Predictive Path Parameterization for Constrained Robot Control

Alberto Bemporad, Tzyh-Jong Tarn, *Fellow, IEEE,* and Ning Xi, *Member, IEEE*

***Abstract*—For robotic systems tracking a given geometric path, this paper addresses the problem of satisfying input and state constraints. According to a prediction of the evolution of the robot from the current state, a discrete-time device called *path governor* (PG) generates on line a suitable time-parameterization of the path to be tracked, by solving at fixed intervals a constrained scalar look-ahead optimization problem. Higher level switching commands are also taken into account by simply associating a different optimization criterion to each mode of operation. Experimental results are reported for a three-degree-of-freedom PUMA 560 manipulator subject to absolute position error, Cartesian velocity, and motor voltage constraints.**

***Index Terms*— Constraints, model predictive control, on-line time-scaling, optimization methods, path parameterization, robots, saturation.**

I. Introduction

TRACKING a given geometric path in the presence of physical and task constraints is a problem which often occurs in robotic manipulation tasks. Physical constraints usually consist of joint torque limits, due to joint-motor voltage saturation, joint velocity and acceleration limits, as well as limits on joint positions for reasons of mechanical construction. Task constraints may include jerk-free and tracking-error constraints, the latter usually due to industrial specifications on the tolerance of manufacture. These constraints can be taken into account in robot motion planning by studying the problem either in joint space, which leads to joint-space trajectory plan and motion control, or in task space through a translation of joint limits to task space. Because of the nonlinearity of the robot dynamic model, this translation often involves strong approximations and simplifications of the original constraints, besides a huge computational load and a consequent difficulty of real-time implementation.

In some joint-space robot motion planning schemes, the original limits are translated into constraints on the only reference trajectory [1]–[3]. For example, torque saturation is converted in constraints on the desired velocity and acceleration. However, this approach entails in assuming perfect tracking, and consequently neglecting part of the robot control system dynamics. Although this approach leads to computationally efficient strategies, it is inadequate in several applications. For instance, limits on the tracking error, which is directly related to the transient behavior of the robot control system, cannot be handled. In addition, in the case of saturating joint torques, existing methods do not leave any room for the amount of torque required by the feedback law; therefore, even if nominally satisfied, during the execution of the task the robot could require a total torque exceeding the limits. More complicated constrained path-planning problems can be formulated taking into account the overall closed-loop dynamics, determined by the adopted feedback torque controller; however, in most cases the resulting computational burden is huge, and the presence of measurement noise and unmodeled dynamics frustrates the effort of such an accurate formulation.

Based on the *time-scaling* concept introduced by [4] (and extended for multirobot configurations by [5]), [3] and, later, [6] suggested *on-line* trajectory time-scaling algorithms which take into account the overall closed-loop dynamics. Basically, given a desired path $q_d(s)$ in joint space, the path acceleration $\ddot{s}(t)$ is selected on-line within a range interval directly derived by the given torque limits and measurements. However, these methods are limited to problems with *input* constraints, and require a previously computed nominal optimal time-parameterization $s_n(t)$ of the desired path.

For a given desired path $Y_d(s)$ to be tracked by the end-effector of the robot, the task level motion planning and control approach described in this paper copes with generic *input/state* constraints—e.g., tracking error, torque, joint/task position constraints—and does not require any previous time-parameterization.

We assume that a feedback controller has been already designed in order to guarantee, in the absence of constraints, nice stability and tracking properties. However, fast reference signals may result in a violation of the constraints. In order to avoid this, we add to the predesigned control system a new discrete-time device, denominated *path governor* (PG), which, on the basis of current position and velocity measurements, generates on line a suitable parameterization $s(t)$ of the desired reference $Y_d(s)$, as depicted in Fig. 1.

The PG attempts to reduce the computational complexity in two ways: First, only a portion of the desired path is considered at a time; second, the resulting subtrajectory depends only on a scalar parameter—its end-point. As for predictive controllers [7]–[10], these simpler planning processes evolve according to a *receding horizon* strategy: The planned parameterization

Manuscript received June 9, 1997. Recommended by Associate Editor, W. Gruver.

A. Bemporad is with the Automatic Control Lab, Swiss Federal Institute of Technology, 8092 Zürich, Switzerland. He is also with the Department of Information Engineering, University of Siena, Italy.

T.-J. Tarn is with the Department of Systems Science and Mathematics, Washington University, St. Louis, MO 63130 USA.

N. Xi is with the Department of Electrical Engineering, Michigan State University, East Lansing, MI 48824-1226 USA.

Publisher Item Identifier S 1063-6536(99)08542-5.

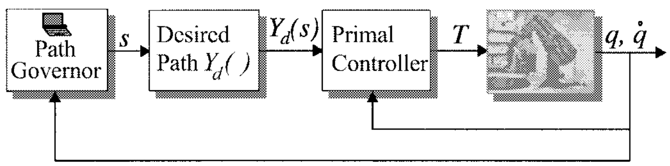

Fig. 1. Path tracking with on-line path parameterization.

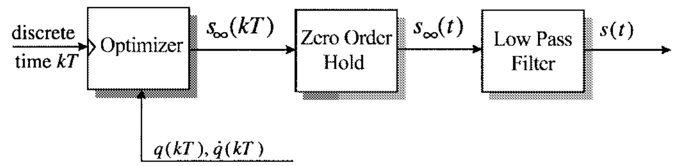

Fig. 2. Path governor (PG).

is applied until new measurements are available. Then, a new parameterization is evaluated which replaces the previous one. This provides the desired robustness against both model and measurement disturbances. The selection of the *temporary* end-point is performed by considering two objectives: 1) minimize the traversal time, i.e., the time required to track the desired path and 2) guarantee that the constraints are and will be fulfilled—i.e., no "blind-alley" is entered. The idea of reducing the complexity of constrained tracking problems was exploited in [11], [12] and, independently, in [13] for linear discrete-time systems, in [14] for nonlinear systems, and in [15] for uncertain systems. Preliminary studies on path governors have appeared in [16].

This paper is organized as follows. In Section II we describe the PG's path-parameterization strategy. In Section III we state the assumptions which are required to prove the main properties of the PG in Section IV. The constrained optimization problem related to the PG is briefly described in Section V, and some extensions are discussed in Section VI to cope with switching commands and partially known desired paths. Finally, experimental results on a three-degree-of-freedom (3-DOF) PUMA 560 manipulator subject (See Fig. 3) to absolute position error, Cartesian velocity, and motor voltage constraints are presented in Section VII.

II. Path Governor Formulation

The robot closed-loop dynamics is expressed by

$$\begin{cases} \dot{x} = \varphi(x, r, \cdots, r^{(h)}) \\ c = \ell(x, r, \cdots, r^{(h)}) \\ r = r(s) \\ x(0) = x_0 \end{cases} \tag{1}$$

where $x = [q' \quad \dot{q}' \quad x_i' \quad x_c']'$ collects the robot positions $q \in \mathbb{R}^m$, velocities $\dot{q}$, possible internal states x_i (e.g., electrical dynamics), and the state x_c of the controller, $x \in \mathbb{R}^n$, and the initial condition $x_0 \in \mathcal{X}_0$ for some compact set $\mathcal{X}_0 \subset \mathbb{R}^n$; $r(s) \in \mathbb{R}^m$ is the reference to be tracked by q, and is a given function of the scalar s, $s_0 \leq s \leq s_f$, determined by the specific task; $r^{(j)} \triangleq (d^j r/dt^j)$, $j = 0, \cdots, h$, where h is the number of derivatives involved in the control law, usually $h = 0$, 1 or 2; $c \in \mathbb{R}^p$ is the vector we wish to satisfy the constraints

