a}_2(s)b(s) + \boldsymbol{a}_3(s)c(s) + \boldsymbol{a}_4(s). \tag{7}$$

Notice that with the definitions of $a(s)$, $b(s)$, and $c(s)$, two additional constraints must be incorporated: (i) $\dot{b}(s) = b'(s)\dot{s} = 2\dot{s}\ddot{s} \Leftrightarrow b'(s) = 2a(s)$ if $\dot{s} > 0$ and (ii) $c(s) = \sqrt{b(s)}$.

On the other hand, using the fact that $\dot{s} > 0$, the total traveling time t_f can be expressed as follows:

$$t_f = \int_0^{t_f} \mathrm{d}t = \int_0^1 \left(\frac{\mathrm{d}s}{\mathrm{d}t}\right)^{-1} \mathrm{d}s = \int_0^1 \frac{1}{c(s)}\, \mathrm{d}s. \tag{8}$$

It follows from (8) that to minimize the traveling time t_f, the pseudo-speed $c(s)$ along the path must be as large as possible. Because several of the aforementioned derivations required $\dot{s} > 0$, we should treat carefully the case when $\dot{s}_0 = \dot{s}_f = 0$, in which case the objective functional (8) is unbounded previously. Therefore, the integral in (8) is defined instead in the interval $[0_+, 1_-]$, where 0_+ and 1_- will be explicitly defined in Section 4; notice that $c(0_+) \neq 0$ and $c(1_-) \neq 0$ even if $c(0) = c(1) = 0$.

With all the aforementioned considerations, we formulate the referred time-optimal trajectory planning problem with full dynamic model (1), as the following mathematical optimization problem:

$$\begin{aligned}
\text{minimize} \quad & \int_{0_+}^{1_-} \frac{1}{c(s)}\, \mathrm{d}s \\
\text{subject to} \quad & b(0) = \dot{s}_0^2, b(1) = \dot{s}_f^2, c(0) = \dot{s}_0, c(1) = \dot{s}_f \\
& \boldsymbol{\tau}_{\mathrm{d}}(s) = \boldsymbol{a}_1(s)a(s) + \boldsymbol{a}_2(s)b(s) + \boldsymbol{a}_3(s)c(s) + \boldsymbol{a}_4(s) \\
& \boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}_{\mathrm{d}}(s) \leqslant \boldsymbol{\tau}_{\max} \\
& \forall s \in [0, 1] \\
& b'(s) = 2a(s), c(s) = \sqrt{b(s)} \\
& b(s), c(s) > 0 \\
& \forall s \in [0_+, 1_-],
\end{aligned} \tag{9}$$

where $a(s)$, $b(s)$, $c(s)$, and $\boldsymbol{\tau}_{\mathrm{d}}(s)$ are the optimization functions and therefore the rest is data. Optimization problem (9) shall be referred to as infinite dimensional because s varies continuously in $[0, 1]$, implying an infinite number of equality and inequality constraints. It can also be viewed as an optimal control problem [15], with control input $a(s)$, linear differential constraint $b'(s) = 2a(s)$, and algebraic state equalities and inequality constraints.

The following properties of formulation (9) are important to point out: (i) The objective functional is convex in $c(s)$;[‡] (ii) The constrained robot dynamics and all inequality constraints are affine; (iii) The differential equality constraint $b'(s) = 2a(s)$ is linear; and (iv) The only nonlinear equality constraint is $c(s) = \sqrt{b(s)}$ $\forall s \in [0_+, 1_-]$. It is therefore deduced that formulation (9) is a non-convex optimization problem [10]. The non-convexity of (9) is due to the nonlinear equality constraint $c(s) = \sqrt{b(s)}$ $\forall s \in [0_+, 1_-]$, which shows up because of viscous friction.

3.2. *Convex relaxation*

Relaxing a problem to obtain a convex approximation is a viable strategy to deal with non-convex problems. Relaxing the problem simply increases the feasible search region. If, even given this larger region, the optimal solution is found in the original region, then the solution to the relaxed problem is a solution to the unrelaxed problem. Notice that because $b(s), c(s) > 0$ for all $s \in [0_+, 1_-]$, the following chain of equivalences is always true:[§]

[‡]Consider the function $f(c) = 1/c, c > 0$, which is trivially a convex function of c. Because the non-negative weighted sum of convex functions is convex [10], it follows that the integral in problem (9) defines a convex objective functional.

[§]We have used the simple fact that, satisfying any equality constraint, say $f_1(x) = f_2(x)$ is equivalent to satisfying two inequality constraints, namely, $f_1(x) = f_2(x) \Leftrightarrow f_1(x) \leqslant f_2(x)$ and $f_1(x) \geq f_2(x)$.

$$\begin{aligned}\sqrt{b(s)} = c(s) &\Leftrightarrow \frac{1}{\sqrt{b(s)}} = \frac{1}{c(s)} \\ &\Leftrightarrow \frac{1}{\sqrt{b(s)}} \leqslant \frac{1}{c(s)} \text{ and } \frac{1}{c(s)} \leqslant \frac{1}{\sqrt{b(s)}} \\ &\Leftrightarrow c(s)^2 - b(s) \leqslant 0 \text{ and } -c(s)^2 + b(s) \leqslant 0.\end{aligned} \tag{10}$$

The inequality constraints $c(s)^2 - b(s) \leqslant 0$ are all convex in $b(s)$ and $c(s)$ for all $s \in [0_+, 1_-]$.¶ On the other hand, all the inequality constraints $-c(s)^2 + b(s) \leqslant 0$ are concave in $b(s)$ and $c(s)$. In an optimization problem, concave inequality constraints imply that the optimization problem is non-convex. Therefore, the proposed convex relaxation of (9) consists in simply dropping all the concave inequality constraints in (10). In other words, we replace the nonlinear equality constraints $c(s) = \sqrt{b(s)}\ \forall s \in [0_+, 1_-]$ in (9) with the convex inequality constraints $1/\sqrt{b(s)} \leqslant 1/c(s)\ \forall s \in [0_+, 1_-]$. Therefore, the following convex relaxation of (9) is obtained:

$$\begin{aligned} \text{minimize} \quad & \int_{0_+}^{1_-} \frac{1}{c(s)}\,\mathrm{d}s \\ \text{subject to} \quad & b(0) = \dot{s}_0^2, b(1) = \dot{s}_f^2, c(0) = \dot{s}_0, c(1) = \dot{s}_f \\ & \boldsymbol{\tau}_\mathrm{d}(s) = \boldsymbol{a}_1(s)a(s) + \boldsymbol{a}_2(s)b(s) + \boldsymbol{a}_3(s)c(s) + \boldsymbol{a}_4(s) \\ & \boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}_\mathrm{d}(s) \leqslant \boldsymbol{\tau}_{\max} \\ & \forall s \in [0, 1] \\ & b'(s) = 2a(s), 1/\sqrt{b(s)} \leqslant 1/c(s) \\ & b(s), c(s) > 0 \\ & \forall s \in [0_+, 1_-]. \end{aligned} \tag{11}$$

In general, there are no guarantees that for all possible scenarios, the optimal solution $\left(a^\star(s), b^\star(s), c^\star(s), \boldsymbol{\tau}_\mathrm{d}^\star(s)\right)$ to convex relaxation (11) will solve exactly the original non-convex problem (9). Nonetheless, the reasoning behind proposing convex relaxation (11) goes as follows: Because $1/c(s) > 0$ for all $s \in [0_+, 1_-]$, and because the integral over $[0_+, 1_-]$ of $1/c(s)$ is minimized, then at optimum, the inequality constraint $1/\sqrt{b(s)} \leqslant 1/c(s)$ is likely to be active. That is, it is likely that $1/\sqrt{b^\star(s)} = 1/c^\star(s)$ for all $s \in [0_+, 1_-]$. In fact, with the aforementioned argument, the referred inequality constraint would indeed be tight in general for all $s \in [0_+, 1_-]$ if $c(s)$ were not constrained to satisfy the equality constraint $\boldsymbol{\tau}_\mathrm{d}(s) = \boldsymbol{a}_1(s)a(s) + \boldsymbol{a}_2(s)b(s) + \boldsymbol{a}_3(s)c(s) + \boldsymbol{a}_4(s)$, which is actually the only additional constraint on $c(s)$. As it turns out, for all the numerical instances reported in this paper and the several more tried out by the authors, exact tightness of the inequality constraint $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$ is attained for all $s \in [0_+, 1_-]$.

Motivated by the aforementioned observations, in Section 6, we choose to incorporate acceleration constraints and a measure of total jerk penalization into problem (11), which shall cope with the drawbacks of pure time-optimal solutions. Importantly, these additions do not explicitly impose more equality or inequality constraints to satisfy by $c(s)$. Clearly, when adding more constraints into problem (11), the feasible search region is reduced, which decreases the likelihood of tightness for $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$ even if $c(s)$ does not explicitly feature in the additional constraints. Nevertheless, for our particular case scenarios, it was observed that when adding constraints into problem (11) that explicitly featured $c(s)$, exact tightness of $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$ was no longer attained. In contrast, when adding constraints that do not explicitly featured $c(s)$, such as the referred acceleration constraints and penalization of total jerk, exact tightness was still achieved.

