# Fast implementation of Model Predictive Control with guaranteed performance

M. Canale, V. Cerone, D. Piga, D. Regruto

*Abstract*— **A fast implementation of a given predictive controller for nonlinear systems is introduced through a piecewise constant approximate function defined over an hyper-cube partition of the system state space. Such a state partition is obtained by maximizing the hyper-cube volumes in order to guarantee, besides stability, an a priori fixed trajectory error as well as input and state constraints satisfaction. The presented approximation procedure is achieved by solving a set of nonconvex polynomial optimization problems, whose approximate solutions are computed by means of semidefinite relaxation techniques for semialgebraic problems.**

## I. INTRODUCTION

In Model Predictive Control (MPC) (see, e.g., [1]) the control move $u(k)$ at each time instant $k$ is computed through the solution to optimization problems aiming at the minimization of a given cost function. This fact introduces serious limitations in using MPC when small sampling times are required which may prevent online solution to the optimization problem. Furthermore, as a consequence of the application of the Receding Horizon (RH) principle, the control move $u(k)$ is a nonlinear static function of the system state $x(k)$, i.e. $u(k) = \mathbf{k}(x(k))$ (see [1]). In general, the function $\mathbf{k}$ is not known explicitly, except when the optimization problem is a linear or a quadratic program (see [2], [3], [4]), in which case, the control law $\mathbf{k}$ is known to be piece-wise affine (PWA) on a polyhedral partition of the state space. While such an approach is quite attractive, since online optimization can be avoided, it may show serious drawbacks. More precisely, since the polyhedral region containing the initial state must be computed between two sampling time instants, and the number of such regions exponentially increases with the control horizon, the time required to perform such a computation may limit the application of the procedure. In order to speed up the evaluation of the region containing the state, an efficient algorithm based on a binary search tree is proposed in [5]. Alternatively, in order to speed up the online computation of the control action in predictive control of linear time invariant (LTI) systems subject to a quadratic cost function and linear constraints, several online optimization algorithms can be found in the literature (see, e.g., [6], [7], [8], [9]). An alternative solution which permits online implementation of predictive controllers for both linear and nonlinear systems is based on approximate control laws $\tilde{\mathbf{k}} \approx \mathbf{k}$ derived from

The authors are with Dipartimento di Automatica e Informatica, Politecnico di Torino, Italy.
e-mail: `massimo.canale@polito.it`, `vito.cerone@polito.it`, `dario.piga@polito.it`, `diego.regruto@polito.it`

a finite number of exact values of $\mathbf{k}$ computed off–line. A first contribution along this line is given in [10], where a neural network approximation of $\mathbf{k}$ is presented. However, since the use of neural networks in approximating the function $\mathbf{k}$ does not allow the evaluation of the guaranteed approximation error, analysis of the approximate control law effects on performance of the closed loop system can not be carried out. In particular, constraints satisfaction on both the system state and input is not guaranteed a priori. Some results in this direction can be found in [11] where the use of a piece-wise constant approximation of $\mathbf{k}$ is investigated in the case of quantized input, and in [12], [13] and [14], where PWA approximations of $\mathbf{k}$ are presented. More precisely, an hyper-rectangular partition of the state space is proposed in [11], [12], [13], while a simplicial partition is used in [14]. In the above works feasibility of the approximate control law is shown and performance degradation is estimated in terms of an upper bound on the difference between the optimal cost function and the one obtained with $\tilde{\mathbf{k}}$. Guaranteed stability and regulation properties depend on this bound, which, on the other hand, can be reduced to a desired level of accuracy by increasing the number of partitions. The main assumption in [12], [13] and [14] is the convexity of the optimization problem involved in the predictive controller design.

Set membership (SM) function approximation theory is employed in [15], [16] to compute tight bounds on the approximation error. From such bounds approximate MPC laws with guaranteed stability, input constraints satisfaction and state regulation to an arbitrarily small neighborhood of the origin can be obtained. Furthermore, accuracy properties and performance are related to the number of the exact control moves employed for the computation of the approximating control law. However, no systematic procedure is introduced in [15] in order to a priori choose the number of the exact control moves needed to obtain a specified state trajectory error. Moreover, state constraints satisfaction is only guaranteed after a finite number of steps. In order to overcome these problems, in this paper, a piecewise constant approximation is introduced over an hyper-cubic partition of the system state space. In particular, an original optimization procedure is presented which allows the computation of a piecewise constant control law function $\tilde{\mathbf{k}}$ that a priori guarantees a given state trajectory error as well as input and state constraints satisfaction. This is achieved by solving nonconvex polynomial optimization problems by means of semidefinite relaxation techniques.