$$c(t) \in \mathcal{C}, \quad \forall\, t \geq 0. \tag{2}$$

The aim of this paper is to design a device, referred to as PG, which on-line selects the parameter $s(t)$ so as to fulfill (2) and minimize the traversal time. Since, as one can expect, this selection involves a nonnegligible amount of computations, this device will operate in discrete time, namely every T seconds. In order to avoid "blind-alleys," rather than selecting $s(t)$ for only $kT < t \leq (k+1)T$, $k \in \mathbb{Z}_+ \triangleq \{0, 1, \cdots\}$, the PG cautiously considers an entire *virtual parameterization* $s(\tau; kT, s_\infty)$, where $\tau \in [0, +\infty)$ represents the prediction time, kT the current time, and s_∞ is a free scalar, $s_\infty = \lim_{\tau\to\infty} s(\tau; t, s_\infty)$, denominated *temporary end-point*. Based on the available measurements $q(kT)$, $\dot{q}(kT)$, the scalar s_∞^k is selected at time kT, by solving a constrained optimization problem. This aims at minimizing the time required to track the desired path, and takes into account that the predicted evolution $c(\tau; kT, s_\infty)$—generated by applying $r(s(\tau; kT, s_\infty))$ from the initial state $x(kT)$—satisfies the given constraints. The algorithm used by the PG can be formulated as follows.

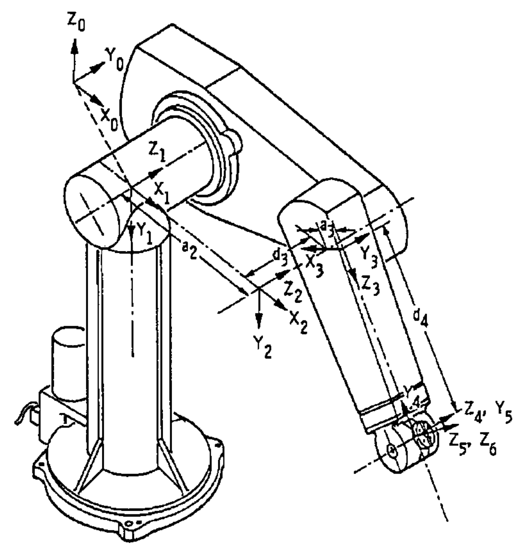

Fig. 3. Link coordinate assignments for PUMA 560.

Algorithm 1:

0) Let Δs_∞, α be fixed positive scalars, and let $s_\infty^{-1} \triangleq s_0$.
1) At time $t = kT$, find the temporary end-point $s_\infty^k \in [s_\infty^{k-1}, s_\infty^{k-1} + \Delta s_\infty]$ which maximizes

$$J(s_\infty) \triangleq s(T; kT, s_\infty) \tag{3}$$

with respect to s_∞ subject to the constraint that the virtual parameterization

$$s(\tau; kT, s_\infty) \triangleq s_\infty + [s(kT) - s_\infty]e^{-\alpha\tau}, \qquad \tau > 0 \tag{4}$$

satisfies the constraints

$$c(\tau; kT, s_\infty) \in \mathcal{C}, \quad \forall\, \tau > 0 \tag{5}$$

where $s(kT) = s(T; (k-1)T, s_\infty^{k-1})$ has been determined at time $(k-1)T$.

2) Apply $r(s(t)) = r(s(t-kT; kT, s_\infty^k))$ to the closed-loop system (1) only for $t \in (kT, (k+1)T]$.
3) Repeat the procedure at time $(k+1)T$ until $s((k+1)T) \geq s_f$.

The PG is sketched in Fig. 2.

We underline the notational difference which will be used hereafter between $s(\tau; kT, s_\infty)$, representing the *virtual* parameter at the prediction time τ, and $s(t)$ which is instead actually used to parameterize the desired path at time t.

Definition 1: At time kT and given the current state $x(kT)$ a temporary end-point s_∞ is *admissible* if the corresponding virtual evolution $c(\tau; t, s_\infty) \in \mathcal{C}$, $\forall \tau > 0$.

Remark 1: Notice that (3) aims at minimizing the time required for tracking the desired path. On the other hand, it also holds that

$$s_\infty^k = \begin{cases} \max\limits_{s_\infty^{k-1} \leq s_\infty \leq s_\infty^{k-1} + \Delta s_\infty} s_\infty \\ \text{subject to } c(\tau; kT, s_\infty) \in \mathcal{C}, \quad \forall \tau > 0. \end{cases} \tag{6}$$

Remark 2: The generated path parameterization is continuous and, where differentiable, satisfies the property

$$\dot{s}(t) > 0, \quad \forall t \geq 0.$$

Remark 3: By setting $T \to 0$, the previous strategy can be regarded as a way to select at each time t the derivative $\dot{s}(t) = \dot{s}(0; t, s_\infty) = \alpha[s_\infty - s(t)]$. Therefore, maximization of s_∞ corresponds to maximization of $\dot{s}$ and hence to minimization of the traversal time

$$t_f = \int_0^{t_f} dt = \int_{s_0}^{s_f} \frac{1}{\dot{s}}\, ds.$$

Remark 4: The upper-bound induced by a finite Δs_∞ prevents that the solution $s_\infty^k = +\infty$. In order to reach the maximum tracking speed, s_∞^k should be determined by the constraints on vector c rather than this artificial upper-bound. Since Δs_∞ can be selected arbitrarily large (for instance, $\Delta s_\infty = 2(s_f - s_0)$), in general one can ensure that the contribution of the constraint $s_\infty^k < +sk-1_infty+\Delta S-\infty$ is arbitrarily irrelevant.

Remark 5: The formulation of Algorithm 1 does not take into account the time required for the computation of s_∞^k, which will be denoted by T_c. Henceforth, for real-time applications, Algorithm 1 should be modified as follows. Let $T > T_c$. At time $(k-1)T$, the current measurement of the state $x((k-1)T)$ is used to predict $x(T; (k-1)T, s_\infty^{k-1})$. This replaces $x(kT)$ in Algorithm 1. Then s_∞^k is computed during the time interval $[(k-1)T, (k-1)T + T_c]$, and is available at time kT for the generation of the desired path $r(s(t))$, $t \in (kT, (k+1)T]$. This modification only introduces a time delay equal to T, as during the first time interval $[0, T]$ $s(t) \equiv s_0$ is applied. Therefore in the following sections we shall neglect this computational aspect, which instead will be discussed in Section VII.

Remark 6: For the sake of simplicity of the proofs, throughout the paper we refer to the reference $r(s)$ given in joint space rather than the desired path $Y_d(s)$ in task space. However, when the primal controller of the robot operates at the task level no actual computation of the joint-space reference by kinematic inversion is needed. Note in fact that, in the implementation of Algorithm 1 for typical task space controllers, the prediction $c(\tau; kT, s_\infty)$ can be computed from $q(kT)$, $\dot{q}(kT)$, $Y_d(s)$, $\dot{Y}_d(s)$, and $\ddot{Y}_d(s)$, and that r can be replaced by Y_d in Step 2).

III. Assumptions

In order to prove nice properties of the PG in Section IV, we consider the class of systems (1) and references r which fulfill the following assumptions. The notation $B(x_0, \epsilon)$ will be used to denote the ball $\{x: \|x - x_0\| \leq \epsilon\}$.

Assumption 1: The reference path $r: [s_0, s_f] \to \mathbb{R}^m$ is continuous and piece-wise differentiable h times, and

$$\left\| \frac{d^j r(s)}{ds^j} \right\| \leq R_j, \qquad \forall j = 0, \cdots, h$$

for some positive R_j.

In particular, Assumption 1 implies that there exists a compact set $\mathcal{R} \subset \mathbb{R}^m$ such that $r(s) \in \mathcal{R}$, $\forall s \in [s_0, s_f]$.