¶In an optimization problem, an inequality constraint of the form $g(x) \leqslant 0$ is said convex if the function $g(x)$ is convex. If on the other hand the function $g(x)$ is concave, then the inequality constraint $g(x) \leqslant 0$ is said concave. By considering $f(b, c) = c^2 - b$, whose Hessian matrix $\partial^2 f/\partial(b, c)^2$ is positive semidefinite, we conclude $f(b, c)$ is a convex function of b, c, and therefore, all the inequality constraints $c(s)^2 - b(s) \leqslant 0$ are convex in $b(s)$ and $c(s)$ for all $s \in [0_+, 1_-]$.

3.3. Infinite-dimensional second-order cone program formulation

The infinite-dimensional problem (11) is reformulated into an infinite-dimensional second-order cone program (SOCP), which is a special class of convex optimization problems [16]. Even though this reformulation might not be necessary, depending really on the solver to be used, its derivation is simple and takes only a couple of lines; besides, lifting up convex problem (11) in the hierarchy of convex problems into an SOCP shall increase the number of available solvers that can be utilized to solve it. Unlike [9] where the reformulation as an SOCP is performed after discretization, we carry out this reformulation directly on infinite-dimensional problem (11), which makes our discretization procedure rather transparent. First, the convex constraints $1/\sqrt{b(s)} \leqslant 1/c(s)\ \forall s \in [0_+, 1_-]$ can be expressed as second-order cone constraints [10, 16]:

$$\frac{1}{\sqrt{b(s)}} \leqslant \frac{1}{c(s)} \Leftrightarrow c(s)^2 \leqslant b(s)\cdot 1 \Leftrightarrow \left\| \begin{bmatrix} 2c(s) \\ b(s)-1 \end{bmatrix} \right\|_2 \leqslant b(s)+1,\ \forall s \in [0_+, 1_-]. \tag{12}$$

To have a linear objective functional, the 'slack' function $d(s) > 0$ is introduced satisfying the following:

$$d(s) \geq \frac{1}{c(s)} \Leftrightarrow 1 \leqslant c(s)d(s) \Leftrightarrow \left\| \begin{bmatrix} 2 \\ c(s)-d(s) \end{bmatrix} \right\|_2 \leqslant c(s)+d(s),\ \forall s \in [0_+, 1_-]. \tag{13}$$

It should then be clear that problem (11) is equivalent to the following problem:

$$\begin{array}{ll} \text{minimize} & \displaystyle\int_{0_+}^{1_-} d(s)\,\mathrm{d}s \\ \text{subject to} & b(0) = \dot{s}_0^2, b(1) = \dot{s}_f^2, c(0) = \dot{s}_0, c(1) = \dot{s}_f \\ & \boldsymbol{\tau}_\mathrm{d}(s) = \boldsymbol{a}_1(s)a(s) + \boldsymbol{a}_2(s)b(s) + \boldsymbol{a}_3(s)c(s) + \boldsymbol{a}_4(s) \\ & \boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}_\mathrm{d}(s) \leqslant \boldsymbol{\tau}_{\max} \\ & \forall s \in [0,1] \\ & \left\| \begin{bmatrix} 2c(s) \\ b(s)-1 \end{bmatrix} \right\|_2 \leqslant b(s)+1 \\ & b'(s) = 2a(s) \\ & \left\| \begin{bmatrix} 2 \\ c(s)-d(s) \end{bmatrix} \right\|_2 \leqslant c(s)+d(s) \\ & b(s), c(s) > 0 \\ & \forall s \in [0_+, 1_-], \end{array} \tag{14}$$

where $a(s),\ b(s),\ c(s),\ \boldsymbol{\tau}_\mathrm{d}(s)$, and $d(s)$ are the optimization functions, while the rest is data.

4. PROBLEM DISCRETIZATION

In order to obtain a numerical solution to optimization problem (14), a discretization of this infinite-dimensional problem needs to be developed. An important difference from the discretization presented in [9], where $a(s)$ is assumed piecewise constant and $b(s)$ piecewise linear, is that we enforce $a(s)$ to be piecewise linear and $b(s)$ piecewise quadratic. These results are reached by strictly following the procedure to solve optimal control problems presented in [17]. First, the independent parameter s in (14) must be discretized by creating a finite grid of N points, $s_1 = 0 < s_2 < \cdots < s_N = 1$. Then, the optimization variables are defined in the following manner: $a_1 = a(s_1), \ldots, a_N = a(s_N)$, $b_1 = b(s_1), \ldots, b_N = b(s_N)$, $c_1 = c(s_1), \ldots, c_N = c(s_N)$, $d_1 = d(s_1), \ldots, d_N = d(s_N)$, and $\boldsymbol{\tau}^1 = \boldsymbol{\tau}_\mathrm{d}(s_1), \ldots, \boldsymbol{\tau}^N = \boldsymbol{\tau}_\mathrm{d}(s_N)$, which represent the functions

$a(s)$, $b(s)$, $c(s)$, $d(s)$, and $\boldsymbol{\tau}_{\mathrm{d}}(s)$ evaluated at the grid points $s_1, s_2, \ldots, s_N$. According to the procedure in [17], we should assume $b(s)$ and $a(s)$ are piecewise cubic and piecewise linear, respectively. Therefore, the pseudo-acceleration $a(s)$ is expressed as follows:

$$a(s) = a_j + (a_{j+1} - a_j)\left(\frac{s - s_j}{s_{j+1} - s_j}\right), \ s \in [s_j, s_{j+1}], \quad j = 1, 2, \ldots, N-1, \tag{15}$$

from which it is indeed true that $a(s_j) = a_j$ and $a(s_{j+1}) = a_{j+1}$. Similarly, $b(s)$ is chosen as follows:

$$b(s) = \sum_{k=0}^{3} \beta_{j,k}\left(\frac{s - s_j}{s_{j+1} - s_j}\right)^k, \ s \in [s_j, s_{j+1}], \quad j = 1, 2, \ldots, N-1, \tag{16}$$

where the polynomial coefficients $\beta_{j,0}$, $\beta_{j,1}$, $\beta_{j,2}$, and $\beta_{j,3}$ need to be determined explicitly. The aforementioned representation for $b(s)$ must satisfy $b(s_j) = b_j$, $b(s_{j+1}) = b_{j+1}$, $b'(s_j) = 2a(s_j)$, and $b'(s_{j+1}) = 2a(s_{j+1})$, which gives four equations in the four unknowns $\beta_{j,0}$, $\beta_{j,1}$, $\beta_{j,2}$, and $\beta_{j,3}$. By explicitly solving this simple system of equations, it is obtained:

$$\begin{aligned} \beta_{j,0} = b_j, \quad \beta_{j,1} = 2\Delta s_j a_j \quad \beta_{j,2} &= 3(b_{j+1} - b_j) - 2\Delta s_j (a_{j+1} + 2a_j) \\ \beta_{j,3} &= 2\Delta s_j (a_{j+1} + a_j) - 2(b_{j+1} - b_j), \end{aligned} \tag{17}$$

where $\Delta s_j := s_{j+1} - s_j$, $j = 1, 2, \ldots, N-1$. Another requirement for $a(s)$ and $b(s)$ is to satisfy $b'(s) = 2a(s)$ at the grid points s_j, $j = 1, \ldots, N$, and at the middle grid points defined as $\bar{s}_j := (s_j + s_{j+1})/2$, $j = 1, \ldots, N-1$. Satisfying $b'(s) = 2a(s)$ at the grid points s_j, $j = 1, \ldots, N$ is already guaranteed with the coefficients $\beta_{j,0}, \beta_{j,1}, \beta_{j,2}$, and $\beta_{j,3}$ obtained previously. The remaining requirements, $b'(\bar{s}_j) = 2a(\bar{s}_j)$, $j = 1, \ldots, N-1$, give after simple algebraic manipulations:

$$b_{j+1} - b_j = \Delta s_j (a_{j+1} + a_j), \quad j = 1, \ldots, N-1. \tag{18}$$

Interestingly, constraints (18) imply that $\beta_{j,3}$ in (17) shall be 0, which means that $b(s)$ must actually be piecewise quadratic. Finally, expecting $c(s)^2 = b(s)$, $c(s)$ is assumed piecewise linear.

