UNIVERSITÀ DEGLI STUDI DI PARMA

Dottorato di Ricerca in Tecnologie dell'Informazione

XXI Ciclo

Optimal Constrained Planning for Complex Mechatronic Systems

Coordinatore:

Chiar.mo Prof. Carlo Morandi

Tutor:

Chiar.mo Prof. Corrado Guarino Lo Bianco

Dottorando: *Oscar Gerelli*

Gennaio 2009

To my beloved parents
and Grandma

Contents

Introduction

This thesis focuses on the challenging problem of the optimal planning for mechatronic systems. The general goal is to find strategies which maximize or minimize some cost criteria defined over a given constrained problem. The planning for mobile or industrial robots is a general framework under which several different open research issues can be found. In fact, the motion planning involves the solution of a variety of optimality problems which range from the optimal path design to the optimal planning of trajectory, or alternatively, of velocity. The aforementioned planning issues can be solved by algorithms that can act either offline or online, i.e., respectively, by designing the overall motion before any movement of the controlled system or by constantly planning or shaping the motion during the task execution. Since, obviously, this is a very wide research field we have limited the scope of our analysis to the cases depicted in Figure 1, which gives a graphical overview of the topics investigated in this work.

Most of the proposed approaches aim at guarantee a perfect path tracking of a generic mobile or industrial robot if trajectories are planned according to the so-called path-velocity decomposition. In this framework, the trajectory to be execute is obtained by first defining a desired geometric path and, only subsequently, by assigning a time law to move along it.

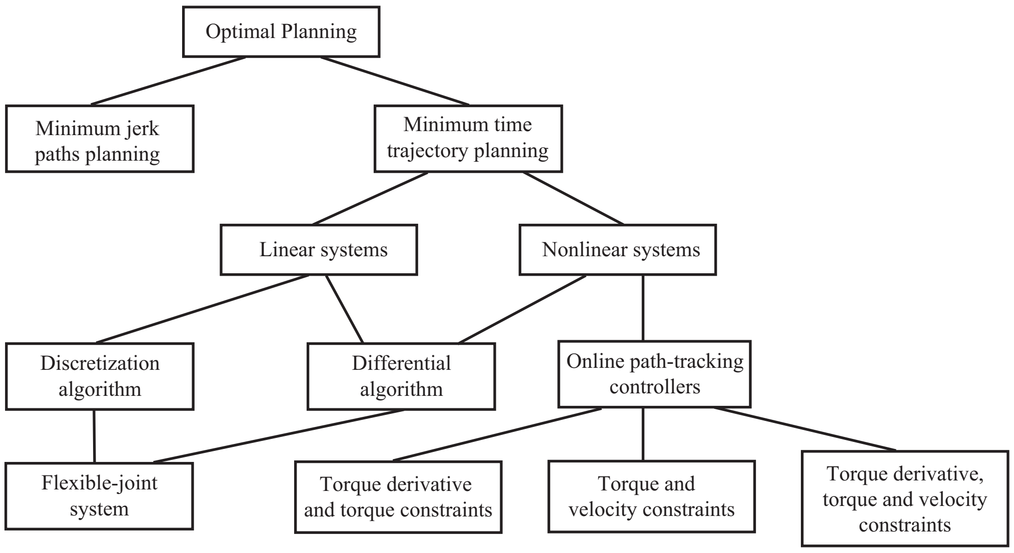

Optimal Planning
Minimum jerk paths planning
Minimum time trajectory planning
Linear systems
Nonlinear systems
Discretization algorithm
Differential algorithm
Online path-tracking controllers
Flexible-joint system
Torque derivative and torque constraints
Torque and velocity constraints
Torque derivative, torque and velocity constraints

Figure 1: A graphical overview of the thesis structure

The first problem analyzed in this thesis is, therefore, the optimal path generation. In particular the attention has been focused on the design of optimal planar paths for mobile robots. There does not exists a unique solution to this issue, since different planning primitives and optimality criterion can be used. In this work we use the η^3-splines, a powerful path planning primitive recently devised by the Control System Group of the University of Parma. One of the features of such primitives is the possibility to modify the shape of the generated path by simply acting on a set of six free parameters. This is both the strength and the weakness of the η^3-splines since it imposes to find an effective procedure for the assignments of the free parameters, in order to generate smooth profiles. In Chapter one, this problem is investigated and heuristic relations, which generate suboptimal paths with minimum curvature derivative, are proposed.

As it was early anticipated, path planning is only one aspect of the optimality problems analyzed in this thesis: dynamics and kinematics constraints have not been considered so far. The planning of the velocity profile represents a crucial step to guarantee the overall trajectory feasibility with respect to the system kinematic and dynamic constraints. Several offline algorithms can be found in the literature to deal with this problem. Our attention has been mainly focused on the analysis of those generating minimum-time trajectories. In particular, our efforts have been spent, in Chapter two and three, to devise control schemes to online shape any desired, possibly unfeasible, trajectory into a new one which fulfills given constraints. Velocity profiles are typically off-line evaluated by means of optimization algorithms which fulfill given dynamic constraints of the systems' models. Obviously, generated solutions are not robust against mismodelling or external perturbations, especially when profiles requiring the maximization of the actuator efforts are planned, such as, e.g., the minimum-time trajectories. This is the reason why online trajectory scaling algorithms are required to avoid that saturations of the control actions could determine a path tracking lost. In particular, Chapter two is devoted to present a novel control scheme, based on a nonlinear filter, able to account for velocity and acceleration/torque constraints while Chapter three extends the approach to account for torque and torque derivative constraints.

As a part of the research on the optimal planning, the last chapters of this thesis presents some contributions to the generation of optimal set-points for constrained nonlinear systems with a particular focus on the minimum-time trajectory planning. More in details, two different approaches are analyzed. The first one, described in Chapter four, uses the discretization to convert the nontrivial optimum problem, for linear systems, into a simpler equivalent set of feasibility tests which can be solved by linear programming algorithms. The method, which can easily manage input, output and state constraints, has been successfully applied to the feedforward control of a flexible joint: its nonlinear model has been linearized around the equilibrium point in oder to use this linear programming approach.

Even if the algorithm of Chapter four has returned very interesting results, the minimum-time problem has been further investigated in Chapter five in order to propose a new pure differential method able to manage also nonlinear systems. The solution is based on the Pontryagin maximum principle and has been tested against the nonlinear model of the flexible joint.

Finally some conclusion and future works recommendations are proposed in the last chapter.

CHAPTER 1

Generation of minimum curvature derivative paths for mobile robots

The path may not be left for an instant.
If it could be left, it would not be the path.
Confucius

Several approaches can be found in the literature in order to generate appropriate paths for autonomous vehicles. They can be roughly divided into two different frameworks. In the first one, usually indicated as “motion planning”, a structured and known environment is considered. Therefore, a path joining two given points can be generated taking into account the obstacle avoidance problem and possibly satisfying some given geometric constraints, e.g. minimizing the maximum path curvature. The first work related to motion planning was proposed by Dubin [1]. In his work a minimum length path was generated as a composition of linear segments and circular arcs. Subsequently, many other works addressed the same problem [2, 3, 4] and only recently it has been enriched by considering the generation of continuous curvature paths [5].

In the second framework, usually indicated with the term of “motion generation”, the planing phase assumes local characteristics being focused on the generation of short distance paths. This framework is generally encountered when a limited information on the vehicle surroundings is available, such in the case of a car vehicle moving along an unknown road or an autonomous robot moving inside an environment with strong dynamics characteristics. Obstacle avoidance is generally handled through an opportune choice of the goal point and of the final robot orientation: if a collision is detected, a different target point is selected.

In a motion generation context, path geometric characteristics are extremely relevant. Several path primitives, which generate continuous curvature paths, were proposed in the past: clothoids, cubic spirals [6], polar polynomials [7], intrinsic splines [8], etc.. Recently, the attention has been focused on planning primitives whose curvature is continuously differentiable [9]. Paths which possess this characteristic are named G^3-paths. G^3-continuity is essential for unicycle-like robots: in [10] it has been proved that G^3-paths are compulsory in order to obtain continuously differentiable control signals. This requirement is not strictly necessary in the case of other autonomous vehicles, however the use of paths whose curvature is continuously differentiable leads to the generation of smooth command signals, which is, undoubtedly, a positive characteristic.

In [11, 12], a new planning primitive, named η^3-splines, has been proposed for the generation of G^3-paths. η^3-splines are planned by means of closed form expressions and always fulfill any arbitrarily assigned set of interpolating conditions. The shape of η^3-splines can be refined by acting on a set of six free parameters which do not affect the curve boundary points: the assigned interpolating conditions are always fulfilled independently from the choice of such parameters. Consequently, given an appropriate shaping criterion, η^3-splines can be considered a powerful tool for the generation of optimal paths. Two main questions arise: which is the most appropriate optimality criterion to be fulfilled? And, moreover: is it possible to devise the optimal shaping parameters by means of a simple method? There is not a single answer to the first question. Since the control strategy proposed in [10] aims at generating smooth and accurate robot movements, the emphasis has been posed on the generation of paths whose curvature derivative is minimized. In this chapter it will be shown that, owing to this choice, lateral solicitations acting on a moving vehicle can be reduced. The answer to the second question is not trivial. If η^3-splines are used in a motion planning context, the optimal planning problem can be offline solved by means of an algorithm for the global semi-infinite optimization which is able to manage nonlinear object functions. This approach is not suited in a motion generation framework since, owing to the problem complexity, evaluation times are not compatible with online applications. As a consequence, the solution must be found through a different approach. The method analyzed in this chapter for the optimal planning of η^3-splines does not require the explicit online solution of an optimization problem and, consequently, can be efficiently used in a real-time framework.

The current chapter is organized as follows. In §1.1, the G^3-interpolation problem is formalized (*Problem 1*) and the closed form expressions (η^3-splines) proposed in [11, 12] for its solution are recalled. The optimal shaping problem (*Problem 2*), is formulated in the same section, while the proposed solution is described in §1.2. The results are verified in §1.3 by means of a path planning and tracking test case.

1.1 Problem formulation

A curve in the Cartesian planar space can be described by means of the function

$$\begin{aligned} \mathbf{p} : [u_0, u_1] &\rightarrow \mathbb{R}^2 \\ u &\rightarrow \mathbf{p}(u) = [\alpha(u)\, \beta(u)]^T \ , \end{aligned}$$

where $[u_0, u_1]$ is a real closed interval. The associated "path" is the image of $[u_0, u_1]$ under the vectorial function $\mathbf{p}(u)$, i.e., $\mathbf{p}([u_0, u_1])$. We say that $\mathbf{p}(u)$ is a regular curve if $\dot{\mathbf{p}}(u)$ is piecewise continuous, i.e., $\dot{\mathbf{p}}(u) \in C_p([u_0, u_1])$, and $\dot{\mathbf{p}}(u) \neq 0$, $\forall u \in [u_0, u_1]$. The arc length or, equivalently, the curvilinear coordinate measured along $\mathbf{p}(u)$, denoted by s, can be evaluated as

$$\begin{aligned} f : [u_0, u_1] &\rightarrow \mathbb{R} \\ u &\rightarrow s = \int_{u_0}^{u} \|\dot{\mathbf{p}}(\xi)\|\, d\xi \end{aligned}$$

where $\|\cdot\|$ denotes the Euclidean norm.

Given any point of a regular curve it can be defined a tangent vector $\theta(u)$ measured along the x-axis, a scalar curvature $\kappa(u)$, and a curvature derivative $\dot{\kappa}(u) := \frac{d\kappa}{ds}(u)$. If $\theta(u)$ and $\kappa(u)$ are continuous functions over $[u_0, u_1]$, then $\mathbf{p}(u)$ is a G^2-curve, i.e., it has a second order geometric continuity. If also $\dot{\kappa}(u)$ is continuous over $[u_0, u_1]$, then $\mathbf{p}(u)$ has a third order geometric continuity and is indicated as a G^3-curve.

Remark 1 *A composite G^3-path can be generated by combining several G^3-curves if it is possible to assign tangents, curvatures, and curvature derivatives at the extreme points of each of them.*

Therefore the following interpolation problem can be stated.

Problem 1 *Assume that two points $\mathbf{p}_A := [x_A\ y_A]^T$ and $\mathbf{p}_B := [x_B\ y_B]^T$ have been assigned in the Cartesian space. Generate a G^3-curve $\mathbf{p}(u)$ between $\mathbf{p}_A$ and $\mathbf{p}_B$ which fulfills given interpolating conditions on the initial and final tangent angles θ_A and θ_B, curvatures κ_A and κ_B, and curvature derivatives $\dot{\kappa}_A$ and $\dot{\kappa}_B$.*

In order to solve *Problem 1*, a new planning primitive, named η^3-splines, has been proposed in [11, 12]. It is given by two seven order polynomial functions defined as follows

$$\mathbf{p}(u) := [\alpha(u)\ \beta(u)]^T, u \in [0,1] \tag{1.1}$$

where

$$\alpha(u) := \alpha_0 + \alpha_1 u + \alpha_2 u^2 + \alpha_3 u^3 + \alpha_4 u^4 + \alpha_5 u^5 + \alpha_6 u^6 + \alpha_7 u^7; \tag{1.2}$$

$$\beta(u) := \beta_0 + \beta_1 u + \beta_2 u^2 + \beta_3 u^3 + \beta_4 u^4 + \beta_5 u^5 + \beta_6 u^6 + \beta_7 u^7\ . \tag{1.3}$$

In the same paper, closed form expressions were proposed in order to efficiently evaluate coefficients α_i and β_i on the basis of the interpolating conditions. For the completeness of the discussion they are recalled in the following.

$$\alpha_0 = x_A \tag{1.4}$$

$$\alpha_1 = \eta_1 \cos\theta_A \tag{1.5}$$

$$\alpha_2 = \frac{1}{2}\eta_3 \cos\theta_A - \frac{1}{2}\eta_1^2 \kappa_A \sin\theta_A \tag{1.6}$$

$$\alpha_3 = \frac{1}{6}\eta_5 \cos\theta_A - \frac{1}{6}\left(\eta_1^3 \dot{\kappa}_A + 3\eta_1\eta_3\kappa_A\right) \sin\theta_A \tag{1.7}$$

$$\begin{aligned}\alpha_4 = {} & 35\,(x_B - x_A) - \left(20\eta_1 + 5\eta_3 + \frac{2}{3}\eta_5\right)\cos\theta_A + \left(5\eta_1^2\kappa_A + \frac{2}{3}\eta_1^3\dot{\kappa}_A + 2\eta_1\eta_3\kappa_A\right)\sin\theta_A \\ & - \left(15\eta_2 - \frac{5}{2}\eta_4 + \frac{1}{6}\eta_6\right)\cos\theta_B - \left(\frac{5}{2}\eta_2^2\kappa_B - \frac{1}{6}\eta_2^3\dot{\kappa}_B - \frac{1}{2}\eta_2\eta_4\kappa_B\right)\sin\theta_B\end{aligned} \tag{1.8}$$

$$\begin{aligned}\alpha_5 = {} & -84\,(x_B - x_A) + (45\eta_1 + 10\eta_3 + \eta_5)\cos\theta_A - \left(10\eta_1^2\kappa_A + \eta_1^3\dot{\kappa}_A + 3\eta_1\eta_3\kappa_A\right)\sin\theta_A \\ & + \left(39\eta_2 - 7\eta_4 + \frac{1}{2}\eta_6\right)\cos\theta_B + \left(7\eta_2^2\kappa_B - \frac{1}{2}\eta_2^3\dot{\kappa}_B - \frac{3}{2}\eta_2\eta_4\kappa_B\right)\sin\theta_B\end{aligned} \tag{1.9}$$

$$\begin{aligned}\alpha_6 = {} & 70\,(x_B - x_A) - \left(36\eta_1 + \frac{15}{2}\eta_3 + \frac{2}{3}\eta_5\right)\cos\theta_A + \left(\frac{15}{2}\eta_1^2\kappa_A + \frac{2}{3}\eta_1^3\dot{\kappa}_A + 2\eta_1\eta_3\kappa_A\right)\sin\theta_A \\ & - \left(34\eta_2 - \frac{13}{2}\eta_4 + \frac{1}{2}\eta_6\right)\cos\theta_B - \left(\frac{13}{2}\eta_2^2\kappa_B - \frac{1}{2}\eta_2^3\dot{\kappa}_B - \frac{3}{2}\eta_2\eta_4\kappa_B\right)\sin\theta_B\end{aligned} \tag{1.10}$$

$$\begin{aligned}\alpha_7 = {} & -20\,(x_B - x_A) + \left(10\eta_1 + 2\eta_3 + \frac{1}{6}\eta_5\right)\cos\theta_A - \left(2\eta_1^2\kappa_A + \frac{1}{6}\eta_1^3\dot{\kappa}_A + \frac{1}{2}\eta_1\eta_3\kappa_A\right)\sin\theta_A \\ & + \left(10\eta_2 - 2\eta_4 + \frac{1}{6}\eta_6\right)\cos\theta_B + \left(2\eta_2^2\kappa_B - \frac{1}{6}\eta_2^3\dot{\kappa}_B - \frac{1}{2}\eta_2\eta_4\kappa_B\right)\sin\theta_B\end{aligned} \tag{1.11}$$

$$\beta_0 = y_A \tag{1.12}$$

$$\beta_1 = \eta_1 \sin\theta_A \tag{1.13}$$

$$\beta_2 = \frac{1}{2}\eta_3 \sin\theta_A + \frac{1}{2}\eta_1^2 \kappa_A \cos\theta_A \tag{1.14}$$

$$\beta_3 = \frac{1}{6}\eta_5 \sin\theta_A + \frac{1}{6}\left(\eta_1^3 \dot{\kappa}_A + 3\eta_1\eta_3\kappa_A\right) \cos\theta_A \tag{1.15}$$

$$\begin{aligned}\beta_4 = {} & 35\,(y_B - y_A) - \left(20\eta_1 + 5\eta_3 + \frac{2}{3}\eta_5\right)\sin\theta_A - \left(5\eta_1^2\kappa_A + \frac{2}{3}\eta_1^3\dot{\kappa}_A + 2\eta_1\eta_3\kappa_A\right)\cos\theta_A \\ & - \left(15\eta_2 - \frac{5}{2}\eta_5 + \frac{1}{6}\eta_6\right)\sin\theta_B + \left(\frac{5}{2}\eta_2^2\kappa_B - \frac{1}{6}\eta_2^3\dot{\kappa}_B - \frac{1}{2}\eta_2\eta_4\kappa_B\right)\cos\theta_B\end{aligned} \tag{1.16}$$

$$\begin{aligned}\beta_5 = {} & -84\,(y_B - y_A) + (45\eta_1 + 10\eta_3 + \eta_5)\sin\theta_A + \left(10\eta_1^2\kappa_A + \eta_1^3\dot{\kappa}_A + 3\eta_1\eta_3\kappa_A\right)\cos\theta_A \\ & + \left(39\eta_2 - 7\eta_4 + \frac{1}{2}\eta_6\right)\sin\theta_B - \left(7\eta_2^2\kappa_B - \frac{1}{2}\eta_2^3\dot{\kappa}_B - \frac{3}{2}\eta_2\eta_4\kappa_B\right)\cos\theta_B\end{aligned} \tag{1.17}$$

$$\beta_6 = 70\,(y_B - y_A) - \left(36\eta_1 + \frac{15}{2}\eta_3 + \frac{2}{3}\eta_5\right)\sin\theta_A - \left(\frac{15}{2}\eta_1^2\kappa_A + \frac{2}{3}\eta_1^3\dot{\kappa}_A + 2\eta_1\eta_3\kappa_A\right)\cos\theta_A$$

$$- \left(34\eta_2 - \frac{13}{2}\eta_4 + \frac{1}{2}\eta_6\right)\sin\theta_B + \left(\frac{13}{2}\eta_2^2\kappa_B - \frac{1}{2}\eta_2^3\dot{\kappa}_B - \frac{3}{2}\eta_2\eta_4\kappa_B\right)\cos\theta_B \quad (1.18)$$

$$\begin{aligned}\beta_7 \;=\; & -20\,(y_B - y_A) + \left(10\eta_1 + 2\eta_3 + \frac{1}{6}\eta_5\right)\sin\theta_A + \left(2\eta_1^2\kappa_A + \frac{1}{6}\eta_1^3\dot{\kappa}_A + \frac{1}{2}\eta_1\eta_3\kappa_A\right)\cos\theta_A \\ & + \left(10\eta_2 - 2\eta_4 + \frac{1}{6}\eta_6\right)\sin\theta_B - \left(2\eta_2^2\kappa_B - \frac{1}{6}\eta_2^3\dot{\kappa}_B - \frac{1}{2}\eta_2\eta_4\kappa_B\right)\cos\theta_B \quad (1.19)\end{aligned}$$

From a rapid analysis of (1.4)–(1.19), it can be observed their dependence on the assigned interpolating conditions $x_A, y_A, x_B, y_B, \theta_A, \theta_B, \kappa_A, \kappa_B, \dot{\kappa}_A$, and $\dot{\kappa}_B$ and on a set of six real parameters η_i. Such parameters, which give their name to the planning primitive, can be packed into a single vector $\eta := [\eta_1\ \eta_2\ \eta_3\ \eta_4\ \eta_5\ \eta_6]^T \in \mathcal{H} \subset (\mathbb{R}^+)^2 \times \mathbb{R}^4$.

Among the other characteristics of the η^3-splines, one, in particular, needs to be highlighted: η^3-splines always fulfill boundary conditions independently from the values of η which, therefore, can be used to shape the curve interior points. This is an important feature of η^3-splines since it introduces flexibility in their design. On the other hand, it forces to find an appropriate method for the selection of η. Different choices are possible: e.g., in motion planning η can be used to avoid obstacles while in a motion generation context, like that considered in this research, η can be assigned to fulfill an appropriate optimality criterion.

The control strategy proposed in [10], [13] aims at obtaining smooth robot movements by generating minimum curvature paths. Indeed, it is well knows that the path shape has a strong impact on the robot lateral solicitations. In particular, lateral accelerations are related to the path curvature while lateral jerks depend on the curvature derivative with respect to s. In order to reduce lateral stresses, η can be selected by solving the following optimization problem.

Problem 2 *Given any set of interpolating conditions $x_A, y_A, x_B, y_B, \theta_A, \theta_B, \kappa_A, \kappa_B, \dot{\kappa}_A$, and $\dot{\kappa}_B$, find the optimal η^3-spline which solves the following semi-infinite minimax problem*

$$\min_{\eta\in\mathcal{H}} \max_{u\in[0,1]} \left\{ \left| \frac{d\kappa}{ds}(u;\eta) \right| \right\} \quad (1.20)$$

subject to

$$\|\dot{\mathbf{p}}(u;\eta)\| > 0, \quad \forall u \in [0,1] \,. \quad (1.21)$$

Constraint (1.21) is added to guarantee the curve regularity.

Problem (1.20), (1.21) is strongly nonlinear and is characterized by a very large number of local minima. For this reason, it can only be solved by means of global optimization algorithms. For example, in this chapter the optimal solution is gained using the hybrid genetic-interval algorithm proposed in [14], [15]. Unfortunately, this approach can only be adopted for off-line cases, since, owing to the problem complexity, evaluation times are normally not compatible with realtime applications. Consequently, it has been necessary to devise an efficient heuristic rule to be used when computational efficiency represents an important issue. Such rule, which returns effective solutions and is characterized by an almost zero evaluation time, is described in the next section. In the same section a comparison is made with a preliminary approach proposed in [11, 12]. In particular, it will be shown how, in most practical cases, the selection method proposed in [11, 12] returns very good results from the point of view of problem (1.20), (1.21), even if better solutions can be achieved by means of the new approach.

1.2 The heuristic rule

Let us indicate by $\Gamma := [x_A\, y_A\, x_B\, y_B\, \theta_A\, \theta_B\, \kappa_A\, \kappa_B\, \dot{\kappa}_A\, \dot{\kappa}_B]^T \in \mathcal{G} \subset \mathbb{R}^4 \times [-\pi, \pi]^2 \times \mathbb{R}^4$ the vector containing the interpolating conditions used to plan a generic η^3-spline. The minimizer η^* of (1.20), (1.21) necessarily depends on Γ, so that it will be indicated in the following as $\eta^*(\Gamma)$. In order to avoid an explicit online solution of (1.20), (1.21) an algebraic function

$$\begin{aligned} \hat{\eta} : \mathcal{G} &\rightarrow \mathcal{H} \\ \Gamma &\rightarrow \hat{\eta}(\Gamma)\,, \end{aligned}$$

which at the best approximates $\eta^*(\Gamma)$, needs to be estimated. Evidently, any effort is spent to guarantee that curves generated by $\hat{\eta}(\Gamma)$ have performance indexes close to those obtained by means of $\eta^*(\Gamma)$.

A preliminary $\hat{\eta}(\Gamma)$ function was proposed in [11, 12]. More precisely, it was selected on the sole basis of the Euclidean norm between $\mathbf{p}_A$ and $\mathbf{p}_B$ according to the following rule

$$\hat{\eta}(\Gamma) := [\|\mathbf{p}_A - \mathbf{p}_B\| \ \|\mathbf{p}_A - \mathbf{p}_B\| \ 0\ 0\ 0\ 0]^T \ .$$

In this section, a new $\hat{\eta}(\Gamma)$ function, which uses all the interpolating conditions, is proposed with the purpose of generating curves with a smaller curvature derivative. The new function $\hat{\eta}(\Gamma)$ is devised through a two steps design. The first step focuses on finding a possible structure for $\hat{\eta}(\Gamma)$. In particular, the structure of $\hat{\eta}(\Gamma)$ is guessed by solving (1.20), (1.21) for a set of appropriate interpolating conditions Γ_i and analyzing the corresponding solutions $\eta^*(\Gamma_i)$. The result of such analysis is a parametric function $\hat{\eta}(\Gamma;\mathbf{k})$, where $\mathbf{k} := [k_1\ k_2\ \dots\ k_{11}]^T \in \mathcal{K} \subset \mathbb{R}^{11}$ is a vector of real parameters used for its "tuning". The first step also returns an initial proposal for $\mathbf{k}$. Subsequently, $\mathbf{k}$ is refined in the second step as the solution of a new optimization problem.

1.2.1 Devising the structure of $\hat{\eta}(\Gamma;\mathbf{k})$

The structure of $\hat{\eta}(\Gamma)$ must be characterized by its simplicity. To this purpose, let us consider some typical planning cases where the solution of problem (1.20), (1.21) is known. Evidently, when $\kappa_A = \kappa_B$, the optimal solution of (1.20), (1.21) is gained when $\frac{d\kappa}{ds}(u;\hat{\eta}) \simeq 0$, i.e., $\kappa(u;\hat{\eta})$ is kept as constant as possible along the curve or, equivalently, the curve at the best approximates a circular arc. In the same way, if $\kappa_A \neq \kappa_B$, the optimal solution is characterized by a function $\kappa(u;\hat{\eta})$ which almost linearly depends on s, so that $\frac{d\kappa}{ds}(u;\hat{\eta})$ is almost constant and the curve at the best approximates a clothoid. Bearing in mind this idea, a set of interpolating conditions Γ_i, compatible with arcs and clothoids, has been generated (see Tables 1.1 and 1.2).

For each configuration Γ_i the optimal solution $\eta^*(\Gamma_i)$ has been found by using the genetic-interval algorithm proposed in [14], [15]. As expected, when the interpolating conditions are compatible with circular arcs, problem (1.20), (1.21) converges toward solutions with $\frac{d\kappa}{ds} \simeq 0$, i.e., η^3-splines almost perfectly emulate circular arcs, while, when clothoids are emulated, it converges toward constant values of $\frac{d\kappa}{ds}$. Moreover, in the case of circular arcs, owing to the symmetry characteristics of such curve ($\kappa_A = \kappa_B$, $\dot{\kappa}_A = \dot{\kappa}_B = 0$), the minimizers show the following relationships: $\eta_1 \simeq \eta_2$, $\eta_3 \simeq -\eta_4$, and $\eta_5 \simeq \eta_6$. Minimizers $\eta^*(\Gamma_i)$, i=1,2,...,12, corresponding to circular arcs, are

Table 1.1: Interpolating conditions Γ_i compatible with circular arcs

	x_A	y_A	x_B	y_B	θ_A	θ_B	κ_A	κ_B	$\dot{\kappa}_A$	$\dot{\kappa}_B$
Γ_1	0	0	1.4142	0.5858	0	$\pi/4$	1/2	1/2	0	0
Γ_2	0	0	3.5355	1.4645	0	$\pi/4$	1/5	1/5	0	0
Γ_3	0	0	5.3033	2.1967	0	$\pi/4$	1/7.5	1/7.5	0	0
Γ_4	0	0	7.0711	2.9289	0	$\pi/4$	1/10	1/10	0	0
Γ_5	0	0	10.6066	4.3934	0	$\pi/4$	1/15	1/15	0	0
Γ_6	0	0	14.1421	5.8579	0	$\pi/4$	1/20	1/20	0	0
Γ_7	0	0	2.0000	2.0000	0	$\pi/2$	1/2	1/2	0	0
Γ_8	0	0	5.0000	5.0000	0	$\pi/2$	1/5	1/5	0	0
Γ_9	0	0	7.5000	7.5000	0	$\pi/2$	1/7.5	1/7.5	0	0
Γ_{10}	0	0	10.0000	10.0000	0	$\pi/2$	1/10	1/10	0	0
Γ_{11}	0	0	15.0000	15.0000	0	$\pi/2$	1/15	1/15	0	0
Γ_{12}	0	0	20.0000	20.0000	0	$\pi/2$	1/20	1/20	0	0

Table 1.2: Interpolating conditions Γ_i compatible with clothoids

	x_A	y_A	x_B	y_B	θ_A	θ_B	κ_A	κ_B	$\dot{\kappa}_A$	$\dot{\kappa}_B$
Γ_{13}	0	0	2.9511	0.7832	0	$\pi/4$	0	1/2	1.5915e-1	1.5915e-1
Γ_{14}	0	0	7.3776	1.9582	0	$\pi/4$	0	1/5	2.5465e-2	2.5465e-2
Γ_{15}	0	0	11.0664	2.9373	0	$\pi/4$	0	1/7.5	1.1318e-2	1.1318e-2
Γ_{16}	0	0	14.7552	3.9165	0	$\pi/4$	0	1/10	6.3662e-3	6.3662e-3
Γ_{17}	0	0	22.1327	5.8747	0	$\pi/4$	0	1/15	2.8294e-3	2.8294e-3
Γ_{18}	0	0	29.5104	7.8329	0	$\pi/4$	0	1/20	1.5915e-3	1.5915e-3
Γ_{19}	0	0	4.9107	2.7091	0	$\pi/2$	0	1/2	7.9577e-2	7.9577e-2
Γ_{20}	0	0	12.2769	6.7727	0	$\pi/2$	0	1/5	1.2732e-2	1.2732e-2
Γ_{21}	0	0	18.4152	10.1590	0	$\pi/2$	0	1/7.5	5.6588e-3	5.6588e-3
Γ_{22}	0	0	24.5538	13.5454	0	$\pi/2$	0	1/10	3.1831e-3	3.1831e-3
Γ_{23}	0	0	36.8305	20.3181	0	$\pi/2$	0	1/15	1.4147e-3	1.4147e-3
Γ_{24}	0	0	49.1075	27.0909	0	$\pi/2$	0	1/20	7.9577e-4	7.9577e-4

reported in Table 1.3.

Table 1.3: Minimizers $\eta^*(\Gamma_i)$ for problem (1.20)–(1.21) when interpolating conditions are congruent with circular arcs

	η_1, η_2	$\eta_3, -\eta_4$	η_5, η_6	$\left\lvert \frac{d\kappa^*}{ds} \right\rvert$
Γ_1	1.1881e+00	2.3650e+00	-5.7853e+00	2.1210e-05
Γ_2	3.6537e+00	1.3173e+00	-1.0960e+00	4.2403e-06
Γ_3	5.6959e+00	1.0188e+00	-3.7426e+00	1.5579e-07
Γ_4	7.6425e+00	1.4546e+00	-9.3034e+00	5.5453e-07
Γ_5	1.1565e+01	1.2535e+00	-8.9196e+00	4.6852e-08
Γ_6	1.5467e+01	1.3156e+00	-1.0510e+01	2.2694e-08
Γ_7	3.1334e+00	1.0140e-01	-8.4748e+00	2.9981e-05
Γ_8	7.5226e+00	2.0679e+00	-2.1859e+01	5.1968e-06
Γ_9	1.0618e+01	5.6298e+00	-1.5491e+01	8.2154e-07
Γ_{10}	1.5179e+01	1.6468e+00	-2.0441e+01	8.0685e-06
Γ_{11}	2.2828e+01	2.3025e+00	-3.3042e+01	3.3372e-06
Γ_{12}	2.9739e+01	8.4987e+00	-6.3444e+01	9.1094e-07

In the case of clothoids, η_1 and η_2 are no more equal, but they remain each other close. The same happens for η_3 and $-\eta_4$, and for η_5 and η_6. For example, for the clothoid whose interpolating conditions are given by Γ_{24} the resulting minimizer is $\eta_1 = 43.8944, \eta_2 = 44.8416, \eta_3 = 34.2107, \eta_4 = -28.1348, \eta_5 = -250.1721, \eta_6 = -253.6511$.

By scrutinizing optimal solutions $\eta^*(\Gamma_i)$ it has been possible to identify some correlations between them and the interpolating conditions reported in Tables 1.1 and 1.2. Such information has been used to propose the following structure for $\hat{\eta}(\Gamma;\mathbf{k})$

$$\eta_1 = k_1 \, \|\mathbf{p}_A - \mathbf{p}_B\| + k_2 \, |\theta_B - \theta_A| + k_3 \, \sqrt{|\kappa_A|}\,, \tag{1.22}$$

$$\eta_2 = k_1 \, \|\mathbf{p}_A - \mathbf{p}_B\| + k_2 \, |\theta_B - \theta_A| + k_3 \, \sqrt{|\kappa_B|}\,, \tag{1.23}$$

$$\eta_3 = k_4 \, \|\mathbf{p}_A - \mathbf{p}_B\|^2 + k_5 \, |\theta_B - \theta_A| + k_6 \, \sqrt{|\kappa_A|} + k_7 \, \sqrt{|\dot{\kappa}_A|}\,, \tag{1.24}$$

$$\eta_4 = -(k_4 \, \|\mathbf{p}_A - \mathbf{p}_B\|^2 + k_5 \, |\theta_B - \theta_A| + k_6 \, \sqrt{|\kappa_B|} + k_7 \, \sqrt{|\dot{\kappa}_B|})\,, \tag{1.25}$$

$$\eta_5 = k_8 \, \|\mathbf{p}_A - \mathbf{p}_B\|^2 + k_9 \, \sqrt{|\theta_B - \theta_A|} + k_{10} \, |\kappa_A| + k_{11} \, \sqrt{|\dot{\kappa}_A|} \, , \tag{1.26}$$

$$\eta_6 = k_8 \, \|\mathbf{p}_A - \mathbf{p}_B\|^2 + k_9 \, \sqrt{|\theta_B - \theta_A|} + k_{10} \, |\kappa_B| + k_{11} \, \sqrt{|\dot{\kappa}_B|} \, , \tag{1.27}$$

where $\|\cdot\|$ indicates the Euclidean norm and $\mathbf{k} := [k_1 \, k_2 \, \dots \, k_{11}]^T \in \mathcal{K} \subset \mathbb{R}^{11}$ is a vector of real parameters. It is easy to verify that, when boundary conditions are compatible with circular arcs, (1.22)–(1.27) correctly return $\eta_1 = \eta_2$, $\eta_3 = -\eta_4$, and $\eta_5 = \eta_6$, while different, but similar, values have to be expected in the case of clothoids. The same selection rule proposed in [11, 12] can be obtained from (1.22)–(1.27) by setting $\mathbf{k} = \mathbf{k}' := [1\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0]^T$,

An initial estimate for $\mathbf{k}$ has been found by means of a least square approach which minimizes the differences between $\eta^*(\Gamma_i)$ and $\hat{\eta}(\Gamma_i;\mathbf{k})$, for $i = 1,2,\dots,24$. The obtained value of $\mathbf{k}$, indicated in the following as $\mathbf{k}''$, is shown in Table 1.4.

1.2.2 Estimating the optimal k

Starting from $\mathbf{k}''$, it is possible to find a more "performing" value of $\mathbf{k}$. To this purpose, let us introduce the following optimization problem

$$\min_{\mathbf{k} \in \mathcal{K}} \{\mathbf{J}(\mathbf{k})\} \, , \tag{1.28}$$

where

$$\mathbf{J}(\mathbf{k}) := \sum_{i=1}^{24} \mathbf{w}_i \left[\frac{d\hat{\kappa}}{ds}(\Gamma_i;\mathbf{k}) - \left| \frac{d\kappa^*}{ds}(\Gamma_i) \right| \right]^2 \tag{1.29}$$

and where $\frac{d\hat{\kappa}}{ds}(\Gamma_i;\mathbf{k}) \; := \; \max_{u\in[0,1]} \left\{ \left| \frac{d\kappa}{ds}[u;\hat{\eta}(\Gamma_i;\mathbf{k})] \right| \right\}$ is the maximum curvature derivative obtained by means of $\hat{\eta}(\Gamma_i;\mathbf{k})$, $\mathbf{w}_i$ is the weight assigned to each interpolating condition Γ_i, while $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|$ represents the maximum curvature derivatives corresponding to the optimal solutions $\eta^*(\Gamma_i)$ of problem (1.20), (1.21). The same interpolating conditions Γ_i used for the first phase have been adopted (see Tables 1.1 and 1.2). Weights $\mathbf{w}_i$ are introduced to take into account the different order of magnitude of minimizer $\eta^*(\Gamma_i)$ (see the last column of Table 1.3). It is worth remembering that $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|$ is equal to zero when interpolating conditions are compatible with circular arcs, while it is equal to the elements of the last column of Table 1.2 in the case of clothoids. Practically, the solution of (1.28), (1.29) generates η^3-splines whose

Table 1.4: Possible optimal parameterizations for (1.22)–(1.27)

	$\mathbf{k}'$	$\mathbf{k}''$	$\mathbf{k}'''$
k_1	1	0,986215955980423	0,9900370309156421
k_2	0	0,04694051539639	0,2338305460827709
k_3	0	0,074863997949512	-0,2337321418102114
k_4	0	0,017994903356811	0,03957912032871749
k_5	0	0,233918712355343	0,1008348340478730
k_6	0	0,674868034806584	1,505166060904769
k_7	0	6,17884077781871	0,5363811172337601
k_8	0	-0,062562404082537	-0,5105585534956896
k_9	0	-35,718866041005704	-4.340011523955019
k_{10}	0	65,80182824188454	-17,91610461019005
k_{11}	0	54,58725230016439	-14,14677605082785

maximum curvature derivative is very close to the minimum achievable for the considered interpolating conditions.

Problem (1.28), (1.29) has been solved with a standard optimization algorithm whose starting point was set equal to $\mathbf{k}''$. The algorithm has converged to solution $\mathbf{k}'''$ shown in Table 1.4, consequently improving the cost index from 5.88589 down to 1.28337e-2.