The paper is organized as follows. The considered model

predictive control problem is introduced in Section II-A, while the performance requirements to be satisfied by the approximate control law are described in Section II-B. In Section III an algorithm to define a suitable piecewise-constant function $\tilde{\mathbf{k}}$ which satisfies the performance specification given in Section II-B is presented. A simulation example is reported in Section IV in order to show effectiveness of the proposed approach.

## II. MODEL PREDICTIVE CONTROL

### A. Nominal formulation

Consider the nonlinear dynamical system

$$x(k+1) = f(x(k), u(k)), \qquad (1)$$

where $x(k) \in \mathbb{R}^n$ and $u(k) \in \mathbb{R}^m$ are the state and the input of the system, respectively. We assume that $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a polynomial function in $x$ and $u$. The aim of the controller is to regulate the system state to the origin under input and state constraints of the kind $u(k) \in \mathcal{U}$ and $x(k) \in \mathcal{X}$, for all $k > 0$, where $\mathcal{U}$ and $\mathcal{X}$ are assumed to be semialgebraic sets containing the origin and defined as

$$\mathcal{U} = \{ u \in \mathbb{R}^m : \ s_j(u) \geq 0, \ j = 1, \ldots, n_s \}, \qquad (2)$$

$$\mathcal{X} = \{ x \in \mathbb{R}^n : \ g_j(x) \geq 0, \ j = 1, \ldots, n_g \}, \qquad (3)$$

with $s_j(u)$ and $g_j(x)$ real-valued polynomials.
The control move is computed by solving, at each time step $k$, the optimization problem

$$\min_U \sum_{h=1}^{N_p-1} \left( \|x(k+h|k)\|_{2,Q} + \|u(k+h-1|k)\|_{2,R} \right) +$$
$$+ \|x(k+N_p|k)\|_{2,P}$$
$$s.t.$$
$$x(k+h|k) \in \mathcal{X}, \ h = 1, \ldots, N_p$$
$$u(k+h|k) \in \mathcal{U}, \ h = 0, \ldots, N_p - 1$$
$$\qquad (4)$$

with $Q, P \succeq 0$, $R \succ 0$; $x(k+h|k) = f(x(k+h-1|k), u(k+h-1|k))$ is the $i$-th step-ahead predicted state at time $h$ for initial condition $x(k)$ and $\|x\|_{2,Q}$ is a shortcut for $x^T Q x$. The vector $U = \left[ u(k|k)^T, \ldots, u(k+N_c-1|k)^T \right]^T$ is the set of control moves to be optimized, $N_p$ and $N_c \leq N_p$ are the prediction horizon and the control horizon, respectively. The remaining predicted control moves $\left[ u(k+N_c|k)^T, \ldots, u(k+N_p-1|k)^T \right]^T$ can be computed according to different strategies, e.g. $u(k+j|k) = u(k+N_c-1|k)$, for all $j = N_c, \ldots, N_p - 1$. The minimizer for problem (4) is denoted as $U^* = \left[ u^*(k|k)^T, \ldots, u^*(k+N_c-1|k)^T \right]^T$ and the control move at time $k$ is chosen according to the Receding Horizon (RH) principle, i.e. $u(k) = u^*(k|k)$. The set of the state $x(k|k)$ for which problem (4) is feasible is denoted as $\mathcal{F} \subseteq \mathbb{R}^n$. The application of the RH strategy leads to a predictive controller which is a static nonlinear function of the current state $x(k)$, i.e. $u(k) = \mathbf{k}(x(k))$. In the sequel, function $\mathbf{k}$ is