On discretizing the objective functional in (14), we explicitly define $0_+ := (1-\alpha)s_1 + \alpha s_2$ and $1_- := (1-\alpha)s_N + \alpha s_{N-1}$, with $\alpha > 0$ being an adjustably small parameter. Assuming $d(s)$ is piecewise linear, we simply use the trapezoidal rule for approximating the definite integrals:

$$\begin{aligned} \int_{0_+}^{1_-} d(s)\,\mathrm{d}s &= \int_{0_+}^{s_2} d(s)\,\mathrm{d}s + \sum_{k=2}^{N-2}\int_{s_k}^{s_{k+1}} d(s)\,\mathrm{d}s + \int_{s_{N-1}}^{1_-} d(s)\,\mathrm{d}s \\ &\approx \frac{1}{2}\Bigg[(1-\alpha)\Delta s_1 (d(0_+) + d_2) \\ &\quad + \sum_{k=2}^{N-2} \Delta s_k (d_k + d_{k+1}) + (1-\alpha)\Delta s_{N-1}(d_{N-1} + d(1_-))\Bigg], \end{aligned} \tag{19}$$

where $d(0_+) = (1-\alpha)d_1 + \alpha d_2$ and $d(1_-) = \alpha d_{N-1} + (1-\alpha)d_N$. The remaining constraints in problem (14) are discretized by evaluating them at the grid points $s = s_j$, $j = 1, \ldots, N$. Those constraints not defined at $s = 0$ and $s = 1$ are evaluated instead at $s = 0_+$ and $s = 1_-$, respectively. With the aforementioned provisos in mind, the following discretization of (14) is obtained, with a_k, b_k, c_k, $\boldsymbol{\tau}^k$ and d_k, $k = 1, \ldots, N$, representing the optimization variables:

$$\begin{aligned}
\text{minimize}\quad & \frac{1}{2}\Bigg[(1-\alpha)\Delta s_1(d(0_+)+d_2) \\
& \quad + \sum_{k=2}^{N-2}\Delta s_k(d_k+d_{k+1}) + (1-\alpha)\Delta s_{N-1}(d_{N-1}+d(1_-))\Bigg] \\
\text{subject to}\quad & b_1 = \dot{s}_0^2, b_N = \dot{s}_f^2, c_1 = \dot{s}_0, c_N = \dot{s}_f \\
& \boldsymbol{\tau}^k = \boldsymbol{a}_1(s_k)a_k + \boldsymbol{a}_2(s_k)b_k + \boldsymbol{a}_3(s_k)c_k + \boldsymbol{a}_4(s_k) \\
& \boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}^k \leqslant \boldsymbol{\tau}_{\max} \\
& \left\|\begin{bmatrix} 2c_k \\ b_k - 1\end{bmatrix}\right\|_2 \leqslant b_k + 1 \\
& k = 1,\dots,N \\
& b_{j+1} - b_j = \Delta s_j(a_{j+1}+a_j) \\
& j = 1,\dots,N-1 \\
& \left\|\begin{bmatrix} 2 \\ c(0_+) - d(0_+)\end{bmatrix}\right\|_2 \leqslant c(0_+) + d(0_+) \\
& \left\|\begin{bmatrix} 2 \\ c_l - d_l\end{bmatrix}\right\|_2 \leqslant c_l + d_l, \quad l = 2,\dots,N-1 \\
& \left\|\begin{bmatrix} 2 \\ c(1_-) - d(1_-)\end{bmatrix}\right\|_2 \leqslant c(1_-) + d(1_-),
\end{aligned} \tag{20}$$

where $c(0_+) = (1-\alpha)c_1 + \alpha c_2$ and $c(1_-) = \alpha c_{N-1} + (1-\alpha)c_N$. The aforementioned optimization problem (20) represents an SOCP, which can be solved using a variety of solvers. To solve problem (20) and the extended formulation that shall be presented in Section 6, we use CVX, a package for specifying and solving convex programs [18, 19], which can be installed as a Toolbox in MATLAB®. Since CVX is a parser front-end that links to several solvers, we have configured it to use SDPT3, mainly because for all our numerical instances, SDPT3 finds the optimal solutions with exact tightness of $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$, which is not always the case when setting SeDuMi as the solver, particularly for the extended formulation with larger number of variables and constraints.

Because a one-to-one relationship between s and t is enforced, there is no ambiguity in recovering the time-optimal solution as a function of t. The time-optimal velocities and accelerations are retrieved as $\dot{\boldsymbol{q}}_\mathrm{d}^\star = \boldsymbol{h}'(s)c^\star(s)$ and $\ddot{\boldsymbol{q}}_\mathrm{d}^\star = \boldsymbol{h}''(s)b^\star(s) + \boldsymbol{h}'(s)a^\star(s)$. To complete the time-optimal solution, it remains to recover time t as a function of s as follows:

$$\frac{ds}{dt} = \dot{s} \Leftrightarrow \frac{dt}{ds} = \frac{1}{c(s)},\ c(s) > 0 \Leftrightarrow t(s) = t(0_+) + \int_{0_+}^{s}\frac{1}{c(u)}\mathrm{d}u.$$

It is then easily shown that

$$t(s_k) = t(s_{k-1}) + \int_{s_{k-1}}^{s_k}\frac{1}{c(u)}\mathrm{d}u, \quad k = 2,\dots,N.$$

Furthermore, at optimum $1/c(s) = d(s)$, thus,

$$t(s_k) = t(s_{k-1}) + \int_{s_{k-1}}^{s_k} d(u)\,\mathrm{d}u = t(s_{k-1}) + \frac{1}{2}\Delta s_{k-1}(d_{k-1}+d_k). \tag{21}$$

Therefore, time t is recovered with the following steps: (i) initialize $t(s_1) = 0$, (ii) for $k = 2,\dots,N$ and compute $t(s_k)$ using the recursive formula (21).

5. APPLICATION TO A SIX-AXIS INDUSTRIAL MANIPULATOR

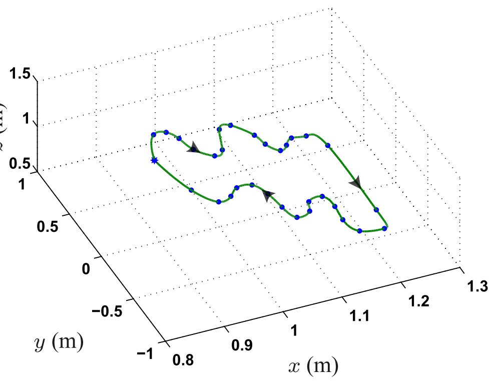
The time-optimal trajectory and feedforward torques $\left(\boldsymbol{q}_{\mathrm{d}}^{\star}(t), \dot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t), \ddot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t), \boldsymbol{\tau}_{\mathrm{d}}^{\star}(t)\right)$ are generated for the six-axis industrial manipulator FANUC M-16*i*B, which is briefly described in Appendix A. The robot end-effector is fully determined in space by its position coordinates (x, y, z) and its orientation, parameterized using Euler angles (ϕ, θ, ψ). An arbitrary Cartesian-space path with non-constant orientation of the end-effector is designed, with the position coordinates (x, y, z) shown in Figure 1(a), where both the initial and final positions are marked with an asterisk '*', corresponding to the robot home position. This path is obtained by choosing 33 control points in Cartesian coordinates (x, y, z), marked with '•' in Figure 1(a), which are then interpolated using cubic splines to satisfy the differentiability requirement on $\boldsymbol{h}(s)$. Likewise, Euler angles (ϕ, θ, ψ) are interpolated using cubic splines from a set of 33 control points that represent intermediate orientations.

The total number of grid points used to discretize the parameter s is $N = 1200$, which means that the positions (x, y, z) and orientations (ϕ, θ, ψ) of the end-effector are known at $N = 1200$ points. The corresponding joint-space path, $\boldsymbol{h}(s_k) \in \mathbb{R}^6$, $k = 1, \ldots, 1200$ is obtained by carrying out computations of inverse kinematics on the 1200 positions and orientations of the robot end-effector, for which the robotics toolbox for MATLAB® has been used [20]. In this manner, Figure 1(b) shows the joint-space path $\boldsymbol{h}(s_k) \in \mathbb{R}^6$ together with its first and second derivatives $\boldsymbol{h}'(s_k), \boldsymbol{h}''(s_k) \in \mathbb{R}^6$, for all six joints of the manipulator. Finally, the dynamic-model vectors $\boldsymbol{a}_1(s_k)$, $\boldsymbol{a}_2(s_k)$, $\boldsymbol{a}_3(s_k)$, and $\boldsymbol{a}_4(s_k)$, $k = 1, \ldots, 1200$, which were defined in (6), are obtained from inverse dynamics computations using again the robotics toolbox for MATLAB®.