The effectiveness of $\mathbf{k}'$, $\mathbf{k}''$, and $\mathbf{k}'''$ is discussed in the following with the help of two performance indexes. In particular we define Mean Squared Deviation (*MSD*) the mean, evaluated over all the interpolating conditions Γ_i, of the squared differences between $\frac{d\hat{\kappa}}{ds}(\Gamma_i;\mathbf{k})$ and $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|$, that is

$$MSD = \frac{1}{n}\sum_i \left[\frac{d\hat{\kappa}}{ds}(\Gamma_i;\mathbf{k}) - \left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|\right]^2 , \tag{1.30}$$

where $\mathbf{k} = \mathbf{k}', \mathbf{k}'', \mathbf{k}'''$, while n is the number of considered interpolating conditions Γ_i. In the same way, we define Maximum Deviation (*MD*) the following index

$$MD = \max_i \left\{\frac{d\hat{\kappa}}{ds}(\Gamma_i;\mathbf{k}) - \left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|\right\} , \tag{1.31}$$

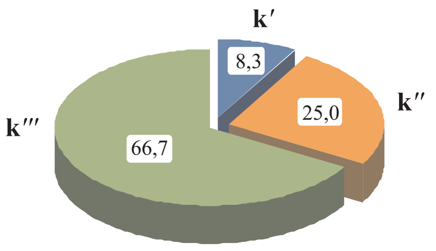
i.e., the maximum difference, evaluated over a set of interpolating conditions Γ_i, between the optimal cost indexes and those obtained by means of (1.22)–(1.27).

Fig. 1.1 shows some statistic results concerning circular arcs. They have been evaluated by considering the set of interpolating conditions of Table 1.1. The pie diagram shows the percentage of best solutions, from the point of view of the curvature derivative, among $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$. In the 66,7% of cases $\mathbf{k}'''$ exhibits the smallest cost index. The histogram in the same figure compares $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$ by means of (1.30) and (1.31), assuming $n = 12$ and $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right| = 0$. Also in this case $\mathbf{k}'''$ represents the best solution since the *MSD* and the *MD* indexes are, respectively, one order and two orders of magnitude smaller that those obtained for $\mathbf{k}'$.

In the case of clothoids, the comparisons are shown in Fig. 1.2. The pie diagram evidences that best solutions are equally spread among $\mathbf{k}'$ and $\mathbf{k}'''$. Nevertheless, some further conclusions can be drawn from the histogram. It has been evaluated by considering $i = 13, \ldots, 24$ and $n = 12$. Necessarily, terms $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|$ depend on the interpolating conditions Γ_i (see the last column in Table 1.2). The histogram reveals that the *MSD* and the *MD* indexes of $\mathbf{k}'''$ are evidently better than those of $\mathbf{k}'$. The reason of this result is that when $\mathbf{k}'$ is characterized by the best cost indexes, $\mathbf{k}'''$ has worst but similar performance indexes, while when $\mathbf{k}'''$ returns the best solutions they are neatly better than those proposed by $\mathbf{k}'$.

Owing to the method used for selecting $\mathbf{k}$, function $\hat{\eta}(\Gamma; \mathbf{k}''')$ generates curves which very well approximate circular arcs and clothoids. It could be interesting to verify what happens in the case of generic interpolating conditions. To this purpose 30 interpolating conditions Γ_i have been randomly chosen belonging to the following intervals: $x_B \in [0, 15], y_B \in [-5, 5], \theta_B \in [-\pi/2, \pi/2], \kappa_A, \kappa_B \in [-0.4, 0.4], \dot{\kappa}_A, \dot{\kappa}_B \in [-0.04, 0.04]$. Without any loss of generality, it has been supposed that $x_A = x_B = \theta_A = 0$ since, according to (1.22)–(1.27), terms η_i are evaluated on the sole basis of differences $\mathbf{p}_B - \mathbf{p}_A$ and $\theta_B - \theta_A$.

For each value of Γ_i an optimal solution $\eta^*(\Gamma_i)$ has been obtained by solving (1.20), (1.21) with the genetic-interval algorithm. The resulting cost indexes $\left|\frac{d\kappa^*}{ds}(\Gamma_i)\right|$ have been compared with the performance indexes $\frac{d\hat{\kappa}}{ds}(\Gamma_i; \mathbf{k})$ evaluated for $\mathbf{k}'$, $\mathbf{k}''$, and $\mathbf{k}'''$. The pie diagram of Fig. 1.3 shows that $\mathbf{k}'''$ can be considered the best solution in the 83% of cases. Nevertheless, $\mathbf{k}'$ and $\mathbf{k}''$ have comparable performance indexes,



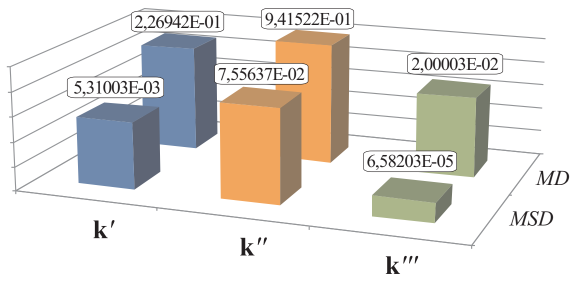
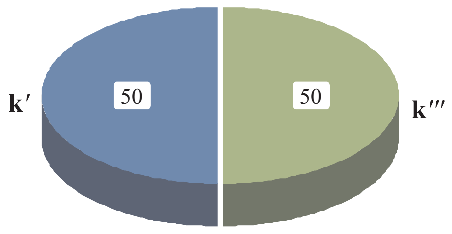


Figure 1.1: A comparison between solutions $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$ in the case of interpolating conditions compatible with circular arcs. The pie diagram reports the percentage of best solutions among $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$, while the histogram compares their Mean Squared Deviation (*MSD*) and the Maximum Deviation (*MD*). A logarithmic scale has been adopted.

as can be deduced from the histogram in the same figure. This conclusion is also confirmed by $\mathbf{J}(\mathbf{k})$: for the three cases it is respectively equal to $\mathbf{J}(\mathbf{k}') = 2,1596$, $\mathbf{J}(\mathbf{k}'') = 2,6015$, and $\mathbf{J}(\mathbf{k}''') = 1,3943$. Fig. 1.4 further proves this assertion by showing a direct comparison, for 7 of the 30 analyzed cases, between the maximum curvature derivatives obtainable with the three proposed methods and those returned by the genetic-interval algorithm. In any situation the best solutions are those devised by the genetic-interval algorithm, but the performance indexes of $\mathbf{k}'$, $\mathbf{k}''$, and $\mathbf{k}'''$ are each other comparable and very close to those of the actual minimizers.

Some conclusions can be drawn form the comparisons. Generally, $\mathbf{k}'''$ generates the smallest curvature derivatives. Even when $\mathbf{k}'$ or $\mathbf{k}''$ are characterized by smaller curvature derivatives, the performance indexes of $\mathbf{k}'''$ are only slightly worse. In the case of generic interpolating conditions $\mathbf{k}'$, $\mathbf{k}''$, and $\mathbf{k}'''$ can be considered equivalent:



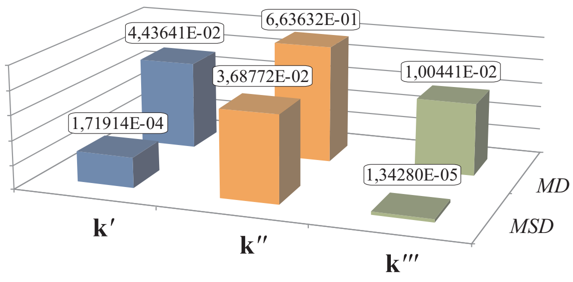


Figure 1.2: A comparison between solutions $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$ in the case of interpolating conditions compatible with clothoids. The pie diagram reports the percentage of best solutions among $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$, while the histogram compares their Mean Squared Deviation (*MSD*) and the Maximum Deviation (*MD*). A logarithmic scale has been adopted.

this result proves that the method originally proposed in [11, 12] for the selection of η represents a sufficiently good solution for problem (1.20), (1.21).

One final doubt is instilled by Fig. 1.4. It seems that, in the case of generic interpolating conditions, the selection of η is not particular critical since the cost indexes of $\mathbf{k}'$, $\mathbf{k}''$, and $\mathbf{k}'''$ are each other comparable and close to those of the global optimal solutions. This is not true, as can be evinced from the example case proposed in the next section where the η-parameters obtained from (1.22)–(1.27) and $\mathbf{k}'''$ are slightly perturbed, thus causing an immediate rise of $\dot{\kappa}$.

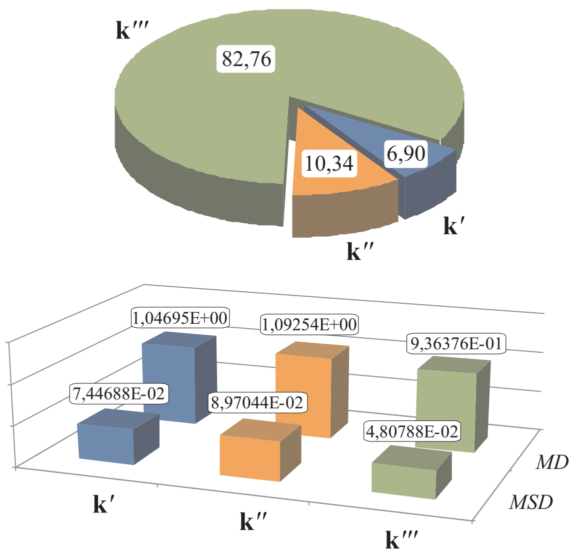


Figure 1.3: A comparison between solutions $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$ in the case of generic interpolating conditions. The pie diagram reports the percentage of best solutions among $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$, while the histogram compares their Mean Squared Deviation (*MSD*) and the Maximum Deviation (*MD*). A logarithmic scale has been adopted.

1.3 An application case

The example case proposed in the following points out the influence exerted by the curvature derivative on the motion performances of a mobile robot. Let us consider an unicycle mobile robot which must move along a composite curve planned by means of η^3-splines. The interpolating conditions used for the generation of the η^3-spline paths are listed in Table 1.5. More in details, the interpolating conditions Γ_{25}, Γ_{27} and Γ_{29} are compatible with a clothoid, a circular arc and a linear segment respectively, while interpolating conditions Γ_{26} and Γ_{28} are not compatible with any standard planning primitive in order to emulate a set of actual data obtained, e.g., from a visual system. It can be immediately evinced from Table 1.5 that the interpolating conditions of each partial curve, i.e., initial and final tangents, curvatures, and curvature derivatives, are selected such to guarantee the required continuity conditions. Necessarily, the overall composite path is G^3-continuous.

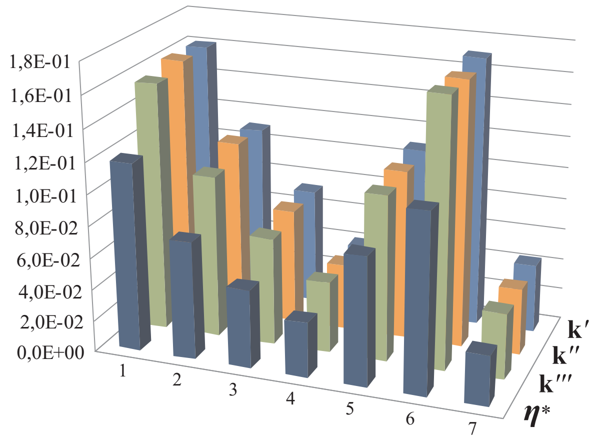


Figure 1.4: A comparison between the performance indexes of η^*, $\mathbf{k}', \mathbf{k}''$, and $\mathbf{k}'''$ for seven generic sets of interpolating conditions. A linear scale has been adopted.

Table 1.5: Interpolating condition Γ_i chosen for the example

	x_A	y_A	x_B	y_B	θ_A	θ_B	κ_A	κ_B	$\dot{\kappa}_A$	$\dot{\kappa}_B$
Γ_{25}	0	0	4.10	1.66	0	$3\pi/8$	0	1/2	0.106	0.106
Γ_{26}	4.10	1.66	7.00	10.00	$3\pi/8$	0	1/2	-0.1	0.106	0
Γ_{27}	7.00	10.00	14.07	7.07	0	$-\pi/4$	-0.1	-0.1	0	0
Γ_{28}	14.07	7.07	15.40	5.00	$-\pi/4$	$-5\pi/8$	-0.1	0	0	0
Γ_{29}	15.40	5.00	15.78	4.08	$-5\pi/8$	$-5\pi/8$	0	0	0	0

In order to verify the relevance of designing curves with minimum curvature derivative, three different scenarios have been considered. In the first case, indicated in the following as the nominal one, the η parameters are evaluated by means of (1.22)–(1.27) and coefficients $\mathbf{k}'''$ shown in Table 1.4. In the second and in the third scenarios, the perturbed cases, the previously evaluated η-parameters are slightly modified. More precisely, η_1 and η_2 have been increased and decreased respectively by the 10% with respect to the nominal case. As a result, three different composite curves satisfying the assigned interpolating conditions have been generated. It is possible to evince from Fig. 1.5 that the three curves have a very similar shape, but a

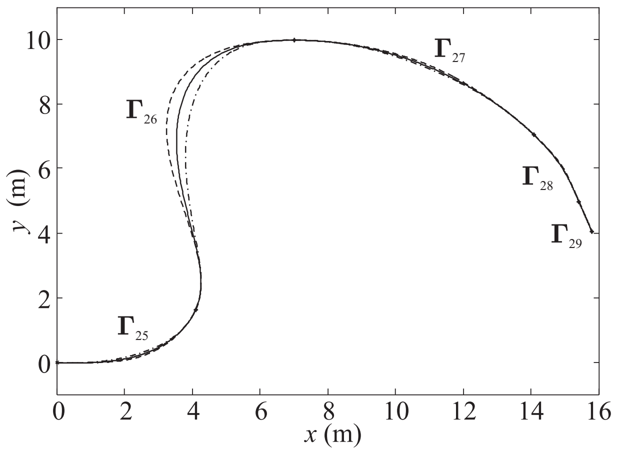


Figure 1.5: The nominal path (continuous curve) compared with the paths obtained by increasing (dashed line) or decreasing (dash-dotted line) η_1 and η_2 by the 10%.

comparison between Fig. 1.6 and Fig. 1.7, which report κ and $\dot{\kappa}$ for the nominal case and one of the two modified cases, highlights how the small perturbations introduced in η_1 and η_2 produce evident changes in the curvature and in the curvature derivative. It is worth noticing from Fig. 1.6, the evidently better emulation of a clothoid and of a circular arc obtained in the nominal case: differently from the perturbed scenario, the curvature derivative is almost constant. As expected, $\dot{\kappa}$ is generally higher in the perturbed case. The situation worsens especially in the case of Γ_{28}, thus demonstrating how the selection of η can be very critical also when generic interpolating conditions are considered.

To better point out the differences between the three composite curves, they have been tracked by an unicycle-like mobile robot driven accordingly to the control strategy proposed in [10]. The robot model used for the simulations takes into account the vehicle dynamics and the existence of sliding effects between wheels and ground. To this purpose, the wheels traction model originally proposed in [16] has been adopted. The vehicle moves at a constant longitudinal velocity, thus the shape of the acceleration profile is similar to the curvature shape, while the jerk profile mimics the curvature derivative profile. Fig. 1.8 shows the lateral acceleration and the lateral jerk acting on the vehicle during its movement along the nominal path. The detail in the

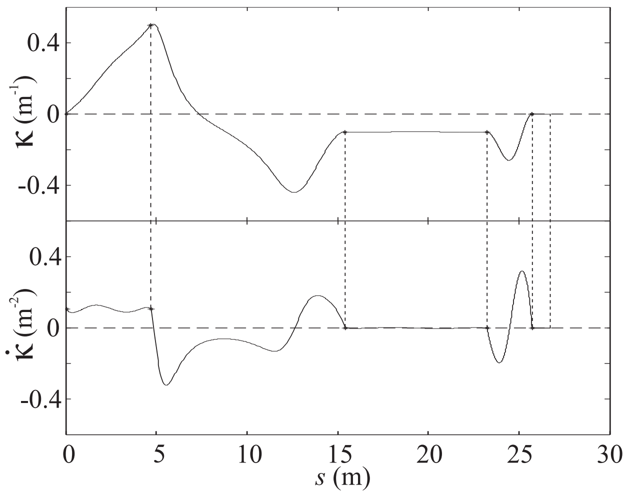


Figure 1.6: The curvature and its derivative for the nominal case, expressed with respect to curvilinear coordinate s.

same figure reveals how the lateral skidding phenomenon can appear every time lateral accelerations and jerks are sufficiently high. As previously asserted, lateral jerk is directly correlated to the curvature derivative and, consequently, the nominal case is characterized by smaller lateral solicitations, being an (almost) optimal solution for problem (1.20), (1.21). On the contrary, Fig. 1.9 reveals that if η_1 and η_2 are increased, the lateral skidding phenomenon can more easily appear owing to the higher lateral stresses acting on the vehicle. The situation does not improve when η_1 and η_2 are decreased with respect to the optimal values, as can be evinced from Fig. 1.10.

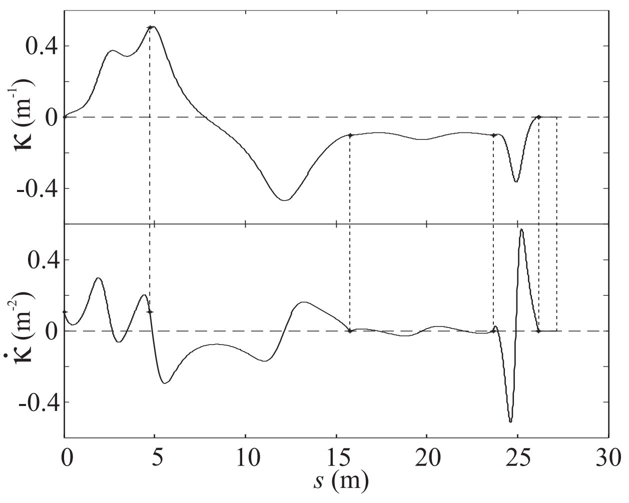


Figure 1.7: The curvature and its derivative obtained by increasing η_1 and η_2, expressed with respect to curvilinear coordinate *s*.

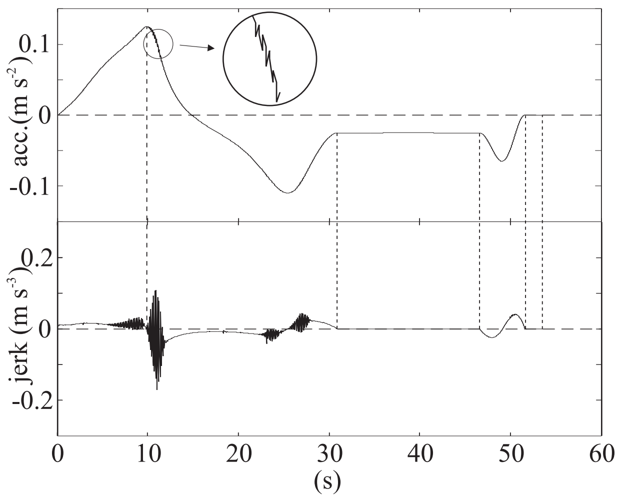


Figure 1.8: The lateral acceleration and jerk along the nominal curve, expressed with respect to the time.

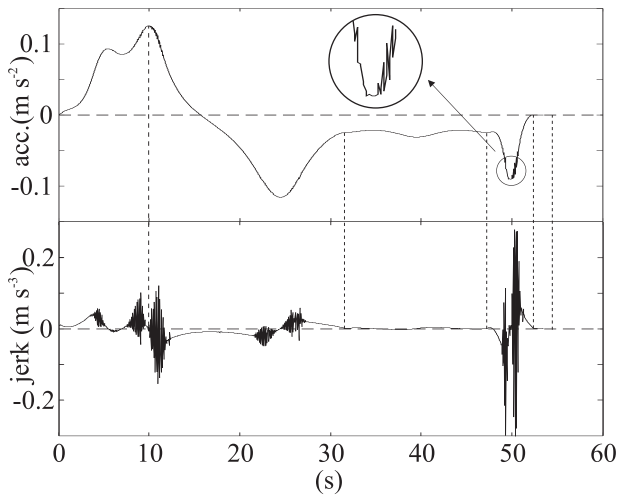


Figure 1.9: The lateral acceleration and jerk along the curve obtained by increasing η_1 and η_2 by the 10%, expressed with respect to the time.

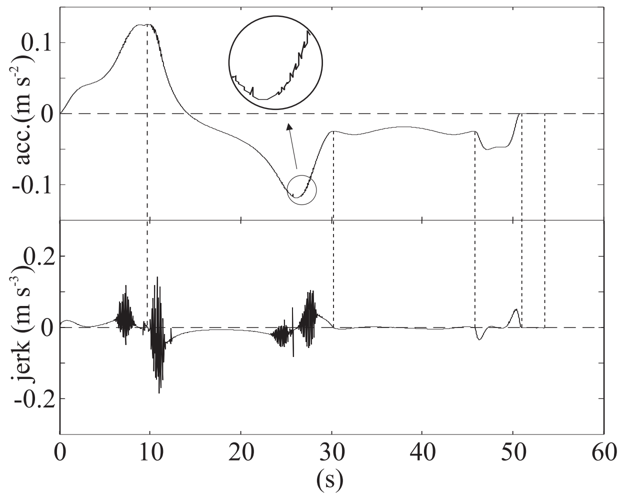


Figure 1.10: The lateral acceleration and jerk along the curve obtained by decreasing η_1 and η_2 by the 10%, expressed with respect to the time.

CHAPTER 2

Online trajectory scaling for robotic manipulators subject to torque and velocity constraints

In the middle of the journey of my life,
I found myself in a dark wood, for I had lost the right path.
Eventually I would find the right path,
but in the most unlikely place.
Dante

Motion control of industrial manipulators requires the generation of appropriate reference signals in order to improve the system performances in terms of precision and time efficiency.

In robotics, great attention has been devoted to design algorithms able to minimize the time required to complete an assigned task. The fulfillment of this requirement is crucial in order to increase the production rate in industrial applications which are often limited by the robot performances rather than the process constraints. Unfortunately, traveling time minimization leads to an increment of the mechanical solicitations. Moreover, the actuators dynamic limits can easily be exceeded, causing a degeneration of the control performances. For this reason, it is important to take into account robots dynamic and kinematic constraints during the trajectory planning phase. In the last decades, several methods have been proposed; they can be roughly divided into two groups: offline and online planners. Methods of the former group devise the optimal trajectory as the outcome of offline computations and can ulteriorly be subdivided into two different approaches. In the first one, constrained optimization algorithms are used, either determining the robot trajectory as a whole, see [17, 18, 19], or by using the path-velocity paradigm [20, 21, 22]. In the second one, a scaling factor is introduced to offline guarantee the feasibility for a given robot trajectory. Main results for nonredundant manipulators can be found in [23], while in [24] the method is extended to robots used in cooperative tasks and in [25] manipulators with elastic joints have been considered. The main drawback of these algorithms is represented by the need of a perfect knowledge of the robot model, a requirement which is often not realistic. Moreover, when minimum-time trajectory are planned, there is always at least one joint working at its torque limits: any external disturbances or robot unmodelled dynamics cannot be compensated by the controller so that tracking is lost.

To overcome these limitations, nominal trajectories are typically online modified by means of appositely devised algorithms. In case of redundant manipulators, the path tracking problem under kinematic and/or dynamic constraints is commonly solved by taking advantage of the redundancy [26, 27, 28, 29]. For nonredundant manipulators the feasibility of a given trajectory is obtained by online scaling the velocity profile used to move along an assigned path. In [30] this method has been adopted to account for joint velocity limits, while accelerations bounds are considered in [31]. Some two-level control algorithms have been proposed in [32, 33, 34] which take into account torque limits. They consist in an inner loop, based on standard feedback controllers, and in an outer loop, which slows down the robot reference velocity when torque saturations are reached.

Previously cited methods have a common denominator: the dynamic constraints are online converted into kinematics bounds on the velocity profile used for the motion along the assigned path. In a broader sense, the nominal velocity profile is somehow filtered in order to generate an output signal which fulfills the assigned bounds. A similar problem has been investigated in the past, dealing with the optimal filtering of rough reference signals for electrical axes [35, 36].

The goal of this chapter is to illustrate an online trajectory tracking control for robotic manipulators subject to torque and velocity constraints. Most of the following results have been presented in [37]. The chapter can be roughly divided in two parts. In the first one, it will be shown how to implement a feedback control scheme able to track at best a given path despite the presence of dynamic and kinematic constraints. To this purpose, a trajectory filter is required. The used filter is then extensively analyzed and improved version of [36] is presented.

The new control scheme introduces several novelties with respect to similar approaches [32, 33, 34]. First of all, not only torque constraints are considered, but also the existence of explicit limits on the maximum joint velocities is taken into account. Secondly, even if the path tracking is still the main target of the controller, now any effort is spent in order to respect the time law assigned for the movement along the curve.

The chapter is organized as follows. The robotic problem is posed in §2.1. In the same section, it is shown how joint torque and velocity constraints can be converted into equivalent kinematic constraints. Such constraints are used to scale the trajectory by means of a dynamic filter: the design and the characteristics of a new discrete-time filter are discussed in §2.2, while a detailed analysis of the filter convergence properties is reported in Section 2.5. Comparisons between the new filter and the one proposed in [32] are made in §2.3, where some practical implementation issues are also discussed. The usefulness of the approach is investigated in §2.4 by means of an example concerning a cartesian manipulator.

2.1 Online trajectory scaling for robotic manipulators

The problem here investigated is similar to that described in [32], where an online trajectory scaling filter has been proposed to account for joint torque constraints. To this purpose a two-level control scheme was designed. At the primary level, a standard feedback controller was adopted, tuned for disturbances rejection and good transient performances. At the secondary level, a dynamic filter was used to modify the nominal, and potentially rough, trajectory in order to fulfill the manipulator torque constraints and track, at the best, a given path.

In our research, the same two-level approach is assumed. The first level is represented by a standard computed torque controller, while a novel filter is used for the optimal trajectory scaling.

Some preliminary definitions can be useful for the discussion. The robot trajectory is defined according to the so-called path-velocity decomposition [20]. For this reason, the path to be followed is described in the joint space by means of a vectorial function $\Gamma(x)$ defined as follows

$$\begin{array}{rcc} \Gamma : [0, x_f] & \rightarrow & \mathbb{R}^n \\ x & \rightarrow & \mathbf{q}_d := \Gamma(x) \ . \end{array} \tag{2.1}$$

where $x \in \mathbb{R}^+$ is the scalar which parametrizes the curve, while $n \in \mathbb{N}$ is the number of the robot independent joints. Without any loss of generality, the path is assumed in the joint space. In fact, it is always possible to convert a task space path into a joint space one by means of an appropriate use of the manipulator Jacobian matrix $\mathbf{J}(\mathbf{q})$.

In the same way, a monotonically increasing time-law, used to move the end effector along $\Gamma(x)$, is defined

$$\begin{array}{rcc} x : [0, t_f] & \rightarrow & [0, x_f] \\ t & \rightarrow & x_d := x(t) \end{array} \tag{2.2}$$

where t_f is the total traveling time. Evidently, the overall robot trajectory is obtained by combining (2.1) and (2.2): $\mathbf{q}_d(t) := \Gamma(x(t))$.

Consider now a serial link rigid-body manipulator. Its generalized forces can be evaluated by means of the classical inverse dynamics equation, so that for each joint $k = 1, 2, \ldots, n$ it follows that

$$\tau_k = \sum_{j=1}^{n} h_{kj}(\mathbf{q})\, \ddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(\mathbf{q})\, \dot{q}_i\, \dot{q}_j + g_k(\mathbf{q}) + f_k(\mathbf{q}, \dot{\mathbf{q}}) \ , \tag{2.3}$$

where

$$c_{ijk} = \frac{1}{2}\left(\frac{\partial h_{kj}}{\partial q_i} + \frac{\partial h_{ki}}{\partial q_j} - \frac{\partial h_{ij}}{\partial q_k}\right) \tag{2.4}$$

are the so called Christoffel symbols of the first order. By defining the generalized force vector as $\tau := [\tau_1\ \tau_2\ \cdots, \tau_n]^T$, and introducing the new terms

$$c_{kj}(\mathbf{q},\dot{\mathbf{q}}) := \sum_{i=1}^{n} c_{ijk}(\mathbf{q})\,\dot{q}_i\ , \tag{2.5}$$

equation (2.3) can be rewritten in the well known matrix form [38]

$$\tau = \mathbf{H}(\mathbf{q})\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{f}(\mathbf{q},\dot{\mathbf{q}})\ . \tag{2.6}$$

As usual, $\mathbf{H}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the symmetric and positive definite inertia matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is the matrix of centripetal and Coriolis terms, $\mathbf{g} \in \mathbb{R}^n$ is the vector of the gravity forces, and $\mathbf{f}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^n$ describes the friction effects. The manipulator is subject to dynamic and kinematic constraints. More precisely, maximum admissible torques are bounded, so that it holds

$$\underline{\tau}_k \le \tau_k \le \overline{\tau}_k, \quad k = 1,2,\dots,n\ , \tag{2.7}$$

where $\underline{\tau}_k$ and $\overline{\tau}_k$ represent the lower and upper bounds on the k-th joint torque. Analogously, maximum joint velocities are bounded, i.e.,

$$\underline{\dot{q}}_k \le \dot{q}_k \le \dot{\overline{q}}_k, \quad k = 1,2,\dots,n\ , \tag{2.8}$$

where $\underline{\dot{q}}_k$ and $\dot{\overline{q}}_k$ represent the lower and upper bounds on the k-th joint velocity. Owing to (2.7) and (2.8) the following tracking problem can be defined.

Problem 3 *Given a manipulator described by (2.6) and a desired trajectory (2.1), (2.2), design a control law to achieve the best possible tracking subject to torque constraints (2.7) and joint velocity constraints (2.8).*

The control scheme proposed to deal with Problem 3 is shown in Figure 2.1. As early anticipated, it is based on a computed torque controller. The controller output is saturated to account for (2.7), while the robot dynamics has been modified in order to introduce the effects of (2.8). If an improper trajectory is used to drive the torque controller, saturations will cause a drastic degeneration of the tracking performances as proved in §2.4. For this reason, the trajectory controller, on the basis of the current

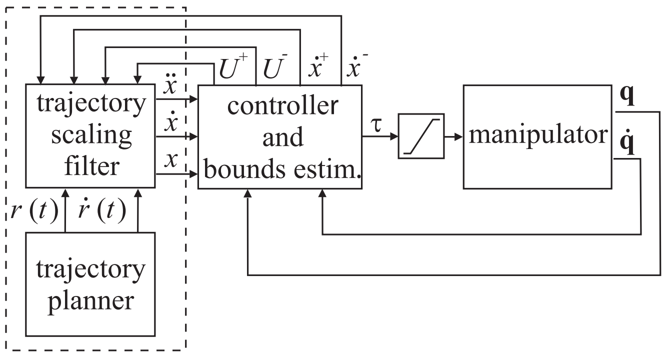


Figure 2.1: Proposed trajectory control scheme

state of motion and considering (2.7) and (2.8), dynamically evaluates equivalent acceleration and velocity bounds which must be fulfilled by time law (2.2). Such bounds are used by the nonlinear filter described in §2.2 to scale any given nominal, but possibly unfeasible, reference signal $r(t)$.

In the following it will be shown how (2.7) and (2.8) can be converted into equivalent constraints for $\dot{x}$ and $\ddot{x}$. By using the chain differentiation rule, it is possible to evaluate the trajectory time derivatives as

$$\dot{\mathbf{q}}_d = \Gamma'(x)\dot{x}\,, \tag{2.9}$$

$$\ddot{\mathbf{q}}_d = \Gamma''(x)\dot{x}^2 + \Gamma'(x)\ddot{x}\,, \tag{2.10}$$

Superscript $'$ indicates a differentiation with respect to x, e.g., $\Gamma(x)' = \frac{d\Gamma(x)}{dx}$, while, as usual, dots indicate time derivatives, e.g., $\dot{x}(t) = \frac{dx(t)}{dt}$. Due to (2.1), (2.9), and (2.10) it is always possible to compute the torque required to track a given path by means of (2.6)

$$\tau = \mathbf{b}_1(x)\ddot{x} + \mathbf{b}_2(x,\dot{x}) \tag{2.11}$$

where

$$\mathbf{b}_1(x) := \mathbf{H}(\Gamma(x))\Gamma'(x)\,, \tag{2.12}$$

$$\begin{aligned}\mathbf{b}_2(x,\dot{x}) := \;& \mathbf{H}(\Gamma(x))\Gamma''(x)\dot{x}^2 + \mathbf{C}(\Gamma(x),\Gamma'(x)\dot{x})\Gamma'(x)\dot{x} \\ & + \mathbf{f}(\Gamma(x),\Gamma'(x)\dot{x}) + \mathbf{g}(\Gamma(x))\,.\end{aligned} \tag{2.13}$$

Let us define $\mathbf{b}_1(x) := [b_{1,1}(x), b_{1,2}(x), \dots, b_{1,n}(x)]^T$ and, in a similar way, $\mathbf{b}_2(x,\dot{x}) := [b_{2,1}(x,\dot{x}), b_{2,2}(x,\dot{x}), \dots, b_{2,n}(x,\dot{x})]^T$. Due to (2.11), constraints (2.7) can be rewritten as follows

$$\underline{\tau}_i \le b_{1,i}(x)\ddot{x} + b_{2,i}(x,\dot{x}) \le \overline{\tau}_i \, , \quad i = 1,2,\dots,n \, . \tag{2.14}$$

In the same way, by using equation (2.9) inequality (2.8) became

$$\underline{\dot{q}}_i \le \Gamma_i'(x)\dot{x} \le \overline{\dot{q}}_i, \quad i = 1,2,\dots,n \, . \tag{2.15}$$

Given two torque bound vectors $\underline{\tau} := [\underline{\tau}_1 \; \underline{\tau}_2 \; \cdots \; \underline{\tau}_n]^T$ and $\overline{\tau} := [\overline{\tau}_1 \; \overline{\tau}_2 \; \cdots \; \overline{\tau}_n]^T$ and two velocity bound vectors $\underline{\dot{\mathbf{q}}} := [\underline{\dot{q}}_1 \; \underline{\dot{q}}_2 \; \dots \; \underline{\dot{q}}_n]^T$ and $\overline{\dot{\mathbf{q}}} := [\overline{\dot{q}}_1 \; \overline{\dot{q}}_2 \; \dots \; \overline{\dot{q}}_n]^T$, it is possible to define the *admissible region* (AR) [21] as the set of points in the $(x,\dot{x})$-plane where (2.15) is satisfied and where there exists at least one value $\ddot{x}$ which fulfills (2.14). It is worth noting that the AR does not depend on time law (2.2). Conversely, it only depends on the path (2.1) and on the robot dynamics (2.6).

A time law $x(t)$ assigned to move along the path is feasible, and the overall robot trajectory is feasible, if and only if all points $(x(t),\dot{x}(t))$ belong to the AR for any $t \in [0,t_f]$. Independently from the adopted controller, trajectory is lost any time a non-feasible velocity profile is used.

In case of online evaluation of the admissible region, more realistic bounds on $\dot{x}$ and $\ddot{x}$ which consider also the output of the feedback controller are required (see also [32]). For example, if a computed torque controller is considered, the output torque τ is evaluated as follows:

$$\tau(\mathbf{q}_d,\dot{\mathbf{q}}_d,\ddot{\mathbf{q}}_d) = \mathbf{H}(\mathbf{q}_d)\ddot{\mathbf{q}}_q + \mathbf{C}(\mathbf{q}_d,\dot{\mathbf{q}}_d)\dot{\mathbf{q}}_d + \mathbf{f}(\mathbf{q}_d,\dot{\mathbf{q}}_d) + \mathbf{g}(\mathbf{q}_d) + \mathbf{k}_p^T\,\mathbf{e} + \mathbf{k}_v^T\,\dot{\mathbf{e}} \tag{2.16}$$

where $\mathbf{k}_p, \mathbf{k}_v \in (\mathbb{R}^+)^n$ are the controller gain vectors and $\mathbf{e} := \mathbf{q} - \mathbf{q}_d$, $\dot{\mathbf{e}} := \dot{\mathbf{q}} - \dot{\mathbf{q}}_d$ are, respectively, the trajectory tracking error and its derivative. Equation (2.16) can be synthetically rewritten as

$$\tau(x,\dot{x},\ddot{x},\mathbf{q},\dot{\mathbf{q}}) = \mathbf{b}_1(x)\ddot{x} + \widetilde{\mathbf{b}}_2(x,\dot{x},\mathbf{q},\dot{\mathbf{q}}) \tag{2.17}$$

where $\mathbf{b}_1(x)$ is defined according to (2.12), while

$$\widetilde{\mathbf{b}}_2(x,\dot{x},\mathbf{q},\dot{\mathbf{q}}) := \mathbf{b}_2(x,\dot{x}) + \mathbf{k}_p^T\mathbf{e} + \mathbf{k}_v^T\dot{\mathbf{e}} \, , \tag{2.18}$$

with $\widetilde{\mathbf{b}}_2(x,\dot{x},\mathbf{q},\dot{\mathbf{q}}) := [\widetilde{b}_{2,1}(x,\dot{x},\mathbf{q},\dot{\mathbf{q}}),\widetilde{b}_{2,2}(x,\dot{x},\mathbf{q},\dot{\mathbf{q}}),\dots,\widetilde{b}_{2,n}(x,\dot{x},\mathbf{q},\dot{\mathbf{q}})]^T$ and $\mathbf{b}_2(x,\dot{x})$ equal to (2.13).

Owing to (2.12) and (2.18), given the current status of motion $(x,\dot{x})$, and torque bounds $\underline{\tau}_k,\overline{\tau}_k$, for each actuated joint the acceleration upper bound ϕ_k and lower bound ψ_k are obtained by rearranging (2.14). In particular, it holds that

$$\phi_k = \begin{cases} \frac{\overline{\tau}_k-\widetilde{b}_{2,k}}{b_{1,k}}, & \text{if } b_{1,k}>0 \\ \frac{\underline{\tau}_k-\widetilde{b}_{2,k}}{b_{1,k}}, & \text{if } b_{1,k}<0 \\ \infty, & \text{if } b_{1,k}=0 \end{cases} \quad \text{and} \quad \psi_k = \begin{cases} \frac{\underline{\tau}_k-\widetilde{b}_{2,k}}{b_{1,k}}, & \text{if } b_{1,k}>0 \\ \frac{\overline{\tau}_k-\widetilde{b}_{2,k}}{b_{1,k}}, & \text{if } b_{1,k}<0 \\ -\infty, & \text{if } b_{1,k}=0 \end{cases}$$

Since bounds ϕ_k and ψ_k must be simultaneously fulfilled $\forall k=1,2\dots,n$, any feasible acceleration $\ddot{x}$ must belongs to the range $[U^-,\, U^+]$ where

$$U^+ := \min_{k=1,\dots,n}\{\phi_k\}\,, \quad U^- := \max_{k=1,\dots,n}\{\psi_k\}\,. \tag{2.19}$$

As long as $U^+ \geq U^-$ current state $(x,\dot{x})$ lies inside the AR.