referred to as the nominal control law. Indeed, the control function $\mathbf{k}$ is defined only in the feasibility set $\mathcal{F}$. It is assumed that, by applying the nominal control law $\mathbf{k}$, system (1) is asymptotically stable at the origin for any initial state $x_0 \in \mathcal{F}$ (see. e.g. [17]). It is worth remarking that, in general, an explicit representation of both the feasibility set $\mathcal{F}$ and of the nominal control law $\mathbf{k}$ is not known. Therefore, at each sample time $k$, problem (4) must be solved in order to compute the control move $u(k)$. Indeed, this leads to a limitation in the practical use of the MPC for fast dynamical systems, whose sampling time is too small for a real-time solution of problem (4). A possible solution to ensure online implementation of a predictive controller is the use of an approximate control law $\tilde{\mathbf{k}} \approx \mathbf{k}$ derived using a finite number of exact values of $\mathbf{k}$ computed off–line. The evaluation of the approximate function $\tilde{\mathbf{k}}$ allows the on-line computation of the control move $\tilde{u}(k) = \tilde{\mathbf{k}}(x(k))$ within the required sampling time. In this work, the optimal control law $\mathbf{k}$ is supposed to be Lipschitz over $\mathcal{F}$, i.e.

$$\|\mathbf{k}(x^{(1)}) - \mathbf{k}(x^{(2)})\|_2 \leq \gamma_L \|x^{(1)} - x^{(2)}\|_2, \ \ \forall x^{(1)}, x^{(2)} \in \mathcal{F},$$
$$\qquad (5)$$

where $\gamma_L$ is the Lipschitz constant, whose a rough upper-bound is assumed to be available.

### B. Approximate control law

Let $x(k)$ and $\tilde{x}(k)$, with $k = 0, 1, 2, \ldots$, be the state trajectories obtained by applying the optimal control law $\mathbf{k}(x(k))$ and the approximate control law $\tilde{\mathbf{k}}(\tilde{x}(k))$, respectively, at each time $k$ and for a given initial condition $x_0 \in \mathcal{F}$. Indeed, state $x(k)$ and $\tilde{x}(k)$ satisfy the following difference equations

$$x(k+1) = f(x(k), \mathbf{k}(x(k))), \ \ \forall k \geq 0; \ x(0) = x_0. \quad (6)$$

$$\tilde{x}(k+1) = f(\tilde{x}(k), \tilde{\mathbf{k}}(\tilde{x}(k))), \ \ \forall k \geq 0; \ \tilde{x}(0) = x_0. \quad (7)$$

In this paper, it will be shown how to compute an approximate control law $\tilde{\mathbf{k}}$ guaranteing that the controlled system (7) satisfies the performance introduced in the following Definition.

*Definition 1:* **Performance properties for the approximate control law $\tilde{\mathbf{k}}$**
**P** 1.1: The control move $\tilde{u}(k) = \tilde{\mathbf{k}}(x(k))$ is required to satisfy the input constraints defined in (2), i.e. $\tilde{u}(k) \in \mathcal{U}$ for all $k = 0, 1, 2, \ldots$
**P** 1.2: The approximate state $\tilde{x}(k)$ in (7) is required to satisfy the state constraints defined in (3), i.e. $\tilde{x}(k) \in \mathcal{X}$ for all $k = 1, 2, \ldots$
**P** 1.3: The distance between the nominal state trajectory in (6) and the approximate state trajectory in (7) is required to be bounded, i.e., for a given $\varepsilon > 0$,

$$\|x(k) - \tilde{x}(k)\|_q \leq \varepsilon \ \ \forall k = 1, 2, \ldots \qquad (8)$$

with $q \geq 1$. In this work, $q = \infty$ is assumed. ∎

*Remark 1:* Satisfaction of Property **P**1.3 implies stability of the closed-loop system. In fact, since by applying the

nominal control law **k** the nominal state trajectory asymptotically converges to the origin, then, the approximate state trajectory asymptotically converges to an arbitrarily small set $\mathbb{B}(0, \varepsilon) = \{x \in \mathbb{R}^n : \|x\|_\infty \leq \varepsilon\}$, i.e.

$$\lim_{k \to \infty} \|\tilde{x}(k)\|_\infty \leq \varepsilon. \qquad (9)$$

■

## III. EVALUATION OF A SUITABLE APPROXIMATE CONTROL LAW

In this section, it is described how to get an approximate control law $\tilde{\mathbf{k}}(x)$ satisfying Properties **P**1.1-**P**1.3. A piecewise constant approximating function $\tilde{\mathbf{k}}(x)$ will be looked for, and the regions where $\tilde{\mathbf{k}}(x)$ is locally constant will be axis-aligned equal-sized boxes in the state space, so that the real-time effort in evaluating in which box a given state $x$ lies is reasonably low. In order to simplify notation single-input systems, i.e. $m = 1$ are considered. Extension to systems with multiple inputs is trivial.