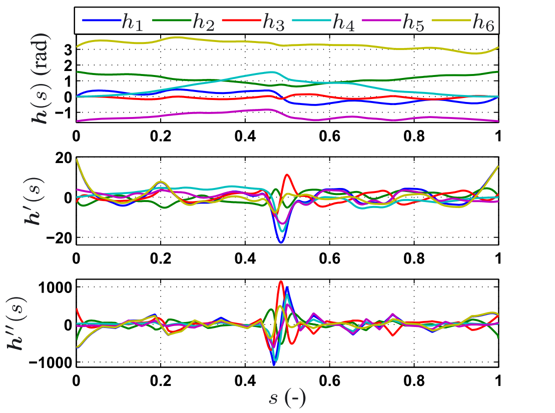
The initial and final pseudo-speeds are enforced to be 0, $\dot{s}_0 = \dot{s}_f = 0$. The torque constraints are symmetric, $\boldsymbol{\tau}_{\min} = -\boldsymbol{\tau}_{\max}$, where $\boldsymbol{\tau}_{\max} = (1782.4\ 1789.7\ 1647.2\ 97.2\ 108.5\ 79.1)^{\top}$ Nm, representing the maximum torques at the link-side, that is, at the gearboxes' output shaft that couple directly to the robot links. The corresponding maximum torques at the motor-side, that is, at the gearboxes' input shaft, are given by $\boldsymbol{u}_{\max} = \boldsymbol{G}^{-1}\boldsymbol{\tau}_{\max} = (10.21\ 10.21\ 8.60\ 4.30\ 1.58\ 1.58)^{\top}$ Nm, where $\boldsymbol{G}$ is a diagonal matrix of gear ratios, $\boldsymbol{G} = \mathrm{diag}(r_1, \ldots, r_6)$. For practical reasons, the optimal feedforward torques shall always be presented in the motor-side, given by $\boldsymbol{u}_{\mathrm{d}}^{\star} = \boldsymbol{G}^{-1}\boldsymbol{\tau}_{\mathrm{d}}^{\star}(t)$.

5.1. *Generated time-optimal trajectories and feedforward torques*

The time-optimal solutions generated when solving problem (20) are presented in Figure 2, yielding an optimal traveling time $t_f = 3.447$ s, which is the minimum time for the robot to move along the path $\boldsymbol{h}(s)$. The motor-side time-optimal torques are shown in Figure 2(a), which feature bang–bang behavior as there is always one actuator that saturates, that is, for all $t \in [0, t_f]$ either $u_1^{\star}$ or $u_2^{\star}$ saturate. The other actuators $u_3^{\star}$, $u_4^{\star}$, $u_5^{\star}$, and $u_6^{\star}$ do not saturate but are required to change suddenly at those time instants when $u_1^{\star}$ and $u_2^{\star}$ exchange being saturated. Likewise, large non-zero

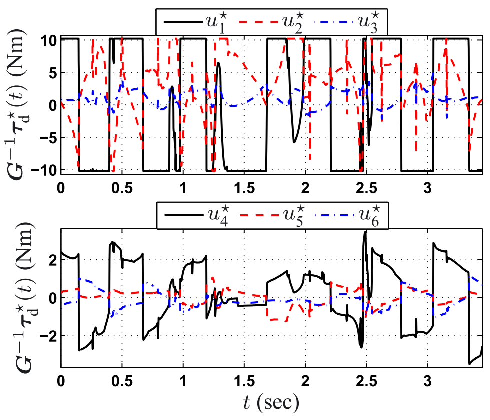


(a) Cartesian-space path with control points

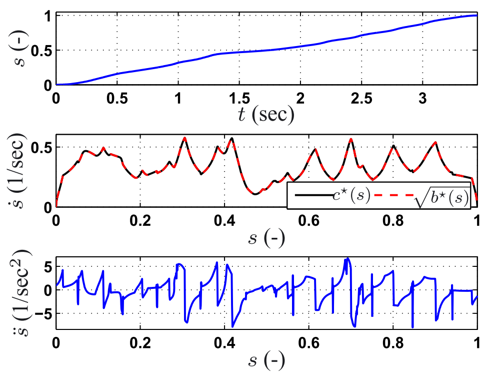


(b) Joint-space path $\boldsymbol{h}(s)$, $\boldsymbol{h}'(s)$ and $\boldsymbol{h}''(s)$ for the 6 axes

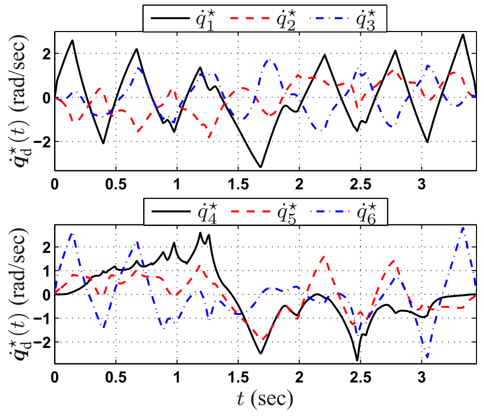
Figure 1. Arbitrary non-trivial path used to test the time-optimal trajectory planning algorithm.



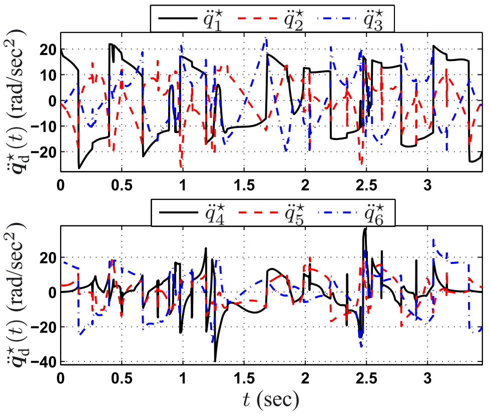
(a) Motor-side time-optimal torques $\boldsymbol{u}_{\mathrm{d}}^{\star} = \boldsymbol{G}^{-1}\boldsymbol{\tau}_{\mathrm{d}}^{\star}(t)$



(b) Parameter s, pseudo-speed $\dot{s}$ and pseudo-acceleration $\ddot{s}$



(c) Time-optimal velocity trajectory $\dot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t)$



(d) Time-optimal acceleration trajectory $\ddot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t)$

Figure 2. Time-optimal solutions generated when solving the optimization problem in (20).

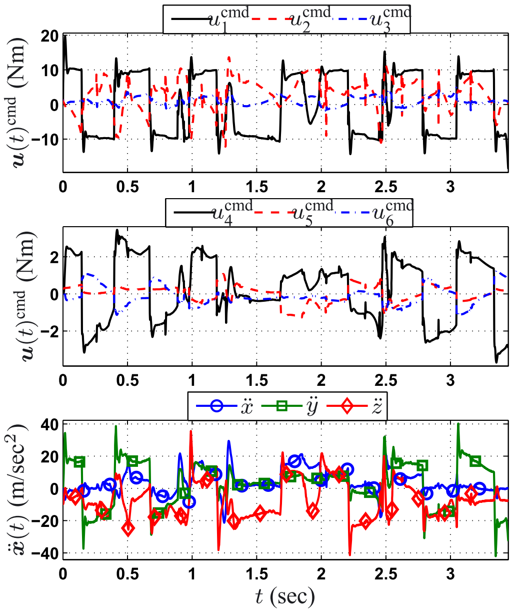
accelerations are required at the beginning and end of the trajectory, with sudden acceleration changes at the intermediate points. The corresponding $s(t)$, pseudo-speed $c(s)$, and pseudo-acceleration $a(s)$ are all presented in Figure 2(b). Both $c^\star(s)$ and $\sqrt{b^\star(s)}$ are plotted together so that they can be compared. Importantly, $c^\star(s) = \sqrt{b^\star(s)}$ for all $s \in [0, 1]$, which means that for the specific conditions of our application, the proposed convex relaxation (14), or equivalently its discretized version (20), solves exactly the original non-convex problem (9).

5.2. *Simulation of time-optimal solution*

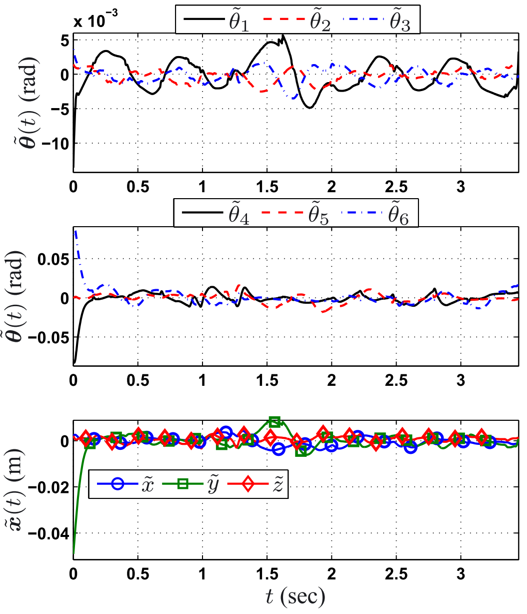
Simulations are carried out in order to study the effects of implementing the time-optimal solution $\left(\boldsymbol{q}_{\mathrm{d}}^{\star}(t), \dot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t), \ddot{\boldsymbol{q}}_{\mathrm{d}}^{\star}(t), \boldsymbol{\tau}_{\mathrm{d}}^{\star}(t)\right)$. A robot simulator in MATLAB® that uses Simulink® and SimMechanics ™ has been developed. This simulator incorporates dynamic effects that are not considered in the dynamic model (1), such as joint flexibility due to indirect drives. The link-side and motor-side joint positions are denoted by $\boldsymbol{q} \in \mathbb{R}^6$ and $\boldsymbol{\theta} \in \mathbb{R}^6$, respectively. The control law implemented in the motor-side corresponds to feedforward plus a PID feedback controller:

$$\boldsymbol{u} = \boldsymbol{u}_{\mathrm{d}}^{\star}(t) + \boldsymbol{K}_{\mathrm{P}}\tilde{\boldsymbol{\theta}} + \boldsymbol{K}_{\mathrm{V}}\dot{\tilde{\boldsymbol{\theta}}} + \boldsymbol{K}_{\mathrm{I}}\int_0^t \tilde{\boldsymbol{\theta}}(v)\,\mathrm{d}v, \tag{22}$$