Assumption 2: At time $t = 0$ the temporary end-point $s_\infty = s_0$ is admissible from the initial state x_0.

This allows to initialize Algorithm 1 by setting $s_\infty^{-1} \triangleq s_0$. As an example, Assumption 2 is satisfied for $q(0) = r(s_0)$, $\dot{q}(0) = 0$, i.e., when the robot is initially at rest on the initial point of the desired path.

Definition 2: The reference path r is extended for $s > s_f$ by setting

$$r(s) \triangleq r(s_f), \qquad \forall s \geq s_f.$$

Notice that the properties in Assumption 1 still hold when r is redefined as in Definition 2.

Assumption 3: $\varphi: \mathbb{R}^n \times \mathcal{R} \times B(0, R_1) \times \cdots \times B(0, R_h) \to \mathbb{R}^n$ is continuous in $(x, r, \cdots, r^{(h)})$.

Assumption 4: $\ell: \mathbb{R}^n \times \mathcal{R} \times B(0, R_1) \times \cdots \times B(0, R_h) \to \mathbb{R}^n$ is uniformly continuous in $(x, r, \cdots, r^{(h)})$.

Assumption 5: For all $x_0 \in \mathcal{X}_0$ and for all $r \in \mathcal{R}$, if $r(s(t)) \equiv r$, $\forall t \geq \bar{t}$ then as $t \to \infty$

$$\begin{cases} q \to r \\ \dot{q} \to 0 \\ x^i \to x_r^i \\ x^c \to x_r^c \end{cases}$$

and x_r^i, x_r^c depend continuously on and are uniquely determined by r. Moreover, by letting $x_r \triangleq [r' \quad 0' \quad x_r'^i \quad x_r'^c]'$ and $c_r \triangleq \ell(x_r, r, 0, \cdots, 0)$, stability is uniform with respect to x_r, in that

$$\begin{aligned} &\forall \epsilon > 0 \quad \exists \rho(\epsilon) > 0: \|x(t_1) - x_r\| \leq \rho(\epsilon) \\ &\qquad \Rightarrow \|x(t) - x_r\| \leq \epsilon, \quad \forall t \geq t_1. \end{aligned} \tag{7}$$

Assumption 6: $\forall r \in \mathcal{R}$, if $r(s(t)) \to r$, and $r^{(j)}(s(t)) \to 0$ as $t \to \infty$, $\forall j = 1, \cdots, h$, then $q(t) \to r$, $\dot{q} \to 0$, $x^c(t) \to x_r^c$.

Notice that often in practical applications, because of finite numerical precision, Assumption 5 also implies Assumption 6. It is clear that, since the PG introduces a further feedback loop (see Fig. 1), stability and tracking properties of the overall system cannot be *a priori* inferred from Assumptions 5 and 6. These properties will be investigated in Section IV.

Assumption 7: The constraint set $\mathcal{C}$ has a nonempty interior.

Assumption 7 requires that there is some "maneuver space" inside $\mathcal{C}$, and that no equality constraints can be handled. A simple instance of $\mathcal{C}$ is a hyper-rectangle having nonzero volume.

Assumption 8: Let δ be a fixed (arbitrarily small) positive real. Then $\mathcal{R}$ is such that $B(c_r, \delta) \subseteq \mathcal{C}, \forall r \in \mathcal{R}$.

Assumption 8 requires that the commanded reference positions, each one taken as a set-point, are restricted to those ones which, in steady-state, give a corresponding constrained vector c_r which "lies away" from the border of $\mathcal{C}$ of at least a distance $\delta > 0$. By Assumption 7 such a δ always exists.

IV. Main Results

In this section, we will study some properties exploited by the path governor formulated in Section II. Lemma 1 will first prove that an admissible temporary end-point s_∞^k can be found at each time kT. Lemma 2 will show that s_∞^k cannot jam on a value between s_0 and s_f, in that a better admissible temporary end-point is always found within a finite time. Lemma 3, on the other hand, will prove that if the generated $s(t)$ converges to a final value $\bar{s}_\infty$, then $s_\infty^k \equiv \bar{s}_\infty$ after a finite time. Theorem 1 will make use of both lemmas to show that $s(t) = s_f$ after a finite time t_f. Theorem 2 will summarize the overall PG properties.

Lemma 1: $\forall k \in \mathbb{Z}_+$ there exists a temporary end-point $s_\infty^k \geq s_\infty^{k-1}$ which is admissible from the current state $x(kT)$.

Proof: The proof easily follows by induction. Assumption 2 states that an admissible s_∞^k can be found at least for $k = 0$. Assume that an admissible temporary end-point s_∞^{k-1} has been found at time $(k-1)T$. Now notice that $s(kT) = s_\infty^{k-1} + [s((k-1)T) - s_\infty^{k-1}]e^{-\alpha T}$ and hence

$$\begin{aligned} s(\tau; kT, s_\infty^{k-1}) &= s_\infty^{k-1} + [s(kT) - s_\infty^{k-1}]e^{-\alpha\tau} \\ &= s_\infty^{k-1} + [s((k-1)T) - s_\infty^{k-1}]e^{-\alpha T}e^{-\alpha\tau} \\ &= s(\tau + T; (k-1)T, s_\infty^{k-1}). \end{aligned}$$

Furthermore, $c(\tau; (k-1)T, s_\infty^{k-1}) \in \mathcal{C}, \forall \tau \geq 0$, in particular $\forall \tau \geq T$. Since also after T seconds the state has moved exactly to $x(kT) = x(T; (k-1)T, s_\infty^{k-1})$, it follows that

$$c(\tau; kT, s_\infty^k) = c(T + \tau; (k-1)T, s_\infty^{k-1}) \in \mathcal{C}, \quad \forall \tau \geq 0.$$

Therefore, at least $s_\infty^k \triangleq s_\infty^{k-1}$ is admissible at time $(k+1)T$ from $x((k+1)T)$. ■

Lemma 1 has proved that the sequence $\{s_\infty^k\}_{k=0}^\infty$ is defined and nondecreasing. Next Lemma 2 shows that such a sequence cannot assume a constant value less or equal than s_f.

Lemma 2: Let s_∞^k be admissible at time kT, $s_\infty^k \leq s_f$. Then, by applying $s(t) = s(t - kT; kT, s_\infty^k)$, there exists a time $\bar{t} \geq kT$ and $\bar{s}_\infty > s_\infty^k$ such that $\bar{s}_\infty$ is admissible at time $\bar{t}$.

Proof: See Appendix A. ■

Lemma 2 proved that, if $s_\infty^k < s_f$ at time kT, then after a finite time another admissible $s_\infty > s_\infty^k$ can be found. Next Lemma 3 shows that if $s(t) \to \bar{s}_\infty$, than this limit value is reached by s_∞^k in a finite time.

Lemma 3: Let $\lim_{t\to\infty} s(t) = \bar{s}_\infty \in [s_0, s_f]$. Then, there exists a finite time $\bar{t} \geq 0$ such that $\bar{s}_\infty$ is admissible at time $\bar{t}$.

Proof: See Appendix A. ■

Next Theorem 1 proves that the path governor generates a desired-path parameterization $s(t)$ such that s_f is reached in a finite time t_f.

Theorem 1: There exists a finite time t_f such that $s(t_f) = s_f$.

Proof: Assume by contradiction that $s(t) < s_f, \forall t \geq t_0$. Since $\dot{s}(t) > 0$, $s(t)$ is a real monotonically increasing and upper-bounded function of the time t, and hence there exists $\bar{s}_\infty \triangleq \lim_{t\to\infty} s(t) \leq s_f$. By Lemma 3 there exists a time $\bar{t}_1$ such that $s_\infty^k = \bar{s}_\infty, \forall kT \geq \bar{t}_1$. Then, by Lemma 2, there exists a time $\bar{t}_2$ such that for $kT \geq \bar{t}_2$ there exists $s_\infty^k > \bar{s}_\infty$ which is admissible. This contradicts the optimality of $\bar{s}_\infty$. Therefore, by continuity of $s(t)$, the proof follows. ■

Next Theorem 2 summarizes the properties of the proposed path governor.