In a similar way, an online strategy to evaluate bounds on admissible velocity $\dot{x}$ can be defined. First of all, since negative velocities along the path are not allowed, it has been assigned $\dot{x}^- := 0$. Instead, the upper bound can be evaluated, due to (2.15), by means of the following relation

$$\rho_k = \begin{cases} \frac{\dot{\overline{q}}_k}{\Gamma_k'(x)}, & \text{if } \Gamma_k'(x)>0 \\ \frac{\dot{\underline{q}}_k}{\Gamma_k'(x)}, & \text{if } \Gamma_k'(x)<0 \\ \infty, & \text{if } \Gamma_k'(x)=0 \end{cases} .$$

Velocity $\dot{x}$ is feasible only if it lies in the interval $[\dot{x}^-,\, \dot{x}^+]$ where

$$\dot{x}^+ := \min_{k=1,\dots,n}\{\rho_k\}\,, \quad \dot{x}^- := 0\,. \tag{2.20}$$

2.2 Nonlinear bounded-dynamics filter

The dynamic system shown in Fig. 2.2 has been successfully used in the past to generate smooth set-points for motion control systems, as reported in [35, 36]. In the following, a new nonlinear filter, based on a similar scheme, is devised to solve Problem 3. It is able to automatically online modify a given scalar trajectory to satisfy limits on $\dot{x}^-, \dot{x}^+, U^-$, and U^+ which derive from (2.19) and (2.20). Since (2.19) does not guarantee symmetric bounds on the acceleration, it has not been possible to directly use the filter proposed in [36], but it has been necessary to completely redesign the control law which drives the double integrator chain.

Let us consider the following design problem

Problem 4 *Design a nonlinear discrete-time filter whose output x tracks "at best" a given reference signal r by fulfilling the following requirements:*

1) the first and second time derivatives of x must be bounded:

$$\dot{x}^- \leq \dot{x} \leq \dot{x}^+, \qquad U^- \leq \ddot{x} \leq U^+ \, , \tag{2.21}$$

where $\dot{x}^-, \dot{x}^+ \in \mathbb{R}$, $U^+ \in \mathbb{R}^+$ and $U^- \in \mathbb{R}^-$.

2) bounds (2.21) can be time-varying and can also change during transients;

3) if (2.21) is not satisfied owing to the filter initial conditions or to a sudden change of the bounds, $\ddot{x}$ must be forced in a single step within the given limits, while $\dot{x}$ must reach the assigned bounds in minimum time;

4) when a reference signal r satisfying (2.21) is applied, the tracking condition $x = r$ is reached in minimum time and without overshoot;

5) when a discontinuous reference signal is applied (or the reference signal has time derivatives larger than the bound values), the tracking is lost. As soon as the reference signal newly satisfies (2.21), tracking is achieved in minimum time;

6) the time derivatives $\dot{x}$ and $\ddot{x}$ of the bounded output must be available for the generation of feedforward actions.

Problem 4 is an optimal minimum-time tracking problem subject to bounded dynamic signals. As early anticipated, its optimal solution is based on a chain of two

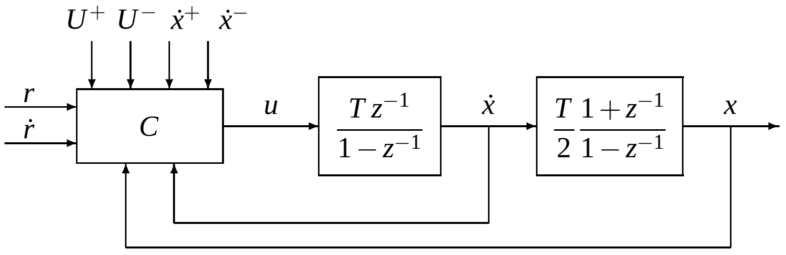

U^+ U^- $\dot{x}^+$ $\dot{x}^-$
r
$\dot{r}$
C
u
$\frac{T z^{-1}}{1-z^{-1}}$
$\dot{x}$
$\frac{T}{2}\frac{1+z^{-1}}{1-z^{-1}}$
x

Figure 2.2: The optimal bounded-dynamics trajectory tracker.

integrators like that shown in Fig. 2.2, whose dynamic equation is

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} u_k \, , \tag{2.22}$$

where T is the system sampling time, $(x, \dot{x})$ is the internal state, while u is the control command of the integrator chain. Subscript k indicates the sample number, so that u_k represents the command signal at time $t_k = kT$.

The integrators are driven by an algebraic discrete-time nonlinear controller C designed by means of variable structure control techniques [39]. In order to meet the requirements imposed by Problem 4, the following control law C is proposed

$$C: \quad u_k := \begin{cases} U^- \mathrm{sat}(\sigma_k) & \text{if } \sigma_k \geq 0 \\ -U^+ \mathrm{sat}(\sigma_k) & \text{if } \sigma_k < 0 \end{cases} \tag{2.23}$$

$$\sigma_k := \dot{z}_k - \dot{\tilde{z}}_k \, , \tag{2.24}$$

where $\dot{z}_k$ and $\dot{\tilde{z}}_k$ are evaluated by means of the following expressions

$$\dot{z}^+ := -\frac{\dot{x}^+ - \dot{r}_k}{TU^-} \, , \tag{2.25}$$

$$z^+ := -\lceil \dot{z}^+ \rceil \left[\dot{z}^+ - \frac{\lceil \dot{z}^+ \rceil - 1}{2} \right] \, , \tag{2.26}$$

$$\dot{z}^- := \frac{\dot{x}^- - \dot{r}_k}{TU^+} \, , \tag{2.27}$$

$$z^- := \lceil -\dot{z}^- \rceil \left[-\dot{z}^- - \frac{\lceil -\dot{z}^- \rceil - 1}{2} \right] \, , \tag{2.28}$$

$$[\alpha\ \beta] := \begin{cases} [U^+\ \ U^-] & \text{if } \dfrac{y_k}{T}+\dfrac{\dot{y}_k}{2} > 0 \\ [U^-\ \ U^+] & \text{if } \dfrac{y_k}{T}+\dfrac{\dot{y}_k}{2} \leq 0 \end{cases}, \tag{2.29}$$

$$z_k := \frac{1}{T\alpha}\left|\frac{y_k}{T}+\frac{\dot{y}_k}{2}\right| , \tag{2.30}$$

$$\gamma_k := \begin{cases} z^+ & \text{if } z_k < z^+ \\ z_k & \text{if } z^+ \leq z_k \leq z^- \\ z^- & \text{if } z_k > z^- \end{cases}, \tag{2.31}$$

$$m_k := \text{Int}\left\lceil \frac{1+\sqrt{1+8\,|\gamma_k|}}{2} \right\rceil , \tag{2.32}$$

$$\dot{\tilde{z}}_k := -\frac{\gamma_k}{m_k} - \frac{m_k-1}{2}\text{sgn}(\gamma_k) , \tag{2.33}$$

$$\dot{z}_k := \begin{cases} \dfrac{\dot{y}_k}{T\,|\alpha|} & \text{if } \left[(z_k \geq 0\ \&\ \dfrac{\dot{y}_k}{T\,|\alpha|} \leq \dot{\tilde{z}}_k)\text{or } (z_k < 0\ \&\ \dfrac{\dot{y}_k}{T\,|\alpha|} \geq \dot{\tilde{z}}_k)\right]; \\ \dfrac{\dot{y}_k}{T\,|\beta|} + \left(\dfrac{m_k-1}{2}+\dfrac{|\gamma_k|}{m_k}\right)\dfrac{\alpha+\beta}{|\beta|} & \text{otherwise}, \end{cases} \tag{2.34}$$

and where r_k is the sampled reference signal, $\dot{r}_k$ is the corresponding discrete-time derivative, $y_k := x_k - r_k$ is the filter tracking error, $\dot{y}_k := \dot{x}_k - \dot{r}_k$ is the filter velocity error. Function $\lceil\cdot\rceil$ provides the upper integer part of its argument, while sat$(\cdot)$ saturates its argument to ± 1. Signals r_k and $\dot{r}_k$ are assumed to be known. Moreover, $\dot{r}_k$ is supposed to be piece-wise constant.

The filter behavior is summarized in the following with the help of Figs. 2.3 and 2.4. The interested reader can find the demonstrations of the filter convergence properties in Section 2.5.

The aim of controller C is to force the system state $(y,\dot{y})$ toward the origin of the phase plane since this implies, according to the definition of y and $\dot{y}$, that a perfect tracking of r is reached. This result must be achieved in minimum time and by satisfying, if possible, the given constraints on the maximum velocity and acceleration. To this purpose, any point in the $(y,\dot{y})$-plane is transformed into an equivalent one in the $(z,\dot{z})$-space by means of (2.25)–(2.34). It is possible to verify that such mapping is bijective and the origin of the two spaces coincides. As a consequence, tracking is achieved if controller C is able to force the state $(z,\dot{z})$ and, in turn, $(y,\dot{y})$

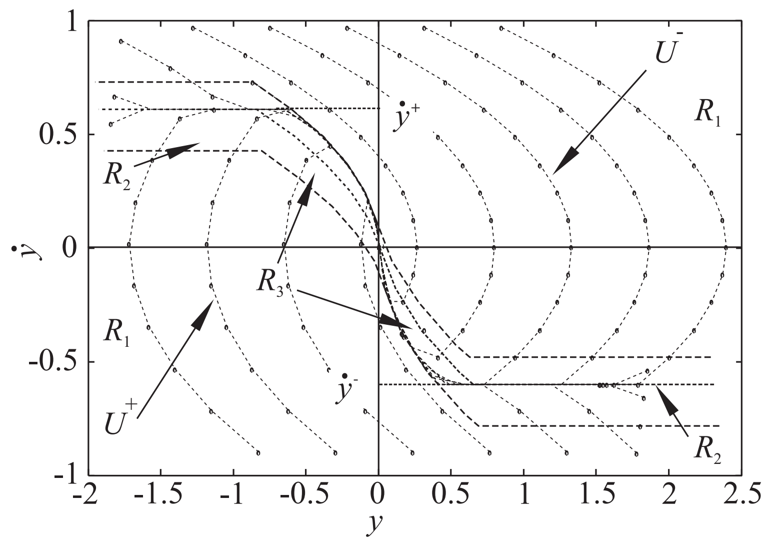


Figure 2.3: The $(y,\dot{y})$ phase plane.

toward the origin. The two constants $\dot{z}^+$ and $\dot{z}^-$ represent the transformed values in the $(z,\dot{z})$-plane of velocity constraints $\dot{x}^+$ and $\dot{x}^-$. Analogously, $\dot{y}^+$ and $\dot{y}^-$ represent the transformed values in the $(y,\dot{y})$-plane of the same constraints. Since the filter stability requires $\dot{z}^+ \in \mathbb{R}^+$ and $\dot{z}^- \in \mathbb{R}^-$ (see Appendix 2.5), the following condition must necessarily hold

$$\dot{x}^- \leq \dot{r} \leq \dot{x}^+. \tag{2.35}$$

From a practical point of view, control law (2.23)–(2.34) creates a sliding surface in the phase plane, whose equation, due to (2.24), is clearly given by (2.33). The sliding surface has been planned such that it monotonically decreases when $z \in [z^+, z^-]$, while it becomes constant and equal to $\dot{\tilde{z}} = \dot{z}^+$ if $z \leq z^+$ or $\dot{\tilde{z}} = \dot{z}^-$ if $z \geq z^-$. Sliding surface $\dot{\tilde{z}}$ is surrounded by a boundary layer (BL): if the filter state is outside such BL, the command signal is $u = U^+$ or $u = U^-$ otherwise u lies in the range $[U^-, U^+]$. In this way, being $\ddot{x} = u$, the constraint on the maximum acceleration is automatically fulfilled.

Fig. 2.3 shows some system trajectories in the $(y,\dot{y})$-plane by considering different starting conditions, while Fig. 2.4 shows the same trajectories in the transformed

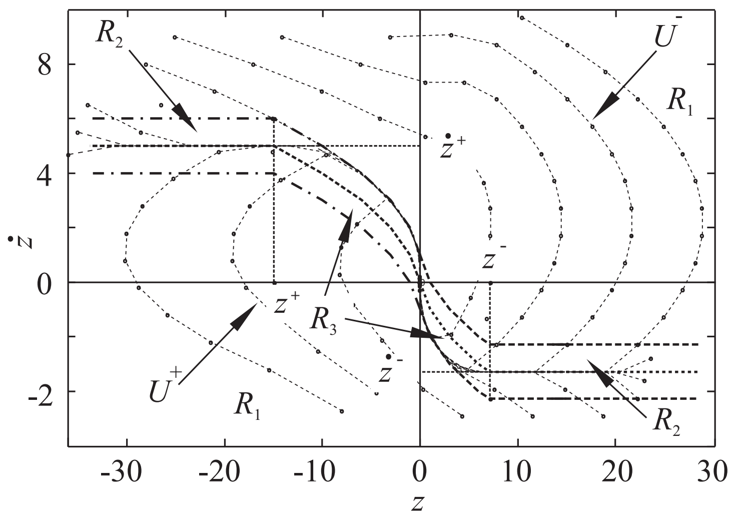


Figure 2.4: The $(z, \dot{z})$ phase plane.

domain. From the two figures it is possible to deduce that the origin of the $(z,\dot{z})$-plane is reached in two steps: the system state is first driven toward $\dot{\tilde{z}}$, then it slides along such surface by pointing to the origin. When outside the BL (region R_1), transients are obtained by applying the maximum command signal: the BL is reached with certainty and in minimum time, as demonstrated in Section 2.5.2. Control law C guarantees that the BL cannot be crossed: as soon as the system state reaches region R_2, with a single step it is forced to the sliding surface and, then, it slides toward the origin with command signal $u = 0$ (see Section 2.5.3). Finally, $(z,\dot{z})$ enters region R_3 and, again with a single step, it is forced to the frontier of the BL: the origin is reached by applying the maximum command signal and with a deadbeat behavior (see Appendix 2.5.4). Apart from the two single-step transients from R_1 to R_2 and from R_2 to R_3, the command signal is always $u \in \{U^-, 0, U^+\}$, i.e., the controller has a bang-zero-bang behavior.

From Figs. 2.3 and 2.4 it can be evinced that if the constraint on the maximum velocity is violated, e.g., $\dot{z} \notin [\dot{z}^-, \dot{z}^+]$ for a sudden change of the given bounds, the system is forced within the new bounds by applying the maximum control action, i.e., in minimum time as required by point 3) of Problem 4.

2.3 Comparison with the Dahl-Nielsen filter and applicative issues

Several details diversify the approach proposed in [32] with the one here devised. The first is particularly clear since in [32] explicit bounds on the joint velocities were not considered. Another one is less evident but very important. In [32] the main emphasis was posed on an accurate path tracking. The time law assumed for the motion along the path was defined through a function $\dot{r}(r)$, i.e., by associating a desired velocity to each point along the path. An accurate path tracking can be achieved by means of that method, but any time-delay caused by saturations cannot be recovered: as long as saturations cease, the system automatically assume the velocity planned for the current path position, so that time-delays accumulate along the trajectory, reducing the robot productivity. The filter proposed in this work assumes a time law directly defined in the time domain according to (2.2). Two advantages descend from this choice. The first is that reference signal $r(t)$ can be generated in a natural way by means of standard planning methods. The second is that any delay accumulated due to saturations is extinguished as soon as dynamic conditions will make it possible: efficiency is preserved and, at the same time, a good path tracking is achieved.

Some remarks can be useful in order to adopt the filter for actual applications. The first issue to be pointed-out is the same already highlighted in [32]. When U^+ approaches U^-, filter state $(x,\dot{x})$ is clearly moving toward the boundary of the AR. This is clearly a dangerous situation since the limitedness of available dynamics – remember that $\ddot{x} \in [U^-,U^+]$ – makes it difficult to move away from the boundary of the AR, so that tracking can easily be lost because torque constraints are violated. In [32] the problem was solved by means of a dynamic filter used to scale reference trajectory $r(t)$ on the basis of the tracking error. Owing to the characteristics of the filter here proposed, several solutions are possible, all of them based on an appropriate reduction of $\dot{x}^+$. For example, in the next section the value of $\dot{x}^+$ obtained by means of (2.20) is replaced by $\dot{\tilde{x}}^+$ where

$$\dot{\tilde{x}}^+ := \begin{cases} \dot{x}^+ & \text{if } \min\{U^+,-U^-\} \geq \overline{U} \\ K\dot{x}^+ \min\{U^+,-U^-\} & \text{if } \min\{U^+,-U^-\} < \overline{U} \end{cases} . \tag{2.36}$$

Table 2.1: Robot link inertial parameters

Link	Mass	Center of gravity			Inertia			Friction
q	m (Kg)	x(m)	y(m)	z(m)	$I_{xx}(Kg.m^2)$	$I_{yy}(Kg.m^2)$	$I_{zz}(Kg.m^2)$	$B(N.s/rad)$
d_1	23.90	0	0	0.090	2.171	2.171	0.358	1.5e-3
d_2	3.88	0	0	0.048	0.336	0.336	0.026	2.8e-3

If dynamic bounds U^+ and $-U^-$ are sufficiently large, velocity bound $\dot{x}^+$ is not scaled, otherwise it is reduced in order to force $(x, \dot{x})$ toward the AR. Constant K is chosen such that $K \min\{U^+, -U^-\} \leq 1$, while $\overline{U}$ represents the activation threshold of the scaling method.

The convergence properties of the filter are valid until the hypothesis (2.35) is fulfilled. This condition cannot be guaranteed a priori since $\dot{x}^+$ and $\dot{x}^-$ are continuously modified. The solution to this problem is straightforward, since it is sufficient to force $\dot{r}$ between the assigned bounds: velocity tracking is lost, but the filter remains stable, so that the nominal reference $r(t)$ is newly gained as soon as $\dot{r}$ returns inside the interval $[\dot{x}^-, \dot{x}^+]$.

2.4 Simulation results

In order to show the effectiveness of the filter when applied to the path tracking problem, a two-link planar robot has been considered. The manipulator dynamic parameters are defined according to Table 2.1.

The manipulator path is an ellipsoid represented by means of a curve in the joint space parametrized with respect to angle $\theta \in [0,\ 2\pi]$, i.e., $\Gamma(\theta) := [\Gamma_1(\theta)\ \Gamma_2(\theta)]^T$, where

$$\begin{cases} \Gamma_1(\theta) & := & 0.4(1-\cos(\theta)) \\ \Gamma_2(\theta) & := & 0.8\sin(\theta) \end{cases} .$$

The trajectory is completely defined once time-law $r(t) = \theta_d(t)$ is assigned. Function $\theta_d(t)$ has been chosen such that it lies within the nominal AR, but, at the same time, it is too demanding with respect to the robot velocity constraints

$$r(t) = \theta_d(t) := \begin{cases} \frac{\pi}{12}t^2, & 0 \leq t \leq 2 \\ \frac{\pi}{3}(t-1), & 2 \leq t \leq 6 \\ \frac{\pi}{6}(t+4), & 6 \leq t \leq 8 \end{cases} . \tag{2.37}$$

Corresponding $\dot{\theta}_d(t)$ can be obtained straightforward.

Simulations are carried out by considering joint velocities and torques constrained between the following bounds: $|\dot{q}_i| \leq 0.65$ s^{-1} and $|\tau_i| \leq 15$ N, $i = 1,2$. The feedback controller is a standard computed torque controller with feedback gains equal to $\mathbf{k}_p = [200\ 200]^T$ and $\mathbf{k}_v = [60\ 60]^T$.

Figure 2.5 shows what happens if (2.37) is directly applied to the robot torque controller. In particular, Figures 2.5b-2.5c highlight that when $\theta \simeq 2.5$ rad, joint velocity $\dot{q}_2$ reaches the maximum admissible value, so that trajectory tracking is lost. After a few time also τ_2 saturates and the situation worsens. The generated path is shown in Figure 2.5a compared with the planned one: the maximum error is equal to $e_{max} = \max_{\theta \in [0,2\pi]}\{\|\mathbf{e}\|\} = 0.1619$ m.

With the use of the proposed filter the situation neatly improves. Figs. 2.6c and 2.6d show that when $\dot{q}_2$ saturates, τ_2 decreases owing to the filter. As soon as $\dot{q}_2$ exits from the saturation condition, τ_1 increases until it saturates: during this phase the system is trying to eliminate the time-delay accumulated with respect to $\theta_d(t)$ due to the trajectory scaling. This conclusion can also be evinced from Fig. 2.7b which compares reference signal $\theta_d(t)$ with the actual $\theta(t)$: the time instant when tracking is lost is clearly shown, as well as the moment when tracking is newly gained.

Figs. 2.6a and 2.6b compare current $\dot{\theta}(t)$ and $\ddot{\theta}(t)$ with the bounds $U^+, U^-, \dot{x}^+$, and $\dot{x}^-$, obtained by means of (2.19) and (2.20). The asymmetry of U^+, U^- is clearly shown and justifies the use of the proposed filter. Since all constraints are always satisfied, a very accurate path tracking is achieved. Fig. 2.7a shows that path tracking error reduces of several order of magnitude with respect to the previous case: in the worst situation it is close to 1.816e-4 m. A further analysis of Figs. 2.6c and 2.6d highlight another filter feature: until reference signal $\theta_d(t)$ lies inside the AR the filter has a bypass behavior, while it starts working when saturations are touched. As early mentioned, when the system saturates the tracking of $\theta_d(t)$ is temporarily lost,

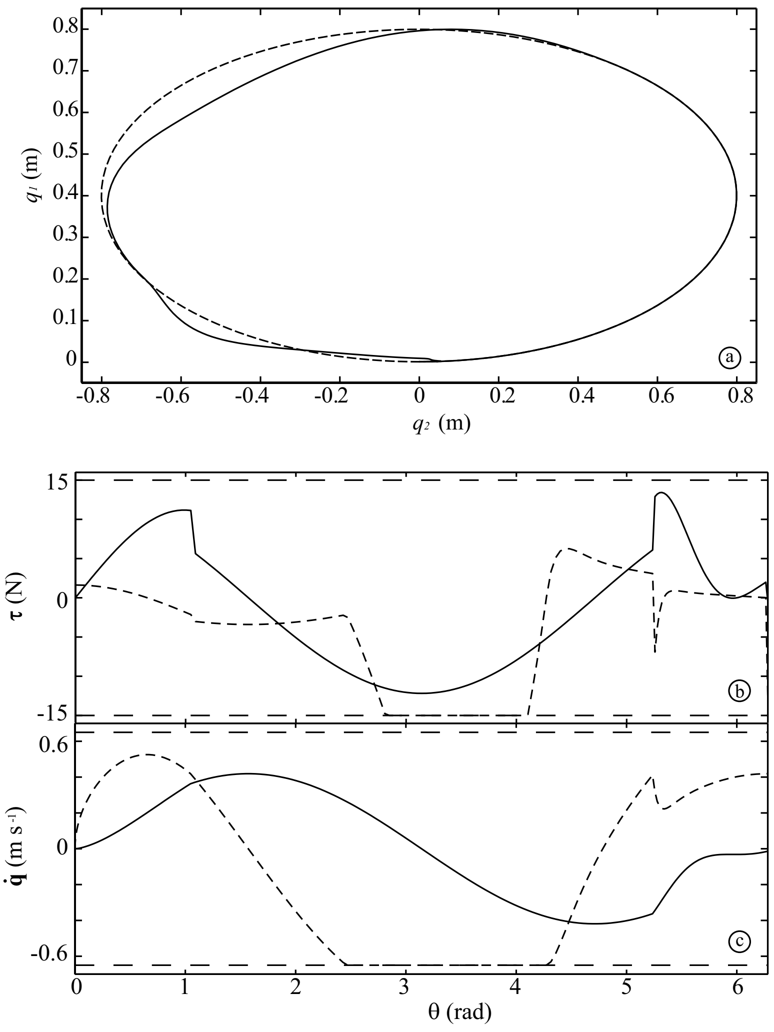


Figure 2.5: Tracking performances of a standard torque controller due to joint velocity and torque saturations: (a) reference robot path (dashed line) compared with the actual robot path (solid line); (b) joint torques τ_1 (solid line) and τ_2 (dashed line); (c) joint velocities $\dot{q}_1$ (solid line) and $\dot{q}_2$ (dashed line).

but the system immediately starts trying to hung-up $\theta_d(t)$ in minimum time, as can be noted by the bang-bang behavior shown in Figs. 2.6c and 2.6d: until $\theta_d(t)$ is not reached there is always one active torque or velocity constraint.

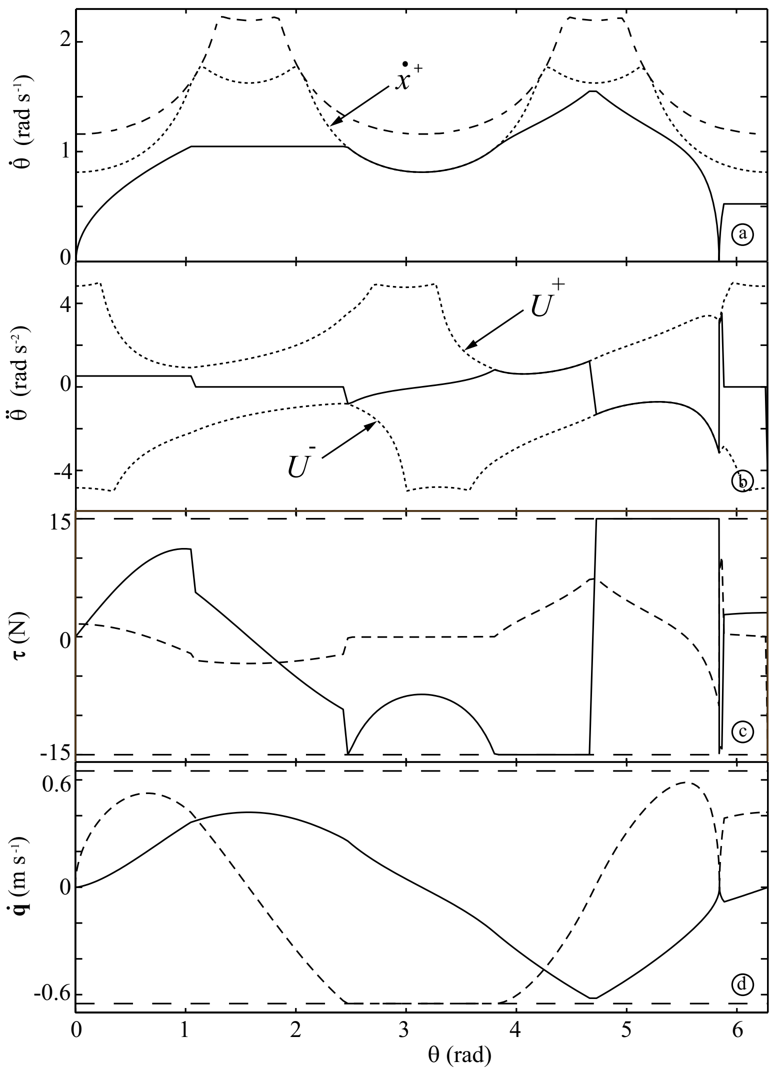


Figure 2.6: Simulation results using the nonlinear trajectory scaling filter: (a) longitudinal velocity $\dot{x}$ (solid line), online evaluated velocity bound $\dot{x}^+$(dotted line) and nominal AR (dashed line); (b) longitudinal acceleration $\ddot{x}$ (solid line) and online acceleration bounds U^+ and U^- (dotted lines); (c) joint torques τ_1 (solid line) and τ_2 (dashed line); (d) joint velocities $\dot{q}_1$ (solid line) and $\dot{q}_2$ (dashed line).

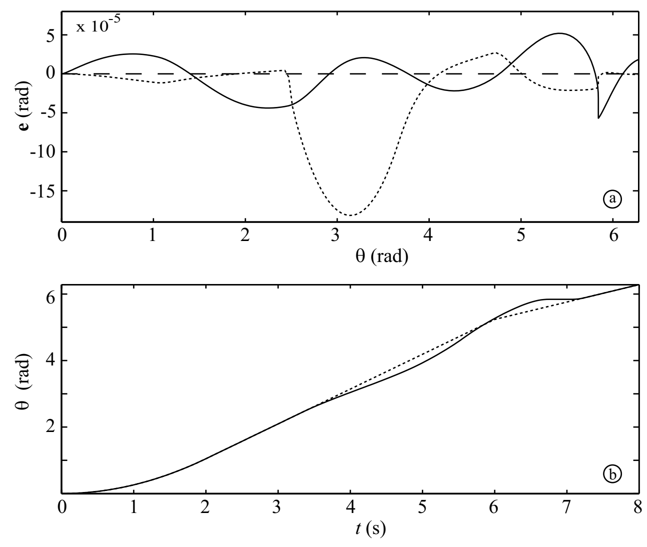


Figure 2.7: Simulation results using the nonlinear trajectory scaling filter: (a) path tracking errors e_1 (solid line) and e_2 (dashed line); (b) comparison between $\theta(t)$ (solid line) and $\theta_d(t)$ (dashed line).

2.5 Stability proofs

In the following, the stability of the proposed filter is proved. Simultaneously, some relevant properties of the same filter are highlighted. The discussion reported hereafter will analyze a system evolution starting from a point $(z, \dot{z})$ located in the left plane of the $(z, \dot{z})$-space, i.e. such that $z \leq 0$. An analogous discussion holds when $z > 0$: the corresponding demonstrations are omitted for conciseness.

2.5.1 General properties

It is easy to verify that, when $z \leq 0$, (2.29) returns $[\alpha\ \beta] := [U^-\ U^+]$, so that equations (2.30)–(2.34) simplify as follows

$$z_k := -\frac{1}{TU^-}\left(\frac{y_k}{T} + \frac{\dot{y}_k}{2}\right), \tag{2.38}$$

$$\gamma_k := \begin{cases} z^+ & \text{if } z_k < z^+ \\ z_k & \text{if } z^+ \leq z_k \leq 0 \end{cases} , \tag{2.39}$$

$$m_k := \text{Int}\left[\frac{1+\sqrt{1-8\gamma_k}}{2}\right] , \tag{2.40}$$

$$\dot{\tilde{z}}_k := -\frac{\gamma_k}{m_k} - \frac{m_k-1}{2}\text{sgn}(z_k) , \tag{2.41}$$

$$\dot{z}_k := \begin{cases} -\dfrac{\dot{y}_k}{TU^-} & \text{if } \left(-\dfrac{\dot{y}_k}{TU^-} \geq \dot{\tilde{z}}_k\right) \\ \dfrac{\dot{y}_k}{TU^+} + \left(\dfrac{m_k-1}{2} - \dfrac{\gamma_k}{m_k}\right)\dfrac{U^+ + U^-}{U^+} & \text{if } \left(-\dfrac{\dot{y}_k}{TU^-} < \dot{\tilde{z}}_k\right) \end{cases} \tag{2.42}$$

The following two properties have general validity and will be used in the last part of the section to prove the system stability.

Property 1 *For any point* $(z_k, \dot{z}_k)$ *lying inside the BL the filter command signal is given by*

$$u_k := -\frac{\dot{y}_k}{T} + \left(\frac{\gamma_k}{m_k} - \frac{m_k-1}{2}\right)U^- . \tag{2.43}$$

Proof. Potentially, two different control laws could apply inside the BL due to (2.42). Let us analyze the switching condition which appears in (2.42) and suppose that

$$-\frac{\dot{y}_k}{TU^-} = \dot{\tilde{z}}_k . \tag{2.44}$$

According to (2.42) it immediately follows that $\dot{z}_k = -\frac{\dot{y}_k}{TU^-} = \dot{\tilde{z}}_k$, so that, due to (2.24), it is possible to conclude that, when (2.44) holds, the considered point is lying on the sliding surface. Practically, (2.42) define two alternative mappings that can be used depending on the position of the considered point with respect to the sliding surface.

Now hypothesize that $-\frac{\dot{y}_k}{TU^-} < \dot{\tilde{z}}_k$, and, equivalently, that $\sigma_k < 0$. Since $(z_k, \dot{z}_k)$ is located inside the BL but below the sliding surface, it is possible to write

$$u_k = -U^+\sigma_k = -U^+(\dot{z}_k - \dot{\tilde{z}}_k) .$$

Equation (2.43) is easily obtained after few algebraic manipulations by means of (2.41) and (2.42).

Similarly, when $-\frac{\dot{y}_k}{TU^-} \geq \dot{\tilde{z}}_k$ or, equivalently, when $\sigma_k \geq 0$, the control law becomes

$$u_k = U^-\sigma_k = U^-(\dot{z}_k - \dot{\tilde{z}}_k) \; .$$

Also in this case, (2.43) is immediately obtained by considering (2.41) and (2.42).

Property 2 *Given any point* $(z_k, \dot{z}_k)$ *lying within the BL, controller C generates a new point such that*

$$z_{k+1} = z_k + \dot{\tilde{z}}_k \; . \tag{2.45}$$

Moreover, the following condition holds

$$\operatorname{sgn}(z_k) = \operatorname{sgn}(z_{k+1}) \; . \tag{2.46}$$

Proof. Being $\dot{r}_k$ piece-wise constant, and assuming that

$$\dot{r}_k = \frac{r_{k+1} - r_k}{T},$$

the discrete-time evolution (2.22) is converted into an equivalent one in the $(y, \dot{y})$-plane defined as follows

$$\begin{bmatrix} y_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_k \\ \dot{y}_k \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} u_k \; . \tag{2.47}$$

By applying command signal (2.43), system (2.47) evolves in the $(y, \dot{y})$-space as follows

$$\dot{y}_{k+1} = \left(\frac{\gamma_k}{m_k} - \frac{m_k - 1}{2}\right) TU^- \tag{2.48}$$

$$y_{k+1} = y_k + \frac{T}{2}\dot{y}_k + \frac{T^2}{2}\left(\frac{\gamma_k}{m_k} - \frac{m_k - 1}{2}\right) U^- \tag{2.49}$$

By considering (2.38), (2.48), and (2.49), it is suddenly possible to write

$$z_{k+1} = -\frac{1}{TU^-}\left(\frac{y_{k+1}}{T} + \frac{\dot{y}_{k+1}}{2}\right) = -\frac{1}{TU^-}\left(\frac{y_k}{T} + \frac{\dot{y}_k}{2}\right) + \frac{m_k - 1}{2} - \frac{\gamma_k}{m_k} \tag{2.50}$$

or, newly due to (2.38),

$$z_{k+1} = z_k + \frac{m_k(m_k - 1) - 2\gamma_k}{2m_k} \; . \tag{2.51}$$

From (2.40) it descends that, when $z_k \leq 0$, the following inequality is verified

$$z_k \leq \gamma_k \leq -m_k + 1\,, \tag{2.52}$$

so that (2.51) implies

$$z_{k+1} \leq \frac{(m_k-1)(2-m_k)}{2m_k}\,. \tag{2.53}$$

Due to definition (2.40) we have that $m_k \in \mathbb{N}\backslash 0$. As a consequence, it is possible to deduce form (2.53) that $z_{k+1} \leq 0$, thus (2.46) holds.

Equation (2.45) immediately descends from (2.50) by considering (2.38) and (2.41). Property 2 practically asserts that any point within the BL cannot abandon the left plane $z \leq 0$. Properties 1 and 2 generically apply to any point within the BL.

2.5.2 Behavior inside region $R1$

Proposition 1 *Given any starting point $(z,\dot{z})$ lying inside region $R1$, the BL which surrounds sliding surface $\dot{\tilde{z}}$ is reached in minimum time and in a finite number of steps.*

Proof. The proof is straightforward since from (2.23) and (2.24) it descends that above the sliding surface $u_k = U^-$, while below $u_k = U^+$. Due to (2.47), it is possible to conclude that $\dot{y}$ monotonically decreases above $\dot{\tilde{z}}$ while it monotonically increases below $\dot{\tilde{z}}$: owing to the shape of the sliding surface region $R2$ or, alternatively, region $R3$ are certainly reached after a finite number of steps (see also Fig. 2.3).

2.5.3 Behavior inside region $R2$

Proposition 2 *Given any point $(z_k,\dot{z}_k)$ lying within the BL and with $z_k < z^+$, controller C generates a command signal such that the system evolves as follows*

$$\begin{bmatrix} z_{k+1} \\ \dot{z}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_k \\ \dot{z}_k \end{bmatrix} + \begin{bmatrix} \dot{z}^+ \\ \dot{z}^+ \end{bmatrix}. \tag{2.54}$$

Proof. Since $z_k < z^+$, due to (2.39) we can write $\gamma_k = z^+$, so that (2.40) and (2.41) become constant and can be rewritten as follows

$$m^+ := \text{Int}\left[\frac{1+\sqrt{1-8z^+}}{2}\right]\,, \tag{2.55}$$

$$\dot{\tilde{z}}_k := -\frac{z^+}{m^+} + \frac{m^+ - 1}{2} \ . \tag{2.56}$$

Due to (2.48) and (2.56) it is possible to write

$$-\frac{\dot{y}_{k+1}}{TU^-} = \frac{m^+ - 1}{2} - \frac{z^+}{m^+} = \dot{\tilde{z}}_k \ . \tag{2.57}$$

It was early anticipated that the sliding surface has been designed such that $\dot{\tilde{z}} = \dot{z}^+$ when $z < z^+$, so that from (2.57) it descends

$$-\frac{\dot{y}_{k+1}}{TU^-} = \dot{z}^+ \ . \tag{2.58}$$

Owing to the shape of the sliding surface (see also Fig. 2.4), it is possible to assert that, in any case, $\dot{\tilde{z}}_{k+1} \leq \dot{z}^+$. Thus, from (2.58) it follows

$$-\frac{\dot{y}_{k+1}}{TU^-} \geq \dot{\tilde{z}}_{k+1} \ . \tag{2.59}$$

Equation (2.59) indicates that $\dot{z}_{k+1}$ must be evaluated according to (2.42) and, consequently, bearing in mind (2.58), we finally obtain, as desired,

$$\dot{z}_{k+1} = -\frac{\dot{y}_{k+1}}{TU^-} = \dot{z}^+ \ . \tag{2.60}$$

The expression for z_{k+1} is obtained straightforward by means of (2.45) and taking into account that $\dot{\tilde{z}}_k = \dot{z}^+$.

Remark 2 *Equation (2.54), implies that when the system state enters into the BL and $z < z^+$, $\dot{z}$ is forced to the sliding surface $\dot{z}^+$ with a single step and there it remains. Moreover, being $\dot{z}^+ \geq 0$, coordinate z increases, i.e., the state slides to the right. Necessarily, after a finite number of steps it reaches region $R3$.*

2.5.4 Behavior inside region $R3$

It is clear that, after a finite number of steps, the system reaches the BL of the region $z^+ \leq z \leq 0$ directly from $R1$ or, alternatively, from $R2$. The following discussion is devoted to demonstrate that the system state cannot abandon the BL and it must move toward the origin of the $(z, \dot{z})$-space.