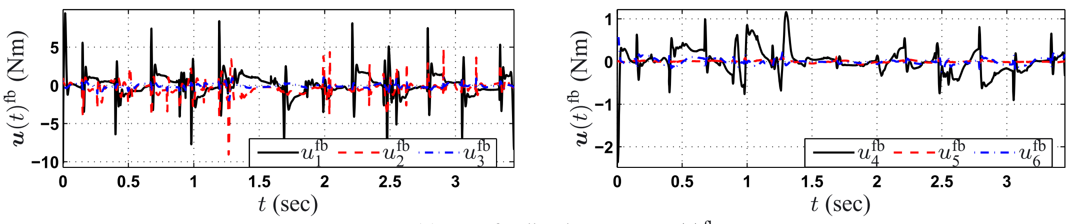
where $\boldsymbol{u}_{\mathrm{d}}^{\star}(t)$ represents the motor-side time-optimal torques. The tracking error in motor-side is $\tilde{\boldsymbol{\theta}}(t) := \boldsymbol{\theta}_{\mathrm{d}}(t) - \boldsymbol{\theta}(t)$, where the motor-side reference $\boldsymbol{\theta}_{\mathrm{d}}(t) = \boldsymbol{G}\boldsymbol{q}_{\mathrm{d}}^{\star}(t)$, with $\boldsymbol{G}$ the diagonal



(a) Torque commands and accelerometer readings



(b) Motor-side and Cartesian-space tracking errors



(c) PID feedback torques $\boldsymbol{u}(t)^{\rm fb}$

Figure 3. Simulation results of the time-optimal trajectory on six-axis industrial robot.

matrix of gear ratios. The feedback gains $\boldsymbol{K}_{\rm P}$, $\boldsymbol{K}_{\rm V}$, and $\boldsymbol{K}_{\rm I}$ are constant matrices to achieve a certain closed-loop servo bandwidth for both simulations and experiments. The motor-side torque commands $\boldsymbol{u}(t)^{\rm cmd} \in \mathbb{R}^6$ are presented in Figure 3(a), which feature large peaks at those instants when $\boldsymbol{u}_{\rm d}^{\star}(t)$ change suddenly. This implies that the actuators' limits, $\boldsymbol{u}_{\rm min}$ and $\boldsymbol{u}_{\rm max}$, are exceeded. Likewise, the readings of a three-axis accelerometer, mounted at the robot's end-effector, exhibit sudden acceleration changes and large overshoots. These acceleration overshoots are due to excitations of the vibration modes related to joint flexibility of indirect drives. The motor-side tracking errors $\tilde{\boldsymbol{\theta}}(t) \in \mathbb{R}^6$ and the Cartesian-space tracking errors $\tilde{\boldsymbol{x}}(t) \in \mathbb{R}^3$ are presented in Figure 3(b). Due to the non-zero initial and final accelerations required by pure time-optimal solutions, the initial and final tracking errors are comparatively large. In Figure 3(c), the feedback PID torques are presented. Because the trajectory changes in accelerations are too fast to follow, the PID feedback controller overreacts to such fast-changing trajectory, generating the peaks shown in Figure 3(c).

6. IMPOSING ACCELERATION CONSTRAINTS AND PENALIZING TOTAL JERK

The aforementioned simulation results suggest that pure time-optimal trajectories and feedforward torques are bound to cause severe degradation of performance. Nevertheless, time-optimal solutions are crucial to increase robot productivity. We therefore incorporate acceleration constraints and penalize a measure of total jerk, leading to near time-optimal solutions that are feasible for implementation at the expense of a modest increase in the traveling time t_f. Consider first incorporating symmetric joint-space acceleration constraints, $-\bar{\ddot{\boldsymbol{q}}}(s) \leqslant \ddot{\boldsymbol{q}}_{\rm d}(s) \leqslant \bar{\ddot{\boldsymbol{q}}}(s)$, where the

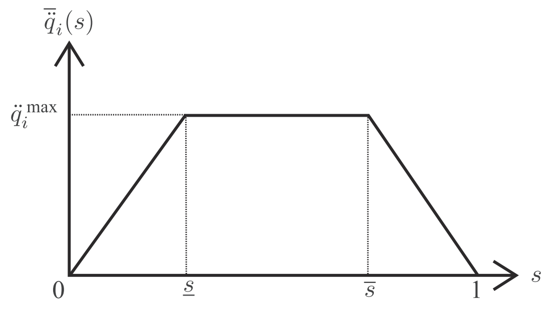


Figure 4. Profile of joint-space acceleration constraints.

inequalities are understood componentwise. For each joint, we propose the acceleration constraint $\overline{\ddot{q}}_i(s)$, $i = 1, \ldots, n$, with the profile shown in Figure 4, where $\ddot{q}_i^{\max}$ is the maximum acceleration allowed for the i-th joint. The parameters $\underline{s}$ and $\overline{s}$ are adjustable and chosen preferably small. From the expression for $\ddot{\boldsymbol{q}}_{\mathrm{d}}(s)$ in (3), the joint-space acceleration constraints are written as follows:

$$-\overline{\ddot{\boldsymbol{q}}}(s) \leqslant \boldsymbol{h}''(s)b(s) + \boldsymbol{h}'(s)a(s) \leqslant \overline{\ddot{\boldsymbol{q}}}(s). \tag{23}$$

Notice that inequalities (23) do not impose additional constraints to explicitly satisfy by $c(s)$, which would otherwise significantly decrease the likelihood of tightness for $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$. However, even when $c(s)$ does not explicitly feature in (23), imposing these constraints reduces the feasible search region, which in turn shall inevitably decrease the likelihood of tightness. Finally, in order to discretize inequalities (23) with the discretization scheme presented in Section 4, we simply evaluate (23) at the grid points, $s_1, s_2, \ldots, s_N$, to obtain the following:

$$-\overline{\ddot{\boldsymbol{q}}}(s_k) \leqslant \boldsymbol{h}''(s_k)b_k + \boldsymbol{h}'(s_k)a_k \leqslant \overline{\ddot{\boldsymbol{q}}}(s_k), \quad k = 1, 2, \ldots, N, \tag{24}$$

which are affine in a_k and b_k, thus preserving convexity when incorporated into problem (20).

6.1. *Penalizing a measure of total jerk*

In order to trade off time-optimality, we use a measure of total jerk $\dddot{\boldsymbol{q}} \in \mathbb{R}^n$. Interestingly, using either jerk $\dddot{\boldsymbol{q}}$ or torque derivative $\dot{\boldsymbol{\tau}}$ will lead in both cases to results that do not destroy convexity of problem (20). However, torque derivative shall lead to a result that explicitly imposes additional constraints to satisfy by $c(s)$, which significantly reduces the likelihood of $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$ for tightness. In our particular application, we initially used torque derivative to trade off time-optimality, which led to numerical results whereby tightness of $1/\sqrt{b^\star(s)} \leqslant 1/c^\star(s)$ was no longer attained. In contrast, using instead jerk $\dddot{\boldsymbol{q}}$ leads to additional constraints that do not explicitly feature $c(s)$. In this case, exact tightness for our particular application was consistently achieved. The forthcoming derivation leads to a different outcome but follows similar lines as those presented in [9] for $\dot{\boldsymbol{\tau}}$. Consider trading off the traveling time t_f against the following measure of total jerk:[‡]

$$J_{\text{jerk}} = \lambda \int_0^{t_f} \|\dddot{\boldsymbol{q}}\|_1 \,\mathrm{d}t, \tag{25}$$

with λ being a weighting parameter and $\|\dddot{\boldsymbol{q}}\|_1 := \sum_{i=1}^n |\dddot{q}_i|$. Notice that J_{jerk} can be simplified:

[‡]Even though the two-norm $\|\dddot{\boldsymbol{q}}\|_2$ is convex in $\dddot{\boldsymbol{q}}$, notice that $\dddot{\boldsymbol{q}}$ is not the optimization variable, but it can be written in terms of the optimization functions as follows:

$$\dddot{\boldsymbol{q}} = \boldsymbol{h}'''(s)b(s)c(s) + \boldsymbol{h}''(s)b'(s)c(s) + \boldsymbol{h}''(s)a(s)c(s) + \boldsymbol{h}'(s)a'(s)c(s),$$

which is not an affine transformation in $a(s)$, $b(s)$, and $c(s)$. By the composition rule of convex functions [10], $\dddot{\boldsymbol{q}}$ would need to be affine in $a(s)$, $b(s)$, and $c(s)$ for $\|\dddot{\boldsymbol{q}}\|_2$ to be convex, which is clearly not the case. It turns out, as shown in the forthcoming derivation, this property can be circumvented when using instead the one-norm $\|\dddot{\boldsymbol{q}}\|_1$.