Theorem 2: Let $s(t)$ be selected according to the path governor (Algorithm 1) formulated in Section II. Then

1) there exists a finite time t_f such that $s(t_f) = s_f$;
2) the constraints $c(t) \in \mathcal{C}$ are fulfilled for all $t \geq 0$ while the robot is driven along the path $r(s(t))$, $t \in [0, t_f]$;
3) $\lim_{t\to\infty} q(t) = r(s_f)$, $\lim_{t\to\infty} \dot{q}(t) = 0$.

Proof: Existence of such a t_f is guaranteed by Theorem 1. Constraint fulfillment follows by the selection criterion for the temporary end-points s_∞^k, in that $c(t, x(t), r(s(t)), \cdots, r^{(h)}(s(t))) = c(t, x(t - kT; kT, s_\infty^k), r(s(t - kT; kT, s_\infty^k)), \cdots, r^{(h)}(s(t - kT; kT, s_\infty^k)), \forall t \in (kT, (k+1)T]$. Convergence of q, $\dot{q}$ follows by Assumption 5. ■

V. Optimization Algorithm

In order to implement the PG described in the previous sections, the optimization problem (6), (24) is solved by using a bisection algorithm over the interval $[s_\infty^{k-1}, s_\infty^{k-1} + \Delta s_\infty]$. Let N denote the number of parameters s_∞ which can be evaluated during the selected period T. Because $\mathcal{C}$ is generic and the model of the robot is nonlinear, no convexity properties of the set of admissible s_∞ can be invoked. Then, the adopted bisection algorithm only provides local minima. By following an approach similar to [14], it can be proved that this does not affect the convergence results proved in Section IV. However, it is clear that if global minimization procedures were adopted in selecting s_∞^k, a smaller traversal time might be achieved, at the expense of an increased computational effort. Testing the admissibility of a given s_∞ requires the evaluation of (5), and consequently the numerical integration of the closed-loop equations (1) from initial state $x(kT)$. In Appendix B we describe how to translate this in a form which is more suitable for algorithmic implementation for general structures of (1). When instead feedback linearization is adopted as primal control strategy, the numerical integration can be carried out on a discrete-time version of the resulting linear system, verifying the (nonlinear) constraints at sample steps.

VI. Dealing with Switching Commands and Partially Known Desired Paths

We present some slight modifications of Algorithm 1 which allow the application of the PG when higher level commands are added to the (autonomous) tracking task, and/or the whole desired path is not completely known in advance.

A. Switching Commands

We wish to take into account higher level commands, which consist of switching the autonomous operation among the following: 1) stop the motion along the trajectory, for example because an unexpected obstacle has been detected; 2) slow down the motion; 3) invert the motion; and 4) resume the normal (autonomous and as fast as possible) execution of the tracking task. This commands can be taken into account by different options in the optimization involved in Algorithm 1, as follows.

- **STOP**: minimize s_∞, $s_\infty \in [s(kT), s_\infty^{k-1}]$.
- **SLOW DOWN**: find the maximum admissible $s_\infty^* \in [s_\infty^{k-1}, s_\infty^{k-1} + \Delta s_\infty]$ and set $s_\infty^k = s_\infty^{k-1} + [s_\infty^* - s_\infty^{k-1}]1/N$, where $N > 1$ is proportional to how much the tracking must be slowed down.
- **GO BACK**: minimize s_∞, $s_\infty \in [s_\infty^{k-1} - \Delta s_\infty, s_\infty^{k-1}]$.
- **GO, RESUME**: maximize s_∞, $s_\infty \in [s_\infty^{k-1}, s_\infty^{k-1} + \Delta s_\infty]$.

Each option also includes (5), and therefore the guarantee of fulfilling the constraints is preserved.

B. Partially Unknown Desired Paths

Assume that the desired path is not completely known in advance, i.e., $r(s)$ is known at time t only for $s \le s(t)+\sigma$, $\sigma \ge 0$. We distinguish two situations: $\sigma = 0$, which corresponds to a task where the end-effector for instance has to track an object whose motion is not known in advance; $\sigma > 0$, for example if new pieces of trajectory are appended before the completion of the tracking task. These modes of operations can be both taken into account by dynamically redefining the desired path. Define recursively, for $s > s(kT)$,

$$r_k(s) \triangleq \begin{cases} r_{k-1}(s), & \text{if } s \le s(kT) + \sigma \\ f_k(s), & \text{otherwise} \end{cases}$$

where $f_k(s)$ is a function which is constructed on-line on the basis of the data available at time kT, and satisfies the following properties:

1) $f_k(s(kT) + \sigma) = r_{k-1}(s(kT) + \sigma)$ (continuity);
2) $f_k(s) = r_{k-1}(s)$, $\forall s \ge s(kT)$, if no admissible temporary end-point can be found otherwise.

For instance, by assuming that the desired path is only known for $s \le s(kT) + \sigma$, consider for $s > s(kT) + \sigma$

$$f_k(s) \triangleq \begin{cases} r(s(kT) + \sigma), & \forall s > s(kT) + \sigma, \text{ if an} \\ & \text{admissible } s_\infty^k \text{ can be found} \\ r_{k-1}(s), & \text{otherwise.} \end{cases}$$

Constraint fulfillment is preserved for all $t \ge 0$, and, in the worst case, as $t \to \infty$, the robot joint coordinates $q(t)$ will jam on $r(s(k_0T) + \sigma)$ for some $k_0 \in \mathbb{Z}_+$.

VII. Experimental Results

The PG scheme has been implemented and experimentally tested on a PUMA 560, 6-DOF robot arm, in the Center for Robotics and Automation at Washington University. The planning and control algorithms have been implemented in a Silicon Graphics SGI 4D/340 VGX workstation, which has four symmetric processors. A multiprocessor SGI IRIX 4D/340VGX allows the parallel real-time computation of the parameter s_∞^k at the PG rate $1/T = 2.5$ Hz, and path generation and primal feedback control at the sampling rate $1/\tau_s = 1000$ Hz. It is interfaced to a Universal Motor Controller (UMC) through a shared memory. The sampling rate for position and velocity measurements is 1000 Hz.

In order to simplify the experiment, only three degrees of freedom have been used. By incorporating the end-effector in the third link, the dynamics is given by the following equation:

$$D(q)\ddot{q} + C(q, \dot{q}) + G(q) = \mathcal{T} \tag{8}$$

where $\mathcal{T}$ is three vector of joint torques, q is the three vector of joint displacements (with $\dot{q}$ and $\ddot{q}$ the first and second derivatives), $D(q)$ is the three-by-three inertia matrix, $C(q, \dot{q})$ is the three vector of centripetal and Coriolis terms, and $G(q)$ is the three vector of gravity terms [17].

The Cartesian task space position Y of the end-effector is related to the joint coordinates q by the direct/inverse kinematic equations reported in [18]. The end-effector has to track the desired path

$$Y_d(s) = \begin{bmatrix} r_0 x_d(s) + p_x \\ r_0 y_d(s) + p_y \\ p_z \end{bmatrix} \tag{9}$$

which is depicted in Fig. 4, where $[x_d, y_d]$ is the quarter of circle defined as

$$\begin{cases} \left[\cos\left(\frac{\pi}{2}s\right), \sin\left(\frac{\pi}{2}s\right)\right] & \text{if } 0 \le s \le 1 \\ [0, 2-s], & \text{if } 1 < s \le 2 \\ [s-2, 0], & \text{if } 2 < s \le 3 \\ [1, 0], & \text{if } s > 3 \end{cases}$$

$r_0 = 0.3$, and $[p_x, p_y, p_z] = [0.3, 0, -0.25]$ (units are given in the MKS system where unspecified). The desired path $r(s) = [\theta_{d1}(s), \theta_{d2}(s), \theta_{d3}(s)]$ in the joint-space is obtained by inverse kinematics, and satisfies Assumption 1.