Due to (2.39), we can assume $\gamma_k = z_k$ when $z^+ \leq z \leq 0$.

Property 3 *Assume that at step k system state $(z_k, \dot{z}_k)$ is lying within the BL, with $z^+ \leq z_k \leq 0$ and it is characterized by m_k. The new state $(z_{k+1}, \dot{z}_{k+1})$ generated by controller C satisfies the following equality*

$$m_{k+1} = m_k - 1\ .$$

Proof. It is possible to rearrange (2.51) as follows

$$z_{k+1} = (m_k - 1)\left(\frac{z_k}{m_k} + \frac{1}{2}\right)\ . \tag{2.61}$$

It is worth noting that (2.40) induces a partition along the z-axis. In particular, owing

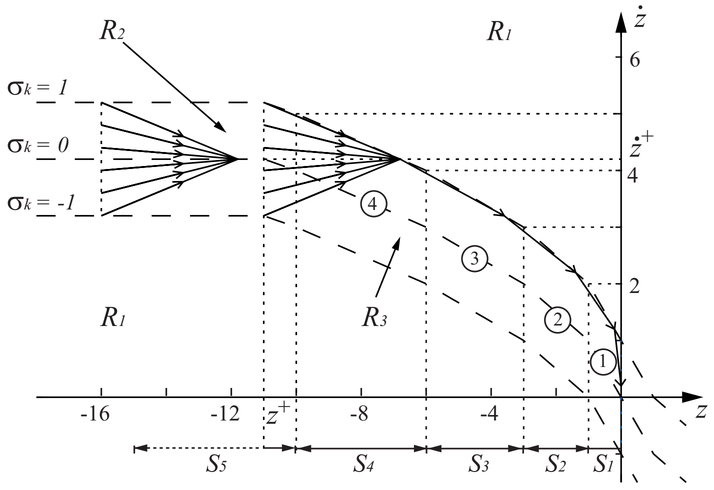


Figure 2.8: Phase-plane in the $(z, \dot{z})$-plane: details in the vicinity of the origin. Circled numbers indicate the corresponding value of m.

to (2.40), associated with any $m \in \mathbb{N}\backslash 0$ there is an interval S_m in z defined as follows (see also Fig. 2.8)

$$S_m := \left\{ z : -\frac{(m+1)m}{2} < z \leq -\frac{m(m-1)}{2} \right\}\ . \tag{2.62}$$

Now hypothesize that current z_k is contained in S_{m_k}, i.e., $z_k \in S_{m_k}$. According to (2.62) it is possible to write

$$-\frac{(m_k+1)m_k}{2} < z_k \leq -\frac{m_k(m_k-1)}{2}\ . \tag{2.63}$$

By taking into account (2.61) it is possible rewrite (2.63) as follows

$$-\frac{(m_k+1)m_k}{2} < \left(\frac{z_{k+1}}{m_k-1}-\frac{1}{2}\right)m_k \le -\frac{m_k(m_k-1)}{2}\ , \tag{2.64}$$

or, equivalently, as

$$-\frac{m_k(m_k-1)}{2} < z_{k+1} \le -\frac{(m_k-2)(m_k-1)}{2}\ . \tag{2.65}$$

Now define

$$m_{k+1} := m_k - 1 \tag{2.66}$$

so that (2.65) can be posed into the form

$$-\frac{(m_{k+1}+1)m_{k+1}}{2} < z_{k+1} \le -\frac{m_{k+1}(m_{k+1}-1)}{2}\ . \tag{2.67}$$

By comparing (2.67) with (2.62) it is immediately possible to conclude that $z_{k+1} \in S_{m_{k+1}}$, where m_{k+1} is defined by (2.66).

Property 4 *Given any point* $(z_k, \dot{z}_k)$ *lying within the BL, with* $z^+ \le z_k \le 0$*, the new point* $(z_{k+1}, \dot{z}_{k+1})$ *generated by controller C is located on the upper frontier of the BL.*

Proof. Due to *Properties 2* and *3*, it is possible to assert that $z_k < z_{k+1} \le 0$, so that the position of the sliding surface corresponding to z_{k+1} can be certainly written, according to (2.41), as follows

$$\begin{aligned}\dot{\tilde{z}}_{k+1} &= -\frac{z_{k+1}}{m_{k+1}} + \frac{m_{k+1}-1}{2} \\ &= -\frac{z_{k+1}}{m_k-1} + \frac{m_k-2}{2}\ ,\end{aligned}$$

or, due to (2.61),

$$\dot{\tilde{z}}_{k+1} = -\frac{z_k}{m_k} + \frac{m_k-3}{2}\ . \tag{2.68}$$

Bearing in mind (2.48), it is possible to assert that

$$-\frac{\dot{y}_{k+1}}{TU^-} = -\frac{z_k}{m_k} + \frac{m_k-1}{2}\ . \tag{2.69}$$

By comparing (2.68) with (2.69) it is possible to conclude that $\dot{z}_{k+1}$ must be evaluated by means of (2.42) since the new point is located above the sliding surface. Consequently

$$\dot{z}_{k+1} = -\frac{z_k}{m_k} + \frac{m_k - 1}{2} \; . \tag{2.70}$$

The position of the new point with respect to the sliding surface is

$$\sigma_{k+1} = \dot{z}_{k+1} - \dot{\tilde{z}}_{k+1} = 1 \; , \tag{2.71}$$

i.e., it exactly lies on the upper frontier of the BL.

The previous properties are used in the following to prove the stability of the filter controller.

Proposition 3 *Given any starting point $(z_k, \dot{z}_k)$ lying within the BL, with $z^+ \leq z_k \leq 0$, controller C forces the system trajectory toward the origin of the $(z, \dot{z})$-plane in minimum time and with a deadbeat dynamics.*

Proof. According to (2.61) and (2.70), the system evolution only depends on the current z_k and m_k. Owing to Property 3, m decreases at each step until it reaches the value $m = 1$. When it happens, owing to (2.61) and (2.70) we have $z_{k+1} = 0$ and $\dot{z}_{k+1} = -z_k$. It is easy to verify by means of (2.40) that m_{k+1} will be still equal to one, so that at the next step (2.61) and (2.70) return $z_{k+2} = 0$ and $\dot{z}_{k+2} = 0$: the origin of the $(z, \dot{z})$-plane is reached with a deadbeat behavior. The system cannot leave the origin during the next sampling times. It is important to note that, due to Property 4, once the system reaches the BL it is forced in a single step toward the frontier of the BL itself. The same Property 4 makes it possible to assert that during the subsequent steps the system does not abandon such frontier, so that the evolution toward the origin is obtained by applying the maximum control command $u_k = U^-$, i.e., in minimum time.

CHAPTER 3

Online trajectory scaling for robotic manipulators subject to generalized forces constraints

If everything seems under control,
you're just not going fast enough.
Mario Andretti

The online path tracking control for robotic manipulators subject to velocity and torque constraints has been devised in the previous chapter, where a non-linear control is used to push toward the origin in minimum-time the tracking error and its first order derivative. In this chapter the previous geometrical ideas are extended to take into account also for torque derivative constraints. Indeed, it is well known that high torques derivatives cause high mechanical stresses and solicit the manipulators unmodelled dynamics, thus decreasing the controller effectiveness: real actuators can only generate limited torque variations, so that path tracking is lost with certainty every time torque derivatives exceed the given limits.

In the first part of the chapter the same controller previously proposed is used at the purpose. Since the controller is of order two, only torque-derivatives and torque constraints are online managed. Therefore, no restrictions are imposed on the robot joint velocities, assuming they are fulfilled by an offline optimization algorithm. Part of the obtained results have been presented in [40].

Finally, the control problem of a chain of three integrator is introduced in order to simultaneously account for torque, torque derivative and velocities constraints.

3.1 Problem formulation

As for the control problem considered in the previous chapter, the solution here proposed is based on the so called path-velocity decomposition [20]: a robot trajectory is obtained by first planning a path to be followed and, then, by generating a velocity profile to move along such path. Paths can be indifferently planned in the task space or in the joint space. For this reason and without any loss of generality, let us define a parametric curve in the joint space by means of a vector function $\Gamma(x)$ and a monotonically increasing time law $x(t)$ according to definition (2.1) and (2.2) respectively.

Consider a serial link rigid-body manipulator, whose standard dynamic is described in (2.6) subject to dynamic and kinematic constraints. More precisely, maximum admissible torques are bounded, so that it is still possible to write inequalities (2.7). Analogously, maximum joint torque-derivatives are bounded, i.e

$$\underline{\dot{\tau}}_k \le \dot{\tau}_k \le \dot{\overline{\tau}}_k, \quad k = 1, 2, \dots, n\,, \tag{3.1}$$

where $\underline{\dot{\tau}}_k$ and $\dot{\overline{\tau}}_k$ represent the lower and upper bounds on the k-th joint torque derivative.

In order to verify the feasibility of the trajectory with respect to (3.1), an analytical representation of $\dot{\tau}$ is required. It can be obtained differentiating (2.6) with respect to time, obtaining for each robot joint $k = 1, 2, \dots, n$

$$\begin{aligned}\dot{\tau}_k \;=\;& \sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial h_{kj}(\mathbf{q})}{\partial q_i}\,\dot{q}_i\ddot{q}_j + \sum_{j=1}^{n} h_{kj}(\mathbf{q})\,\dddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n}\sum_{l=1}^{n}\frac{\partial c_{ijk}(\mathbf{q})}{\partial q_l}\,\dot{q}_l\,\dot{q}_i\,\dot{q}_j + \sum_{j=1}^{n}\frac{\partial g_k(\mathbf{q})}{\partial q_j}\dot{q}_j + \\ & 2\sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(\mathbf{q})\,\dot{q}_i\,\ddot{q}_j + \sum_{j=1}^{n}\frac{\partial f_k(\mathbf{q},\dot{\mathbf{q}})}{\partial q_j}\dot{q}_j + \sum_{j=1}^{n}\frac{\partial f_k(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{q}_j}\,\ddot{q}_j\,.\end{aligned} \tag{3.2}$$

By defining

$$
\begin{aligned}
\dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}}) &:= \sum_{i=1}^{n} \frac{\partial h_{kj}(\mathbf{q})}{\partial q_i}\,\dot{q}_i\,,\\
d_{kj}(\mathbf{q},\dot{\mathbf{q}}) &:= \sum_{i=1}^{n}\sum_{l=1}^{n} \frac{\partial c_{ijk}(\mathbf{q})}{\partial q_l}\,\dot{q}_l\,\dot{q}_i\,,\\
b_{kj}(\mathbf{q},\dot{\mathbf{q}}) &:= \frac{\partial g_k(\mathbf{q})}{\partial q_j} + \frac{\partial f_k(\mathbf{q},\dot{\mathbf{q}})}{\partial q_j}\,,\\
e_{kj}(\mathbf{q},\dot{\mathbf{q}}) &:= \frac{\partial f_k(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{q}_j}\,,
\end{aligned}
$$

equation (3.2) is synthetically rewritten as follows

$$
\begin{aligned}
\dot{\tau}_k \;=\; & \sum_{j=1}^{n} \dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{q}_j + \sum_{j=1}^{n} h_{kj}(\mathbf{q})\,\dddot{q}_j + \sum_{j=1}^{n} d_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\dot{q}_j + \\
& 2\sum_{j=1}^{n} c_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{q}_j + \sum_{j=1}^{n} b_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\dot{q}_j + \sum_{j=1}^{n} e_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{q}_j\,. \qquad (3.3)
\end{aligned}
$$

Finally, the following matrix form can be obtained from (3.3)

$$
\dot{\tau} = \dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q})\,\dddot{\mathbf{q}} + \mathbf{D}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + 2\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{\mathbf{q}} + \mathbf{B}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{E}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{\mathbf{q}}\,. \qquad (3.4)
$$

The first two terms represent the component of the generalized force derivative which are due to the system inertia. In the same way, the second two terms are due to the Coriolis and centripetal components, while the last two refer to the gravity and friction effects. Owing to (2.7) and (3.1) the following tracking problem can be defined

Problem 5 *Given a manipulator described by (2.6) and a desired trajectory (2.1), (2.2), design a control law to achieve the best possible tracking compatibly with torque constraints (2.7) and torque derivative constraints (3.1).*

The following question immediately arises: given a trajectory $\mathbf{q}_d := \Gamma(x(t))$, is it possible to verify its feasibility with respect to (2.7) and (3.1)?

By taking into account the chain differentiation rule, the trajectory time derivatives till the third order are

$$
\begin{aligned}
\dot{\mathbf{q}}_d &= \Gamma^{'}(x)\dot{x}\,, && (3.5)\\
\ddot{\mathbf{q}}_d &= \Gamma^{''}(x)\dot{x}^2 + \Gamma^{'}(x)\ddot{x}\,, && (3.6)\\
\dddot{\mathbf{q}}_d &= \Gamma^{'''}(x)\dot{x}^3 + 3\Gamma^{''}(x)\dot{x}\ddot{x} + \Gamma^{'}(x)\dddot{x}\,. && (3.7)
\end{aligned}
$$

where superscript $'$ indicates a differentiation with respect to x, e.g., $\Gamma(x)' = \frac{d\Gamma(x)}{dx}$, while, as usual, dots indicate time derivatives, e.g., $\dot{x}(t) = \frac{dx(t)}{dt}$.

Due to (3.5)–(3.7), equations (2.6) and (3.4) can be expressed in function of x and its derivatives

$$\tau(x,\dot{x},\ddot{x}) = \mathbf{b}_1(x)\ddot{x} + \mathbf{b}_2(x,\dot{x}) \tag{3.8}$$

$$\dot{\tau}(x,\dot{x},\ddot{x},\dddot{x}) = \mathbf{c}_1(x)\dddot{x} + \mathbf{c}_2(x,\dot{x},\ddot{x}) \tag{3.9}$$

where $\mathbf{b}_1(x) := [b_{1,1}\ b_{1,2}\ \cdots\ b_{1,n}]^T \in \mathbb{R}^n$ and $\mathbf{b}_2(x,\dot{x}) := [b_{2,1}\ b_{2,2}\ \cdots\ b_{2,n}]^T \in \mathbb{R}^n$ are defined according to (2.12) and (2.13) respectively, while

$$\mathbf{c}_1(x) := \mathbf{H}(\Gamma(x))\Gamma'(x) \tag{3.10}$$

$$\begin{aligned}\mathbf{c}_2(x,\dot{x},\ddot{x}) := {} & \dot{\mathbf{H}}(\Gamma(x),\Gamma'(x)\dot{x})\,[\Gamma''(x)\dot{x}^2 + \Gamma'(x)\ddot{x}] + \\ & \mathbf{H}(\Gamma(x))\,[\Gamma'''(x)\dot{x}^3 + 3\Gamma''(x)\dot{x}\ddot{x}] + \mathbf{D}(\Gamma(x),\Gamma'(x)\dot{x})\,\Gamma'(x)\dot{x} + \\ & 2\mathbf{C}(\Gamma(x),\Gamma'(x)\dot{x})\,[\Gamma''(x)\dot{x}^2 + \Gamma'(x)\ddot{x}] + \mathbf{B}(\Gamma(x),\Gamma'(x)\dot{x})\,\Gamma'(x)\dot{x} + \\ & \mathbf{E}(\Gamma(x),\Gamma'(x)\dot{x})\,[\Gamma''(x)\dot{x}^2 + \Gamma'(x)\ddot{x}].\end{aligned} \tag{3.11}$$

being $\mathbf{c}_1(x) := [c_{1,1}\ c_{1,2}\ \cdots\ c_{1,n}]^T \in \mathbb{R}^n$ and $\mathbf{c}_2(x,\dot{x},\ddot{x}) := [c_{2,1}\ c_{2,2}\ \cdots\ c_{2,n}]^T \in \mathbb{R}^n$.

By means of (3.8), (3.9), constraints (2.7) and (3.1) can be used to check the feasibility of a given trajectory: for each joint $k = 1,2,\ldots,n$ the following inequalities must be satisfied

$$\underline{\tau}_k \le b_{1,k}(x)\ddot{x} + b_{2,k}(x,\dot{x}) \le \overline{\tau}_k\,, \tag{3.12}$$

$$\underline{\dot{\tau}}_k \le c_{1,k}(x)\dddot{x} + c_{2,k}(x,\dot{x},\ddot{x}) \le \overline{\dot{\tau}}_k\,. \tag{3.13}$$

Assigned $\underline{\tau}_k$ and $\overline{\tau}_k$, it is possible to verify, for any pair x, $\dot{x}$, if there exists at least one value $\ddot{x}$ which fulfills (2.14): in this way a region in the plane $(x,\dot{x})$, where at least one feasible solution exists, can be found (see, e.g., the area delimited by the continuous line in Fig. 3.1). Analogously, assigned $\underline{\dot{\tau}}_k$ and $\overline{\dot{\tau}}_k$, it is possible to verify, for any triplet x, $\dot{x}$, and $\ddot{x}$, if there exists at least one value $\dddot{x}$ which fulfills (3.13): in this way it is possible to find a volume $\mathcal{A}$, an admissible region AR, in the space $(x,\dot{x},\ddot{x})$ where a feasible solution is defined (see, e.g., the volume delimited by the two surfaces in Fig. 3.1).

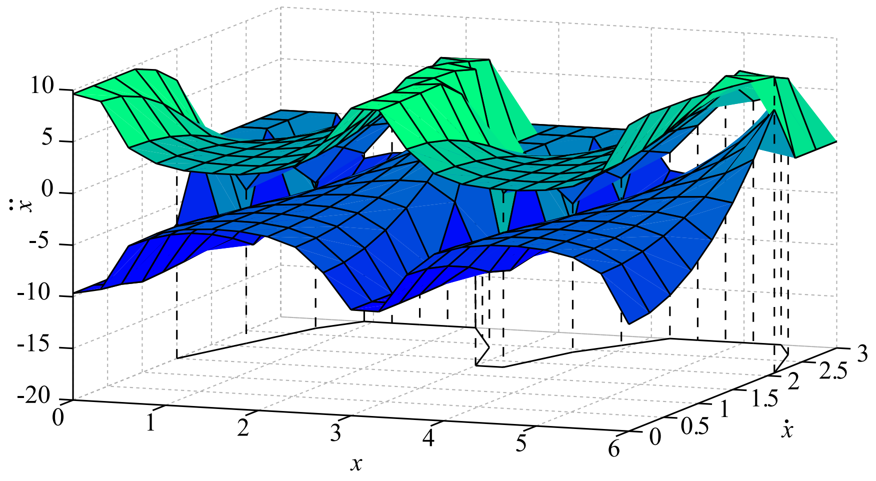


Figure 3.1: An example of the AR $\mathcal{A}$ corresponding to the manipulator proposed in Section 3.6

Definition 1 *Given a curve $\Gamma(x)$ and a time law $x(t)$, the resulting trajectory is feasible if and only if triplet $(x(t),\dot{x}(t),\ddot{x}(t))$ belongs to $\mathcal{A}$ for any $t \in [0,t_f]$.*

As already remarked in Chapter 2, independently from the adopted controller, trajectory tracking is lost any time a non-feasible trajectory is planned. This can mainly happen for two reasons. In the first scenario the trajectory is planned by optimizing a performance index. For example, it is very common to plan time optimal trajectories which minimize the robot traveling time. The resulting trajectory has bang-bang characteristics, that is, there is always at least one robot joint working at its dynamic limits. This corresponds to a point $(x(t),\dot{x}(t),\ddot{x}(t))$ which is constantly moving along the boundary surfaces of region $\mathcal{A}$. Due to model uncertainties or external disturbances, the point could abandon the feasible area, so that trajectory tracking is lost. Lost of tracking can also arise when the trajectory is programmed by an operator. Normally, in this case dynamic constraints are not considered during the planning, thus the resulting trajectory could be unfeasible.

When path tracking is a priority, online trajectory scaling algorithms are able to overcome these issues. A sketch of the proposed control strategy is given in Fig. 3.2. The manipulator is driven by a torque controller whose output signals are saturated

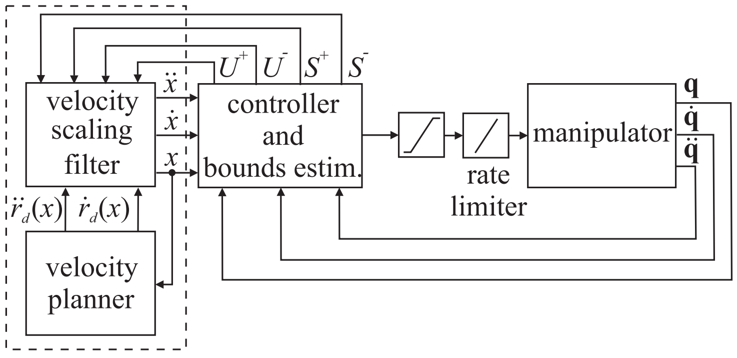


Figure 3.2: Proposed trajectory control scheme

both in amplitude and slew rate. Any standard torque controller can be used at the purpose since it can be parametrized with respect to a scalar value x by means of (3.5)–(3.7). In the following sections, two of the most used feedback controller are considered: the Feedforward Controller with Position and Velocity feedback (FCPV) and the Inverse Dynamics Controller (IDC). The same torque controller evaluates, depending on the current status of motion, appropriate bounds on the longitudinal acceleration and jerk in order to fulfill the dynamic constraints on the maximum torque and torque derivative. The velocity scaling filter modifies the reference trajectory in order to satisfy such bounds.

3.2 FCPV controller parametrization

In this section, the parametrization with respect to the scalar value x is obtained for a feedforward controller with position and velocity feedback. In Chapter 2, the standard formulation of the controller has been recalled in (2.16), while its torque parametrization has been already presented in (2.17).

It is worth recalling that the two vectors $\mathbf{b}_1(x) := [b_{1,1}\ b_{1,2}\ \cdots\ b_{1,n}]^T$ and $\widetilde{\mathbf{b}}_2(x,\dot{x}) := [\widetilde{b}_{2,1}\ \widetilde{b}_{2,2}\ \cdots\ \widetilde{b}_{2,n}]^T$ are evaluated at each iteration of the control algorithm on the basis of the reference position along the path $x(t)$ and the tracking error $\mathbf{e}$. Typically, an efficient iterative Newton-Euler algorithm [41] is used to this purpose. Vectors $\mathbf{b}_1(x)$ and $\widetilde{\mathbf{b}}_2(x,\dot{x})$ are used for the online evaluation of the admissible bounds on the longitudinal acceleration according to the technique proposed in [32], which yield to condition (2.19).

In this chapter the control strategy is improved by considering the boundedness of the torque derivatives. By means of (3.5)–(3.7) its parametric form is given by

$$\dot{\tau}(x,\dot{x},\ddot{x},\dddot{x},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = \mathbf{c}_1(x)\dddot{x} + \widetilde{\mathbf{c}}_2(x,\dot{x},\ddot{x}) \tag{3.14}$$

where $\mathbf{c}_1(x)$ is defined according to (3.10), while

$$\widetilde{\mathbf{c}}_2(x,\dot{x},\ddot{x},\dot{\mathbf{q}},\ddot{\mathbf{q}}) := \mathbf{c}_2(x,\dot{x},\ddot{x}) + \mathbf{k}_p^T\dot{\mathbf{e}} + \mathbf{k}_v^T\ddot{\mathbf{e}}\,. \tag{3.15}$$

with $\mathbf{c}_2(x,\dot{x},\ddot{x})$ defined by (3.11).

In order to avoid huge online computations, the two terms $\mathbf{c}_1(x) := [c_{1,1}\; c_{1,2}\; \cdots\; c_{1,n}]^T \in \mathbb{R}^n$ and $\widetilde{\mathbf{c}}_2(x,\dot{x},\ddot{x}) := [\widetilde{c}_{2,1}\; \widetilde{c}_{2,2}\; \cdots\; \widetilde{c}_{2,n}]^T \in \mathbb{R}^n$ are evaluated by means of the extended iterative Newton-Euler algorithm recently proposed in [42], which returns the manipulator generalized forces. The additional computational burden needed for their evaluation is comparable with the one required for $\mathbf{b}_1(x)$ and $\mathbf{b}_2(x,\dot{x},\mathbf{q},\dot{\mathbf{q}})$: the resulting overall procedure is therefore suitable to be used online.

3.3 IDC controller parametrization

In this section, the parametrization with respect to the scalar value x is obtained for an inverse dynamic controller whose equation, according to [38], is given as follows

$$\tau = \mathbf{H}(\mathbf{q})\,\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{f}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{k}_p^T\,\mathbf{e} + \mathbf{k}_v^T\,\dot{\mathbf{e}}\,. \tag{3.16}$$

As previously, $\mathbf{e} := \mathbf{q}_d - \mathbf{q}$ and $\dot{\mathbf{e}} := \dot{\mathbf{q}}_d - \dot{\mathbf{q}}$ respectively represent the tracking errors and their first derivatives while $\mathbf{k}_p \in \mathbb{R}^n$ and $\mathbf{k}_v \in \mathbb{R}^n$ are the gain vectors of the feedback action. Differently from the FCPV controller (2.16), the sole dependence on the desired trajectory, $\ddot{\mathbf{q}}_d$, is related to the inertial effects; the other terms depend on the manipulator current status of motion, $\mathbf{q},\dot{\mathbf{q}}$. A detailed analysis, regarding the converge properties of the controller (3.16), can be found in [38].

The controller equation can be reparametrized by means of (3.6) in the form

$$\tau(u,\dot{u},\ddot{u};\mathbf{q},\dot{\mathbf{q}}) = \mathbf{b}_1(u;\mathbf{q})\ddot{u} + \widetilde{\mathbf{b}}_2(u,\dot{u};\mathbf{q},\dot{\mathbf{q}})\;, \tag{3.17}$$

where, this time,

$$\mathbf{b}_1(u;\mathbf{q}) := \mathbf{H}(\mathbf{q})\mathbf{f}'(u)\,, \tag{3.18}$$

$$\widetilde{\mathbf{b}}_2(u,\dot{u};\mathbf{q},\dot{\mathbf{q}}) := \mathbf{H}(\mathbf{q})\mathbf{f}''(u)\dot{u}^2+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{g}(\mathbf{q})+\mathbf{f}(\mathbf{q},\dot{\mathbf{q}})+\mathbf{k}_p^T\,\mathbf{e}+\mathbf{k}_v^T\,\dot{\mathbf{e}}. \tag{3.19}$$

Analogously, differentiating (3.16) with respect to time and by using (3.6) and (3.7), the following parametric representation of the torque derivative can be found

$$\dot{\tau} = \mathbf{c}_1(u;\mathbf{q})\dddot{u}+\widetilde{\mathbf{c}}_2(u,\dot{u},\ddot{u};\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\,, \tag{3.20}$$

where

$$\mathbf{c}_1(u;\mathbf{q}) := \mathbf{H}(\mathbf{q})\mathbf{f}'(u)\,, \tag{3.21}$$

$$\begin{aligned}\widetilde{\mathbf{c}}_2(u,\dot{u},\ddot{u};\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) := {} & \dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})\,[\mathbf{f}''(u)\dot{u}^2+\mathbf{f}'(u)\ddot{u}]+\mathbf{H}(\mathbf{q})\,[\mathbf{f}'''(u)\dot{u}^3+3\mathbf{f}''(u)\dot{u}\ddot{u}]\\ & +\mathbf{D}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}+2\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{\mathbf{q}}+\mathbf{B}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}\\ & +\mathbf{E}(\mathbf{q},\dot{\mathbf{q}})\,\ddot{\mathbf{q}}+\mathbf{k}_p^T\,\dot{\mathbf{e}}+\mathbf{k}_v^T\,\ddot{\mathbf{e}}\,.\end{aligned} \tag{3.22}$$

Term $\mathbf{c}_1(u;\mathbf{q})$ is already known since it coincides with (3.18), while the evaluation of $\widetilde{\mathbf{c}}_2(u,\dot{u},\ddot{u};\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ is not straightforward. Equation (3.22) reveals it is possible to compute $\widetilde{\mathbf{c}}_2(u,\dot{u},\ddot{u};\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ only if the derivative of the inertia matrix, i.e., $\dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})$, is available. To this purpose, a method for the online evaluation of $\dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})$ is proposed. Such method is general and therefore can be also used in scenarios different from the one here considered.

3.3.1 Evaluation of the inertia matrix derivative

The proposed solution evaluates the coefficient of $\dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})$ with a two step algorithm. In the first step, terms $c_{kj}(\mathbf{q},\dot{\mathbf{q}})$ of the Coriolis/centripetal matrix $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ are computed, then the second step devises $\dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}})$ of $\dot{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})$.

Let us indicate the unit vectors of a standard orthonormal base as $\mathbf{e}_j \in \mathbb{R}^n$, $j = 1,2,\ldots,n$: only the j-th component of $\mathbf{e}_j$ is equal to one while the other terms are null. In the following, friction and gravity coefficients are always set equal to zero, so that (2.3) and (3.2) simplify as follows

$$\tau_k = \sum_{j=1}^{n} h_{kj}(\mathbf{q})\,\ddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(\mathbf{q})\,\dot{q}_i\,\dot{q}_j \,, \tag{3.23}$$

$$\dot{\tau}_k = \sum_{j=1}^{n} \dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}})\ddot{q}_j + \sum_{j=1}^{n} h_{kj}(\mathbf{q})\dddot{q}_j + \sum_{j=1}^{n} d_{kj}(\mathbf{q},\dot{\mathbf{q}})\dot{q}_j + 2\sum_{j=1}^{n} c_{kj}(\mathbf{q},\dot{\mathbf{q}})\ddot{q}_j \tag{3.24}$$

As a first step, the Newton-Euler algorithm is invoked n times with $\ddot{\mathbf{q}} = 0$, $\dot{\mathbf{q}} = \mathbf{e}_j; j = 1,2,\dots,n$. From (3.23) it can be immediately evinced that, under these conditions, the recursive algorithm returns all the Christoffel symbols which have the same first two indexes, i.e.,

$$y_{kj} := \tau_{kj} = c_{jjk}(\mathbf{q}) \,. \tag{3.25}$$

Subsequently, the inverse dynamics is newly evaluated with $\ddot{\mathbf{q}} = 0$, $\dot{\mathbf{q}} = \mathbf{e}_j + \mathbf{e}_i; i,j = 1,2,\dots,n; i \neq j$. This time its output is

$$\widetilde{y}_{ijk} := \tau_{ijk} = c_{jjk}(\mathbf{q}) + c_{jik}(\mathbf{q}) + c_{ijk}(\mathbf{q}) + c_{iik}(\mathbf{q}) \,. \tag{3.26}$$

Since $c_{jik}(\mathbf{q}) = c_{ijk}(\mathbf{q})$, and remembering that terms $c_{jjk}(\mathbf{q}) = y_{kj}$ have already been computed, rearranging equation (3.26) it holds that

$$c_{ijk}(\mathbf{q}) = \frac{\widetilde{y}_{ijk}(\mathbf{q}) - y_{kj}(\mathbf{q}) - y_{ki}(\mathbf{q})}{2} \,. \tag{3.27}$$

Once all Christoffel symbols $c_{jjk}(\mathbf{q})$ have been evaluated, elements $c_{kj}(\mathbf{q},\dot{\mathbf{q}})$ of matrix $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ are computed by means of (2.5).

The second step of the procedure is based on the use of the extended Newton-Euler algorithm [42]. If we assume $\dddot{\mathbf{q}} = 0$, $\ddot{\mathbf{q}} = \mathbf{e}_j; j = 1,2,\dots,n$, it is possible to evince from (3.24) that the algorithm returns

$$w_{kj}(\mathbf{q},\dot{\mathbf{q}}) := \dot{\tau}_k = \dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}}) + 2c_{kj}(\mathbf{q},\dot{\mathbf{q}}) + \sum_{j=1}^{n} d_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\dot{q}_j \,. \tag{3.28}$$

Analogously, by assuming $\dddot{\mathbf{q}} = 0$, $\ddot{\mathbf{q}} = 0$, from (3.24) it is possible to evince that

$$\widetilde{w}_{kj}(\mathbf{q},\dot{\mathbf{q}}) := \dot{\tau}_k = \sum_{j=1}^{n} d_{kj}(\mathbf{q},\dot{\mathbf{q}})\,\dot{q}_j \,. \tag{3.29}$$

By rearranging (3.28) and considering (3.29), we finally obtain the components of matrix $\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})$

$$\dot{h}_{kj}(\mathbf{q},\dot{\mathbf{q}}) = w_{kj}(\mathbf{q},\dot{\mathbf{q}}) - \widetilde{w}_{kj}(\mathbf{q},\dot{\mathbf{q}}) - 2c_{kj}(\mathbf{q},\dot{\mathbf{q}}) \,. \tag{3.30}$$

3.4 Online bounds evaluation

To satisfy condition (3.1), independently from the adopted controller, it is necessary to impose

$$\underline{\dot{\tau}}_i \le c_{1,i}\dddot{x} + \widetilde{c}_{2,i} \le \overline{\dot{\tau}}_i, \quad i = 1,2,\ldots,n\,. \tag{3.31}$$

Evidently, $\dddot{x}$ is feasible if $\dddot{x} \in \bigcap_{i=1}^{n} [\delta_i\ ,\gamma_i]$, with

$$\gamma_i = \begin{cases} \frac{\overline{\dot{\tau}}_i - \widetilde{c}_{2,i}}{c_{1,i}}, & \text{if } c_{1,i} > 0 \\ \frac{\underline{\dot{\tau}}_i - \widetilde{c}_{2,i}}{c_{1,i}}, & \text{if } c_{1,i} < 0 \\ \infty, & \text{if } c_{1,i} = 0 \end{cases} \quad \text{and} \quad \delta_i = \begin{cases} \frac{\underline{\dot{\tau}}_i - \widetilde{c}_{2,i}}{c_{1,i}}, & \text{if } c_{1,i} > 0 \\ \frac{\overline{\dot{\tau}}_i - \widetilde{c}_{2,i}}{c_{1,i}}, & \text{if } c_{1,i} < 0 \\ -\infty, & \text{if } c_{1,i} = 0 \end{cases} \tag{3.32}$$

or, equivalently, if $\dddot{x} \in [S^-\ ,S^+]$ where

$$S^+ := \min_{i=1,\ldots,n} \{\gamma_i\}\,, \quad S^- := \max_{i=1,\ldots,n} \{\delta_i\}\,. \tag{3.33}$$

As for condition (2.19), also in this case $S^+ > S^-$ only if triplet $(x,\dot{x},\ddot{x})$ is feasible, otherwise there does not exist any solution which fulfills the torque derivative constraints.

It is worth remarking again that S^+, S^-, U^+, and U^- are evaluated by simultaneously considering the manipulator dynamics and the feedback controller actions. This means that feasible volume $\mathcal{A}$ is online reshaped to account for any deed of the feedback controller.

Moreover, the described version of the filter does not manage velocity constraints, so that they are indirectly considered during the planning phase by designing velocity profiles compatible with the maximum admissible joint velocities.

3.5 Online trajectory scaling

Bounds on longitudinal jerk (3.33) and acceleration (2.19) are used to online scale the robot trajectory. To this purpose, the trajectory scaling filter shown in Fig. 3.3 has been developed. The filter behavior is the same of the one extensively described in Chapter 2. Practically, the filter output $\dot{x}(t)$ exactly coincides with $\dot{x}_d(t)$ only if the

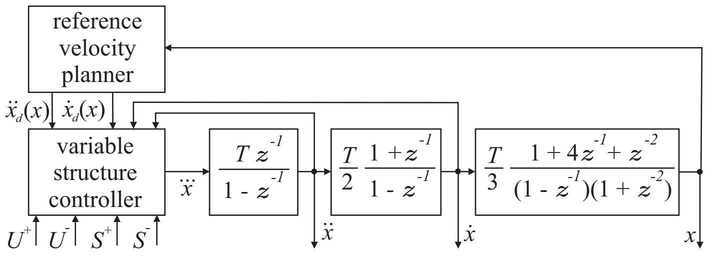

reference velocity planner
$\ddot{x}_d(x)$
$\dot{x}_d(x)$
variable structure controller
$\dddot{x}$
$\frac{T z^{-1}}{1 - z^{-1}}$
$\frac{T}{2}\frac{1 + z^{-1}}{1 - z^{-1}}$
$\frac{T}{3}\frac{1 + 4z^{-1} + z^{-2}}{(1 - z^{-1})(1 + z^{-2})}$
$\ddot{x}$
$\dot{x}$
x
U^+ U^- S^+ S^-

Figure 3.3: Block diagram of the nonlinear filter used for the trajectory scaling.

assigned bounds are fulfilled, i.e., if $\ddot{x}_d \in [U^-, U^+]$ and $\dddot{x}_d \in [S^-, S^+]$, while tracking is voluntarily lost every time such bounds are violated. In this case a new velocity profile $\dot{x}$, which satisfies the given constraints, is generated. The dynamic filter is designed such that $\dot{x}$ robustly converges in minimum time toward $\dot{x}_d$ as soon as $\dot{x}_d$ newly fulfills the dynamic constraints.

The output of the variable-structure controller is evaluated according to the discrete time law presented in Section 2.2, where all the derivative signals related to x and its transformed values in the z-plane are augmented of one order. In this way, the filter is used to regulate the tracking error on jerk and acceleration, while the scalar feedback x is obtained as the outcome of the chain of the three discrete integrators depicted in Figure 3.3 whose state-space representation is equal to

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \\ \ddot{x}_{k+1} \end{bmatrix} = \mathbf{A} \begin{bmatrix} x_k \\ \dot{x}_k \\ \ddot{x}_k \end{bmatrix} + \mathbf{b}\dddot{x}_k = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \\ \ddot{x}_k \end{bmatrix} + \begin{bmatrix} \frac{T^3}{6} \\ \frac{T^2}{2} \\ T \end{bmatrix} \dddot{x}_k , \qquad (3.34)$$

where T is the sampling time and subscript k represents the current data sample.

3.6 Simulation results on a planar PP robot

The trajectory controller has been evaluated considering the same two link planar manipulator introduced in Section 2.4, whose dynamic parameters are defined according to Table 2.1, moved along an assigned ellipsoid path parametrized in the joint space.

The following nominal velocity profile has been assumed

$$\dot{x}_d(x) = \begin{cases} -K_1(x-a)^2 + K_2, & 0 \le x \le a \\ 4a, & a \le x \le b \\ 2a, & b \le x \le 2b \\ -K_3(x-c)^2 + K_4, & 2b \le x \le c \\ 3a, & \text{otherwise,} \end{cases} \tag{3.35}$$

where $a = 0.5$, $b = 1.8$, $c = 3.9$ and $K_1 = 7.6$, $K_2 = 2$, $K_3 = 5.56$ and $K_4 = 1.5$. The corresponding nominal acceleration can easily be computed by considering the differentiation chain rule

$$\ddot{x}_d(x) = \frac{d\dot{x}_d(x)}{dt} = \frac{d\dot{x}_d(x)}{dx}\dot{x}_d(x). \tag{3.36}$$

Clearly $\dot{x}_d(x)$ is too demanding with respect to the dynamic constraints. Indeed, at $x = b$, an infinite acceleration is required, so that such profile could only be tracked if an infinite torque is available.