$$\begin{aligned} J_{\text{jerk}} &= \lambda \int_0^{t_f} \|\dddot{\boldsymbol{q}}\|_1 \, \mathrm{d}t = \lambda \sum_{i=1}^{n} \int_0^{t_f} |\dddot{q}_i| \, \mathrm{d}t = \lambda \sum_{i=1}^{n} \int_0^{t_f} \left| \frac{\mathrm{d}\ddot{q}_i}{\mathrm{d}t} \right| \, \mathrm{d}t = \lambda \sum_{i=1}^{n} \int_0^{1} \left| \frac{\mathrm{d}\ddot{q}_i}{\mathrm{d}s} \right| \, \mathrm{d}s \\ &\approx \lambda \sum_{i=1}^{n} \sum_{j=1}^{N-1} \left| \ddot{q}_i(s_{j+1}) - \ddot{q}_i(s_j) \right| \;\propto\; \lambda \sum_{i=1}^{n} \sum_{j=1}^{N-1} \frac{\left| \ddot{q}_i(s_{j+1}) - \ddot{q}_i(s_j) \right|}{\ddot{q}_i^{\max}}, \end{aligned} \tag{26}$$

where dividing by $\ddot{q}_i^{\max}$ aims to non-dimensionalize the objective function, which improves numerical stability when trying out a wide range of values for λ, from very small to rather large.

The objective function (26) is nonlinear because the absolute value function $|\cdot|$ is nonlinear, that is, piecewise linear. In order to have a linear objective function, consider introducing the following slack variables: e_{ij}, $i = 1, \ldots, n$, and $j = 1, \ldots, N-1$, such that $\forall i, j$ $\left| \ddot{q}_i(s_{j+1}) - \ddot{q}_i(s_j) \right| \leqslant \ddot{q}_i^{\max} e_{ij}$. Thus, the objective function (26) is equivalent to a linear objective and inequality constraints:

$$J_{\text{jerkLin}} = \lambda \sum_{i=1}^{n} \sum_{j=1}^{N-1} e_{ij}, \text{such that } \left| \ddot{q}_i(s_{j+1}) - \ddot{q}_i(s_j) \right| \leqslant \ddot{q}_i^{\max} e_{ij}, \ \forall i, j. \tag{27}$$

The constraints $\left| \ddot{q}_i(s_{j+1}) - \ddot{q}_i(s_j) \right| \leqslant \ddot{q}_i^{\max} e_{ij}$, $i = 1, \ldots, n$, and $j = 1, \ldots, N-1$ are expressed compactly by defining $\boldsymbol{e}_j := \begin{pmatrix} e_{1j} & e_{2j} & \cdots & e_{nj} \end{pmatrix}^{\top} \in \mathbb{R}^n$, $j = 1, \ldots, N-1$. Therefore, these constraints can be written in vector form:

$$-\boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max} \leqslant \ddot{\boldsymbol{q}}_{\mathrm{d}}(s_{j+1}) - \ddot{\boldsymbol{q}}_{\mathrm{d}}(s_j) \leqslant \boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max}, \quad j = 1, \ldots, N-1,$$

where the notation $\boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max}$ means element-wise vector multiplication. By explicitly substituting $\ddot{\boldsymbol{q}}_{\mathrm{d}}(s) = \boldsymbol{h}''(s)b(s) + \boldsymbol{h}'(s)a(s)$, it is obtained for all $j = 1, \ldots, N-1$:

$$-\boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max} \leq \boldsymbol{h}''(s_{j+1})b_{j+1} + \boldsymbol{h}'(s_{j+1})a_{j+1} - \boldsymbol{h}''(s_j)b_j - \boldsymbol{h}'(s_j)a_j \leq \boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max}. \tag{28}$$

As mentioned beforehand, inequalities (28) do not explicitly impose any additional constraints to satisfy by $c(s_k)$, which would not be the case had we considered instead torque derivative $\dot{\boldsymbol{\tau}}$. Therefore, the following final convex formulation is obtained, which makes it possible to incorporate acceleration constraints with the profile of Figure 4, and to trade off time-optimality against a measure of total jerk through the weighting parameter λ:

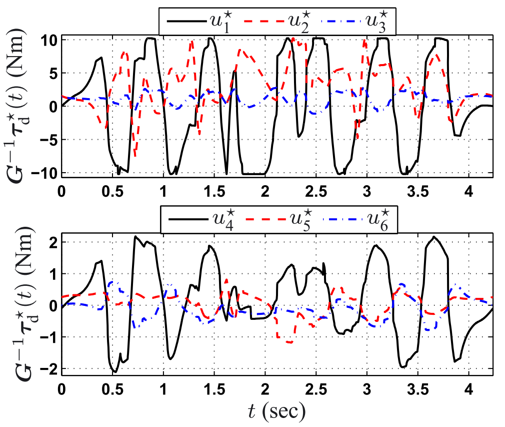
$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} \Bigg[(1-\alpha)\Delta s_1 (d(0_+) + d_2) + \sum_{k=2}^{N-2} \Delta s_k (d_k + d_{k+1}) \\ & \quad + (1-\alpha)\Delta s_{N-1}(d_{N-1} + d(1_-)) \Bigg] + \lambda \sum_{i=1}^{n} \sum_{j=1}^{N-1} e_{ij} \\ \text{subject to} \quad & b_1 = \dot{s}_0^2, b_N = \dot{s}_f^2, c_1 = \dot{s}_0, c_N = \dot{s}_f \\ & \boldsymbol{\tau}^k = \boldsymbol{a}_1(s_k)a_k + \boldsymbol{a}_2(s_k)b_k + \boldsymbol{a}_3(s_k)c_k + \boldsymbol{a}_4(s_k) \\ & \boldsymbol{\tau}_{\min} \leqslant \boldsymbol{\tau}^k \leqslant \boldsymbol{\tau}_{\max} \\ & \left\| \begin{bmatrix} 2c_k \\ b_k - 1 \end{bmatrix} \right\|_2 \leqslant b_k + 1 \\ & -\overline{\ddot{\boldsymbol{q}}}(s_k) \leqslant \boldsymbol{h}''(s_k)b_k + \boldsymbol{h}'(s_k)a_k \leqslant \overline{\ddot{\boldsymbol{q}}}(s_k) \\ & k = 1, \ldots, N \end{aligned}$$

$$\begin{aligned}
& b_{j+1} - b_j = \Delta s_j (a_{j+1} + a_j) \\
& -\boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max} \leq \left[\boldsymbol{h}''(s_{j+1}) b_{j+1} + \boldsymbol{h}'(s_{j+1}) a_{j+1} \right. \\
& \qquad \left. - \boldsymbol{h}''(s_j) b_j - \boldsymbol{h}'(s_j) a_j \right] \leq \boldsymbol{e}_j * \ddot{\boldsymbol{q}}^{\max} \\
& j = 1, \dots, N-1 \\
& \left\| \begin{bmatrix} 2 \\ c(0_+) - d(0_+) \end{bmatrix} \right\|_2 \leqslant c(0_+) + d(0_+) \\
& \left\| \begin{bmatrix} 2 \\ c_l - d_l \end{bmatrix} \right\|_2 \leqslant c_l + d_l, \quad l = 2, \dots, N-1 \\
& \left\| \begin{bmatrix} 2 \\ c(1_-) - d(1_-) \end{bmatrix} \right\|_2 \leqslant c(1_-) + d(1_-),
\end{aligned} \tag{29}$$

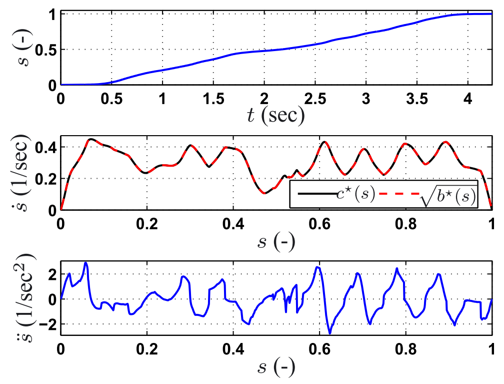
where $a_k,\ b_k,\ c_k,\ \boldsymbol{\tau}^k,\ d_k$, and $\boldsymbol{e}_j$ are the optimization variables and the rest is all data.