In order to allow the end-effector's Cartesian position Y to track the reference signal $Y_d(s(t))$ in (9), the nonlinear feedback task controller (NFTC) reported in [19]

$$\begin{aligned} \mathcal{T} = {} & D(q)J(q)^{-1} \\ & \cdot [\ddot{Y}_d(s(t)) + K_p(Y_d - Y) + K_d(\dot{Y}_d - \dot{Y}) - \dot{J}(q, \dot{q})\dot{q}] \\ & + C(q, \dot{q}) + G(q) \end{aligned} \tag{10}$$

is adopted as primal controller, with $K_p = 6000 \cdot I_3$, $K_d = 80 \cdot I_3$, As a general rule to design controllers to be used

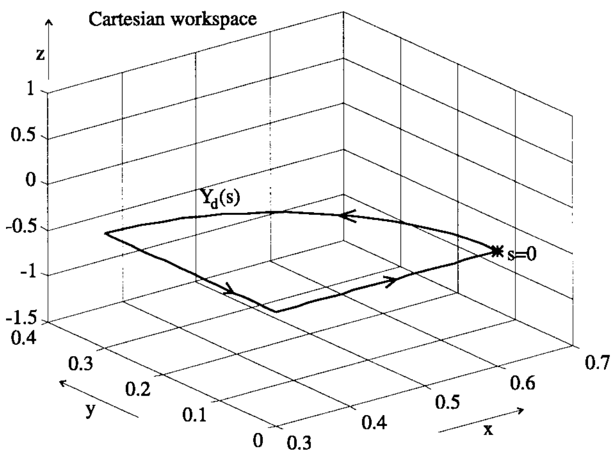

Fig. 4. Desired geometric path $Y_d(s)$. Units are expressed in MKS.

in connection with a PG, in order to maximize the tracking properties one should try to select a primal controller which provides a fast closed-loop response of system (1). Usually this corresponds to large violations of the constraints, which therefore can be enforced by inserting a PG. The closed-loop equations (1) resulting from (8) and (10)

$$\ddot{e} + K_d\dot{e} + K_p e = 0 \tag{11}$$

are linear, and therefore it is easy to show that Assumptions 3 and 5–8 are satisfied. However, these are fulfilled for a wider class of closed-loop systems. Consider for instance simple individual joint PD controllers with gravity compensation

$$\mathcal{T} = -K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d) + G(q) \tag{12}$$

let $x \triangleq [q' \quad \dot{q}']'$, $q_d(t) \equiv q_d$, and consider the following function:

$$V(x) = \tfrac{1}{2}\dot{q}'D(q)\dot{q} + \tfrac{1}{2}(q_d - q)'K_p(q_d - q)$$

which is a Lyapunov function for (8)–(12) [20]. Since its derivative along the trajectories of the system

$$\dot{V}(x) = -\dot{q}'K_d\dot{q} > 0$$

and $V(x) = 0$ iff $x = [q_d' \quad 0']$, the first part of Assumption 5 is satisfied. Uniform stability is proved as follows. By contradiction, suppose that there exists an $\epsilon > 0$ such that, $\forall \rho > 0$, there exists x_r and t_1 with $\|x(0) - x_r\| \leq \rho$ and $\|x(t_1) - x_r\| > \epsilon$. Since $\gamma_1 I \leq H(x) \leq \gamma_2 I$ for some positive γ_1, γ_2, by denoting by $\lambda_m(K_p)$, $\lambda_M(K_p)$ the minimum and maximum eigenvalue of K_p, respectively, and by setting $\gamma_3 \triangleq \min\{\lambda_m(K_p), \gamma_1\}$, $\gamma_4 \triangleq \max\{\lambda_M(K_p), \gamma_3\}$, it follows that $\|x(t_1) - x_r\| \leq (2/\gamma_3)V(x(t_1)) \leq (2/\gamma_3)V(x(0)) \leq (\gamma_4/\gamma_3)\rho$ for any arbitrary positive ρ, a contradiction. Moreover, in practice, the reference $r(s(t))$ is expressed by a finite numerical precision, and therefore, if $r(s(t))$ tends toward r, after a finite time $r(s(t)) \equiv r$, and Assumption 6 is verified. It is easy to check that Assumptions 3, 7, and 8 are satisfied as well.

We wish to impose the following constraint on the absolute position error:

$$\|e\| = \sqrt{(x - x_d)^2 + (y - y_d)^2 + (z - z_d)^2} \leq 5 \text{ mm} \tag{13}$$

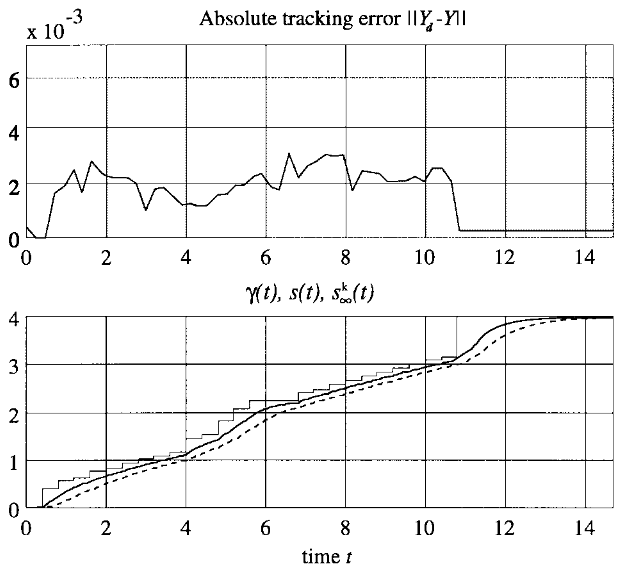

Fig. 5. Absolute tracking error, generated $s(t)$ (thick line), s^k_∞ (thin line), and $\gamma(t)$ (dashed line). Units are expressed in MKS.

on the Cartesian velocity

$$\|\dot{Y}\| \leq 0.6 \text{ m/s} \tag{14}$$

and on the motor voltages

$$V_i = \left(\frac{R_i}{N_i K_i^t}\right)\mathcal{T}_i + (K_i^b N_i)\dot{q}_i \leq 10 \text{ V} \tag{15}$$

where the values of armature resistance R_i, gear ratio N_i, torque constant K_i^t, and back EMF constant K_i^b are reported in [19]. Notice that, because of the choices (9) and (13)–(15), the constrained vector $c \triangleq [\|e\|^2, \|\dot{Y}\|^2, V_1, V_2, V_3]'$ fulfills Assumption 4.

On-line optimization has been performed by using the bisection method mentioned in Section V. For numerical integration of (1), the linear system (11) has been discretized with sampling period $T_{\text{pred}} = 0.002$ s in order to predict $x(T; (k-1)T, s_\infty^{k-1})$, and $T_{\text{constr}} = 0.005$ s for constraint checking. The resulting program is executed on one CPU in 0.06 to 0.28 s, thus allowing the selection of the PG period $T = 0.4$ s. The initial condition $q(0) \approx q_d(0)$, $\dot{q} = 0$, and $s_\infty = 0$ satisfies Assumption 2. The parameters $\alpha = 2$, $\Delta s_\infty = 0.03$, and $N = 5$ evaluations per period have been selected. No switching command was issued during the experiment.