The torque controller used for the evaluation is the FCPV, whose gains are $\mathbf{k}_p = [200\ 200]^T$ and $\mathbf{k}_v = [60\ 60]^T$ respectively. The path tracking performance has been firstly analyzed by assuming a perfect knowledge of the robot model and, subsequently, considering a perturbed system. In both cases, torques and torque derivatives have been constrained between the following bounds: $|\tau_k| \le 30\ N$ and $|\dot{\tau}_k| \le 350\ N\,s^{-1}$, $k = 1,2$.

3.6.1 Perfect knowledge of the manipulator model

In the first example the manipulator model is supposed to be completely known, so that control law (2.17) is implemented. The control improvements due to the time scaling filter are highlighted by considering three different scenarios:

Case 1 - the filter is disabled and reference velocity (3.35) and acceleration (3.36) are directly used to drive the manipulator controller (dotted lines);

Case 2 - the time scaling filter is activated but only to account for torque constraints (2.7) (dashed lines), thus mimicking [32] where jerk bounds were not considered;

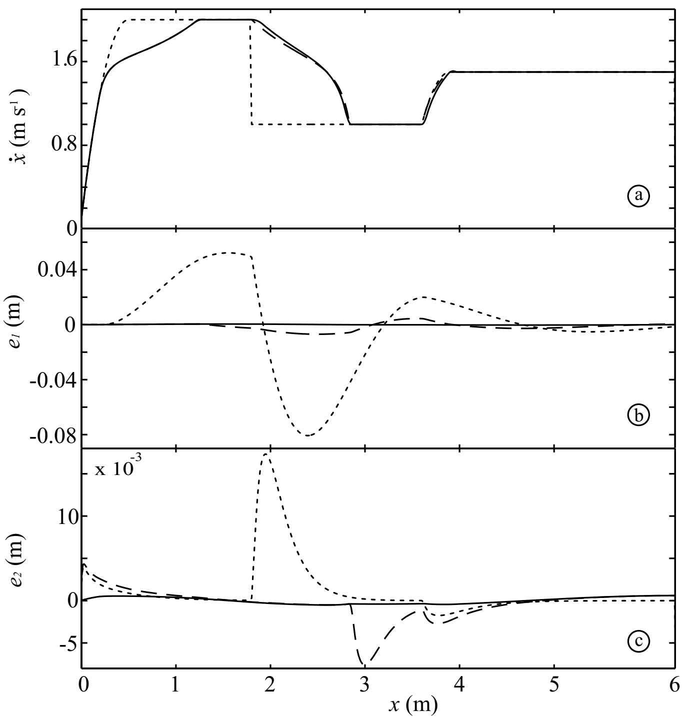


Figure 3.4: Exact model knowledge: (a) actual reference velocities; (b) and (c) tracking errors for the two joints. Dotted lines refer to *Case* 1, dashed lines refer to *Case* 2, while continuous lines refer to *Case* 3.

Case 3 - the filter is fully activated to simultaneously fulfill (2.7) and (3.1) (continuous lines).

Since (3.35) and (3.36) are not feasible, i.e., the nominal trajectory does not always belong to region $\mathcal{A}$, path tracking is evidently lost when the scaling strategy is not used. This conclusion is immediately confirmed by Fig. 3.5, where the robot path is plotted using a dotted line. As expected, tracking errors drastically reduce if the time scaling filter is activated. More precisely, from Fig. 3.4, errors detected for *Case* 2 are almost one order of magnitude smaller than those obtained for *Case* 1. Fig. 3.4 also demonstrates that, when the proposed filter is used to account for (2.7) and (3.1) (*Case* 3), maximum tracking errors reduce of almost two orders of magnitude with

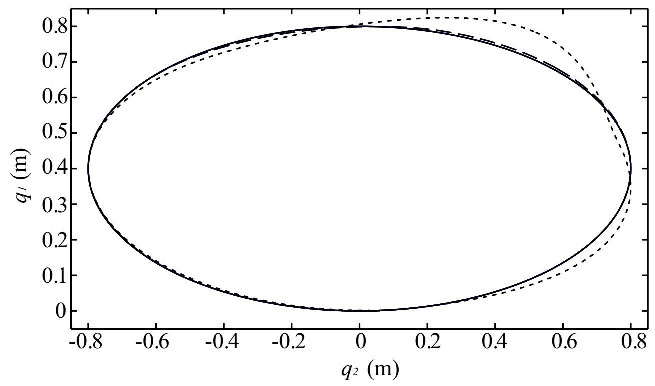


Figure 3.5: Exact model knowledge: dotted line refers to *Case* 1, dashed line to *Case* 2, while continuous line refers to *Case* 3. The reference path is perfectly shadowed by the path obtained by means of the time scaling filter.

respect to *Case* 1, yielding to $e_{max} = \max_{x\in[0,2\pi]}\{\|\mathbf{q}(x)-\mathbf{f}(x)\|\} = 6.012\cdot 10^{-4}$ m where $\mathbf{q}(x) = (q_1, q_2)$ is the actual path followed by the manipulator while $\mathbf{f}(x)$ is the desired path. Paths $\mathbf{q}(x)$ obtained for the three cases are shown in Fig. 3.5.

Fig. 3.6 specifically refers to *Case* 3. More precisely, Fig. 3.6a and Fig. 3.6b respectively show the real time evaluated bounds on $\ddot{x}$ and $\dddot{x}$ (dotted lines) compared with the actual manipulator longitudinal accelerations and jerks (continuous lines): constraints are evidently active in several points along the path. Finally, Fig. 3.6c and 3.6d, respectively show the controller output torques and torque derivatives: dynamic constraints are always fulfilled despite any interference of the feedback controller.

3.6.2 Approximate knowledge of the manipulator model

The behavior of the control scheme has also been verified when only a partial and wrong knowledge of the robot model is available. The following controller, based on an insufficient dynamics knowledge, is assumed

$$\tau(x,\dot{x},\ddot{x},\mathbf{q},\dot{\mathbf{q}}) = \hat{\mathbf{H}}(\Gamma(x))[\Gamma''(x)\dot{x}^2 + \Gamma'(x)\ddot{x}] + \mathbf{k}_p^T\mathbf{e} + \mathbf{k}_v^T\dot{\mathbf{e}} \tag{3.37}$$

where $\hat{\mathbf{H}}(\cdot)$ denotes an estimated inertia matrix obtained by perturbing the coefficients of nominal $\mathbf{H}(\cdot)$ by the 5%. In practice, the robot is considered as a pure inertial sys-

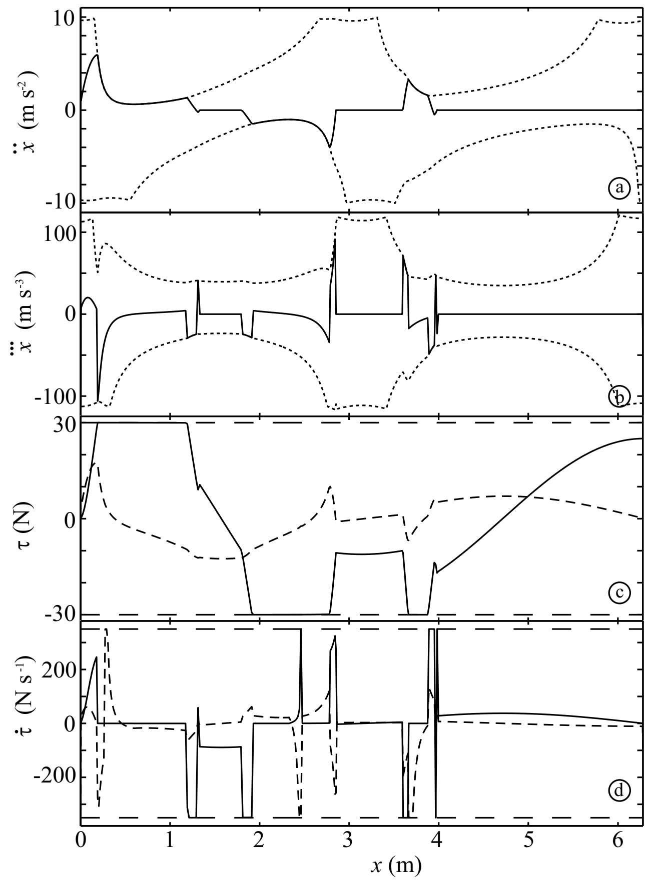


Figure 3.6: Simulation results when the time scaling filter is used: (a) longitudinal acceleration (continuous line) and acceleration bounds (dotted lines); (b) longitudinal jerk (continuous line) and jerk bounds (dotted lines); (c) joint 1 torque (continuous line), joint 2 torque (dashed line); (d) joint 1 torque derivative (continuous line), joint 2 torque derivative (dashed line).

tem and its nonlinear dynamics are neglected. The use of this simplified robot model is justified by the fact that the identification of the whole manipulator parameters, especially for systems with many degree of freedom, can be quite a demanding task. On the contrary, the inertia matrix can be obtained with practical recursive algorithms, such as the one proposed in [38].

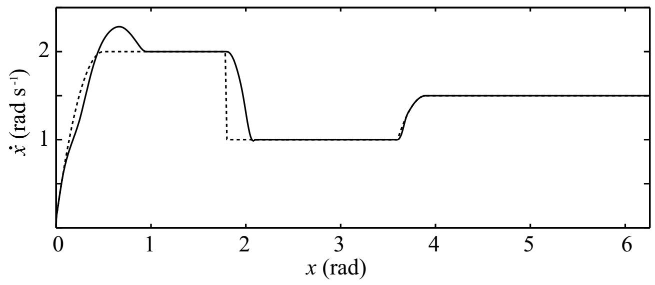


Figure 3.7: Velocity reference signal $\dot{u}_d$ (dashed line) compared with the filter output $\dot{u}$ (solid line).

For the example herein considered, the maximum tracking error only slightly increases with respect to *Case* 3, i.e., $e_{max} = 6.144 \cdot 10^{-4}$ m. As early asserted, the filter constantly forces the system inside current region $\mathcal{A}$ by scaling the trajectory. As a consequence, tracking tolerance does not depend on its behavior but it is mainly due to the performances of the inner controller (3.37). Tracking tolerance considerations could be performed by extending the techniques proposed in [43].

3.7 Simulation results on a planar RP robot

In order to test the behavior of the control scheme even in presence of non-negligible Coriolis and centrifugal effects, a different manipulator has been simulated. Moreover, in order to remark the possibility of using different torque controllers within the same framework, the IDC parametrization described in Section 3.3 has been adopted.

The chosen robot is a RP planar manipulator characterized by the dynamic parameters reported in Table 3.1. The path to be tracked is an ellipse parametrized as follows

$$\mathbf{f}(u) = \begin{bmatrix} \theta_1 \\ d_2 \end{bmatrix} := \begin{bmatrix} \mathrm{Atan2}(0.8 \sin x, 0.4 \cos x) \\ \sqrt{0.4^2 \cos^2 x + 0.8^2 \sin^2 x} \end{bmatrix}, \quad x \in [0, 2\pi] \,. \tag{3.38}$$

The following tuning parameters have been selected for the controller: $\mathbf{k}_p = [500\;400]^T$, $\mathbf{k}_v = [10\;60]^T$. The velocity reference is shown in Fig. 3.7 by means of a dashed line and defined as in the previous section. Once again the reference signal is chosen such

Table 3.1: Robot inertial parameters

Link	Mass	Center of gravity			Inertia			Friction
q	m (Kg)	x(m)	y(m)	z(m)	$I_{xx}(Kg.m^2)$	$I_{yy}(Kg.m^2)$	$I_{zz}(Kg.m^2)$	$B(N.s/rad)$
θ_1	23.90	0	0.10	0	2.521	1.671	1.358	1.5e-3
d_2	3.88	0	-0.30	0	0.336	0.336	0.026	2.8e-3

to be unfeasible with respect to the robot dynamic constraints that are supposed active on both joints. In particular, the following limits have been used for the torques and the torque derivatives: $\tau_1, \tau_2 \in [-13, 13]$, $\dot{\tau}_1 \in [-200, 200]$, and $\dot{\tau}_2 \in [-150, 150]$.

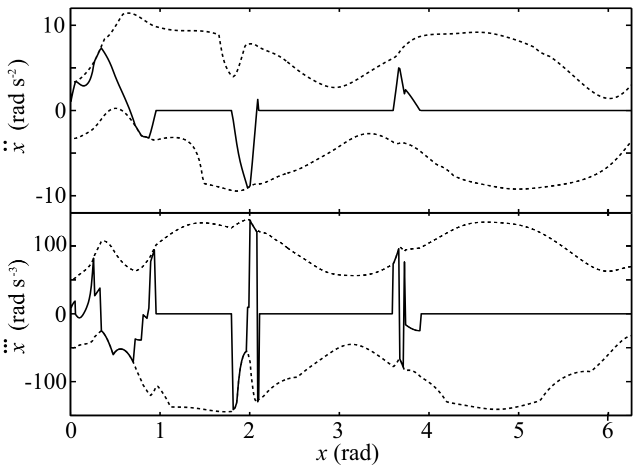


Figure 3.8: Velocity and acceleration bounds online evaluated

Fig. 3.8 is useful to understand the system behavior. Dashed lines correspond to upper an lower bounds on $\ddot{x}$ and $\dddot{x}$ evaluated by means of (2.19) and (3.33): the time scaling system generates an output signal $\dot{x}$ whose first and second derivatives fulfill the imposed constraints. A comparison between the original $\dot{x}_d$ and $\dot{x}$ is shown in Fig. 3.7. The feasibility of the generated profile is proven by Fig. 3.10: actual τ and $\dot{\tau}$ always satisfy the given constraints. It is relevant to note that every time the

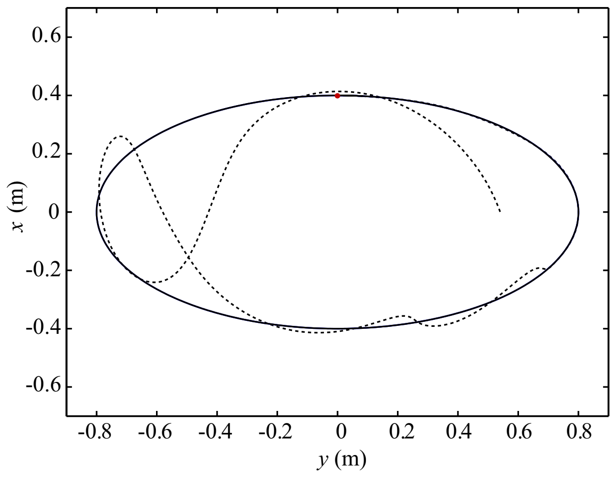


Figure 3.9: Paths generated by adopting the filter (solid line) and without the filter (dashed line). The manipulator is clockwise moving starting from the red point.

constraints on τ and $\dot{\tau}$ are touched, the velocity tracking is lost in order to maintain a correct path tracking.

The overall accuracy of the controller is verified by measuring the path tracking error defined as the Euclidean distance, expressed in function of x, between the manipulator tool frame and the reference path. Fig. 3.11 compares the errors detected with and without the filter: the maximum error without the filter is equal to 3.770e-2 m, while it decreases to 6.946e-4 m when the filter is used.

The relevance of the constraints on the generalized force derivatives are highlighted in Fig. 3.9: when the filter is not used, it is sufficient to reduce the bounds on $\dot{\tau}_1$ to ± 160 Nms^{-1} to obtain a complete tracking lost.

3.8 Minimum time tracking problem for a chain of three integrators with bounded input

In the path tracking problems analyzed so far, it has always been used the stabilization filter devised in Chapter 2. However, when dealing with the online trajectory scaling of manipulators subject to generalized forces and velocities constraints, it is

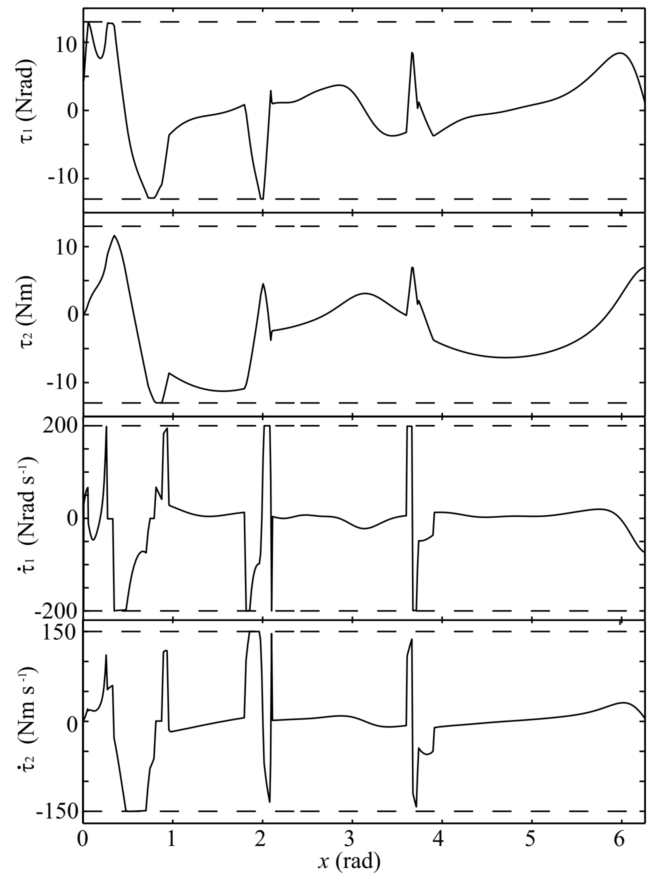


Figure 3.10: Generalized forces and their derivatives for the two joints

interesting to study the stabilization problem of a constrained discrete set of three integrators.

This problem has been already extensively studied in the continuous time case, also for a chain of arbitrary order with saturated input [44], where the solution is found on the basis of the Pontryagin Maximum Principle. A discrete time solution has been recently proposed in [45], where only the boundedness of the control input is considered. By defining the normalized tracking error and its derivatives as

$$y = \frac{x-r}{U},\ \dot{y} = \frac{\dot{x}-\dot{r}}{U},\ \ddot{y} = \frac{\ddot{x}-\ddot{r}}{U},$$

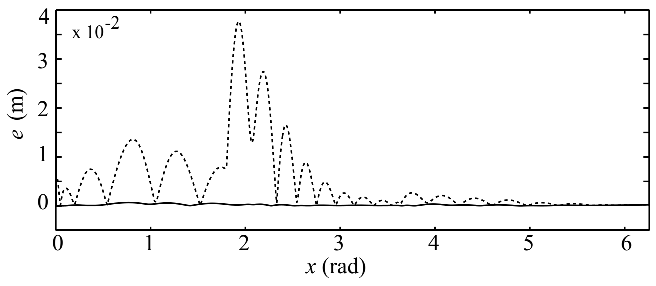


Figure 3.11: The path tracking error without (dashed line) and with (solid line) the velocity scaling filter.

where r, $\dot{r}$ and $\ddot{r}$ are, respectively, the reference input and its derivatives, where x, $\dot{x}$, $\ddot{x}$ are the system state space variables, as depicted in Figure 3.3, and where U is the input constraint, the following problem has been solved.

Problem 6 *Design the nonlinear static function* $u_k = u_k(y_k)$ *such that, starting from any initial condition* $\mathbf{y}_0 = [y_0, \dot{y}_0, \ddot{y}_0]^T$*, system*

$$\mathbf{y}_{k+1} = \mathbf{A}\mathbf{y}_k + \mathbf{b}u_k,$$

with $\mathbf{A}$ *and* $\mathbf{b}$ *defined according to (3.34), is controlled to the origin in the minimum time compatible with the constraint* $|u_k| \leq 1$*, being* u_k *the normalized control action.*

The problem solution is investigated for the equivalent formulation obtained by applying a state space bijective transformation in order to eliminate any dependencies of the discrete system from the sampling time T. The new state space variables, at time instant k, are indicated with the vector $\mathbf{z}_k := [z_{1,k}, z_{2,k}, z_{3,k}]^T$. By using the transformation $\mathbf{z}_k = \mathbf{T}\mathbf{y}_k$, where

$$\mathbf{T} = \frac{1}{T^2}\begin{bmatrix} \frac{1}{T} & 1 & \frac{T}{3} \\ 0 & 1 & \frac{T}{2} \\ 0 & 0 & T \end{bmatrix},$$

the following equivalent system is obtained

$$\mathbf{z}_{k+1} = \bar{\mathbf{A}}\mathbf{z}_k + \bar{\mathbf{b}}u_k, \tag{3.39}$$

with

$$\bar{\mathbf{A}} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad \bar{\mathbf{b}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

According to [45], the following definitions holds.

Definition 2 *Let $B^-_{k,0}$ and $B^+_{k,0}$ denote the points of the state space $\mathbf{z}_k$ from which it is possible to reach the origin in a number k of sampling periods when the constant normalized inputs $u_k = -1$ and $u_k = 1$, respectively, are applied.*

Definition 3 *Let $B^-_{h,k}$ and $B^+_{h,k}$ denote the points of the state space from which it is possible to reach the point $B^+_{k,0}$ and $B^-_{k,0}$ respectively, in h sampling periods by using, respectively, the constant inputs $u_k = -1$ and $u_k = 1$.*

For example, the set of points $B^+_{h,k}$ is obtained by inverting the system (3.39) and solving it with control input equal to $u_k = 1$ and initial condition $\mathbf{z}_0 = B^-_{k,0}$. It's closed form expression is given by

$$\begin{aligned} B^+_{h,k} &= \mathbf{A}^h B^-_{k,0} - \textstyle\sum_{n=0}^{h-1} \mathbf{A}^{n-1}\mathbf{b} = \mathbf{A}^h B^-_{k,0} - B^-_{h,0} = \\ &= \begin{bmatrix} \frac{k(k-1)(k-2)}{6} + \frac{k(k-1)h}{2} + \frac{h(h-1)k}{2} - \frac{h(h-1)(h-2)}{6} \\ \frac{h(h-1)}{2} - hk - \frac{k(k-1)}{2} \\ k-h \end{bmatrix}, \end{aligned} \tag{3.40}$$

where $h,k \geq 0$. In a similar way it is possible to devise the set of points $B^-_{h,k}$.

It is worth noting that the points $B^+_{h,k}, B^+_{h,k+1}, B^+_{h+1,k}$ and $B^+_{h+1,k-1}$, with $h \geq 0$ and $k \geq 1$, define a parallelogram in the state space. Denote this parallelogram by using the symbol $\mathbf{P}^+_{h,k}$. The union of all the parallelograms clearly describe a surface from which it is possible to reach a point belonging to $\mathbf{P}^+_{0,k}$ in h steps, by applying the maximum positive control $u_k = 1$. Denote this surface with the symbol σ^+_d. Analogous considerations apply to definition of the parallelograms $\mathbf{P}^-_{h,k}$ and the surface σ^-_d.

The discrete time control law C which solves Problem 6, as proved in [45], is given by

$$C: \quad u_k := -\mathrm{sat}(\sigma_k) \tag{3.41}$$

$$\sigma_k \;:=\; z_{3,k} + \frac{2h+k-1}{h(h+k)} z_{2,k} + \frac{2}{h(h+k)} z_{1,k} + \tag{3.42}$$

$$+\frac{2h^3+k^3+3h^2k-3hk-3h^2+h-k}{6h(h+k)}\eta \tag{3.43}$$

where

$$(h,k,\eta) = \begin{cases} (h,k,1) & \text{if } (z_{1,k}, z_{2,k}) \in \bar{\mathbf{P}}^{+}_{h,k}, \\ (h,k,-1) & \text{if } (z_{1,k}, z_{2,k}) \in \bar{\mathbf{P}}^{+}_{h,k}, \end{cases}$$

and where $\bar{\mathbf{P}}^{\pm}_{h,k}$ is the projection of the parallelogram $\mathbf{P}^{\pm}_{h,k}$ on the plane (z_1, z_2).

Equation σ_k is obtained computing the distance, along the component z_3, of the system state $\mathbf{z}_k$ from the middle of the boundary layer identified once σ_d^+ and σ_d^- are known. The integer parameters h, k and η are completely determined given the state $\mathbf{z}_k$. To this purpose the following iterative search has been implemented.

3.8.1 Algorithm for computing h,k,η

The algorithm is mainly subdivided into two different steps: the first one determines the value of η, i.e. if the projection of the current point $\mathbf{z}_k$ belongs to $\bar{\mathbf{P}}^+$ or $\bar{\mathbf{P}}^-$. Then, the second step iteratively searches the polygon $\bar{\mathbf{P}}_{h,k}$ inside which the current point is located.

In practice, as depicted in Figure 3.12, the algorithm moves the operating point $\mathbf{z}$ along the sliding surface represented by a solid blu line. At each iteration of the algorithm, the value of the parameter h is kept constant to zero, while the other integer parameter k is increased. The couple (h,k) is then used in (3.40) to compute the new operating point. The sliding along the surface continues until a sign change in the first component of $\mathbf{z}$ is detected, represented in Figure 3.12 by two solid red lines. Let us indicate with $\mathbf{z}_p$ and $\mathbf{z}_a$ the system point before and after the stop condition, respectively. The value of η is then determined as the sign of the cross product between the displacement vector $(\mathbf{z}_a - \mathbf{z}_p)$ and the error vector between the current operative point $\mathbf{z}$ and the input data $\mathbf{z}_k$.

Once the sign of η has been computed, the second stage of the algorithm is executed. In particular, the couple of values (h,k), identifying the parallelogram $\bar{\mathbf{P}}_{h,k}$ inside which $\mathbf{z}_k$ is located, are determined by the following computations.

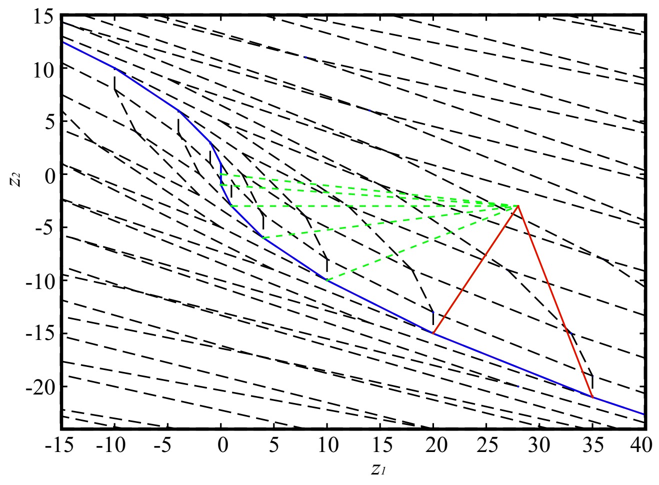


Figure 3.12: Example of the iterative search for the η value in the (z_1, z_2)-plane

Assigned a value of h, which identifies in the (z_1, z_2)-plane one of the parabolas depicted in Figure 3.12, the algorithm searches with a bisection method the value of k which minimizes the distance between the input point $\mathbf{z}_k$ and the one identified by the triplet (h, k, η). A boolean flag is also returned to indicate if the desired point $\mathbf{z}_k$ is located above or bottom with respect to the base of the polygon identified by the current values of h and k. Based on this information, an upper and lower bound couple of values, $(\overline{h}, \overline{k})$ and $(\underline{h}, \underline{k})$, are stored. Then, a new candidate value of h is chosen within the previous range, according to a bisection method. The algorithm iterates until the distance between the upper and lower bound on h, i.e. $(\overline{h} - \underline{h})$, is less or equal to one. Figure 3.13 illustrates the described procedure: green lines are the error vectors computed for the different points investigated by the algorithm, the red dot is the input point $\mathbf{z}_k$ while the red polygon $\overline{P}_{h,k}$ is the one computed using the returned values (h, k, η).

The solution (3.41)-(3.42) introduced by [45], stabilizes the integrator chain by considering the presence of a single constraint, namely the one on the control input. However, as already remarked throughout this chapter, in order to use this nonlinear filter in the context of the trajectory scaling problem for robotic manipulators, it is necessary to account also for acceleration and velocity constraints, i.e., with reference

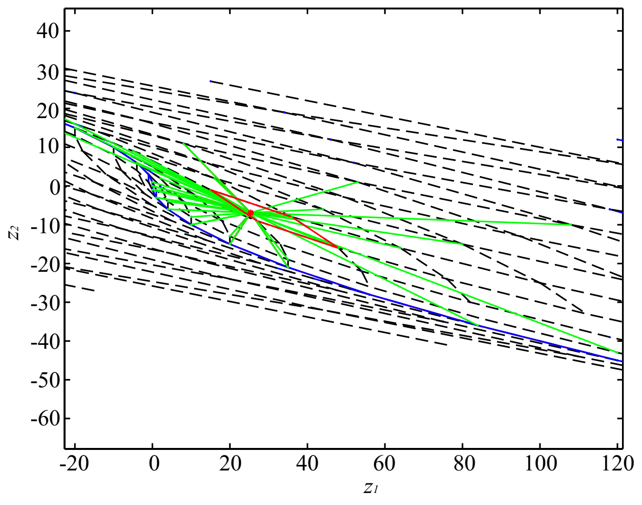


Figure 3.13: Example of the iterative search for the values of (h,k) in the $(z_1,\ z_2)$-plane

to Figure 3.3, considering also the boundedness of $\ddot{x}$ and $\dot{x}$. For this reason, the author is currently involved in extending the previously described method in order to initially account for acceleration constraints. Even if not yet completely formalized from a mathematical point of view, good and promising results have been already achieved, which will be part of next publications.

CHAPTER 4

A minimum-time feed-forward control of a flexible joint

He who controls the present, controls the past.
He who controls the past, controls the future.
George Orwell

In many applications, such as robotic manipulators, disk-drive heads, or pointing systems, sophisticated control algorithms are required to make optimal use of the maximum torque available for rapid maneuvers, [46] [47]. Unfortunately, any minimum time performance is usually achieved by maximizing the actuators dynamic efforts possibly leading to undesirable results in the case of standard feedback controllers. Indeed, due to saturations, the system behavior could be characterized by overshoots and oscillations, as extensively remarked in the previous chapters. These effects are even more relevant for robotic manipulators showing a significant elastic coupling between joints, like those designed to share their workspace with human beings. In such cases the use of elastic joints increases the system safety by reducing the arm stiffness. In fact, as stated in [48], by decoupling the actuators inertia from the inertia of the links, it is possible to reduce the end-effector impact force such to limit potential danger to the operator. In recent literature a considerable attention has been given to the control of robot with flexible joints, see for instance [49, 50, 51], or [52] for a survey.

A major drawback of manipulators with a significant elastic coupling is that the output reacts slowly to the input, thus degrading the manipulator performances. Hence, it is interesting to consider the minimum-time control problem, that is to find the control input that allows performing a desired rest-to-rest transition for the end-effector, by minimizing at the same time the robot traveling time. This makes it possible to improve the resulting control performances despite the elastic coupling.

However, for such kind of robots, any sudden torque change, an implicit requirement of minimum-time motions, can excite the oscillatory dynamics. It is therefore important to introduce, together with the usual input constraints considered in the robotic literature, also output constraints. In this chapter a time-optimal solution for an electrically driven flexible joint arm is proposed. Explicit bounds on the motor feeding voltage are considered but, at the same time, a zero overshoot solution is required.

The minimum-time transition is obtained by discretizing the continuous-time model of the flexible joint and formulating an equivalent discrete-time optimization problem solved by means of linear programming techniques. More precisely, upper and lower bounds on the input voltage, as well as those on output overshoot and undershoot, are expressed by linear inequalities on a vector $\mathbf{u}$, representing the input voltages at sampling times. The optimization method searches the input vector $\mathbf{u}$ such that the end-effector performs a rest-to-rest transition in a number of steps less or equal than an initial guess n, while fulfilling the input and output constraints. Hence, the minimum-time problem is reformulated as a feasibility test for a linear programming (LP) problem and the minimum number of steps required to complete the given rest-to-rest transition can be found through a simple bisection algorithm. Since the sampling time T is fixed, minimizing the number of steps implies achieving the minimum-time solution which fulfills the given constraints.

The use of linear programming techniques for solving minimum-time problems for linear discrete-time systems, subject to bounded inputs, dates back to Zadeh [53].

Subsequently, many contributions have appeared focusing on various improvements. For example a faster algorithm is proposed in [54]. Work [55] presents a more general linear programming algorithm for solving optimal control problems for linear systems under generic constraints. In [56] a feasibility test is presented to improve the algorithm speed. For what concerns time-optimal control for continuous time systems, a related result, under different hypotheses, is presented in [57]. It applies a comparison principle to a time-optimal control problem for a class of state-constrained second-order systems.

The chapter is organized as follows. In §4.1 the dynamic model of a flexible joint is devised. It will be used for the synthesis and the validation of the proposed control technique. In §4.2 the control problem is proposed and a solution is obtained in the subsequent section by means of a linear programming algorithm. An experimental test case is discussed in §4.4.

Notation: Given a sequence $u(k) : \mathbb{Z} \to \mathbb{R}$, $U(z) = \mathcal{Z}\{u(k)\}$ represents its Z-transform, $\|u(k)\|_\infty = \max\{|u(k)| : k \in \mathbb{Z}\}$ is the infinity norm of $u(k)$. For $x \in \mathbb{R}$, $\lfloor x \rfloor = \max\{i \in \mathbb{Z} | i < x\}$ is the floor of x and $1_n \in \mathbb{R}^n = (1, 1, \dots, 1)^T$. Given a matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$, $\|\mathbf{M}\|_2 = \max\{\|Ax\| : x \in \mathbb{R}^n \text{ with } \|x\| = 1\}$ is the 2-norm.

4.1 Flexible joint model

The minimum-time control problem is solved for a single flexible joint device produced by Quanser Consulting. Fig. 4.1 shows the top view of the considered system: a rigid arm is connected, through a flexible joint, to a rotating "body", which is actuated by a dc servo motor. Both the body and the arm can rotate around the vertical axis "O" of Fig. 4.1. The elastic coupling between the body and the arm is obtained by means of two springs whose stiffness is K_e and whose unstretched length is l_0.

The control technique proposed in §4.3 is based on the knowledge of the system model. For this reason, an accurate nonlinear model, mainly used for simulation purposes, is proposed in the following. The linearized version of the same model, to be used for the controller synthesis, is then devised.

Spring forces $\mathbf{f}_1$ and $\mathbf{f}_2$ cover an important role in the system dynamics. In order to

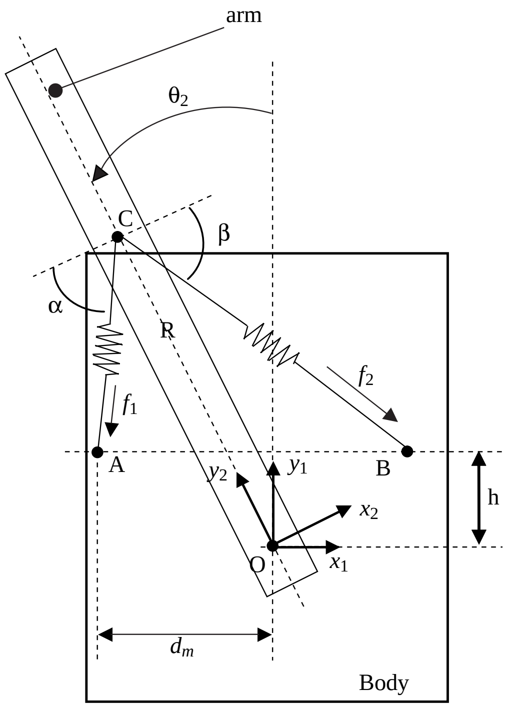


Figure 4.1: Flexible joint experiment: Top view.

evaluate their amplitude, let us assign a reference frame $\{1\}$ whose origin is located in *"O"* and integral with the body. Moreover, let us assign a further frame $\{2\}$, located in *"O"* but integral with the arm, and indicate by θ_2 the joint angle between the two frames. Angle θ_2 is counterclockwise positive. In the same way, let us indicate by θ_1 the counterclockwise positive joint angle between the body frame $\{1\}$ and a given stationary frame.

The three points *"A"*, *"B"*, and *"C"* shown in Fig. 4.1 can be described with respect to frame $\{1\}$ by means of three vectors $\mathbf{p}_a := [-d_m\ h]^T$, $\mathbf{p}_b := [d_m\ h]^T$, and $\mathbf{p}_c := [-R\sin\theta_2\ R\cos\theta_2]^T$ where d_m, h, and R are the geometrical dimensions reported in the same figure.

The spring force norms, i.e., $f_1 := \|\mathbf{f}_1\|$ and $f_2 := \|\mathbf{f}_2\|$, depend on the spring lengths l_1 and l_2 according to equations

$$f_1 \;=\; K_e(l_1 - l_0)\,, \tag{4.1}$$

$$f_2 \;=\; K_e(l_2 - l_0)\,, \tag{4.2}$$

where l_1 and l_2 can be evaluated as follows

$$\begin{aligned} l_1 &= \|\mathbf{p}_c - \mathbf{p}_a\| \\ &= \sqrt{R^2 + d_m^2 + h^2 - 2R(d_m \sin\theta_2 + h\cos\theta_2)}\,, \end{aligned} \tag{4.3}$$

$$\begin{aligned} l_2 &= \|\mathbf{p}_c - \mathbf{p}_b\| \\ &= \sqrt{R^2 + d_m^2 + h^2 + 2R(d_m \sin\theta_2 - h\cos\theta_2)}\,. \end{aligned} \tag{4.4}$$

Forces acting on point *"C"* can be described with respect to frame $\{2\}$ leading to

$$\begin{bmatrix} f_{1_x} \\ f_{1_y} \end{bmatrix} = \begin{bmatrix} f_1\cos(\alpha) \\ f_1\sin(\alpha) \end{bmatrix} = \begin{bmatrix} -K_e(l_1 - l_0)\cos(\alpha) \\ -K_e(l_1 - l_0)\sin(\alpha) \end{bmatrix}$$

and

$$\begin{bmatrix} f_{2_x} \\ f_{2_y} \end{bmatrix} = \begin{bmatrix} f_2\cos(\beta) \\ f_2\sin(\beta) \end{bmatrix} = \begin{bmatrix} K_e(l_2 - l_0)\cos(\beta) \\ -K_e(l_2 - l_0)\sin(\beta) \end{bmatrix}$$

where $\alpha, \beta \in \mathbb{R}^+$ are the two auxiliary angles shown in Fig. 4.1 which can be evaluated by means of the following equations

$$\begin{aligned} \alpha(\theta_2) &= \arctan\left[\frac{R\cos(\theta_2) - h}{d_m - R\sin(\theta_2)}\right] - \theta_2\,, \\ \beta(\theta_2) &= \arctan\left[\frac{R\cos(\theta_2) - h}{d_m + R\sin(\theta_2)}\right] + \theta_2\,. \end{aligned}$$

Elastic forces induce an elastic nonlinear torque in the arm that can be expressed as

$$\tau_e(\theta_2) = \left[-f_{1_x}(\theta_2) - f_{2_x}(\theta_2)\right] R\,. \tag{4.5}$$

It is worth noting that components f_{1_y} and f_{2_y} do not generate any torque with respect to *"O"*.