7. EXPERIMENTAL RESULTS ON SIX-AXIS INDUSTRIAL MANIPULATOR

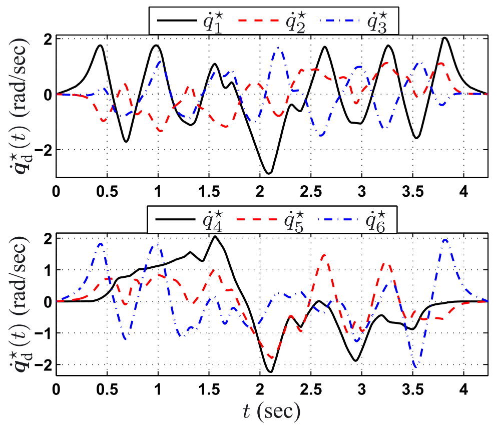
We proceed to carry out experiments on the real six-axis industrial manipulator FANUC M16*i*B, which is described in Appendix A. The near time-optimal trajectory and feedforward torques $\left(\boldsymbol{q}_{\rm d}^\star(t), \dot{\boldsymbol{q}}_{\rm d}^\star(t), \ddot{\boldsymbol{q}}_{\rm d}^\star(t), \boldsymbol{\tau}_{\rm d}^\star(t)\right)$ are generated using formulation (29), for which the same baseline



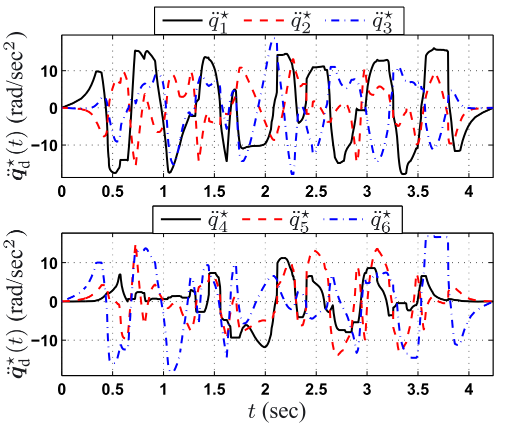
(a) Motor-side optimal torques $\boldsymbol{u}_{\rm d}^\star = \boldsymbol{G}^{-1}\boldsymbol{\tau}_{\rm d}^\star(t)$



(b) Parameter s, pseudo-speed $\dot{s}$, and pseudo-acceleration $\ddot{s}$



(c) Optimal velocity trajectory $\dot{\boldsymbol{q}}_{\rm d}^\star(t)$



(d) Optimal acceleration trajectory $\ddot{\boldsymbol{q}}_{\rm d}^\star(t)$

Figure 5. Near time-optimal solutions generated when solving problem (29) for $\lambda = 0.02$.

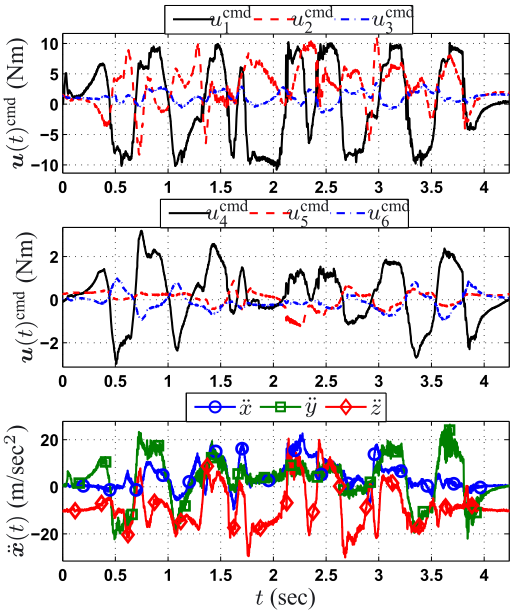
parameters as in Section 5 are used. Additionally, the parameters for acceleration constraints of Figure 4 are set to $\underline{s} = 0.02$, $\overline{s} = 0.98$, and $\ddot{\boldsymbol{q}}^{\max} = (60\ 60\ 60\ 30\ 30\ 30)^{\top}$ rad/sec^2.

The near time-optimal solutions generated when solving problem (29) for $\lambda = 0.02$ are presented in Figure 5, which should be compared against the pure time-optimal solutions of Figure 2. Notice from the near time-optimal torques in Figure 5(a) that $u_1^{\star}$ is still required to saturate at some time instants because the solutions are near time-optimal. Nevertheless, the transitions occur smoothly in a non-zero amount of time. Likewise, these torques are not required to saturate at the beginning and end of the trajectory, but instead, smoothly grow from and decay to the required gravity torques for home position. These important benefits come at the cost a modest increase in the traveling time, $t_f = 4.238$ s, that is, this near time-optimal solution is slower than the pure time-optimal solution by only 0.791 s.

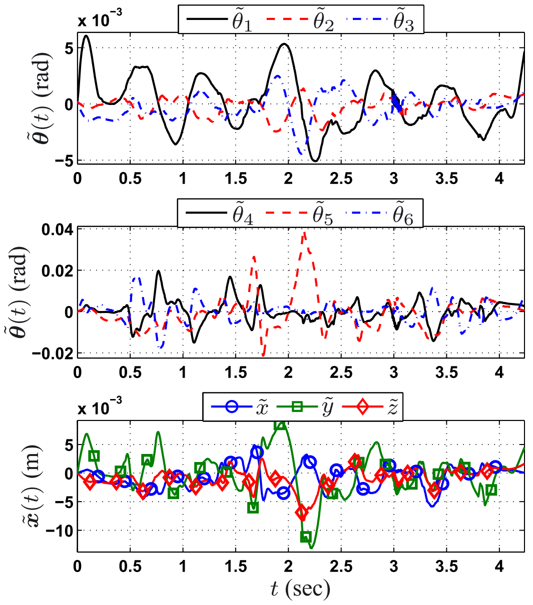
Importantly, exact tightness is still attained, that is, $c^{\star}(s) = \sqrt{b^{\star}(s)}\ \forall s \in [0, 1]$, which means that for our specific conditions, optimization problem (29) generates solutions that are dynamically feasible with respect to the full dynamic model (1). Likewise, the pseudo-acceleration $\ddot{s}$ has been rendered smaller and smoother. As clearly shown in Figure 5(d), this has the effect on $\ddot{\boldsymbol{q}}_{\mathrm{d}}^{\star}$ of effectively eliminating the sudden acceleration changes. Finally, notice that exact zero accelerations are indeed enforced at the beginning and end of the trajectory, with smooth growth from and decay to 0.

7.1. *Experimental results of near time-optimal solutions*

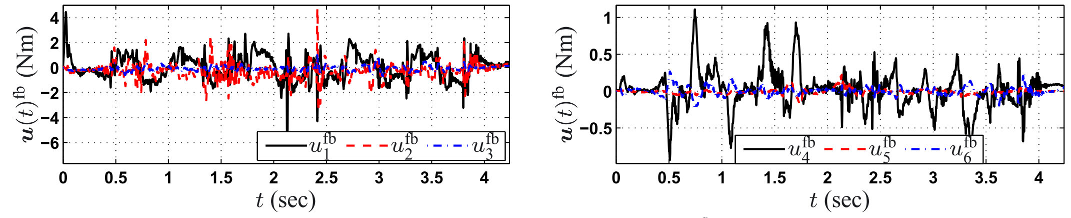
The near time-optimal solutions of Figure 5 are implemented on the real robotic system running at 1 kHz sampling rate. We implement the exact same control law given in (22), where the feedback



(a) Torque commands and accelerometer readings



(b) Motor-side and Cartesian-space tracking errors



(c) PID feedback torques $\boldsymbol{u}(t)^{\mathrm{fb}}$

Figure 6. Experimental results for the near time-optimal solutions with $\lambda = 0.02$.

gains $\boldsymbol{K}_{\mathrm{P}}$, $\boldsymbol{K}_{\mathrm{V}}$, and $\boldsymbol{K}_{\mathrm{I}}$ are exactly the same for simulations and experiments. The experimental results are given in Figure 6. Notice that the torque commands $\boldsymbol{u}(t)^{\mathrm{cmd}}$ from Figure 6(a) are rather consistent with the optimal feedforward torques of Figure 5(a). Even though these near time-optimal torques and trajectories represent the fastest motions achievable by the actual manipulator, the torque commands do not seem to exceed the maximum and minimum torque limits.

The joint-space motor-side tracking errors $\tilde{\boldsymbol{\theta}}(t) \in \mathbb{R}^6$ are presented in Figure 6(b), all of which are visibly better than the ones obtained in simulations for the pure time-optimal solution. The Cartesian-space tracking errors $\tilde{\boldsymbol{x}}(t) \in \mathbb{R}^3$ in Figure 6(b), which are measured with a CompuGauge 3D measurement system briefly described in Appendix A, seem also superior than the tracking errors obtained in simulation for the pure time-optimal solution. Notice that the accelerometer readings do not exhibit those large acceleration overshoots because of joint flexibility. Finally, the experimental PID feedback torques $\boldsymbol{u}(t)^{\mathrm{fb}}$ are presented in Figure 6(c). The role of the feedback part of the control law is to compensate for uncertainty and disturbances not considered in dynamic model (1). Clearly, the feedback controller does not excessively overreact in order to track this fast-changing trajectory.