The trajectories recorded during the experiment are depicted in Figs. 5 and 6.[1] The traversal time is about 11 s. The voltage constraints are slightly violated, because of the mismatching between the predictive model and the real system. However, it is clear that one can easily assign more stringent limitations in order to still guarantee constraint fulfillment. Since the derivatives $\dot{s}(t)$, $\ddot{s}(t)$ are only piece-wise continuous, $s(t)$ has been smoothed out by a low-pass filter before being used to parameterize the reference $Y_d(s)$. The resulting filtered signal $\gamma(t)$ is also depicted in Fig. 5. Notice that, because

[1] A real-time movie of the experiment can be retrieved at http://control.ethz.ch/~bemporad/dsi/images/puma560.mov.

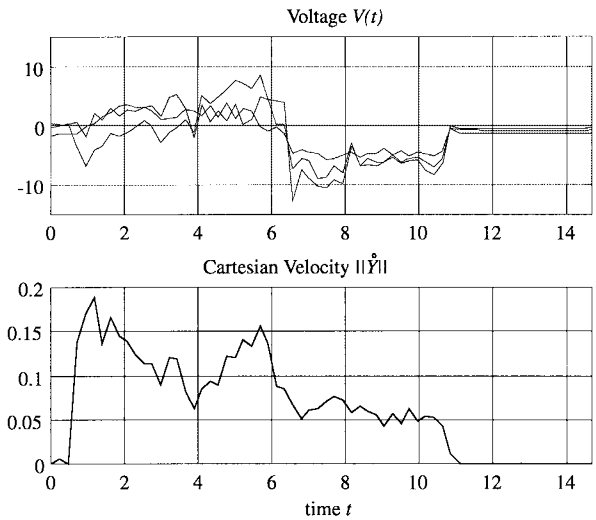

Fig. 6. Motor voltage V_i, $i = 1, 2, 3$, and Cartesian velocity $\|\dot{Y}\|$ for the trajectories of Fig. 5. Units are expressed in MKS.

of the adopted receding horizon strategy and the particular structure (4), the resulting path parameterization $s(t)$ is only near-minimum time. The performance is also affected by the available computational power, which sets a lower bound on T, and therefore on how often the PG can receive feedback and provide new temporary end-points.

Finally, we point out that the complexity of the PG mainly depends on the complexity of the numerical method adopted to integrate the differential equations of the robot. Although a thorough investigation of the numerical complexity of the PG is beyond the scope of the paper, one should expect that the complexity of the PG and the complexity of integrating the differential equations of the robot scale in the same way with the number of degrees of freedom.

VIII. Conclusions

For robotic systems, this paper has addressed the problem of tracking a given geometric path while satisfying constraints on the variables of the system. A time-parameterization of the path is generated on-line by performing at fixed intervals a scalar constrained optimization based on the integration of the robot's dynamic equations, and the method has been shown to be implementable in real-time.

The proposed strategy introduces a new design approach. Tedious trial-and-error sessions to tune the primal controller parameters in order to satisfy physical constraints—e.g., on motor voltages—are no longer required: Once the primal controller's design knobs have been roughly selected, the constraints are automatically enforced by the PG. Because of the general class of robot models, primal controllers, and physical constraints considered in this paper, we believe that the proposed approach is versatile enough to cope with many different practical robotic applications.

Appendix A

Proof of Lemma 2: Let $r_\infty^k \triangleq r(s_\infty^k)$, $x_\infty^k \triangleq x_{r_\infty^k}$. Since $s(t) = s(t-kT; kT, s_\infty^k) \to s_\infty^k$ as $t \to \infty$, by Assumption 1 $r(s(t)) \to r_\infty^k$, and, by Assumption 6, $x(t) \to x_\infty^k$. By Assumption 4, $c(t) \to c_\infty^k \triangleq \ell(x_\infty^k, r_\infty^k, 0, \cdots, 0)$, and also $\exists \epsilon = \epsilon(\delta) > 0$ such that, $\forall x \in \mathbb{R}^n$, $\forall r, \cdots, r^{(h)}$ satisfying Assumption 1, $\forall \overline{s}_\infty \in [s_0, s_f]$, $\overline{r}_\infty \triangleq r(\overline{s}_\infty)$, $\overline{x}_\infty \triangleq x_{\overline{r}_\infty}$

$$\|x - \overline{x}_\infty\| + \|r - \overline{r}_\infty\| + \sum_{j=1}^{h} \|r^{(h)} - 0\| \le \epsilon \qquad (16)$$

implies $\|c - \overline{c}_\infty\| \le \delta$, where $c = \ell(x, r, \cdots, r^{(h)})$, and $\overline{c}_\infty = \ell(\overline{x}_\infty, \overline{r}_\infty, 0, \cdots, 0)$. We wish to find a time $\overline{t}$ and a parameter $\overline{s}_\infty > s_\infty^k$ such that (16) holds for $x = x(\tau; \overline{t}, \overline{s}_\infty)$, $r = r(s(\tau; \overline{t}, \overline{s}_\infty))$, and $\forall \tau > 0$, in order to claim that $\overline{s}_\infty$ is admissible from $x(\overline{t})$. In order to accomplish this task, let t_a such that

$$\|x(t) - x_\infty^k\| \le \frac{1}{2}\rho\left(\frac{\epsilon}{h+2}\right)$$

for all $t \ge t_a$, where the function ρ is defined in (7). By virtue of Assumption 5 (continuity of x_r^c with respect to r) and Assumption 1 (continuity of $r(s)$), let $s_\infty^a > s_\infty^k$ such that the corresponding equilibrium state $x_\infty \triangleq x_{r(s_\infty)}$ satisfies

$$\|x_\infty - x_\infty^k\| \le \frac{1}{2}\rho\left(\frac{\epsilon}{h+2}\right)$$

$\forall s_\infty \in (s_\infty^k, s_\infty^a]$. By continuity of $r(s)$, there exists a $\Delta s_\infty^b > 0$ such that

$$\|r(s_\infty) - r_\infty^k\| \le \frac{\epsilon}{2(h+2)}$$

for all $s_\infty \in [s_\infty^k - \Delta s_\infty^b, s_\infty^k + \Delta s_\infty^b]$. Because $s(t) \to s_\infty^k$, take t_b such that $s(t) \ge$ '$s_\infty^k - \Delta s_\infty^b$ for all $t \ge t_b$. Since, for every temporary end-point s_∞ and time t, $s(\tau; t, s_\infty)$ monotonically increases from $s(t)$ to s_∞ as τ increases, the conditions $t \ge t_b$ and $s_\infty \in [s_\infty^k - \Delta s_\infty^b, s_\infty^k + \Delta s_\infty^b]$ imply

$$\|r(s(\tau; t, s_\infty)) - r_\infty^k\| \le \frac{\epsilon}{2(h+2)}, \quad \forall \tau \ge 0.$$