It is now possible to propose the dynamic equation of the rigid arm described with respect to *"O"*

$$J_{load}(\ddot{\theta}_2 + \ddot{\theta}_1) = \left[-f_{1_x}(\theta_2) - f_{2_x}(\theta_2)\right] R - B_{eq}^{L}\dot{\theta}_2 \tag{4.6}$$

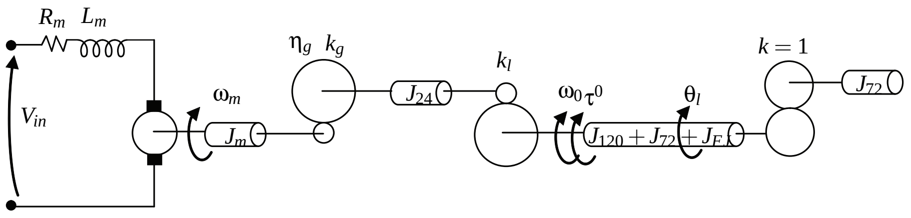


Figure 4.2: Inertia and gears ratio chain view from motor rotor axis

where J_{load} is the arm inertia evaluated with respect to *"O"*, while B_{eq}^L is the friction coefficient associated to angular velocity $\dot{\theta}_2$. Practically, arm dynamics takes into account torques which are due to inertia, friction and elasticity.

It is similarly possible to devise the dynamic equation of the "body". It is made of a chain of inertial loads and reduction gears driven by a permanent magnet dc motor according to the scheme shown in Fig. 4.2. More precisely, the motor, characterized by an inertia J_m, is connected through a chain of reduction gears to the output shaft. Each reduction gear is characterized by a reduction ratio, see e.g. k_G, k_l, and an inertia, see e.g. J_{120}, J_{72}, and J_{24}. The first reduction gear is characterized by an efficiency coefficient η_G, while the the body inertia is J_{FJ}. Output angle θ_1 is measured through a potentiometer coupled to the output shaft by means of a gear which has reduction ratio $k=1$.

The system is affected by torques which are due to inertia, friction and elasticity, yielding to

$$J_{eq}^0\ddot{\theta}_1 = \tau^0 - B_{eq}^0\dot{\theta}_1 - \left[-f_{1_x}(\theta_2) - f_{2_x}(\theta_2)\right] R + B_{eq}^L\dot{\theta}_2 \,, \tag{4.7}$$

where J_{eq}^0 is the equivalent inertia of the system composed by motor, reduction gears, and "body", τ^0 is the motor torque reflected through the gears ratios, while B_{eq}^0 is the friction coefficient associated to angular velocity $\dot{\theta}_1$. All quantities in (4.7) are referred to the output shaft of the system. For the system of Fig. 4.2 the equivalent inertia can be expressed as

$$J_{eq}^0 = \left[J_m k_g^2 k_l^2 \eta_g + J_{24} k_l^2 + J_{120} + 2J_{72} + J_{FJ}\right] .$$

The dynamics of a dc motor is given by the following equations, [58]

$$\begin{cases} L\frac{di}{dt} &= -R_m i - k_m \omega_m + v_{in}; \\ \tau_m &= k_m i, \end{cases} \tag{4.8}$$

where τ_m is the torque at the output shaft of the dc motor, L_m is the armature inductance, ω_m is the motor angular velocity, k_m is the motor electric constant, R_m is the motor winding resistance, and v_{in} is the motor feeding voltage.

Due to (4.8), the motor electrical pole is equal to

$$\frac{R_m}{L_m} \simeq 1 \cdot 10^4 \quad \text{rad s}^{-1}$$

which is negligible with respect to the mechanical pole equal to 11 rad s^{-1}. Therefore (4.8) can be approximated as follows

$$\tau_m \simeq k_m \left(\frac{v_{in} - k_m \omega_m}{R_m} \right).$$

Hence, according to Fig. 4.2, the output torque τ^0 can be expressed as

$$\tau^0 = \tau_m (k_g k_l \eta_g \eta_m) = \frac{k_g k_l k_m \eta_g \eta_m}{R_m} v_{in} - \frac{k_g^2 k_l^2 k_m^2 \eta_g \eta_m}{R_m} \dot{\theta}_1 \tag{4.9}$$

where η_m is the motor efficiency.

Bearing in mind (4.9), (4.7) can be rewritten as follows

$$J_{eq}^0 \ddot{\theta}_1 = -G\dot{\theta}_1 + B_{eq}^L \dot{\theta}_2 - [-f_{1_x}(\theta_2) - f_{2_x}(\theta_2)]\, R + H v_{in}\,, \tag{4.10}$$

where

$$G = \frac{k_g^2 k_l^2 k_m^2 \eta_g \eta_m}{R_m} + \beta_{eq}^0\,, \tag{4.11}$$

$$H = \frac{k_g k_l k_m \eta_g \eta_m}{R_m}\,. \tag{4.12}$$

Equations (4.6) and (4.10) represent the complete nonlinear dynamic model of the flexible joint and are used to simulate the system behaviour. For the synthesis of the control technique proposed in §4.3, an equivalent linear model is devised. Elastic torque τ_e is the sole nonlinear term which appears in (4.6) and (4.10). It can be linearized in $\theta_2 = 0$ leading to $\tau_e \simeq -K_{stiff}\theta_2$, where

$$K_{stiff} = -\frac{d\tau_e(\theta_2)}{d\theta_2}\,|_{\theta_2=0} = \frac{2R^2 d_m^2 k_l}{(R-h)^2 + d_m^2}$$

is the stiffness constant. Consequently, (4.6) and (4.10) can be rewritten as

$$J_{eq}^0 \ddot{\theta}_1 = -G\dot{\theta}_1 + B_{eq}^L \dot{\theta}_2 + K_{stiff}\theta_2 + Hv_{in}\ , \quad (4.13)$$

$$J_{load}(\ddot{\theta}_2 + \ddot{\theta}_1) = -B_{eq}^L \dot{\theta}_2 - K_{stiff}\theta_2\ . \quad (4.14)$$

The output of the system is given by the angle formed by the end-effector with respect to the stationery frame. Hence, $y = \theta_1 + \theta_2$, i.e, the sum of the angle between the body and the stationary frame and the angle formed by the arm with respect to the body. Finally, it is possible to rewrite (4.13) and (4.14), into a state-space form

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}v_{in} \\ \mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{d}v_{in} \end{cases}$$

by assuming $\mathbf{x} := [x_1 x_2 x_3 x_4]^T = [\theta_1 \theta_2 \dot{\theta}_1 \dot{\theta}_2]^T$ and defining

$$\mathbf{A} := \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{K_{stiff}}{J_{eq}^0} & -\frac{G}{J_{eq}^0} & \frac{B_{eq}^L}{J_{eq}^0} \\ 0 & -\frac{K_{stiff}(J_{load}+J_{eq}^0)}{J_{load}J_{eq}^0} & \frac{G}{J_{eq}^0} & -\frac{B_{eq}^L(J_{load}+J_{eq}^0)}{J_{load}J_{eq}^0} \end{bmatrix}, \quad \mathbf{b} := \begin{bmatrix} 0 \\ 0 \\ \frac{H}{J_{eq}^0} \\ -\frac{H}{J_{eq}^0} \end{bmatrix}$$

$$\mathbf{C} := \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{d} := 0 \quad (4.15)$$

The corresponding discrete-time system is obtained from (4.15) by the zero-order hold equivalence, yielding to

$$\begin{cases} \dot{\mathbf{x}}_{n+1} = \mathbf{A}_0\mathbf{x}_n + \mathbf{b}_0 v_{in} \\ \mathbf{y}_{n+1} = \mathbf{C}_0\mathbf{x}_{n+1} + \mathbf{d}_0 v_{in}, \end{cases} \quad (4.16)$$

where $\mathbf{A}_0 = e^{AT}$, $\mathbf{b}_0 = \int_0^T e^{A\tau}\mathbf{b}d\tau$, $\mathbf{C}_0 = \mathbf{C}$, $\mathbf{d}_0 = \mathbf{d}$ and T is the sampling period.

4.2 Problem formulation

In this section, the minimum-time feedforward control problem is stated for scalar discrete-time systems in a general case and then for the specific case of the flexible joint presented in § 4.1.

4.2.1 General formulation

A linear discrete-time system Σ_d is described by the proper scalar transfer function

$$H(z) = \frac{b(z)}{a(z)} = \frac{b_m z^m + b_{m-1} z^{m-1} + \cdots + b_0}{a_n z^n + a_{n-1} z^{n-1} + \cdots + a_0} \quad . \tag{4.17}$$

$a(z)$, $b(z)$ are coprime, Σ_d is stable, and its static gain $H(1) \neq 0$. The system input and output sequences are denoted by $u(k)$ and $y(k)$ respectively, $k \in \mathbb{Z}$.

The behavior $\mathcal{B}_d$ of system Σ_d is the set of all input-output pairs $(u(\cdot), y(\cdot))$, where $u(\cdot), y(\cdot) : \mathbb{Z} \to \mathbb{R}$, satisfying the difference equation:

$$\begin{aligned} & a_n y(k+n) + a_{n-1} y(k+n-1) + \cdots + a_0 y(k) = \\ & b_m u(k+m) + b_{m-1} u(k+m-1) + \cdots + b_0 u(k) \, . \end{aligned} \tag{4.18}$$

The set of input-output equilibrium points of Σ_d is $\mathcal{E} := \{(u, y) \in \mathbb{R}^2 : y = H(1)u\}$ and the set $\mathcal{K}_s \subset \mathcal{B}_d$ of all rest-to-rest constrained transitions from $(0,0) \in \mathcal{E}$ to $(\frac{y_f}{H(1)}, y_f) \in \mathcal{E}$ is defined as follows.

Definition 4 *Given the parameter set* $\mathbf{s} := \{U_c, Y_c, y_f\}$*, where* $U_c = [u_c^-, u_c^+]$ *and* $Y_c = [y_c^-, y_c^+]$ *are the constraint intervals for the input and output respectively and* y_f *is the final rest value of the output,* $\mathcal{K}_s$ *is the set of all pairs* $(u(\cdot), y(\cdot)) \in \mathcal{B}_d$ *for which there exists* $k_f \in \mathbb{N}$ *such that:*

$$\begin{aligned} u(k) &= 0 \ \forall k < 0 \, , \quad u(k) = \frac{y_f}{H(1)} \ \forall k \geq k_f \, , & (4.19) \\ u(k) &\in U_c \ \forall k \in \mathbb{Z} \, , & (4.20) \\ y(k) &= 0 \ \forall k < 0 \, , \quad y(k) = y_f \ \forall k \geq k_f \, , & (4.21) \\ y(k) &\in Y_c \ \forall k \in \mathbb{Z} \, . & (4.22) \end{aligned}$$

The minimum-time feedforward constrained control problem for discrete-time systems consists in finding the optimal input sequence $u^*(k)$, $k = 0, 1, \ldots, k_f^* - 1$ for which the pair $(u^*(\cdot), y^*(\cdot)) \in \mathcal{K}_s$ is a minimizer for the optimization problem:

$$k_f^* = \min_{(u(\cdot),y(\cdot))\in\mathcal{K}_s} K_f(u(\cdot),y(\cdot)) \ . \tag{4.23}$$

where

$$K_f(u(\cdot),y(\cdot)) := \min\{k_1 \in \mathbb{N} : u(k) = \frac{y_f}{H(1)},\, y(k) = y_f \ , \forall k \geq k_1\} \ .$$

is the rest-to-rest transition time associated to pair $(u(\cdot), y(\cdot))$.

4.2.2 An approximated solution to the continuous time problem using discretization

Given a continuous system $H(s)$, a time-optimal constrained control problem can be converted into the previously defined discrete-time one through the following procedure:

- find the discretized system $H(z)$ using a zero-order equivalence, with sampling period T, by applying relation $H(z) = (1 - z^{-1})\mathcal{Z}\{\frac{H(s)}{s}\}$;
- find the time-optimal input sequence $u^*(k)$ such that (4.23) is satisfied;
- transform the discrete sequence $u^*(k)$ into the continuous function $u_c(t)$ by using a zero-order hold, that is the signal is kept constant between one sampling time and the next one;
- apply the input function $u_c(t)$ to the continuous-time system.

Fig. 4.3 gives a representation of the signals involved in the discretization. Due to the procedure given before, the control found with this method is optimal only with respect to the class of input functions which are constant in each sampling period. Hence, the resulting transition time is higher than the minimum one achievable with continuous time input functions.

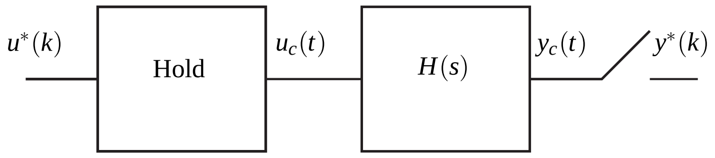


Figure 4.3: Zero-order hold equivalence

Moreover note that $y^*(k)$ fulfills for certainty the prescribed constraints, so that the output $y_c(t)$ of the continuous system satisfies the following condition

$$y_c(kT) \in Y_c, \forall k \in \mathbb{Z} ,$$

while it is not guaranteed that $y_c(t) \in Y_c$ if the time t is not multiple of the sampling period T. In other word, the output may exceed the prescribed bounds between two consecutive sampling times. Obviously, the maximum constraints violation of $y_c(t)$ is strictly related to the choice of the sampling period T. In § 4.3.1 a bound on maximum violation is found and, in turn, considerations on the choice of T are presented.

4.2.3 Problem formulation for the flexible joint system

Consider the system obtained by discretizing the rotary flexible joint system introduced in §4.1. The problem to be solved is the following one.

Problem 7 (Minimum time control problem for the flexible joint) *Consider the discrete-time system (4.16) and intervals $U_c = [u_c^-, u_c^+]$ of admissible values for the input voltage v_{in} and $Y_c = [y_c^-, y_c^+]$ of admissible values for the output angle $y = \theta_1 + \theta_2$, where y_c^- and y_c^+ represent maximum undershoot and overshoot specifications. Find the input sequence $u^*(k)$ that minimizes the time required for the rest-to-rest transition of the output $y^*(k)$ from the initial angle 0 to the desired final angle y_f, while satisfying the input and output constraints*

$$u^*(k) \in U_c, \, y^*(k) \in Y_c, \, \forall k > 0 \, .$$

4.3 Problem resolution

In this section a general method is proposed for the solution of the rest-to-rest control problem for scalar systems with bounded input and output. In the next section it will be applied to the linearized flexible-joint system.

The following theorem proposes a feasibility condition for the existence of a solution of the constrained rest to rest transition problem, which is equivalent to the non-emptiness of set $\mathcal{K}_s$ defined in Definition 4.

Theorem 1 *Set $\mathcal{K}_s$ is not empty if*

$$\{0, \frac{y_f}{H(1)}\} \subset (u_c^-, u_c^+) \quad \textit{and} \quad \{0, y_f\} \subset (Y_c^-, Y_c^+)\,, \tag{4.24}$$

with the convention that $\frac{y_f}{H(1)} = 0$ if $H(s)$ has a pole in 1.

Proof.: see Appendix 4.5.

The following proposition allows to convert the time-optimal problem into a LP-problem.

Proposition 4 *The set $\mathcal{K}_s$ of all rest-to-rest constrained transitions is not empty if and only if there exist $k_f \in \mathbb{N}$ and a vector $\mathbf{u} \in \mathbb{R}^{k_f}$ for which the following LP problem is feasible:*

$$y_c^- \cdot \mathbf{1}_{k_f} \quad \leq \mathbf{H}\mathbf{u} \leq y_c^+ \cdot \mathbf{1}_{k_f} \tag{4.25}$$

$$u_c^- \cdot \mathbf{1}_{k_f} \quad \leq \mathbf{u} \leq u_c^+ \cdot \mathbf{1}_{k_f} \tag{4.26}$$

$$\bar{\mathbf{H}} \left[\begin{array}{c} \mathbf{u} \\ \hline \frac{y_f}{H(1)} \cdot \mathbf{1}_n \end{array} \right] \quad = y_f \cdot \mathbf{1}_n \tag{4.27}$$

where $\mathbf{H} \in \mathbb{R}^{k_f \times k_f} = (h_{i,j})$ is defined by $h_{i,j} := h(i-j)$ and $\bar{\mathbf{H}} \in \mathbb{R}^{n \times (k_f+n)} = \bar{h}(i,j)$ by $\bar{h}_{i,j} := h(i + k_f - j)$.

Proof.(Necessity) Assume that there exists a vector $\mathbf{u}$ for which equations (4.25)–(4.27) are satisfied. Define the input sequence

$$u(k) = \begin{cases} 0 \text{ if } k < 0 \\ \mathbf{u}(k) \text{ if } 0 \leq k < k_f \\ \frac{y_f}{H(1)} \text{ if } k \geq k_f\,, \end{cases} \tag{4.28}$$

which satisfies Properties (4.19) and (4.20) of Definition 4. The output is given by $y(k) = \sum_{i=0}^{\infty} u(k-i)h(i)$, where $h(k)$ is the impulse response of the discrete system. Setting $\mathbf{y} \in \mathbb{R}^{k_f} = (y_1, y_2, \dots, y_{k_f})^T : y_i = y(i)$ and $\bar{\mathbf{y}} \in \mathbb{R}^n : \bar{y_i} = y(k_f + i)$, it is

$$\mathbf{y} = \mathbf{H}\mathbf{u} \;,\; \bar{\mathbf{y}} = \bar{\mathbf{H}} \left[\begin{array}{c} \mathbf{u} \\ \hline \frac{y_f}{H(1)} \cdot \mathbf{1}_n \end{array} \right] \;,$$

and, by virtue of (4.25), $y(k)$ satisfies Property (4.22) of Definition 4, $\forall k < k_f$. Finally $y(k) = y_f$, $\forall k \geq k_f$ because of Lemma 1 (see Appendix 4.5).
(Sufficiency) Assume that for a given k_f, the set $\mathcal{K}_\mathbf{s}$ is not empty, therefore it contains at least a pair $(u(k), y(k))$. If $\mathbf{u}$ and $\mathbf{y}$ are defined as above, due to (4.20) and (4.22) it follows that

$$\begin{array}{c} u_c^- \cdot \mathbf{1}_{k_f} < \bar{u} < u_c^+ \cdot \mathbf{1}_{k_f} \\ y_c^- \cdot \mathbf{1}_{k_f} < \bar{y} < y_c^+ \cdot \mathbf{1}_{k_f} \;, \end{array}$$

moreover, being $y(k) = \sum_{i=0}^{+\infty} h(k-i)u(i)$,

$$\left[\begin{array}{c} \mathbf{y} \\ \hline \bar{\mathbf{y}} \end{array} \right] = \left[\begin{array}{c} \mathbf{H} \,|\, 0 \\ \hline \bar{\mathbf{H}} \end{array} \right] = \left[\begin{array}{c} \bar{u} \\ \hline \frac{y_f}{H(1)} \cdot \mathbf{1}_n \end{array} \right] \;,$$

therefore equations (4.25)–(4.27) are satisfied. □

By virtue of Proposition 1, the minimum-time k_f^* and an associated optimal feed-forward input $u^*(k)$, $k = 0, 1, \dots k_f^* - 1$ can be determined by means of a sequence of LP feasibility tests, defined by (4.25)-(4.27)), through the simple bisection algorithm reported below. In this algorithm $LPP(\mathbf{s}, H(z), k_f, \mathbf{u})$ denotes a linear programming procedure that solves problem (4.25)-(4.27): if the problem is feasible it returns a Boolean true value along with a solution $\mathbf{u}$.

Algorithm 1: Compute the minimum-time feedforward control with input and output constraints for discrete-time systems

```
input  : $H(z)$ and $\mathbf{s}$
output: $k_f^*$ and $u^*(k)$, $k = 0, 1, \ldots, k_f^* - 1$
begin
    $k_f \longleftarrow 1$;
    $l \longleftarrow 0$;
    while $\sim LPP(\mathbf{s}, H(z), k_f, \mathbf{u})$ do
        $l \longleftarrow k_f$;
        $k_f \longleftarrow 2k_f$
    $h \longleftarrow k_f$;
    while $h - l > 1$ do
        $k_f \longleftarrow \lfloor \frac{h+l}{2} \rfloor$;
        if $\sim LPP(\mathbf{s}, k_f, \mathbf{u})$ then
            $l \longleftarrow k_f$;
        else
            $h \longleftarrow k_f$
    $k_f^* \longleftarrow h$;
    $u^*(k) \longleftarrow \mathbf{u}$
end
```

Note that parameter set $\mathbf{s} := \{U_c, Y_c, y_f\}$ (see Definition 4) contains the input and output constraints and the desired final output value.

4.3.1 Choice of the sampling period

The choice of sampling period T is critical for the proposed algorithm, since larger values of T allow a faster computation but less accurate results. The continuous time input signal $u_c(t)$ is obtained, from the optimal discrete-time sequence $u^*(k)$, through a zero-order hold, that is the signal is kept constant between one sampling time and the next one:

$$u_c(t) = u^*(k) \text{ , where } k = \{\max i | iT \le t\} \text{ .}$$

Let $y_c(t)$ be the system output and $y^*(k)$ the corresponding sampled signal. The proposed approach guarantees that $y^*(k)$ satisfies the prescribed constraints, that is

$$y_c(kT) \in Y_c, \forall k \in \mathbb{Z} \, .$$

It is worth noting that there is not any certainty that $y_c(t) \in Y_c$ for $t \neq kT$, since the optimization algorithm only checks the constraints at the sampling times. The following proposition shows that the maximum excursion of the continuous time signal $y_c(t)$ from the prescribed constraints is bounded by a term that goes to 0 as the sampling time T approaches to 0.

Proposition 5 *Consider the continuous-time scalar system*

$$\begin{aligned} \dot{x} &= \mathbf{A}x + \mathbf{b}u \\ y &= \mathbf{C}x \, , \end{aligned}$$

where A is a nonsingular matrix and let $Y_c = [y_c^-, \, y_c^+]$ be two nonempty intervals. If $u(t)$ is constant in $[kT, (k+1)T[$, $\forall k \in \mathbb{Z}$, and $y(kT) \in Y_c$, $\forall t = kT$, then for any integer $l \in \mathbb{Z}$, $l \geq 2$ the following inequality is satisfied

$$\begin{aligned} \max_{t \in \mathbb{R}} \{y(t) - y_c^+ , \, y_c^- - y(t)\} \leq & \left(\|C\|(e^{\|A\|T} - 1 - \|A\|T) - \sum_{i=2}^{l} \frac{T^i}{i!}(\|C\|\|A\|^i - \|CA^i\|\right) \\ & \cdot(\max_k \|x(kT)\| + \|B\|\|A\|^{-1} \max_k \|u(kT)\|) \, . \end{aligned} \tag{4.29}$$

Proof.: see Appendix 4.6.

Remark 3 *Proposition 5 gives a set of estimates for the maximum output constraints violation* $\max_{t\in\mathbb{R}}\{y(t) - y_c^+ , \, y_c^- - y(t)\}$*. The estimates depend on the integer parameter l and become more accurate as l increases and the sample time T decreases. Choosing $l = 2$, for instance, the following bound is obtained*

$$\begin{aligned} \max_{t \in \mathbb{R}} \{y(t) - y_c^+ , \, y_c^- - y(t)\} \leq & \left(\|C\|(e^{\|A\|T} - 1 - \|A\|T) - \frac{T^2}{2}(\|C\|\|A\|^2 - \|CA^2\|\right) \\ & \cdot(\max_k \|x(kT)\| + \|B\|\|A\|^{-1} \max_k \|u(kT)\|) \, . \end{aligned} \tag{4.30}$$

Note that the proposed approach guarantees the discretized system reaches the desired equilibrium at the final sample k_f^*, but this does not necessarily imply that also the underlying continuous-time system reaches the equilibrium. The following proposition shows that this requirement is fulfilled if a restriction on sampling time T is imposed.

Proposition 6 *Let be given a continuous-time system Σ with transfer function*

$$H(s) = \frac{b_m s^m + b_{m-1}s^{m-1} + \ldots + b_0}{s^n + a_{n-1}s^{n-1} + \ldots + a_0} ,$$

where $n > m$, $T > 0$, $t_0 \in \mathbb{R}$. Consider an input-output pair $(u(t), y(t)) \in \mathcal{B}$ such that

$$u(t) = \frac{y_f}{H(0)},\ \forall t \geq t_0 ,$$

$$y(t_0 + kT) = y_f,\ \textit{for}\ k = 0, \ldots, n-1 , \tag{4.31}$$

and for which the distinct roots $p_1, \ldots, p_l$ of the characteristic polynomial $s^n + a_{n-1}s^{n-1} + \ldots + a_0$ satisfy

$$p_i - p_r \neq k\frac{2\pi j}{T},\ \forall i, r = 1, \ldots, l,\ \forall k \in \mathbb{Z} - \{0\} , \tag{4.32}$$

then the following condition is satisfied

$$y(t) = y_f,\ \forall t \geq t_0 .$$

Proof.: see Appendix 4.7.

Remark 4 *Condition (4.32) is satisfied if*

$$T < \frac{2\pi j}{\max_i\{Im\{p_i\}\}} \forall i = 1, \ldots, l, \tag{4.33}$$

that is the sampling time is less that 2π divided by the largest imaginary part among the system poles.

In conclusions the sampling time T must be chosen sufficiently small such that condition (4.33) is satisfied and bound (4.29) is sufficiently small.

4.4 Simulation and experimental results

In this section the control method proposed in §4.3 is applied to the case of the flexible joint model derived in §4.1.

Simulation are executed on a P4 3.0Ghz computer within Matlab programming environment. The freely available library GPLK (GNU Linear Programming Kit) [59] is used as linear programming solver and interfaced with Matlab through [60]. Experimental results are obtained by interfacing the flexible joint device with Matlab through the Quanser Q4 PCI data acquisition board.

Electrical data			**Gears parameters**			**Viscous frictions**	
R_m	k_m	η_m	k_g	k_l	η_g	B_{eq}^0	B_{eq}^L
2.6	$7.67\ 10^{-3}$	0.69	14	5	0.9	$14.99\ 10^{-3}$	$11.42\ 10^{-3}$

Inertias					
J_m	J_{120}	J_{72}	J_{24}	J_{FJ}	J_{load}
$0.386\ 10^{-6}$	$0.440\ 10^{-6}$	$5.274\ 10^{-6}$	$0.195\ 10^{-6}$	$2.10\ 10^{-3}$	$11.03\ 10^{-3}$

Table 4.1: Flexible joint parameters

By substituting the flexible joint parameters defined in Table 4.1 in state-space model (4.15), the following numerical representation for the plant is obtained

$$\mathbf{A} := \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 379.9 & -56.65 & 2.956 \\ 0 & -512.9 & 56.65 & -3.99 \end{bmatrix}, \quad \mathbf{b} := \begin{bmatrix} 0 \\ 0 \\ 93.74 \\ -93.74 \end{bmatrix}$$

$$\mathbf{C} := \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{d} := 0. \tag{4.34}$$

Two different cases have been considered. In the first one, output constraints have been imposed only on the end-effector angle. According to the flexible joint model in §4.1, this imply to set a limit on the sum of the rotation angle of the body, θ_1, plus the displacement θ_2 induced by the joint elasticity, i.e. $y = \theta_1 + \theta_2$. In the second case, additional constraints on the joint displacement θ_2 have been added. In fact, limiting the angle induced by the joint flexibility between the arm and the rotating body, allows keep bounded the torsion moment on the joint itself, which in turn, implies reducing the reflected joint solicitation torque.

4.4.1 Control without constraints on θ_2

A rest-to-rest transition from $y = 0$ to $y = y_f = \pi/4$ is considered. The flexible joint is driven with an amplifier whose maximum bipolar voltage is equal to $\pm 5\,V$. Therefore, the input constraint is $\|u(t)\|_\infty \leq 5$, so that $U_c = [-5, +5]$. A strong requirement has been set on the output function: a maximum of 0.1% overshoot and undershoot is allowed on y, so that $Y_c = [-7.8539 \cdot 10^{-4}, \pi/4\ + 7.8539 \cdot 10^{-4}]$. First of all note that condition (4.24) of Theorem 1 is satisfied. In fact, since the flexible joint discretized transfer function has two poles in $z = 1$, condition (4.24) reduces to

$$\{0\} \in U_c, \frac{\pi}{4} \in Y_c\ .$$

This mean that set $\mathcal{K}_s$ is nonempty.

Condition (4.33) must be satisfied in order to ensure that the continuous-time system reaches the equilibrium with the same transient time of the discretized one. This implies that $T < 0.57\ s$. In all the simulation examples, the sampling time has been chosen equal to $T = 0.001\ s$. Simulation results, obtained with the algorithm described in §4.3, are shown in Figs. 4.4 and 4.5.

Fig. 4.4 highlights the bang-bang control input which makes it possible to obtain a rest-to-rest transition time of $t_f^* = 0.305\ s$. Fig. 4.5 plots a comparison between the ideal simulated output and the real behavior of the flexible joint. The real output shows a small overshoot and undershoot: this is due to the small mismatching between the real plant and the flexible joint model devised in §4.1.

The maximum error on the output constraints for the continuous-time system, obtained with (4.29) for a sampling time $T = 0.001\ s$, is given by $\max_{t\in\mathbb{R}}\{y(t) - y_c^+, y_c^- - y(t)\} \leq 0.00113\ rad$.

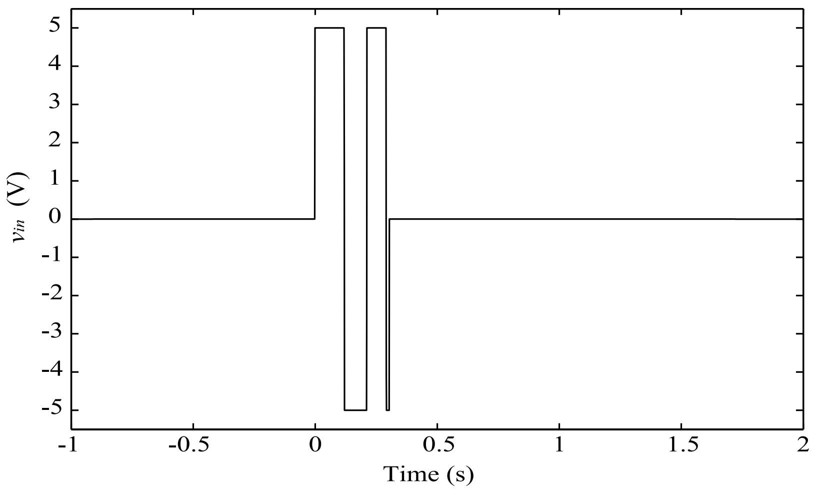


Figure 4.4: Optimal input signal devised with the proposed approach.

The herein proposed approach has been compared with that presented in [61] and [62], where a time-optimal control is found by means of dynamic inversion based on the so called “transition polynomials” (see [61]). For brevity we recall here only the general expression of this type of interpolating polynomials that allows an arbitrarily smooth transition between two constant output values (in this case 0 and $\pi/4$):

$$y(t;\tau) = \begin{cases} 0 \quad \text{if} \quad t \leq 0, \\ \frac{(2k+1)!}{k!\tau^{2k+1}} \sum_{i=0}^{k} \frac{(-1)^{k-i}}{i!(k-i)!(2k-i+1)} \tau^i t^{2k-i+1} \quad \text{if} \quad 0 \leq t \leq \tau, \\ \pi/4 \quad \text{if} \quad t \geq \tau \end{cases}$$

where y is the desired output function, k is the relative order of the plant transfer function and τ is the minimum transition time. In this case the plant transfer function, from (4.34), is equal to:

$$H(s) = \frac{96.97s + 1.247 \cdot 10^4}{s^4 + 60.64s^3 + 571.5s^2 + 7534s}$$

thus the relative order is $k = 3$.

Results obtained by applying that planning method are shown in Figs. 4.6 and 4.7. Comparing Fig. 4.6 and Fig. 4.4, it is clearly visible that the approach based on “transition polynomials” allows to generate smoother input control. However, the

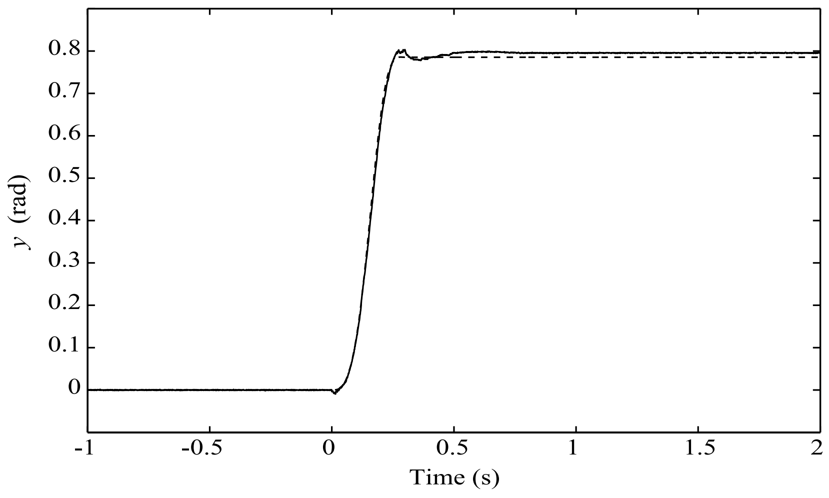


Figure 4.5: Expected system output y (dashed line) and measured plant output (solid line)

output function presents a slightly oscillatory behavior, causing an increase of the real transition time with respect to the simulated one, which was equal to $t_f^* = 0.360$ s. Hence, the two methods are both suitable for the constrained control of the considered flexible joint. However, when the application requires rapid maneuvers, the proposed bang-bang control allows to perform the required output transition in a smaller time.

4.4.2 Control with constraints on θ_2

The minimum-time control law used in the previous simulations and experiments does not take care of the solicitation torque induced to the joint by the deflection angle θ_2: the only constraint that has been imposed is related to the end-effector position, i.e. the sum of $\theta_1 + \theta_2$.

On a real flexible-joint robot it can be interesting to devise a time-optimal transition control also constraining the maximum admissible displacement between the link position and the joint position, thus reducing the mechanical solicitation on the joint itself.

Thus, under the same constraints used in previous section, a limit on θ_2 angle has

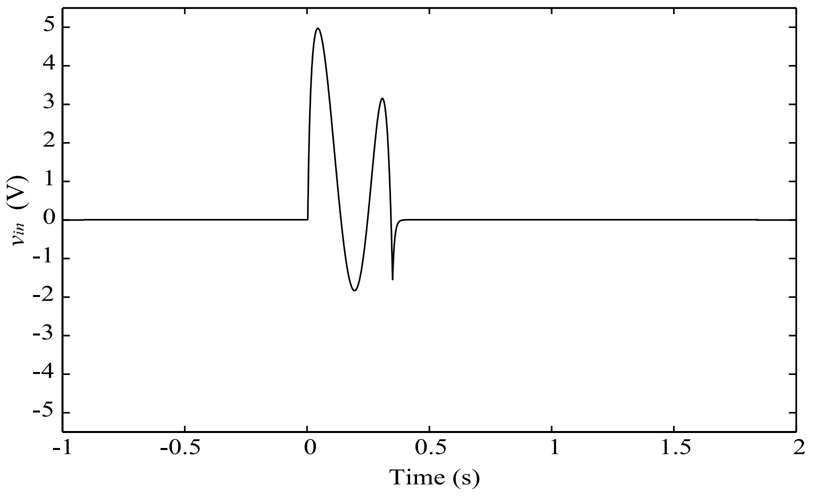


Figure 4.6: Optimal transition polynomial input signal

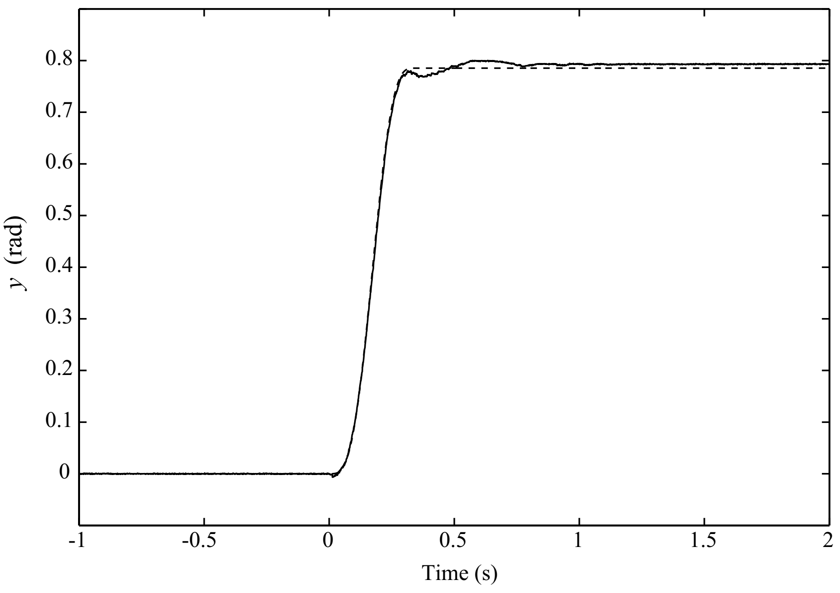


Figure 4.7: Expected system output y (dashed line) and measured plant output (solid line)

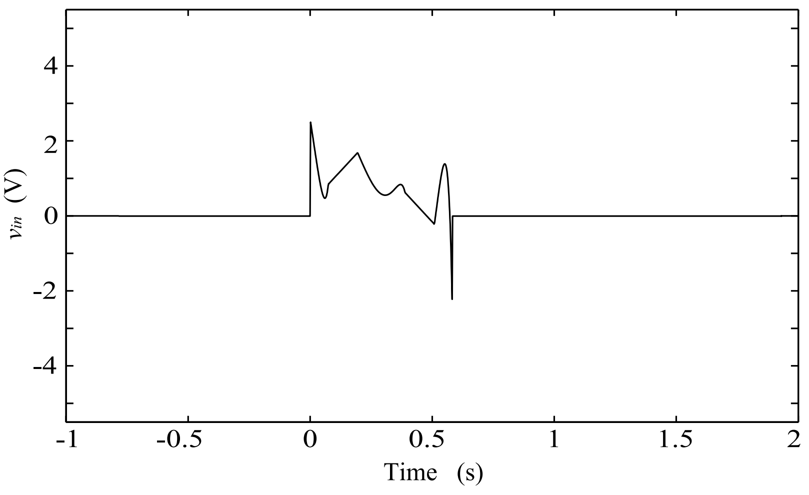


Figure 4.8: The time-optimal control with angle limit on θ_2

been added such that $\theta_2 \in [-5\pi/180, 5\pi/180]$.
Simulated and experimental results are reported in Figs. 4.8 and 4.9. In particular in Fig. 4.10 is reported the time-waveform of the relative displacement between the arm and the rotating body. As it is shown, the θ_2 angle is constantly saturated to the imposed constraint value, and this is the reason why the optimal control is no longer a bang-bang function. Clearly the optimal transition time increases: in this case $t_f^* = 0.59$ s.