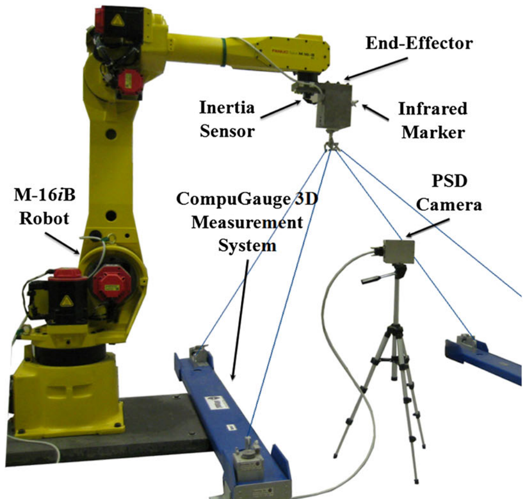
8. CONCLUSIONS

In this paper, we presented the problem of time-optimal trajectory planning of robot manipulators along predetermined geometric paths. We aimed to improving existing algorithms, by requiring that the time-optimal trajectories and feedforward torques be dynamically feasible with respect to the full nonlinear dynamic model, which included viscous friction. We presented the formulation as a mathematical optimization problem, which strategically differed from existing formulations, so that it would allow us to readily propose a convex relaxation with a reasonable likelihood to solve exactly the original non-convex problem. We then developed a discretization scheme to obtain a numerical solution. Even though there is no guarantee of exact tightness for all possible scenarios, in our particular case scenario, the proposed convex relaxation solved exactly the problem of time-optimal trajectory planning with full dynamic model. Throughout the paper, the effects of such fast-changing trajectories on three crucial variables were analyzed, namely, the tracking errors, applied torques, and a three-axis accelerometer mounted at the robot's end-effector.

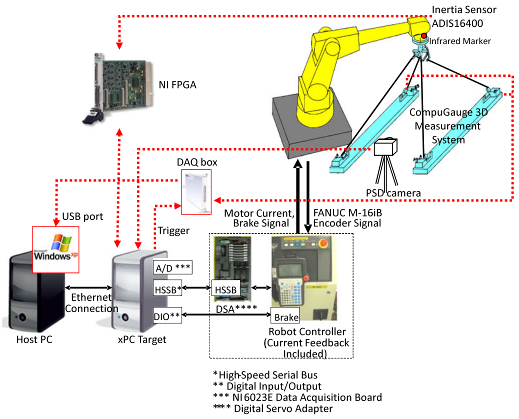
An extended problem formulation was then presented that incorporated acceleration constraints and traded off time-optimality against a measure of total jerk. These incorporations were chosen so as to preserve convexity and to prevent significantly decreasing the likelihood of exact tightness, although it was admitted that likelihood of tightness is still decreased because adding constraints will inevitably reduce the feasible search region. Nevertheless, for our particular application, exact tightness was consistently attained under many different parameters tried out by the authors. Acceleration constraints and penalization of the measure of total jerk both proved useful from real experiments on a six-axis industrial manipulator. The near time-optimal solutions represent the fastest optimal solutions achievable by the real robot manipulators without seriously degrading system performance. This brings modest contributions to existing algorithms, but also an important experimental study on this classical subject. For other applications with trouble scenarios whereby exact tightness might not be attained, the optimal solution to our convex relaxation could be used as the initial guess for a sequential convex optimization procedure.

APPENDIX A: EXPERIMENTAL SETUP OF SIX-AXIS INDUSTRIAL MANIPULATOR

Throughout this paper, we apply the proposed algorithms on a six-axis industrial manipulator, namely, the FANUC M-16*i*B/20 robot, courtesy of FANUC Corporation. This multi-joint manipulator is shown in Figure A.1(a) together with additional hardware components; it is a medium-size industrial robot capable of carrying objects with weights up to 20 kg at a maximum speed of 2000 mm/s. We have attached to the robot's end-effector an 'L'-shape payload, made from steel and weighting 18.37 kg. Each motor of the M-16*i*B robot is equipped with a built-in encoder to measure joint position in the motor-side, that is, at the gearboxes' input shaft. The M-16*i*B robot commercial controller utilizes position and velocity feedback for control, which is fixed and does not offer flexibility in the modification of control algorithms. The connection diagram of hardware components in



(a) FANUC M-16iB robot



(b) Hardware connection

Figure A.1. Industrial robot FANUC M-16iB and its hardware connection diagram.

Figure A.1(b) makes it possible to bypass the commercial robot controller and implement our own control algorithms.

The control algorithms are designed in MATLAB® and Simulink® on the host PC that runs a Windows platform. Then these control algorithms are implemented for experiments on the target PC, for which the MATLAB® toolbox xPC Target ™ is conveniently used. The host PC and target PC are connected via Ethernet as depicted by the hardware diagram in Figure A.1(b). After the controller design is performed in the host PC, the corresponding real-time code is loaded into the target PC through the Ethernet connection. The motor torque commands are computed by our own control algorithms, running on the target PC, and then converted to current commands for the FANUC robot controller, which delivers such a current. When the control algorithms are running on the target PC, the connection between the two computers is automatically disabled such that real-time execution of the algorithms in the target PC is guaranteed. The sampling rates of all the sensor signals as well as the real-time controller are set to 1 kHz.

A three-axis accelerometer mounted on the robot's payload is used to monitor the effects of near time-optimal trajectories. For this purpose, we use an inertial measurement unit from Analog Devices ADIS16400, which includes a three-axis accelerometer and a three-axis gyroscope. Likewise, a three-dimensional position measurement system, known as CompuGauge 3D [21], is utilized to measure the coordinates (x, y, z) of the end-effector tool center point, which gives the ground truth for performance evaluation.

ACKNOWLEDGEMENTS

The first author gratefully acknowledges UC MEXUS-CONACYT for providing him with a 5-year doctoral fellowship to support his PhD studies at Berkeley. Likewise, the financial and technical supports from FANUC Corporation were greatly appreciated. Finally, the reviewers of this paper are acknowledged for engaging in critical but constructive revisions aimed always at improving the final manuscript.

REFERENCES

1. LaValle SM. *Planning Algorithms*. Cambridge University Press: Cambridge, U.K., 2006.
2. R. Tedrake. Underactuated robotics: Learning, planning, and control for efficient and agile machines. *Course Notes for MIT 6.832, MIT 32-380*, 32 Vassar Street, Cambridge, MA 02139, USA, October 2009.
3. Bobrow JE, Dubowsky S, Gibson JS. Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotics Research* 1985; **4**(3):3–17.
4. Shin KG, McKay ND. Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control* 1985; **30**(6):531–541.

5. Pfeiffer F, Johanni R. A concept for manipulator trajectory planning. *Robotics and Automation, IEEE Journal of* 1987; **3**(2):115–123.
6. Shiller Z. Time-energy optimal control of articulated systems with geometric path constraints. *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on,* IEEE, San Diego, California, USA, 1994; 2680–2685.
7. Betts JT. *Practical Methods for Optimal Control Using Nonlinear Programming, Advances in Design and Control*, Vol. 3. Society for Industrial and Applied Mathematics (SIAM): Philadelphia, PA, 2001.
8. Costantinescu D, Croft E. Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *Journal of Robotic Systems* 2000; **17**(5):233–249.
9. Verscheure D, Demeulenaere B, Swevers J, De Schutter J, Diehl M. Time-optimal path tracking for robots: a convex optimization approach. *IEEE Transactions on Automatic Control* 2008; **54**:2318–2327.
10. Boyd S, Vandenberghe L. *Convex Optimization* (1st edn.) Cambridge University Press: Cambridge, MA, 2004.
11. Lam D, Manzie C, Good M. Model predictive contouring control. *Decision and Control (CDC), 2010 49th IEEE Conference on,* IEEE, 2010; 6137–6142.
12. Siciliano B, Sciavicco L, Villani L, Oriolo G. *Robotics: Modelling, Planning and Control* (1st edn.), Advanced Textbooks in Control and Signal Processing. Springer-Verlag: London, 2009.
13. O'Reilly OM. *Intermediate Dynamics for Engineers: A Unified Treatment of Newton–Euler and Lagrangian Mechanics*. Cambridge University Press: Cambridge, MA, 2008.
14. Spong MW, Hutchinson S, Vidyasagar M. *Robot Modeling and Control*. John Wiley & Sons Hoboken (N.J.), 2006.
15. Bryson AE. *Applied Optimal Control: Optimization, Estimation, and Control*. Taylor & Francis: New York, 1975.
16. Lobo MS, Vandenberghe L, Lebret H, Boyd S. Applications of second-order cone programming. *Linear Algebra and its Applications* 1998; **284**:193–228.
17. Stryk OV. Numerical solution of optimal control problems by direct collocation, ed. *In Optimal Control - Calculus of Variation, Optimal Control Theory and Numerical Methods, 111, 129-143, International Series of Numerical Mathematics, Birkhäuser* 1993; **111**:129–143.
18. Grant M, Boyd S. CVX: MATLAB software for disciplined convex programming, version 2.1, March 2014. (Available from: http://cvxr.com/cvx) [Accessed on April 2013].
19. Grant M, Boyd Sb. Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, Blondel V, Boyd S, Kimura H (eds)., Lecture Notes in Control and Information Sciences. Springer-Verlag Limited, 2008; 95–110.
20. Corke PI. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine* 1996; **3**(1):24–32.
21. CompuGauge 3D, July 2009. (Available from: http://www.dynalog-us.com/solutions) [Accessed on May 2012].