Consider now

$$\begin{aligned}
&\frac{dr}{dt}(s(\tau; t, s_\infty))\\
&\quad = \frac{dr}{ds}\dot{s}(\tau; t, s_\infty)\\
&\quad = \frac{dr}{ds}\alpha[s_\infty - s_\infty^k + (s(kT) - s_\infty^k)e^{-\alpha(t-kT)}]e^{-\alpha\tau}
\end{aligned}$$

since $\|dr/ds\| \le R_1$, one can find $s_\infty^c > s_\infty^k$ and t_c such that $\|\dot{r}(s(\tau; t, s_\infty))\| \le (\epsilon/2+h), \forall \tau \ge 0, \forall s_\infty \in (s_\infty^k, s_\infty^c]$, and $\forall t \ge t_c$. Similarly, since by Assumption 1 $\|(d^j r/ds^j)\| \le R_j$, $\forall j = 1, \cdots, h$, one can select s_∞^c and t_c so that $\|r^{(j)}(s(\tau; t, s_\infty))\| \le (\epsilon/2+h)$, $\forall \tau \ge 0$, $\forall j = 1, \cdots, h$, $\forall s_\infty \ge s_\infty^c$, $\forall t \ge t_c$. Then, by selecting

$$\overline{s}_\infty \triangleq \min\{s_\infty^a, s_\infty^k + \Delta s_\infty^b, s_\infty^c\}$$

and

$$\overline{t} \triangleq \max\{t_a, t_b, t_c\}$$

one has

$$\begin{aligned}&\|x(\tau;\bar t,\bar s_\infty)-\bar x_\infty\|+\|r(s(\tau;\bar t,\bar s_\infty))-\bar r_\infty\|\\&\quad+\sum_{j=1}^{h}\|r^{(h)}(s(\tau;\bar t,\bar s_\infty))\|\\&\le\|x(\tau;\bar t,\bar s_\infty)-x_\infty\|+\|x_\infty-\bar x_\infty\|\\&\quad+\|r(s(\tau;\bar t,\bar s_\infty))-r_\infty\|+\|r_\infty-\bar r_\infty\|\\&\quad+\sum_{j=1}^{h}\|r^{(h)}(s(\tau;\bar t,\bar s_\infty))\|\le\epsilon\\&\forall\,\tau\ge 0\end{aligned}$$

and thus $\|c(\tau;\bar t,\bar s_\infty)-\bar c_\infty\|\le\delta$, $\forall\,\tau\ge 0$. By Assumption 8, $c(\tau;\bar t,\bar s_\infty)\in\mathcal{C}$, $\forall\,\tau\ge 0$, follows. Then, $\bar s_\infty > s_\infty^k$ is admissible at time $\bar t$. ■

Proof of Lemma 3: Since

$$s((k+1)T)=s_\infty^k+[s(kT)-s_\infty^k]e^{-\alpha T}$$

it follows that

$$\lim_{k\to\infty}s_\infty^k=\lim_{k\to\infty}\frac{s((k+1)T)-s(kT)e^{-\alpha T}}{1-e^{-\alpha T}}=\bar s_\infty.$$

By setting $\bar r\triangleq r(\bar s_\infty)$, $\bar x\triangleq x_{\bar r}$, $\bar c\triangleq\ell(\bar x,\bar r,0,\cdots,0)$, and following arguments similar to those used in the proof of Lemma 2, we can find a time $\bar t$ such that:

$$\begin{aligned}&\|x(\tau;\bar t,\bar s_\infty)-\bar x\|+\|r(\tau;\bar t,\bar s_\infty)-\bar r\|\\&\quad+\sum_{j=1}^{h}\|r^{(h)}(\tau;\bar t,\bar s_\infty)\|\le\epsilon\end{aligned}\tag{17}$$

for all $\tau\ge 0$, or equivalently $\|c(\tau;\bar t,\bar s_\infty)-\bar c\|\le\delta$. By Assumption 8, $c(\tau;\bar t,\bar s_\infty)\in\mathcal{C}$, $\forall\,\tau\ge 0$ follows, or, equivalently, $\bar s_\infty$ is admissible at time $\bar t$. ■

APPENDIX B

At each time kT the PG must solve the optimization problem (6). Despite the simple structure of the cost function (3), the problem involves continuous state constraints (5) over an infinite horizon. We translate (6) in a general form which is more suitable for algorithmic implementation via Runge–Kutta methods. Let the set $\mathcal{C}$ be defined as

$$\mathcal{C}=\{c\in\mathbb{R}^p:\varphi_i(c)\le 0,\ i=1,\cdots,q\}.$$

Then, the constraints in (5) can be expressed in the form

$$g_i(x,s,\dot s,\cdots,s^{(h)})\le 0,\qquad i=1,\cdots,q\tag{18}$$

where the functions g_i derive from the composition of φ_i, ℓ, the desired path $r(s)$, and its derivatives (d^jr/ds^j), $j=0,\cdots,h$. For the sake of simplicity, we assume that the constrained vector c does not depend on the derivatives of the reference r, $c=\ell(x,r)$, which allows us to drop the dependence on the derivatives of s in (18). At a fixed time $t=kT$, system (1), (4) can be rewritten as

$$\begin{cases}\dot x=\varphi(x,r(s),\cdots,r^{(h)}(s))\\ \dot s=\alpha[s_\infty-s]\\ x(\tau=0)=x(kT)\\ s(\tau=0)=s(kT).\end{cases}\tag{19}$$

Then, the constraints in (5) become

$$\begin{aligned}&g_i(x(\tau;kT,s_\infty),s(\tau;kT,s_\infty))\le 0\\&\forall\,\tau>0,\qquad i=1,\cdots,q.\end{aligned}\tag{20}$$

According to the procedure in [21], the condition (20) is equivalent to the scalar constraint equality

$$\begin{aligned}G(s_\infty)\triangleq&\sum_{i=1}^{q}\int_0^{+\infty}\\&\cdot\max\{g_i(x(\tau;kT,s_\infty),s(\tau;kT,s_\infty)),0\}\,d\tau\\=&0.\end{aligned}\tag{21}$$

By defining for a small $\epsilon>0$ the function $\sigma_\epsilon:\mathbb{R}\mapsto\mathbb{R}$

$$\sigma_\epsilon(g)\triangleq\begin{cases}g, & \text{if } g>\epsilon\\ \dfrac{(g+\epsilon)^2}{4\epsilon}, & \text{if } |g|\le\epsilon\\ 0, & \text{if } g<-\epsilon\end{cases}$$

the fulfillment of the constraint (21) is guaranteed by the condition

$$\begin{aligned}&G_\epsilon(s_\infty)\\&\triangleq\sum_{i=1}^{q}\int_0^{+\infty}\sigma_\epsilon(g_i(x(\tau;kT,s_\infty),s(\tau;kT,s_\infty)),0)\,d\tau\\&=0\end{aligned}\tag{22}$$

which ensures better numerical conditioning, and allows the derivative (dG_ϵ/ds_∞) to be analytically computed, when this is required by gradient-based optimization algorithms. However, the evaluation of G_ϵ still requires the integration of (19) over an infinite horizon. This can be avoided by integrating the differential equations

$$\begin{cases}\dfrac{dx}{ds}=\dfrac{\varphi(x,r(s),\cdots,r^{(h)}(s))}{\alpha(s_\infty-s)}\\ x(s(kT))=x(kT)\end{cases}\tag{23}$$

over the finite interval $s\in[s(kT),s_\infty]$. Similarly, (22) is transformed in the finite integral

$$\begin{aligned}&G_\epsilon(s_\infty)\\&\triangleq\sum_{i=1}^{q}\int_{s(kT)}^{s_\infty}\sigma_\epsilon(g_i(x(s;kT,s_\infty),s),0)\frac{1}{\alpha(s_\infty-s)}\,ds\\&=0.\end{aligned}\tag{24}$$

Notice that, as a consequence of Assumption 8, convergence of the integral (21) or, equivalently, (24), can be guaranteed by choosing ϵ sufficiently small so that $B(c,\delta)\subset\mathcal{C}\Rightarrow\varphi_i(c)\le\epsilon, i=1,\cdots,q$. In this case, after a finite time $\tau^*\triangleq 1/\alpha\log((s(kT)-s_\infty)/(s^*-s_\infty))$, where $s^*\in(s(kT),s_\infty)$, the function to be integrated is zero.

REFERENCES

[1] J. Bobrow, S. Dubowsky, and J. Gibson, "Time-optimal control of robotic manipulators along specified paths," *Int. J. Robot. Res.*, vol. 4, no. 3, pp. 3–17, 1985.