4.4.3 Computational complexity

In this section, some considerations are given for what concern the computational complexity of the time-optimal algorithm. In particular, Table 4.2 shows the computational time required by the proposed approach to devise the time-optimal control sequence. Symbol $\Delta\theta$ has been used to represent the overall rest-to-rest transition, while T indicates, as always, the sample time required by the discretization phase. As it can be seen, performances strongly depend on the used sampling time: by reducing T, which means sampling the continuous-time system with an higher frequency, the dimension of the resulting LP problem increases, thus causing an increment of the

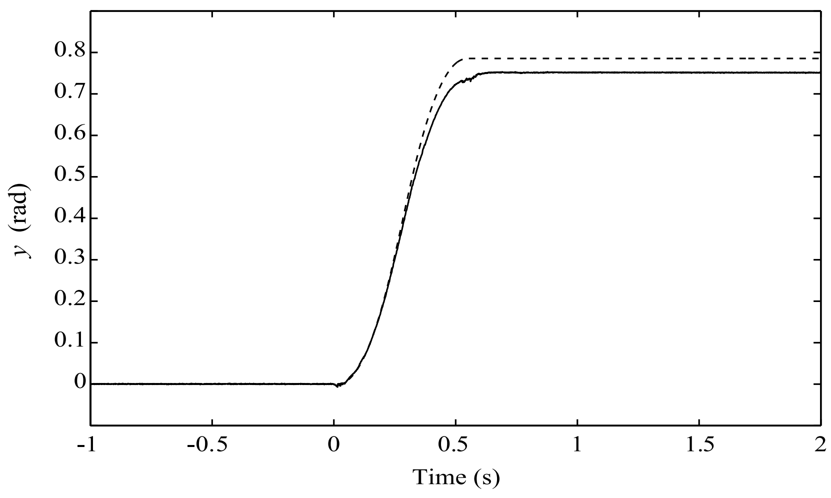


Figure 4.9: Expected system output y (dashed line) and measured plant output (solid line)

total computational time. Considering the computational complexity, Karmarkar has shown in [63] that a linear programming problem can be solved by means of an algorithm with running time proportional to $n^{3.5}$, where n is the number of inequalities. In our case this would means that each feasibility test would require a time proportional to $n_s^{3.5}$, where n_s is the total number of samples. The complexity of the bisection search, with respect to the minimum number of samples, is given by $O(\log n_s)$, therefore the total complexity of the proposed algorithm is given by $O(\log n_s n_s^{3.5})$. In our tests the dual simplex method has been used. It is well known (see [64]) that the simplex method complexity is theoretically exponential with respect to n, due to the existence of special worst cases, but, in practice, the complexity is almost linear with respect to n. This would mean that the "practical" complexity of the proposed algorithm is $O(\log n_s n_s)$. In any case, it is important to keep the number of samples (which is inversely proportional to the sampling time T) as small as possible.

Generally the time required by the algorithm to obtain the optimal control has an order of magnitude of a few seconds and can be further improved if the algorithm is directly coded in C/C++. Since the algorithm performances are predictable, once the sampling time is set, the proposed approach can be used in a real-time context.

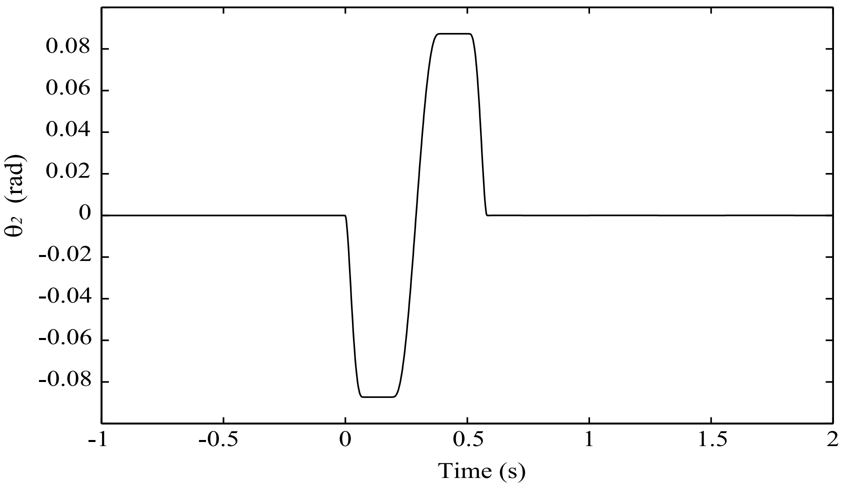


Figure 4.10: θ_2 angle

4.5 Proof of Theorem 1

The following lemma will be used in the proof.

Lemma 1 *Consider system (4.17) and be the pair $(u(k), y(k)) \in \mathcal{B}_d$. If*

$$\begin{array}{ll} y(i+N) = y_f & \textit{for } i = 0, \ldots, n-1 \\ u(i+N) = \frac{y_f}{H(1)} & \textit{for } i \geq 0 \,, \end{array}$$

then

$$y(i) = y_f, \forall i \geq N. \tag{4.35}$$

Proof of the lemma Consider the input-output pair $(u_2(k), y_2(k)) = \left(u(k) - \frac{y_f}{H(0)}, y(k) - y_f\right)$. Since $u_2(k) = 0$, $\forall k \geq N$, therefore, for $k \geq N$, $y_2(k)$ satisfies the following difference equation

$$\begin{cases} a_n y_2(k+n) = -a_{n-1} y(k+n-1) - a_{n-2} y_2(k+n-2) - \cdots - a_0 y_2(k) \\ y_2(N) = y_2(N+1) = \cdots = y_2(N+n-1) = 0 \,, \end{cases}$$

which has solution $y_2(k) = 0$, $\forall k \geq N$. Consequently, (4.35) follows. $\square$

$\Delta\theta$ (rad)	$T(s)$	Computation time (s)	Transition time $t_f^*(s)$
$\pi/4$	$1\cdot10^{-3}$	$7.943\cdot10^{0}$	$3.05\cdot10^{-1}$
	$1\cdot10^{-2}$	$1.023\cdot10^{0}$	$3.10\cdot10^{-1}$
	$5\cdot10^{-2}$	$6.854\cdot10^{-1}$	$4.00\cdot10^{-1}$
$\pi/2$	$1\cdot10^{-3}$	$9.275\cdot10^{0}$	$3.88\cdot10^{-1}$
	$1\cdot10^{-2}$	$6.882\cdot10^{-1}$	$3.90\cdot10^{-1}$
	$5\cdot10^{-2}$	$8.906\cdot10^{-1}$	$5.00\cdot10^{-1}$

Table 4.2: Algorithm Perfomances

Proof of the theorem Define a continuous function $l(t)$ which has the following properties:

$$\begin{cases} l(t)=0 \text{ if } t\le 0 \\ l(t)=\frac{y_f}{H(1)} \text{ if } t\ge 1 \\ 0\le l(t)\le \frac{y_f}{H(1)} \;\; \forall t\in[0,1] \end{cases}$$

Impose $u_N(k)=l(\frac{k}{N})$ and let $U_N(z)$ be the corresponding Z-transform. Moreover, be $Y_N(z)=U_N(z)H(z)$ and $y_N(k)=\mathcal{Z}^{-1}\{Y_N(z)\}$.

First of all it is proved that

$$\lim_{N\to+\infty}||H(1)\,u_N(k)-y_N(k)||_\infty=0 \tag{4.36}$$

Indeed,

$$H(1)\,U_N(z)-Y_N(z)=H(1)\,U_N(z)-H(z)\,U_N(z)=(H(1)-H(z))U_N(z)\;.$$

Being $H(1)-H(z)|_{z=1}=0$, function $H(1)-H(z)$ has a zero in $z=1$. Hence, $H(1)-H(z)=(z-1)H'(z)$, where $H'(z)$ has the same poles as $H(z)$. Therefore $(H(1)-H(z))U_N(z)=H'(z)(z-1)U_N(z)$ and

$$\lim_{N\to+\infty}||H(1)\,u_N(k)-y_N(k)||_\infty\le\lim_{N\to\infty}||\mathcal{Z}^{-1}\{H'(z)\}||_2\;\;||u_N(k+1)-u(k)||_\infty=0\;,$$

in fact

$$\lim_{N\to+\infty} ||u_N(k+1) - u(k)||_\infty = 0$$

because function $l(t)$ is uniformly continuous and $\|\mathcal{Z}^{-1}\{H'(z)\}\|_2$ is finite because $H'(z)$ is stable.

Equation (4.36) shows that as N approaches infinity, the output $y_N(k)$ becomes equal to input $u_N(k)$ multiplied by the static gain $H(1)$ and, for $k \geq N$, the difference $y_N(k) - y_f$ tends to zero. In the following a correcting term $\bar{y}_N(k)$ is introduced such that $y_N(k) + \bar{y}_N(k) = y_f$, $\forall k \geq N$. Define the error vector $\mathbf{e}_N \in \mathbb{R}^n = (e_{N,0}, e_{N,1}, \dots, e_{N,n-1})^T$ as

$$e_{N,i} = y_N(N+i) - y_f, \;\; i = 0, \dots, n-1 \; ,$$

let $\mathbf{M} \in \mathbb{R}^{n\times N} = (m_{i,j})$ be such that

$$m_{i,j} = h(j-i), \;\; i = 1, \dots, n, \;\text{ and }\; j = 1, \dots, N \; ,$$

where $h(k) = \mathcal{Z}^{-1}\{H(z)\}$ denotes the system impulse response. Set $\bar{\mathbf{u}}_N = (\bar{u}_{N,0}, \bar{u}_{N,1}, \dots, \bar{u}_{N,n-1})^T$ as

$$\bar{\mathbf{u}}_N = -\mathbf{M}^+\mathbf{e}_N \; ,$$

where $\mathbf{M}^+ = \mathbf{M}^t(\mathbf{M}^t\mathbf{M})^{-1}$ is the pseudo-inverse of $\mathbf{M}$.

Define the correcting input vector $\bar{u}_N$ as

$$\begin{cases} \bar{u}_N(N+k) = \mathbf{u}_{N,k} \text{ if } 0 \leq k < n-1 \; , \\ 0 \text{ otherwise} \end{cases}$$

and let $\bar{y}_N(k)$ be the corresponding output. Consider as input $u_N(k) + \bar{u}_N(k)$, the corresponding output is $y_N(k) + \bar{y}_n(k)$. The following conditions are satisfied:

$$\begin{aligned} y_N(k) + \bar{y}_N(k) &= y_f, \; \forall k \geq N \; , & (4.37) \\ \lim_{N\to+\infty} \|\bar{u}_N(k)\|_\infty &= 0, & (4.38) \\ \lim_{N\to+\infty} \|\bar{y}_N(k)\|_\infty &= 0. & (4.39) \end{aligned}$$

If fact (4.37) follows from the fact that

$$y_N(N+k) + \bar{y}(N+k) = y_f + e_N(k) - e_N(k) = y_f. \;\; k = 0, \dots, n-1 \; ,$$

and $y_N(k)+\bar{y}_N(k)=y_f$, $\forall k \geq N$ as a consequence of Lemma 1. Conditions (4.38) and (4.39) follows from the following inequalities

$$\lim_{N\to+\infty} \|\bar{u}_N(k)\|_\infty \leq \|\mathbf{M}^+\|_2 \lim_{N\to+\infty} \|e_n\|_\infty = 0 \;,$$
$$\lim_{N\to+\infty} \|\bar{y}_N(k)\|_\infty \leq \|h(k)\|_2 \|\mathbf{M}^+\|_2 \lim_{N\to+\infty} \|e_n\|_\infty = 0 \;,$$

being $\lim_{N\to\infty} \|e_n\|_\infty = 0$ by (4.36).

Therefore

$$\lim_{N\to\infty} \max \Big\{ u_N(k)+\bar{u}_N(k)-\frac{y_f}{H(1)}, -u_N(k)-\bar{u}_N(k),$$
$$y_N(k)+\bar{y}_N(k)-y_f, -y_N(k)-\bar{y}_N(k) \Big\} = 0$$

and, because of (4.24), for N sufficiently large the following property holds

$$\begin{aligned} \max\big\{ & u_N(k)+\bar{u}_N(k)-u_c^+, -u_N(k)-\bar{u}_N(k)-u_c^-, \\ & y_N(k)+\bar{y}_N(k)-y_c^+, -y_N(k)-\bar{y}_N(k)-y_c^- \big\} < 0 \;, \end{aligned} \tag{4.40}$$

therefore all properties (4.19)-(4.22) are verified. In fact (4.19) is verified by construction, (4.21) comes from (4.37) and (4.20), (4.22) follow from (4.40). □

4.6 Proof of Proposition 5

For any $k \in \mathbb{Z}$ and $\tau \in [0,1]$, set

$$e(\tau) = y(kT+\tau T) - [\tau y((k+1)T) + (1-\tau) y(kT)] \;, \tag{4.41}$$

and note that $e(0)=e(1)=0$. Since $y(kT+\tau T) = \mathbf{C}e^{\mathbf{A}\tau T}x(kT) + \mathbf{C}\int_0^{\tau T} e^{\mathbf{A}h}\mathbf{B}u(kT)dh$ equation (4.41) becomes

$$e(\tau) = \mathbf{C}(e^{\mathbf{A}\tau T} - I - \tau(e^{\mathbf{A}T} - I))x(kT) + \mathbf{C}\left(\int_0^{\tau T} e^{\mathbf{A}h}dh - \tau \int_0^{T} e^{\mathbf{A}h}dh\right)\mathbf{B}u(kT) \;, \tag{4.42}$$

It is known that

$$\int_0^x e^{\mathbf{A}h}dh = (e^{\mathbf{A}x} - I)\mathbf{A}^{-1}$$

therefore (4.42) can be rewritten as follows

$$e(\tau) = \mathbf{C}(e^{\mathbf{A}\tau T} - I - \tau(e^{\mathbf{A}T} - I))(x(kT) + \mathbf{A}^{-1}\mathbf{B}u(kT)) \; . \tag{4.43}$$

Define $z(\tau) = \mathbf{C}(e^{\mathbf{A}\tau T} - I - \tau(e^{\mathbf{A}T} - I))$. It is easily verifed that $z(0) = 0$ and $\frac{dz}{d\tau}(0) = 0$. Consequently $z(\tau)$ can be expanded as follows

$$z(\tau) = \mathbf{C}\sum_{i=2}^{+\infty} \frac{\mathbf{A}^i T^i \tau^i}{i!} \; , \tag{4.44}$$

i.e. the two first element of the series are missing. Equation (4.44) can be manipulated leading to

$$\begin{aligned} z(\tau) &\le \sum_{i=2}^{+\infty} \frac{\|\mathbf{C}\mathbf{A}^i T^i\|\tau^i}{i!} \le \sum_{i=2}^{l} \frac{\|\mathbf{C}\mathbf{A}^i T^i\|}{i!} + \|\mathbf{C}\| \sum_{i=l+1}^{+\infty} \frac{\|\mathbf{A}T\|^i}{i!} = \\ &= \mathbf{C}(e^{\|\mathbf{A}\|T} - I - \|\mathbf{A}\|T) - \sum_{i=2}^{l} (\|\mathbf{C}\|\|\mathbf{A}\|^i - \|\mathbf{C}\mathbf{A}^i\|)\frac{T^i}{i!} \; . \end{aligned} \tag{4.45}$$

Finally, substituting (4.45) in (4.43) and using the fact that $\max_{t\in\mathbb{R}}\{y(t) - y_c^+, \, y_c^- - y(t)\} \le \max_{t\in\mathbb{R}} |e(t)|)$ we obtain the thesis. □

4.7 Proof of Proposition 6

Define $\bar{y} = y - y_f$ and $\bar{u} = u - \frac{y_f}{H(0)}$, then the following differential equation is satisfied $\forall t \ge t_0$:

$$D^n \bar{y}(t) + a_{n-1}D^{n-1}\bar{y}(t) + \ldots + a_0\bar{y}(t) = 0 \; . \tag{4.46}$$

Indicate with ρ_i, $i = 1, \ldots, l$ the multiplicities respectively associated to any root p_i, then, $\forall t \ge t_0$ the solution of (4.46) can be expressed in the following form

$$\begin{aligned} y(t) = &\sum_{i=1,\ldots,l} \sum_{j=0,\ldots,\rho_i - 1} c_{i,j} \cdot \\ &\cdot \frac{(t-t_0)(t-t_0-T)\cdots(t-t_0-(j-1)T)}{T^j (j-1)!} e^{p_i(t-t_0-jT)} \; , \end{aligned} \tag{4.47}$$

where $C_{i,j}$ are suitable constants. Due to (4.47), (4.31) can be written as follows

$$0 = \mathbf{V}\mathbf{C} \; , \tag{4.48}$$

where $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_l]$ and

$$\mathbf{V}_i = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ e^{Tp_i} & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \\ e^{(\rho_i-1)Tp_i} & (\rho_i-1)e^{(\rho_i-1)Tp_1} & \frac{(\rho_i-2)(\rho_i-1)}{2}e^{(\rho_i-1)Tp_i} & \ldots & 1 \end{bmatrix},$$

$$\mathbf{C} = \left(c_{1,0},\ c_{1,1},\ \ldots,\ c_{1,\rho_1-1},\ c_{2,0},\ \ldots,\ c_{l,\rho_l-1}\right)^T .$$

$\mathbf{V}$ is the *generalized Vandermonde matrix* and, as stated in [65], $det(\mathbf{V}) = \prod_{1\le i<j\le l}(e^{p_iT} - e^{p_jT})^{\rho_i\rho_j}$. Because of (4.32), $det\mathbf{V} \neq 0$, so that from (4.48) it follows that $\mathbf{C} = 0$, and therefore $y(t) = 0$, $\forall t \geq t_0$. $\square$

CHAPTER 5

Minimum-time feedforward control based on convexity

What is now proved was once only imagined.
William Blake

In the previous chapter the optimal control problem has been solved by discretizing the linear or linearized dynamic of the considered system, converting the original constrained problem into a set of feasibility tests of an equivalent linear programming formulation. The proposed approach allows to simplify the search of the optimal minimum-time solution for a constrained rest-to-rest transition but, as previously remarked, it has two major drawbacks: it is designed only to deal with linear systems and, due to the discretization step, the choice of the sampling time is critical in order to fulfill given constraints between two sampling instant.

For these reasons efforts have been spent in order to define a computational algorithm able to solve the minimum-time problem with a pure differential method. Moreover the original goal has been to consider a wider class of dynamic systems, also nonlinear, satisfying some necessary conditions for the algorithm convergence.

In the general case the solution of minimum time problems can be formulated using the Pontryagin's Maximum Principle (PMP) for which, under some hypothesis, every optimal solution can be generated with the knowledge of two parameters: the transition time, t^*, and the final costate, q_1, which is the normal vector to the boundary of the set reachable at time t^* at the final state. Generally the analytical solution of PMP is quite hard to find unless the system is of low order, time invariant and linear.

Such problems may be solved numerically, and a number of time-optimal bang-bang control algorithms have been proposed in the literature, such as the shooting method or other iterative procedures. The shooting method for time optimal control was originally proposed in [66] for a class of simple systems for which an initial good guess of the costate values were possible. In general, the shooting method has shown an high convergence sensitivity to the costate initial guess. Other proposed approaches make use of geometrical considerations. In particular, for the scope of this chapter, one can recall those introduced in [67] and [68]. The geometric ideas at the basis of the approach described in the following are similar.

The proposed algorithm, based on PMP, is in fact able to find the right values of t^* and q that guarantee to reach the final state x_1, through a geometric method that makes use of the convexity of the system reachable sets. The algorithm is based on a differential equation that determines a function $x(\lambda)$ which is a vector that converges to the final state x_1. For every λ, $x(\lambda)$ belongs to the boundary of the set reachable from the initial state x_0 in a time that grows with λ. The error function, defined as the norm of the distance between the state $x(\lambda)$ and the final state x_1 is monotonically decreasing. A proof of convergence is presented.

The chapter is organized as follows. In §5.1 the control problem is proposed and the solution is obtained in the subsequent section. In §5.3 some considerations about numerical issues are presented; then in §5.4 some simulations are discussed.

Notation: For any two vectors $v, w \in \mathbb{R}^n$, $< v,\ w >= \sum_{i=1}^{n} v_i w_i$ denotes the scalar product. Given $v \in \mathbb{R}^n$, $\hat{v} = \frac{v}{\|v\|}$ denotes the unit vector having the same direction as v. Given a set $\mathcal{A} \subset \mathbb{R}^n$, $\partial\mathcal{A}$ denotes the boundary of $\mathcal{A}$ and $I(\mathcal{A})$ denotes its internal part. Given a differential manifold $\mathcal{M} \in \mathbb{R}^n$, $T_x(\mathcal{M})$ denotes the tangent space of $\mathcal{M}$ at x. The set $S^n \subset \mathbb{R}^{n+1} = \{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$ is the unit ball of $\mathbb{R}^{n+1}$.

5.1 Preliminaries

Consider a time independent non linear system of the following form

$$\begin{cases} \dot{x} = f(x,u) \\ x(0) = x_0 \end{cases} \tag{5.1}$$

where $x \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$. It is assumed that the input function $u(t)$ satisfies the following condition

$$u(t) \in U, \forall t \geq 0,$$

where $U \subset \mathbb{R}^m$ is an arbitrary convex set. The notation $x_u(t)$ denotes the solution of (5.1) for a given input function $u(t)$.

The *time-optimal problem* consists in minimizing the time needed for a transition from an initial state x_0 to a final state x_f

$$\min_u \{t^* | x_u(t^*) = x_f\} \ , u(t) \in U, \ \forall t \geq 0 \ .$$

5.1.1 Characterization of the optimal solution

One of the most important tools for optimal control is the Pontryagin's Maximum Principle (PMP), which gives a necessary condition for optimality. In the case of minimum-time problem it can be formulated as follows.

Theorem 2 (PMP) *If $u^*(t)$ is an admissible control for system (5.1) that is a solution of the time-optimal problem with final time t^*, then there exists a Lipschitz function*

$$q(t) \in \mathbb{R}^n, \ q(t) \neq 0, \forall t \in [0,t^*]$$

such that, almost everywhere on $[0,t^*]$,

$$\begin{aligned} &< q, \ f(x,u^*) >= \max_{u \in U} < q, \ f(x,u) > , \\ &\dot{q}^T(t) = -q^T \tfrac{df}{dx}|_{x = x_{u^*}(t)} \ , \\ &< q, \ f(x,u^*) >= 1 \ . \end{aligned} \tag{5.2}$$

Function $H(x,q,u) =< q,\ f(x,u) >$ is called Hamiltonian and its value is constantly 1 along the time-optimal solution.

In the case of linear systems Equation (5.1) takes the form

$$\begin{aligned} &\dot{x} = Ax + bu \\ &x(0) = x_0 \end{aligned}$$

and the costate equation reduces as follows

$$\begin{aligned} &\dot{q} = -A^T q \\ &< q,\ f(x,u^*) >= \max_{u \in U} < q,\ Bu > \ . \end{aligned}$$

The reachable set $\mathcal{A}_{x_0}(t)$ represents the states that can be reached at time t, starting from the initial condition x_0.

Definition 5 *The reachable set of system (5.1) is*

$$\mathcal{A}_{x_0}(t) = \{x_u(t) | u \in L^\infty([0,t],U)\} \ .$$

Remark 5 *If state $x_f \in \mathbb{R}^n$ is reached in minimum time t^* from the initial state x_0, then it must be*

$$x_f \in \partial \mathcal{A}_{x_0}(t^*) \ ,$$

that is x_f belongs to the boundary of the set accessible from x_0 in time t^*.

5.1.2 Characterization of convexity

A subset $\mathcal{C}$ of $\mathbb{R}^n$ is strictly convex when the internal part of the segment joining any couple of points of $\mathcal{C}$ belongs to its interior.

Definition 6 *A set $\mathcal{C} \in \mathbb{R}^n$ is strictly* convex *if $\forall x,y \in \mathcal{C}$, $x + \lambda(y-x) \in I(\mathcal{C})$, $\forall \lambda \in (0,1)$.*

A vector w is said to be normal to a convex subset $\mathcal{C}$ at a point x, where $x \in \mathcal{C}$, if w does not make an acute angle with any line segment in $\mathcal{C}$ with x as endpoint, i.e $< x-y,\ w > \geq 0$ for every $y \in \mathcal{C}$. The set of all vectors w normal to $\mathcal{C}$ in x is called the normal cone to $\mathcal{C}$ at x, as reported in Definition 2, chapter 5 of [69].

Definition 7 *The normal cone at $x \in \mathcal{C}$ where $\mathcal{C}$ is a convex subset of $\mathbb{R}^n$, is given by*

$$N_{\mathcal{C}}(x) = \{p \in \mathbb{R}^n | < p,\, x - y > \geq 0 \;, \forall y \in \mathcal{C}\} \;. \tag{5.3}$$

Proposition 7 *If $\mathcal{C}$ is a closed, bounded and strictly convex subset of $\mathbb{R}^n$ then for any $q \in S^{n-1}$ there exist one and only one $x \in \mathbb{R}^n$ such that $q \in N_{\mathcal{C}}(x)$.*

Proof. Given $q \in S^{n-1}$ define the family of hyperplanes normal to q and parameterized by $\lambda \in \mathbb{R}$

$$\mathcal{H}(\lambda) = \lambda q + \{x| < x,\, q >= 0\}.$$

Because of the boundedness of $\mathcal{C}$ the following maximum is well-defined

$$\overline{\lambda} = \max\{\lambda > 0 | \mathcal{H}(\lambda) \cap \mathcal{C} \neq \emptyset\}$$

moreover $\mathcal{H}(\lambda) \cap \mathcal{C}$ contains only one vector $x_0 \in \mathbb{R}^n$. In fact if there existed $x_1,\, x_2 \in \mathcal{H}(\lambda) \cap \mathcal{C}$, with $x_1 \neq x_2$, then, being $\mathcal{H}(\lambda) \cap \mathcal{C} \in \partial C$ a convex set,

$$x_1 + \lambda(x_2 - x_1) \in \mathcal{H}(\lambda) \cap \mathcal{C} \;\; \forall \lambda \in [0,\, 1]$$

subset $\mathcal{C}$ cannot be strictly convex because the segment connecting x_1 and x_2 belongs to its boundary. By the Proposition 7, the mapping

$$T(q) = x, \text{ if } q \in N_{\mathcal{C}}(x) \;, \tag{5.4}$$

is well defined. Mapping $T(q)$ associates to every possible normal vector $q(t) \in S^{n-1}$ the vector x which lies on the boundary of manifold $\mathcal{C}$ such that q belongs to the normal cone of $\mathcal{C}$ at x.

Proposition 8 *For any $q_1,\, q_2 \in S^{n-1}$ mapping T satisfies the following property*

$$< T(q_2) - T(q_1),\, q_2 - q_1 > \geq 0. \tag{5.5}$$

Proof. By the definition of T, $q_2 \in N_C(T(q_2))$, therefore it satisfies

$$< q_2,\ T(q_2) - T(q_1) > \geq 0, \tag{5.6}$$

which comes from Definition 7, setting $x = T(q_2)$ and $y = T(q_1)$. In the same way it is

$$< q_1,\ T(q_2) - T(q_1) > \leq 0, \tag{5.7}$$

by subtracting (5.7) to (5.6) the thesis follows.

The *shape operator* is a linear operator that is associated to the derivative of the normal vector to a differential manifold with respect to the position on the surface and is defined as follows.

Definition 8 *Consider a $n-1$ dimensional differentiable manifold $\mathcal{M}$, embedded in $\mathbb{R}^n$, represented by the image of the differentiable function $M : \Omega \subset \mathbb{R}^{n-1} \to \mathbb{R}^n$, i.e.*

$$\mathcal{M} = M(\Omega)\ ,$$

let $\hat{n}(x) \in S^{n-1}$ be the normal unit vector to $\mathcal{M}$ at $x \in \mathcal{M}$, then the shape operator *associated to $\mathcal{M}$ is the mapping $\mathcal{S} : T_x\mathcal{M} \to T_x\mathcal{M}$, such that*

$$\mathcal{S}_x(v) = -\frac{d\hat{n}(x+\lambda v)}{d\lambda}\ , \tag{5.8}$$

where $v \in T_x\mathcal{M}$.

The shape operator is related to the curvature of trajectories defined on a manifold as follows: let $\gamma(t) \in \mathcal{M}$ be a smooth arc-length parametrized curve such that $\gamma(0) = x$ and $\dot{\gamma} = v$, then

$$< \ddot{\gamma},\ \hat{n}(x) > = < v,\ \mathcal{S}_x v > \ .$$

The shape operator defines a quadratic form $< v, \mathcal{S}_x v >$ that represents the component of the curvature of $\gamma(t)$ normal to the manifold $\mathcal{M}$.

Proposition 9 *Given a convex subset $\mathcal{C} \subset \mathbb{R}^n$, let $x \in \partial\mathcal{C}$ and $q \in S^{n-1}$ be such that $x = T(q)$, where T is the mapping defined in (5.4), and let $\mathcal{S}_x(v)$ be the shape operator defined on x. Then if T is differentiable in q it is*

$$\frac{dT(q)}{dq} = \mathcal{S}_x^{-1}(v)$$

Proof. By (5.4)

$$T(q) = x, \text{ if } q \in N_C(x).$$

Assume $q = q(x)$, if T is differentiable in q then

$$\frac{dT}{dq}\frac{dq}{dx} = I.$$

From the definition of shape operator in (5.8), it follows that

$$\frac{dT}{dq}\mathcal{S}_x = I$$

Proposition 10 *Given a closed, bounded and strictly convex manifold $\mathcal{C} \subset \mathbb{R}^n$, if mapping T defined in (5.4) is differentiable then*

$$\frac{dT(q)}{dq} \geq 0$$

that is $\frac{dT(q)}{dq}$ is positive semi-definite.

Proof. Given $q \in S^{n-1}$, and $V \in T_q S^{n-1}$ and let q_i be a succession of values in S^{n-1} such that

$$\begin{array}{rcl} \lim_{i\to\infty} q_i & = & q \\ \lim_{i\to\infty} V_i & = & V, \end{array}$$

where $V_i = \frac{q_i - q}{\|q_i - q\|}$, then it is

$$< V, \frac{dT}{dq}V >= \lim_{i\to\infty} < V_i, \frac{T(q_i) - T(q)}{\|q_i - q\|} >=< \frac{q_i - q}{\|q_i - q\|}, \frac{T(q_i) - T(q)}{\|q_i - q\|} >$$

Since (5.5) holds, the thesis follows.

A consequence of this result is the following property (see (3.2), chapter 7 of [70]).

Proposition 11 *If $\mathcal{C}$ is a closed, bounded and strictly convex subset of $\mathbb{R}^n$, then on its boundary the shape operator is positive semi-definite, i.e.*

$$< V, \mathcal{S}_x(x)V > \geq 0 , \forall x \in \partial\mathcal{C} , \forall V \in T_x^m \partial\mathcal{C}. \tag{5.9}$$

5.1.3 Problem formulation

Definition 9 *Given an initial state $x_0 \in \mathbb{R}^n$, the* final costate mapping $\gamma_f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$*, is given by*

$$\gamma_f(q_1, t) = x(t),$$

where $x(t)$ is the solution at time t of the augmented system (5.1) + (5.2) with initial state $x(0) = x_0$ and final costate condition $q(t) = q_1$.

The initial costate mapping $\gamma_i : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$*, is given by*

$$\gamma_i(q_0, t) = x(t),$$

where $x(t)$ is the solution at time t of the augmented system (5.1) + (5.2) with initial condition $x(0) = x_0$ and initial costate condition $q(0) = q_0$.

The only difference between functions γ_i and γ_f lies on the fact that the boundary condition on the costate is given on the initial and, respectively, the final state. The relations between the two functions is given by following proposition.

Proposition 12 *Let $\phi(t) \in \mathbb{R}^{n\times n}$ be the solution of system*

$$\begin{cases} \dot{\phi} = -\frac{df(x,u^*)^T}{dx}\big|_{x=x_{u^*}(t)}\phi, \\ \phi(0) = I\ , \end{cases}$$

where u^* is given by (5.2), then it is

$$\gamma_f(\phi(q_0), T) = \gamma_i(q_0, T).$$

The general problem considered in this chapter is the following

Problem 8 (Shooting problem) *Given a nonlinear system of the form (5.1) and a final state x_1, find a initial costate q_0 and a time T such that*

$$x_1 = \gamma_i(q_0, T)$$

5.2 Main result

The basic geometric idea of the proposed algorithm is depicted in Fig. 5.1. Given a final costate q_1 and a final time t, consider the final state $x_1 = \gamma_f(q_1,t)$. Remark that q_1 represents the normal vector at x_1 to the set of states reachable in time t. The error vector is defined as $e = x_f - x_1$ and it is decomposed as follows

$$e_N =< e,\ \hat{q}_1 > ;\ e_T = e - < e,\ \hat{q}_1 > \hat{q}_1,$$

where e_N is the error component parallel to q_1 and e_T is parallel to the tangent space to the boundary of the reachable set $\mathcal{A}_{x_0}(t)$ at x.

If q_1 is varied by the small quantity δq_1 it follows that the final state varies by

$$\delta x_1 = \mathcal{S}^{-1}\delta q_1,$$

which satisfies $< \delta x_1,\ \delta q_1 > \geq 0$, because of the convexity of the reachable set.

Therefore if δq_1 is proportional to the tangential error, i.e. $\delta q_1 = K e_T$, the tangential error is reduced. On the other hand the normal error can be reduced by increasing the final time t by a term proportional to the normal error itself, exploiting the fact that the state derivative $f(x,u^*)$ is always directed outwards with respect to the reachable set, as a consequence of the third equation of (5.2).

A key technical fact is that the error vector will always be inside a cone with axis q and semi-aperture $\arcsin\sqrt{1-\beta^2}$, where β is a tuning parameter close to 1.

The following theorem is the main contribution of this paper and present and algorithm for solving Problem 8.

Theorem 3 *Let $\hat{t}$ be a time greater than the optimal time t^*, and let K, β, α, M, χ be positive real constants that satisfy the properties*

$$0 < \chi < \min_{x\in\mathcal{A}_0(\hat{t})} \{< \hat{q},\ \hat{f}(x,u^*) >\}\ , \tag{5.10}$$

$$\beta - \chi^{-1}\sqrt{1-\beta^2} > M > 0 \tag{5.11}$$

$$\frac{1-\beta^2}{\beta} < \alpha < \frac{1-\beta^2}{\beta(1-\beta M)}\ . \tag{5.12}$$

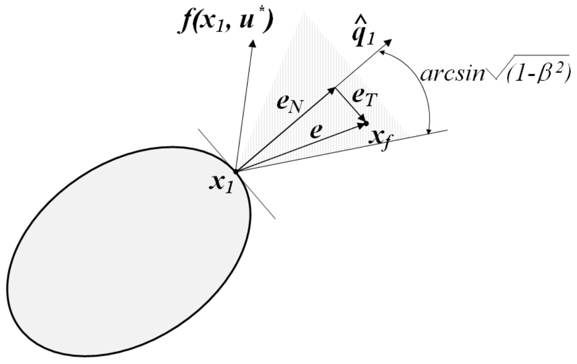


Figure 5.1: A schematic representation of the control technique.

Consider system (5.2), with the associated final state mapping $x_1 = \gamma_f(q_1, t)$. *and define the error function as* $e(q,t) = x_f - S(q_1, t)$. *Consider the following differential system*

$$\begin{array}{l} \frac{dt}{d\lambda} = K\alpha \frac{<e,\, \hat{q}_1>}{<f,\, \hat{q}_1>} \|e\| \\ \frac{d\hat{q}_1}{d\lambda} = K(e - <e,\, \hat{q}_1> \hat{q}_1)\ , \end{array} \tag{5.13}$$

then if the reachable sets $\mathcal{A}_{x_0}(t)$ *are convex for all* $t \geq 0$ *and if*

$$<\hat{e}(0),\, \hat{q}_1(0)>> \beta\ , \tag{5.14}$$

then

$$<\hat{e}(t),\, \hat{q}_1(t)>> \beta\ , \forall t \geq 0\ . \tag{5.15}$$

moreover it is

$$\lim_{\lambda \to \infty} e(\lambda) = 0\ . \tag{5.16}$$

Proof. Equation (5.15) is equivalent to

$$<e,\, \hat{q}_1> -\beta \|e\| \geq 0\ , \ \forall t \geq 0\ ,$$

deriving the above expression, it follows that

$$\frac{d <e,\, \hat{q}_1> -\beta\|e\|}{d\lambda} =< \dot{e},\, \hat{q}_1> + <e,\, \dot{\hat{q}}_1> -\beta <\dot{e},\, \hat{e}>\ , \tag{5.17}$$

where the dot denotes the derivatives with respect to λ.

Rewrite the first of the three terms in (5.17) taking into account (5.13)

$$< \dot{e},\, \hat{q}_1 >= -K\alpha < e,\, \hat{q}_1 > \|e\| = -K\alpha < \hat{e},\, \hat{q}_1 > \|e\|^2 \,.$$

The second term in (5.17) is given by

$$< e,\, \dot{\hat{q}}_1 >=< e,\, K(e- < e,\, \hat{q}_1 > \hat{q}_1) >= K\|e\|^2(1- < \hat{e},\, \hat{q}_1 >^2)\,,$$

and the third one by

$$\begin{aligned} -\beta < \dot{e},\, \hat{e} >= \beta\Big(&\frac{K\alpha < f,\, \hat{e} >< e,\, q_1 >}{< f,\, \hat{q}_1 >}\|e\| + \\ &+K < \hat{e}- < \hat{e},\, \hat{q}_1 > \hat{q}_1,\, S_{x_1}^{-1}(e- < e,\, \hat{q}_1 > \hat{q}_1) > \Big) \end{aligned}$$

where S^{-1} is the inverse shape operator computed on x_1 which lies on the boundary of the reachable set $\mathcal{A}_{x_1}(t_1)$. Being the border of the reachable set convex, matrix S is negative definite by Proposition 11. Moreover

$$< f,\, e >=< \hat{q}_1,\, f >< \hat{q}_1,\, e > + < f- < \hat{q}_1,\, f > \hat{q}_1,\, e >$$

and equation (5.17) can be bounded as follows

$$\frac{d < e,\, \hat{q}_1 > -\beta\|e\|}{d\lambda} \geq K\|e\|^2(1- < \hat{e},\, \hat{q} >^2 - \alpha < \hat{e},\, \hat{q}_1 > + \beta\alpha < f,\, \hat{e} >< \hat{e},\, q_1 >)$$

evaluating this expression for $< \hat{e},\, \hat{q}_1 >= \beta$ it follows that

$$\frac{d < e,\, \hat{q}_1 > -\beta\|e\|}{d\lambda} \geq K\|e\|^2(1-\beta^2-\alpha\beta+\beta\alpha(\beta-\chi^{-1}\sqrt{1-\beta^2}))\,.$$

Applying (5.11) and (5.12) it follows that

$$\frac{d < e,\, \hat{q}_1 > -\beta\|e\|}{d\lambda} \geq K\|e\|^2(1-\beta^2-\alpha\beta+\beta\alpha M) \geq 0 \;\text{ if } \|e\| \geq 0\,,$$

therefore (5.15) must hold.

Consider now the equation for the normal error $< e,\, \hat{q}_1 >$, it is $\frac{d}{d\lambda} < e,\, \hat{q}_1 >=< \dot{e},\, \hat{q}_1 > + < e,\, \dot{\hat{q}}_1 >$, which corresponds to the first two terms of (5.17) and it follows that

$$\begin{aligned} &\frac{d}{d\lambda} < e,\, \hat{q}_1 >= -K\alpha < e,\, \hat{q}_1 > \|e\| + K\|e\|^2[1- < e,\, \hat{q}_1 >^2] \leq \\ &\leq -\|e\|^2 < \hat{e},\, \hat{q}_1 > K[(1-\beta^2)-\alpha\beta)] \leq -\|e\|^2 < e,\, \hat{q}_1 > Kc \leq - < e,\, \hat{q}_1 >^3 Kc\,. \end{aligned}$$

where property (5.12) had been applied and $c > 0$ is a positive constant. Therefore by the comparison lemma

$$\lim_{\lambda\to\infty} < e,\, q_1 >= 0,$$

and, being, by (5.15), $\|e\| \leq \| < e,\, \hat{q}_1 > \| \sqrt{1+\beta^2}$, (5.16) follows.

Remark 6 *Theorem 3 represents a procedure that can be used for the computation of the time-optimal control for systems whose reachable sets are convex. The convexity is crucial because allows the inverse of the shape operator S_{x_1} to be semidefinite positive. The property of convexity is enjoyed by various kind of systems, for example linear time-varying systems, some bilinear systems [71], nonlinear systems with small inputs [72].*

Remark 7 *Conditions (5.11), (5.12) can always be satisfied for some value of β, α and K provided that the value of χ in (5.10) is found. Doing this may be difficult, because it requires an a priori estimate of a reachable set containing the final state, x_f. In practice, if the algorithm does not converge, the term χ can be reduced (making β closer to* 1*) until convergence is achieved.*

5.3 Numerical implementation

The approach devised in §5.2 has been numerically implemented as reported in Algorithms 2 and 3.

The input parameters are the following: $K \in \mathbb{R}^+$ is a gain constant, x_f is the final state, ε_e represent the error tolerance and α, β are the parameters appearing in Theorem 3. Moreover $\phi(q,t) \in \mathbb{R}^{n\times n}$ is the solution of

$$\begin{cases} \dot{\phi}(t) & = -\frac{df(x,u^*)^T}{dx}\big|_{x=x_{u^*}(t)}\phi(t) \\ \phi(0) & = I \end{cases}$$

where f is the system function, $x \in \mathbb{R}^n$ is the state-vector and u^* is found with (5.2). The algorithm is a direct application of Theorem 3. In particular Equation (5.13) is solved using the initial state $t(0) = 0$ and $q_0(0) = q_1(0) = x_1$, such that

$< \hat{e}(0),\ \hat{q_1}(0) >= 1$, and (5.14) holds. The algorithm ends when the norm of the error between the final state x_1 and the current state $\gamma_i(q_0, t^*)$ is less than the tolerance ε_e.

Algorithm 2: Compute the minimum-time feedforward control

```
input  : $x_f$, $\varepsilon_e$, $\alpha$, $\beta$, $K$
output: $q_1$ and $t^*$
begin
    $t_0 = 0$;
    $q_0 = \frac{x_1 - x_0}{\|x_1 - x_0\|}$;
    repeat
        $(\frac{dt_1}{d\lambda}, \frac{d\hat{q}_1}{d\lambda}) \longleftarrow G(q_0, t)$;
        $q_0 \longleftarrow q_0 + \phi(q_0, t)^{-1} \frac{d\hat{q}_1}{d\lambda}$;
        $t \longleftarrow t + \frac{dt}{d\lambda}$;
    until $e \geq \varepsilon_e$ ;
    $q_1 \longleftarrow \phi(q_0, t) q_0$;
    $t^* = t$;
    $u^*(t) \longleftarrow \mathrm{sgn}(q_1 B)$;
end
```

Remark 8 *The Euler algorithm is used here only for simplicity, but any kind of differential equation solver can be adopted, i.e. Runge-Kutta method.*

Algorithm 3: $G(q,t)$: Compute the derivative of time t and the initial costate q

```
input  : $t_1$, $q_0$ and $e$
output: $\frac{dt}{d\lambda}$ and $\frac{d\hat{q}_1}{d\lambda}$
begin
    $x_1 \longleftarrow \gamma_1(q_0, t)$;
    $e \longleftarrow \|x_f - x_1\|$;
    Compute $\phi(q_0, t)$;
    $\frac{dt}{d\lambda} \longleftarrow K\alpha \frac{<e,\ \hat{q}_1>}{<f,\ \hat{q}_1>} \|e\|$;
    $\frac{dq}{d\lambda} \longleftarrow K(e - < e,\ \hat{q}_1 > \hat{q}_1)$;
end
```

5.4 Simulation and experimental results

Three different types of systems have been used to evaluate the correctness and the convergence of the proposed approach. Simulations have been performed in Matlab, while experimental results have been obtained using the WinCon real-time extension.

First of all the algorithm has been tested with a dummy problem, a double order integrator system. Then simulations and experimental results have been carried on the linearized model of a flexible joint device, and finally the algorithm has been tested on a non-linear system, without any modification.

5.4.1 Double order integrator

It is given by $\dot{x} = Ax + Bu$, where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix} . \tag{5.18}$$

The reachable set $\mathcal{A}_0(1) = \gamma_i(S^1, 1)$ is shown in Fig. 5.2: the set is convex, with two non-differentiable points respectively in $x = [-2.5\ , -5]^T$ and $x = [2.5\ , 5]^T$.

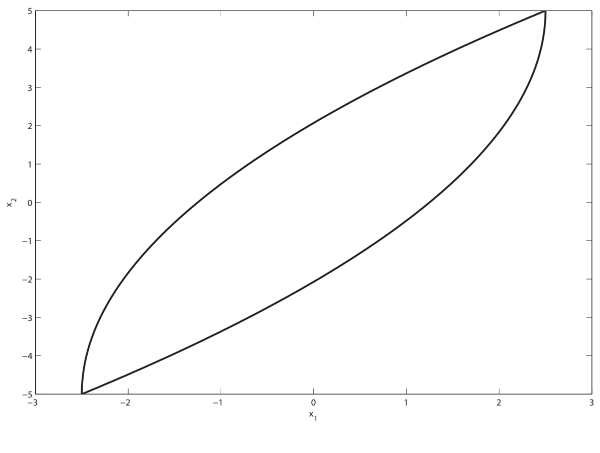
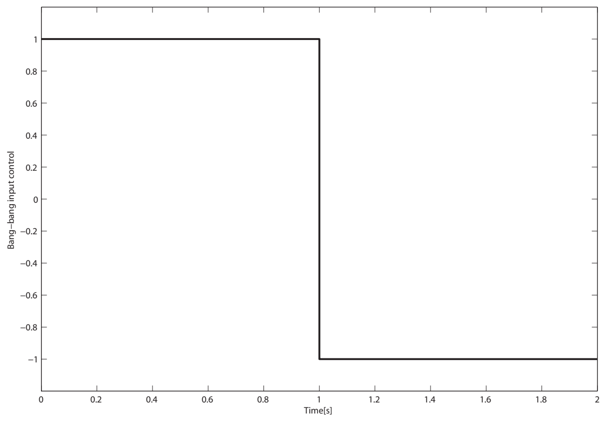
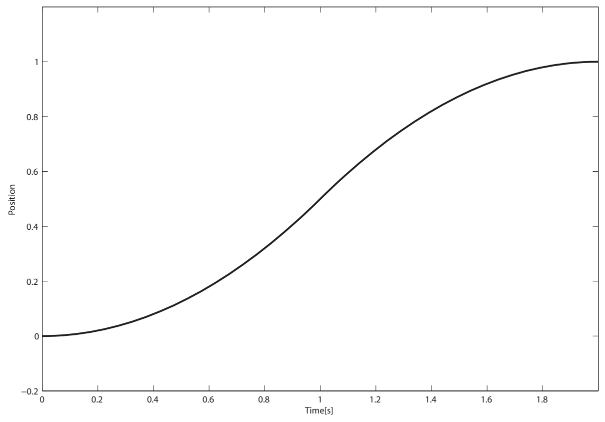


Figure 5.2: The reachable set at $t_f^* = 1$ s for the double order integrator

The control law allowing to reach the final state $x_f = [1,\ 0]^T$ has been computed with the input constraint $\|u(t)\|_\infty \leq 1$. Simulation results are shown in Fig. 5.3. As



(a) Bang-bang input control



(b) System output

Figure 5.3: Simulation of a double order integrator subject to input constraint for a transition to final state $\mathbf{x} = [1\ 0]^T$.

expected the input control is a standard bang-bang signal; in the first half of the optimal transition the system accelerates at the maximum rate and then it decelerates, always at the maximum admissible rate.

5.4.2 Flexible joint system

To further validate the effectiveness of the proposed algorithm, a mechanical simulator of a flexible joint has been used. It's mathematical model is described in Chapter 4 and published in [73]. The system state space model is $\dot{x} = Ax + Bu$, where

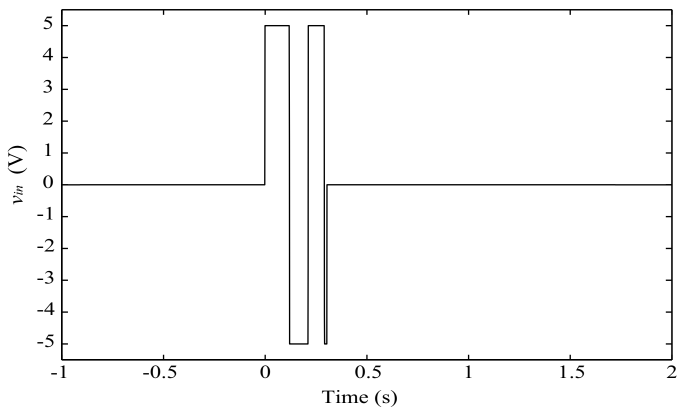
$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 379.9 & -56.65 & 2.956 \\ 0 & -512.9 & 56.65 & -3.99 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ 93.74 \\ -93.74 \end{bmatrix}.$$

Time-optimal feedforward control $u^*(t)$ has been found by means of the algorithm described in §5.3, to get a rest-to-rest transition from $x_0 = [0,0,0,0]^T$ to $x_f = [\pi/4,0,0,0]^T$ with the input constraint $\|u(t)\|_\infty \leq 5$ Volts.

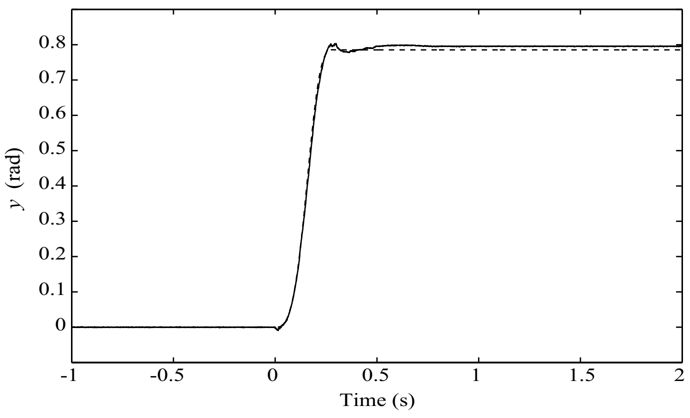
The optimal transition is performed in $t^* = 0.31$ s and the related control signal is reported in Fig. 5.4. Fig. 5.4(b) shows the comparison between the simulated plant output and the real one. Since the proposed approach has been tested on the linearized model of the flexible joint, the behavior difference is mainly due to friction and other neglected system non-linearities. As validation of the correctness of the devised solution, this result has been successfully compared with the one obtained by the algorithm described in [73].

5.4.3 Mass on a cart

Consider now the mechanical system made of a mass M on a linear cart subject to an external force u. The state space vector is given by $x = [x_1,\, x_2]$, where x_1 is the cart position and x_2 is its linear velocity. The control problem is to devise a bounded input u in order to move the cart in minimum-time from an initial state $x_0 = [x_{10},\, x_{20}]$ to a desired final state $x_f = [x_{11},\, x_{21}]$.



(a) Bang-bang input control



(b) Expected system output y (dashed line) and measured plant output (solid line)

Figure 5.4: Experimental results of the control technique applied to the linearized model of a rotary flexible joint subject to input constraint for a rest-to-rest transition to final state $x_f = [\pi/4\ ,0\ ,0\ ,0]^T$.

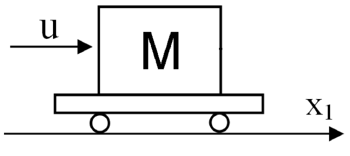
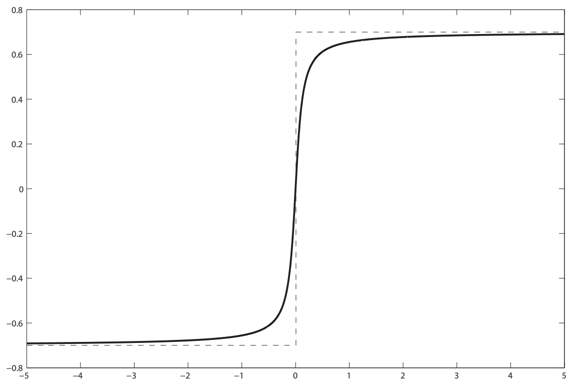
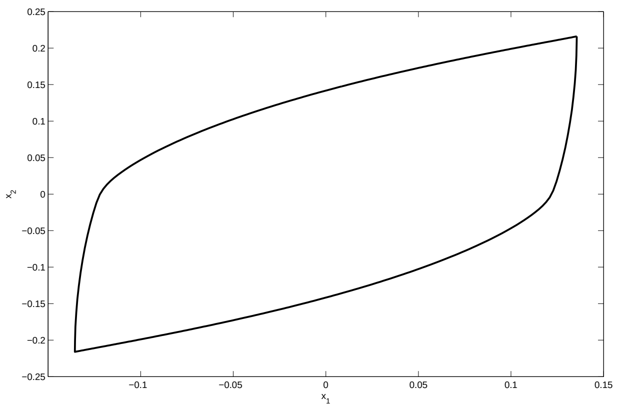


Figure 5.5: A schematic representation of the mass on a cart system.



(a) Approximation of the static friction ($K_s = 0.7$, $\alpha = 10$)



(b) The reachable set at $t_f^* = 1$ s

Figure 5.6: Static friction and reachable set for a mass on a cart.

In order to remark the ability of the proposed algorithm to deal also with non-linear systems, static and dynamic frictions are taken into account in the modeling phase. Since the augmented system (5.2) has to be differentiable, the static friction τ_s defined as

$$\tau_s = \begin{cases} K_s \text{ if } x_2 \geq 0 \\ -K_s \text{ otherwise} \end{cases}$$

has been approximated, see Fig. 5.6(a), with the function

$$\tau_s = K_s \arctan(\eta x_2) \frac{2}{\pi},$$

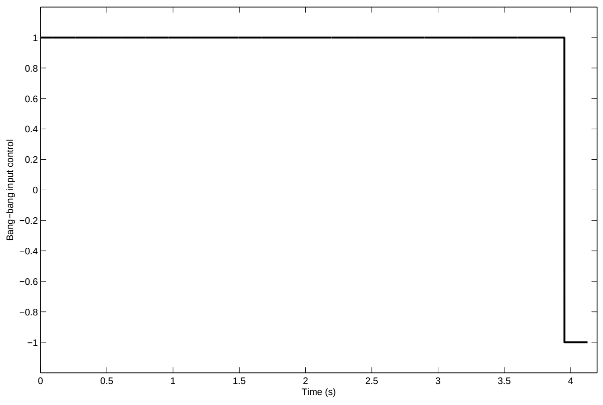
which depends on the parameter η: the higher is its value, the more accurate is the similarity with the friction heaviside function.

The system model is then equal to:

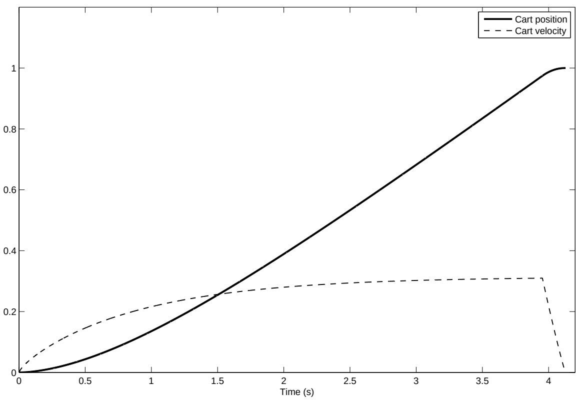
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{K_v}{M} x_2 - \arctan(\alpha x_2) \frac{2}{\pi} \frac{K_s}{M} + \frac{u}{M} \end{cases}$$

where K_v is the dynamic friction constant, K_s is the static friction and M is the mass.

Fig. 5.6(b) shows the reachable set $\mathcal{A}_0(1) = \gamma_i(S^1, 1)$ which is, as expected, convex. The Time-optimal feedforward control $u^*(t)$ has been obtained with the algorithm described in §5.3, to get a rest-to-rest transition from $x_0 = [0,\ 0]$ to $x_f = [1,\ 0]$, satisfying the input constraint given by $\|u(t)\|_\infty \leq 1$. Simulation parameters are $K_s = 0.7$, $K_v = 1$, $\alpha = 100$ and $M = 1$. The optimal transition is performed in $t^* = 4.1279$ s and the related control and output signals are reported in Fig. (5.7).



(a) Bang-bang input control



(b) System output

Figure 5.7: Results of the control technique applied to the non-linear model of a mass on a cart for a rest-to-rest transition to final state $x_f = [1\ ,0]^T$.

Conclusion and Future Work

The goal of this thesis was to develop new efficient strategies for the optimal planning of mechatronic systems and, more in details, to deeply analyze the problem of optimal path generation and online path tracking. In this section, we discuss to what extent these research goals have been accomplished.

In Chapter 1, an effective solution to the optimal path planning problem for mobile robots has been described. To this purpose, it has been used a power planning primitive, called η^3-splines, which allows to generate paths with a third order geometric continuity. The shape of the η^3-splines can be modeled to fulfill a given optimality criterion by acting on a vector η of freely tunable parameters. The selection of η represents a key point for the generation of optimal paths: a wrong choice can easily introduce undesired vehicle solicitations. In particular, it has been shown how, by acting on η, it is possible to generate curves with minimum curvature derivative with the purpose of minimizing the vehicle lateral jerk. In order to avoid the execution of huge online optimizations, an heuristic method has been proposed for the optimal selection of η. When interpolating conditions are compatible with circular arcs and clothoids, the devised expressions generate curves which at the best emulate such primitives. In the case of generic interpolating conditions, the maximum curvature derivative is very close to the actual achievable minimum. To show the strong impact of the lateral solicitations on the motion performances, a test case based on a unicycle-like mobile robot has been simulated also considering the traction forces generated at the interface between the wheels and the ground. Results have shown that later skidding phenomena can be drastically reduced when the proposed planning is used.

In Chapters 2 and 3, the problem of the trajectory scaling for constrained path tracking of robotic manipulators has been studied. In particular, it has pointed out, and proved by simulations, that it is essential to design control schemes able to online shape any given desired input trajectory in order to fulfill robot kinematic and dynamic constraints, thus allowing a good path tracking. This requirement is not only important when trajectories are planned by an operator but it is still fundamental when offline optimization algorithms are used in the planning phase to design minimum-time trajectories.

More in details, in Chapter 2 the problem originally proposed by Dahl and Nielsen has been improved by also considering explicit constraints on the manipulator joint velocities and torques. To this purpose, a newly devised discrete-time filter has been used to online scale any nominal trajectory, which could be infeasible. The proposed control scheme requires minor adaptations of standard manipulators controllers, since the desired result is obtained by simply inserting the new filter between a reference signal and the controller itself. Simulation results demonstrate that path tracking performances neatly improves and, simultaneously, also the velocity reference signal is followed at best, compatibly with the manipulator constraints.

The same strategy, in Chapter 3, has been improved to account for manipulator high-order dynamic constraints, namely torque and torque derivatives. This analysis, using the same filter structure devised in Chapter 2, has required to use an efficient algorithm for the evaluation of the high order manipulator dynamics and, for the controller parametrization, an algorithm for the efficient online evaluation of the robot mass matrix derivative has been devised.

Simulation results have proved that, in both cases, a good path tracking is achieved even when reference trajectories are not physically feasible. The algorithms have also been tested in presence of model uncertainties and also by using two different manipulator standard torque controllers.

Finally, in the last part of the thesis, corresponding to Chapters 4 and 5, the design of algorithms for the constrained minimum-time control of both linear and nonlinear systems has been studied. Differently from the previous methods, where the path-velocity paradigm has been used, in these chapters the aim of the proposed algorithms is to generate the optimal trajectory controls as a whole, given the system model.

Initially, the problem has been solved by using a discretization method which converts the minimum-time problem for linear systems into a set of feasibility tests solvable by standard linear programming methods. The described approach has been successfully applied to the control of a flexible joint device. Even if its model is clearly nonlinear, very good results have been achieved by using the proposed feed-forward method by linearizing the system around its equilibrium point. A comparison with an inversion-based feedforward control has confirmed the effectiveness of the new approach. Moreover, it applies to any stable linear plant, so that it is foreseeable an extension of the technique to the more challenging cases of systems with unstable zero-dynamics like, for example, flexible links [74]. Nevertheless, general nonlinear systems cannot be managed and, moreover, the sampling time assumes a critical role in the fulfillments of the system output constraints. For these reasons, a new way to obtain the optimal solution has been investigated.

In Chapter 5 an algorithm able to devise the minimum-time control for nonlinear systems has been described. The approach proposes a geometric invariant in time-optimal control for an input constrained transition based on the convexity of the system reachable sets. Therefore, it is useful for those systems whose reachable sets are convex, such as linear systems, weakly nonlinear systems and a class of bilinear systems. The described method is based on the solution of two nested differential equations: the inner one computing the initial state mapping and the outer one computing the movement in the state parametrization in order to reduce both the normal and the tangential error vector. A proof of convergence has been devised and experimental results have been presented for three different plants. Among them also the system model of the flexible joint has been used as a benchmark. Good results have been obtained from the proposed method, whose main novelty is represented by the complete differential approach.

Recommendations for future works

The answers to the research problems treated by this thesis have led to more questions, and hence several directions for future research. In particular, for what concerns the path generation for mobile or industrial robots, it could be interesting improve the mathematical definition of the η^3-splines by adding a third dimension, thus allowing to plan 3D paths while maintaining all the good features of the current primitives. Moreover, the proposed approach has been tested uniquely in simulation; hence, it could be interesting having a test bed with a real robot in order to directly measure the path tracking improvements due to the reduction of the lateral vehicle skidding. In the same way it could be also possible to measure the tyre shear stresses and therefore experimentally validate the model devised in [75].

For what concern the online strategies for the path tracking problem, an interesting enhancement of the current methods is certainly to continue the research on the three stage integrator filter in order to simultaneously deal with velocities, torque and torque derivative constraints. Moreover, we have so far defined the robot trajectory in the joint space. Even if from a theoretical point of view it is always possible to convert a task space path into a joint space one, from a practical point on view this poses some difficulties inside the considered framework: it requires to evaluate the high order derivative of the manipulator jacobian matrix, which is still an open research issue. Another improvement, as for the previous case, is to test the proposed control scheme with a newly bought six degree of freedom industrial robot.

Finally, for what concern the time optimal control algorithm, an interesting improvement is to speed up the convergence by solving some numerical problems highlighted during the conducted simulations.

Bibliography

[1] L.E. Dubins. On curves of minimal length with a constraint on average curvature and withprescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497–517, 1957.

[2] J.A. Reeds and R.A. Shepp. Optimal paths for a car that goes both forward and backward. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.

[3] J.-D. Boissonnat, A. Cérézo, and J. Leblond. Shortest paths of bounded curvature in the plane. In *Proc. of the 1992 IEEE Int. Conf. on Robotics and Automation*, pages 2315–2320, Nice, France, May 1992.

[4] P. Souères and J.-P. Laumond. Shortest paths synthesis for a car-like robot. *IEEE Transaction on Automatic Control*, 41(5):672–688, May 1996.

[5] T. Fraichard and A. Scheuer. From Reeds and Shepp's to continuous-curvature paths. *IEEE Trans. on Robotics*, 20(6):1025–1035, Dec. 2004.

[6] Y. Kanayama and B.I. Hartman. Smooth local path planning for autonomous vehicles. In *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA89*, volume 3, pages 1265–1270, Scottsdale, AZ (US), May 1989.

[7] W. Nelson. Continuous-curvature paths for autonomous vehicles. In *Proc. of the IEEE Conf. on Robotics and Automation*, volume 3, pages 1260–1264, Scottsdale, AZ, May 1989.

[8] H. Delingette, M. Hébert, and K. Ikeuchi. Trajectory generation with curvature constraint based on energy minimization. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 206–211, Osaka, Japan, November 1991.

[9] J. Reuter. Mobile robots trajectories with continuously differentiable curvature: anoptimal control approach. In *Proc. of the 1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 1, pages 38–43, Victoria, B.C., Canada, Oct. 1998.

[10] C. Guarino Lo Bianco, A. Piazzi, and M. Romano. Smooth motion generation for unicycle mobile robots via dynamic path inversion. *IEEE Trans. on Robotics*, 20(5):884–891, Oct. 2004.

[11] A. Piazzi, M. Romano, and C. Guarino Lo Bianco. G^3-splines for the path planning of wheeled mobile robots. In *Proc. of the 2003 European Control Conference, ECC 2003*, Cambridge, UK, September 2003.

[12] A. Piazzi, C. Guarino Lo Bianco, and M. Romano. η^3-splines for the smooth path generation of wheeled mobile robots. *IEEE Trans. on Robotics*, 23, NO. 5:1089–1095, 2007.

[13] C. Guarino Lo Bianco. Optimal velocity planning for autonomous vehicles under kinematic constraints. In *8th Int. IFAC Symp. on Robot Control, SYROCO 2006*, Bologna, Italy, Sept. 2006.

[14] C. Guarino Lo Bianco and A. Piazzi. A hybrid genetic/interval algorithm for semi-infinite optimization. In *Proc. of the 35th Conf. on Decision and Control*, pages 2136–2138, Kobe, Japan, December 1996.

[15] C. Guarino Lo Bianco and A. Piazzi. A hybrid algorithm for infinitely constrained optimization. *Int. J. of Systems Science*, 32(1):91–102, January 2001.

[16] H. Dugoff, P. S. Fancher, and L. Segel. An analysis of tyre traction properties and their influence on vehicle dynamicperformance. *SAE Paper NO. 700377*, pages 1219–1243, 1970.

[17] C.-S. Lin, P.-R. Chang, and J.Y.S. Luh. Formulation and optimization of cubic polynomial joint trajectories forindustrial robots. *IEEE Trans. Automatic Control*, AC-28(12):1066–1074, 1983.

[18] A. De Luca, L. Lanari, and G. Oriolo. A sensitivity approach to optimal spline robot trajectories. *Automatica*, 27(3):535–539, 1991.

[19] C. Guarino Lo Bianco and A. Piazzi. Minimum-time trajectory planning of mechanical manipulators under dynamicconstraints. *Int. J. of Control*, 75(13):967–980, 2002.

[20] K. Kant and S.W. Zucker. Toward efficient trajectory planning: the path-velocity decomposition. *Int. J. of Robotics Research*, 5(3):72–89, 1986.

[21] K. G. Shin and N. D. McKay. Minimum-time control of robotic manipulators with geometric path constriants. *IEEE Transactions on Automatic Control*, AC-30(6):531–541, 1985.

[22] J. E. Bobrow, S. Dubowsky, and J .S. Gibson. Time-optimal control of robotic manipulators along specified paths. *International Journal of Robotics Research*, 4(3):554–561, 1985.

[23] J. M. Hollerbach. Dynamic scaling of manipulator trajectories. *Journal of Dynamic Systems, Measurement, and Control*, 106:102–106, 1984.

[24] S.B. Moon and S. Ahmad. Time scaling of cooperative multirobot trajectories. *IEEE Transaction System Man and Cybernetics*, 21:900–908, 1991.

[25] A. De Luca and R. Farina. Dynamic scaling of trajectories for robots with elastic joints. In *Proc. of the IEEE-Int. Conf. on Robotics and Automation*, May, 2002.

[26] S.I. Choi and B.K. Kim. Obstacle avoidance control for redundant manipulators using collidability measure. *Robotica, Cambridge University Press*, 18:143–151, 2000.

[27] K. Ohishi, H. Nozawa, and T. Miyazaki. Robust motion control with the consideration algorithm of joint torque saturation for a redundant manipulator. *Advanced Robotics*, pages 345–359, 2002.

[28] B. Daachi and A. Benallegue. A neural network adaptive controller for end-effector tracking of redundant robot manipulators. *Journal of Intelligent Robot Systems*, 46:245–262, 2006.

[29] D. Omrcen, L. Zlajpah, and Nemec B. Compensation of velocity and/or acceleration joint saturation applied to redundant manipulator. *Robotics and Autonomous Systems*, 55:337–344, 2007.

[30] T. Sugie, K. Fujimoto, and Y. Kito. Obstacle avoidance of manipulators with rate constraint. *IEEE Trans. Robotics and Automation*, 19:162–167, 2003.

[31] G. Antonelli, S. Chiaverini, and G. Fusco. A new online algorithm for inverse kinematics of robot manipulators ensuring path tracking capabilities. *IEEE Trans. Robotics and Automation*, 19:162–167, 2003.

[32] O. Dahl and L. Nielsen. Torque-limited path following by on-line trajectory time scaling. *IEEE Trans. on Robotics and Automation*, 6(5):658–669, 1990.

[33] O. Dahl. Path-constrained robot control with limited torques-experimental evaluation. *IEEE Trans. on Robotics and Automation*, 10(5):658–669, 1994.

[34] H. Arai, K. Tanie, and S. Tachi. Path tracking control of a manipulator considering torque saturation. *IEEE Transaction on Industrial Electronics*, 41:25–31, February 1994.

[35] R. Zanasi, C. Guarino Lo Bianco, and A. Tonielli. Nonlinear filters for the generation of smooth trajectories. *Automatica*, 36(3):439–448, March 2000.

[36] C. Guarino Lo Bianco and R. Zanasi. Smooth profile generation for a tile printing machine. *IEEE Trans. on Ind. Electronics*, 50(3):471–477, 2003.

[37] O. Gerelli and C. Guarino Lo Bianco. Nonlinear variable structure filter for the online trajectory scaling for robotic manipulators. *Submitted to IEEE Transaction on Industrial Electronics*, 2008.

[38] L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag, Berlin, Germany, 2000.

[39] V.I. Utkin. Variable structure systems with sliding modes. *IEEE Transactions on Automatic Control*, 22:212–222, 1977.

[40] O. Gerelli and C. Guarino Lo Bianco. Real-time path-tracking control of robotic manipulators with bounded torques and torque-derivatives. In *Proceedings of the 2008 IEEE International Conference on Intelligent Robots and Systems*, Nice, France, September 2008.

[41] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul. On-line computational scheme for mechanical manipulators. *ASME J. of Dynamic Systems, Measurement, and Control*, 102:69–76, 1980.

[42] C. Guarino Lo Bianco and E. Fantini. A recursive newton-euler approach for the evaluation of generalized forcesderivatives. In *IEEE 12th Int. Conf. on Methods and Models in Automation and Robotics*, pages 739–744, Miedzyzdroje, Poland, August 2006.

[43] J. Kieffer, A. Cahill, and M. James. Robust and accurate time-optimal path-tracking control for robot manipulators. *IEEE Trans. on Robotics and Automation*, 13(6):880–890, 1997.

[44] A.R. Teel. Global stabilization and restricted tracking for multiple integrators with bounded controls. *Systems and Control Letters*, 18:165–171, 1992.

[45] R. Zanasi and R. Morselli. Discrete minimum time tracking problem for a chain of three integrators with bounded input. *Automatica*, 39:1643–1649, 2003.

[46] K. S. Ananthanarayanan. Third-order theory and bangbang control of voice coil actuators. *IEEE Trans. Magnetics*, 1982.

[47] R. Sinha B. Wie and Q. Liu. Robust time-optimal control of uncertain structural dynamic systems. *AIAA J. Guid. Control Dyn.*, 1993.

[48] Bicchi A and Tonietti G. Fast and "soft-arm" tactics. *IEEE Robotics & Automation Magazine*, 11(2):22–33, 2004.

[49] Palli G, Melchiorri C, and De Luca A. On the feedback linearization of robots with variable joint stiffness. In *2008 IEEE International Conference on Robotics and Automation*, pages 1753–1759, 2008.

[50] Albu-Schäffer A, Ott C, and Hirzinger G. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *The International Journal of Robotics Research*, 26(1):23–39, 2007.

[51] O. Dahl. Path constrained motion optimization for rigid and flexible joint robots. *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 223–229 vol.2, May 1993.

[52] Ozgoli S and Taghirad H D. A survey on the control of flexible joint robots. *Asian Journal of Control*, 8(4):1–15, 2006.

[53] L. A. Zadeh. On optimal control and linear programming. *IRE Transactions on Automatic Control*, pages 45–46, 1962.

[54] Bashein G. A simplex algorithm for on-line computation of time optimal controls. *IEEE Transactions on Automatic Control*, 1971.

[55] Scott M. Time/fuel optimal control of constrained linear discrete systems. *Automatica*, 1986.

[56] M. H. Kim and S. Engell. Speed-up of linear programming for time-optimal control. *Proc. of the 1994 American Control Conference, ACC94*, pages 2667–2670, 1994.

[57] Kim S., Choi D., and Ha I. A comparison principle for state-constrained differential inequalities and its application to time-optimal control. *Automatic Control, IEEE Transactions on*, 2005.

[58] Chiasson J. *Modeling and High-Performance Control of Electric Machines*. Wiley-Interscience, 2005.

[59] GNU Foundation. Gplk, gnu linear programming kit. Available at http://www.gnu.org/software/glpk/.

[60] Giorgietti N. Gplkmex, a matlab mex interface for the gplk library. Available at http://glpkmex.sourceforge.net/.

[61] A. Piazzi and A. Visioli. Optimal noncausal set-point regulation of scalar systems. *Automatica*, 37(1):121–127, January 2001.

[62] C. Guarino Lo Bianco and A. Piazzi. A servo control system design using dynamic inversion. *Control Engineering Practice*, 10(8):847–855, August 2002.

[63] Karmarkar N. A new polynomial-time algorithm for linear programming. *Report, AT&T Bell Laboratories*, 1984.

[64] Shamir R. The efficiency of the simplex method: A survey. *Management Science,*, 33(3):301–334, 1987.

[65] Dan Kalman. The generalized vandermonde matrix. *Mathematics Magazine*, 57(1):15–21, 1984.

[66] G.J. Lastman. A Shooting Method for Solving Two-Point Boundary-Value Problems Arising From Non-Singular Bang-Bang Optimal Control Problems. *Int. J. Control*, 27(4):513–524, 1978.

[67] J.H. Eaton. An Iterative Solution to Time-Optimal Control. *Journal of Mathematical Analysis and Applications*, (5):329–344, 1962.

[68] G.Meyer and E. Polak. A Decomposition Algorithm for Solving a Class of Optimal Control Problems. *Journal of Mathematical Analysis and Applications*, (32):118–140, 1970.

[69] J.P. Aubin and A. Cellina. *A Series of Comprehensive Studies in Mathematics*. Springer-Verlag, 1979.

[70] M. Spivak. *A Comprehensive Introduction to Differential Geometry*. Perish, 1979.

[71] M. V. Topunov. The reachable set of a quasi-commutative bilinear system: Its convexity. *Automation and Remote Control*, 2003.

[72] B.T. Polyak. The convexity principle and its applications. *Bulletin of the Brazilian Mathematical Society*, 34(1):59–75, April 2003.

[73] L. Consolini, O. Gerelli, C. Guarino Lo Bianco, and A. Piazzi. Minimum-time control of flexible joints with input and output constraints. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, Rome, Italy, April 2007.

[74] A. Piazzi and A. Visioli. End-point control of flexible link via optimal dynamic inversion. In *Proceedings of the 2001 IEEE/ASME International Conference on Advanced IntelligentMechatronics*, pages 936–941, Como, Italy, July 2001.

[75] C. Guarino Lo Bianco and O. Gerelli. An alternative model for the evaluation of tyre shear forces under steady-state conditions. In *Proceedings of the 2008 IEEE International Conference on Intelligent Robots and Systems*, Nice, France, September 2008.

Acknowledgments

My thesis and my university path has finally come to its very end, so now it's time for the acknowledgments.

First and foremost, I would like to thank my supervisor professor Corrado Guarino Lo Bianco for having supported and motivated me during these years and for having provided his invaluable help in any part of the research results discussed in this thesis.

Even though Corrado has been my first promoter, I have to express my gratitude to my dear friend Luca for his rentless and unstoppable enthusiasm for everything, from girls to geometry and optimal control. To me, any of your inputs and our discussions have been really helpful. I would also like to thank the head of our small research group, professor Aurelio Piazzi, for his inspiring presence.

Besides Corrado and Luca, I would like to remember all the other colleagues I had the pleasure to share this experience with, in particular my office mates: I know that I will miss the always "optimistic" Jacopo, the endless knowledge of Dario and the european conference calls of Michele.

Then, a very big goodbye to all the folks of the Control System Laboratory I spend wonderful moments with. I have been lucky to meet such a crazy B.Sc. and M.Sc. students and coordinating part of their thesis work. Luca M, Marco Z, Matteo, Stefano D, Ergys, Antonio, Fabio, Luca S, Michele, Marco T, Alessandro, Nicola, Mauro, Lorenzo, Stefano S, Tommy and all the others I'm now forgetting, I wish you all to succeed in any of your current activities.

I would like also to express my gratitude to all the friends I have met during my stay in Enschede, The Netherlands, which I recall as one of the best moment ever. Many thanks to professors Job van Amerongen, head of the control lab, and to Stefano Stramigioli, my local supervisor, for the opportunity they gave me to work in their wonderful group.

Finally, many thanks to my parents who have made this experience possible and to all my relatives for their direct or indirect support. You have always been, and you will always be, a guide and a secure harbor.

Following a Tommy idea, if in this section I forgot to recall someone and if you are disappointed for this, there is still a lot of empty space in this page, so feel free to add your name.