[2] K. Shin and N. D. McKay, "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Trans. Robot. Automat.*, vol. 30, pp. 531–541, 1985.

[3] O. Dahl and L. Nielsen, "Torque-limited path following by on-line trajectory scaling," *IEEE Trans. Robot. Automat.*, vol. RA-6, pp. 554–561, 1990.

[4] J. M. Hollerbach, "Dynamics scaling of manipulator trajectories," *ASME J. Dynamic Syst., Measurement, Contr.*, vol. 105, pp. 102–106, 1984.

[5] S. B. Moon and S. Ahmad, "Time scaling of cooperative multirobot trajectories," *IEEE Trans. Syst., Man, Cybern.,* vol. 21, pp. 900–908, 1991.

[6] A. Kumagai, D. Kohli, and R. Perez, "Near-minimum time feedback controller for manipulators using on-line time scaling of trajectories," *ASME J. Dynamic Syst., Measurement, Contr.,* vol. 118, pp. 300–308, 1996.

[7] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—A survey," *Automatica,* vol. 25, no. 3, pp. 335–348, 1989.

[8] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Trans. Automat. Contr.,* vol. 35, pp. 814–824, 1990.

[9] E. Mosca, J. M. Lemos, and J. Zhang, "Stabilizing I/O receding-horizon control," in *Proc. 29th IEEE Conf. Decision Contr.,* 1990, pp. 2518–2523.

[10] D. W. Clarke and R. Scattolini, "Constrained receding-horizon predictive control," *Proc. IEE,* vol. 140, pp. 247–354, 1991.

[11] A. Bemporad and E. Mosca, "Constraint fulfillment in feedback control via predictive reference management," in *Proc. 3rd IEEE Conf. Contr. Applicat.,* Glasgow, U.K., 1994, pp. 1909–1914.

[12] A. Bemporad, A. Casavola, and E. Mosca, "Nonlinear control of constrained linear systems via predictive reference management," *IEEE Trans. Automat. Contr.,* vol. AC-42, pp. 340–349, 1997.

[13] E. G. Gilbert, I. Kolmanovsky, and K. Tin Tan, "Discrete-time reference governors and the nonlinear control of systems with state and control constraints," *Int. J. Robust Nonlinear Contr.,* pp. 487–504, 1995.

[14] A. Bemporad, "Reference governor for constrained nonlinear systems," *IEEE Trans. Automat. Contr.,* vol. AC-43, pp. 415–419, 1998.

[15] A. Bemporad and E. Mosca, "Fulfilling hard constraints in uncertain linear systems by reference managing," *Automatica,* vol. 34, no. 4, pp. 451–461, 1998.

[16] A. Bemporad and T. J. Tarn, "On-line path parameterization for manipulators with input/state constraints," in *IEEE/ASME Int. Conf. Advanced Intell. Mechatronics'97,* Tokyo, Japan, June 1997.

[17] T. J. Tarn, A. K. Bejczy, and X. Yun, "Dynamic equations for PUMA 560 robot arm," Dept. Syst. Sci. Math., Washington Univ., St. Louis, MO, Robot. Lab. Rep. SSM-RL-85-02, 1985.

[18] T. J. Tarn, A. K. Bejczy, G. T. Marth, and A. K. Ramadorai, "Kinematic characterization of the PUMA 560 manipulator," Dept. Syst. Sci. Math., Washington Univ., St. Louis, MO, Robot. Lab. Rep. SSM-RL-91-15, 1991.

[19] ——, "Performance comparison of four manipulator servo schemes," *IEEE Contr. Syst. Mag.,* pp. 22–29, Feb. 1993.

[20] H. Asada and J.-J. E. Slotine, *Robot Analysis and Control.* New York: Wiley, 1986.

[21] K. L. Teo and L. S. Jennings, "Nonlinear optimal control problems with continuous state inequality constraints," *J. Optimization Theory Applicat.,* vol. 63, no. 1, 1989.

[22] P. I. Corke, "The unimation puma servo system," MTM-226, 1994. Available ftp://janus.cat.csiro.au/pub/pic/pumaservo.ps.Z

[23] T. J. Tarn, N. Xi, and A. K. Bejczy, "Path-based approach to integrated planning and control for robotic systems," *Automatica,* vol. 32, no. 12, pp. 1675, Dec. 1996.

[24] T. J. Tarn, N. Xi, C. Guo, and A. K. Bejczy, "Human/machine sharing control for telerobotic systems," *Intell. Robot. Syst.,* pp. 373–380, 1995.

Alberto Bemporad was born in Florence, Italy, in 1970. He received the Ph.D. degree in control engineering in 1997 and the M.S. degree in electrical engineering in 1993 from the University of Florence, Italy.

He spent the academic year 1996 to 1997 at the Center for Robotics and Automation, Department Systems Science and Mathematics, Washington University, St. Louis, MO, as a Visiting Researcher. Since 1997, he is helding a postdoctoral position at the Automatic Control Lab, ETH, Zurich, Switzerland and in 1999 was appointed Assistant Professor at the University of Siena, Italy. He has published papers in the area of model predictive control, constrained control, robotics, and hybrid systems, and is involved in developing the Model Predictive Control Toolbox for Matlab.

Dr. Bemporad received the IEEE Centre and South Italy section "G. Barzilai" and the AEI (Italian Electrical Association) "R. Mariani" best graduate awards.

Tzyh-Jong Tarn (M'71–M'83–F'85) received the D.Sc. degree in control system engineering from Washington University, St. Louis, MO.

He is currently a Professor in the Department of Systems Science and Mathematics and the Director of the Center for Robotics and Automation at Washington University.

Dr. Tarn served as the President of the IEEE Robotics and Automation Society from 1992 to 1993, the Director of the IEEE Division X (Systems and Control) from 1995 to 1996, and a member of the IEEE Board of Directors from 1995 to 1996. At present, he serves as a member of the Nomination Committee of the IEEE Board of Directors. He also serves as both Chairman of the Management Committee and an Editor of the IEEE/ASME TRANSACTIONS ON MECHATRONICS and as a Board member of the IEEE Neural Network Council. He received the NASA Certificate of Recognition for the creative development of a technical innovation on *Robot Arm Dynamic Control by Computer* in 1987. The Japan Foundation for the Promotion of Advanced Automation Technology presented him with the Best Research Article Award in March 1994. He also received the Best Paper Award at the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, and the Distinguished Member Award from the IEEE Control Systems Society in 1996. He is the first recipient of both the Nakamura Prize (in recognition and appreciation of his contribution to the advancement of the technology on intelligent robots and systems over a decade) at the tenth anniversary of the IROS in Grenoble, France, 1997, and the Ford Motor Company Best Paper Award at the Japan/USA Symposium on Flexible Automation, Otsu, Japan, 1998. In addition, he was featured in the Special Report on Engineering of the 1998 Best Graduate School issue of the *U.S. News and World Report*.

Ning Xi (S'80–M'92) received the B.S. degree in electrical engineering from Beijing University of Aeronautics and Astronautics, the M.S. degree in computer science from Northeastern University, Boston, MA, and the D.Sc. degree in systems science and mathematics from Washington University, St. Louis, MO, in December 1993.

Currently, he is an Assistant Professor in the Department of Electrical and Computer Engineering at Michigan State University, East Lansing. Currently, his research interests include robotics, manufacturing automation, intelligent control and systems.

Dr. Xi received the Excellent Research Award from the Japan Foundation for the Promotion of Advanced Automation Technology in March 1995. He also received the Best Paper Award in IEEE/RSJ International Conference on Intelligent Robots and Systems in August 1995. Xi was the recipient of Anton Philips Best Student Paper Award in 1993 IEEE International Conference on Robotics and Automation. In addition, he has received National Science Foundation CAREER Award.