### A. Overview of the method

For the sake of clarity, a general overview of the algorithm used to define a piecewise-constant control law $\tilde{\mathbf{k}}(x)$ satisfying Properties **P**1.1-**P**1.3 is first presented.

*Algorithm 1:* **Definition of a piecewise-constant approximate control law $\tilde{\mathbf{k}}(x)$ with guaranteed performance**

A1.1 Pick a box $X_0 \supseteq \mathcal{X}$ such that the center of $X_0$ is the origin of the state space.

A1.2 Solve problem (4) for $x(k|k) = x_0^c$, where $x_0^c$ is the center of the box $X_0$. If problem (4) is not feasible go to step A1.3, else go to step A1.4.

A1.3 If the volume of the box $X_0$ is smaller than a given tolerance, mark $X_0$ as infeasible, else split $X_0$ into $3^n$ axis-aligned equal-sized boxes $X_i$, with $i = 1, \ldots, 3^n$. For every $i = 1, \ldots, 3^n$, set $X_0 = X_i$ and go to step A1.2.

A1.4 Let $u_0^c$ be the optimal control law computed at step A1.2 of the algorithm. If, for any state $x(k|k) \in X_0$, the control move $u_0^c$ is such that Properties **P**1.2-**P**1.3 are satisfied, mark $X_0$ as feasible. Else, if the volume of the box $X_0$ is smaller than a given tolerance, mark $X_0$ as infeasible, else split $X_0$ into $3^n$ axis-aligned equal-sized boxes $X_i$, with $i = 1, \ldots, 3^n$. For every $i = 1, \ldots, 3^n$, set $X_0 = X_i$ and go to step A1.2.

A1.5 Collect all the boxes marked as feasible in an $N$-dimensional list $\mathcal{F}_0$, where $N$ is the number of boxes marked as feasible.

By using Algorithm 1, the feasibility set $\mathcal{F}$ is inner-approximated by $\mathcal{F}_{in} = \bigcup_{j=1}^{N} X_j$, where $X_j$ denotes a component of the list $\mathcal{F}_0$. Besides, for every $x(k|k) \in X_j$,

when the input $u(k) = u_j^c$ (corresponding the optimal control law associated to the center of the box $X_j$ and computed at stage A1.2 of the algorithm), the predicted state $x(k+1|k) = f(x(k|k), u_j^c)$ belongs to the state-constraint set $\mathcal{X}$ and the distance between the approximate state trajectory and the nominal state is smaller than a given tolerance $\varepsilon$. In other words, the piecewise-constant control law $\tilde{\mathbf{k}} : \mathcal{F}_{in} \to \mathbb{R}$ defined as:

$$\tilde{\mathbf{k}}(x) = u_j^c \text{ when } x \in X_j \qquad (10)$$

guarantees satisfaction of properties **P**1.2-**P**1.3. Besides, since $u_j^c$ is the solution of problem (4) for a given state value $x(k) = x_0^c$, it follows that $u_j^c \in \mathcal{U}$ for all $j = 1, \ldots, N$. Therefore, $u(k) = \tilde{\mathbf{k}}(x(k))$ satisfies also Property **P**1.1. It is worth noting that the proposed algorithm implies that the optimal control move is applied when the approximate state trajectory $\tilde{x}(k)$ reach the origin of the state space which is assumed to be an equilibrium point for the nominal controlled system. This is due to the fact that the initial box $X_0$ is centered at the origin and subsequently split into $3^n$ axis-aligned equal-sized boxes.

In order to reduce the complexity in evaluating which region $X_j$ of $\mathcal{F}_0$ a given state $x(k|k)$ belongs to, the set $X_j$ is partitioned into $N_j$ axis-aligned boxes $X_j^i$ of edges of length $\{\Delta_1, \ldots, \Delta_n\}$, where $\Delta_i$, with $i = 1, \ldots, n$, are the lengths of the edges of the smallest box in $\mathcal{F}_0$. In such a way, the set $\mathcal{F}_{in}$ is partitioned into $M = \sum_{j=1}^{N} N_j$ equal-sized boxes and $\tilde{\mathbf{k}}$ in (10) can be rewritten as

$$\tilde{\mathbf{k}}(x) = u_j^c \text{ when } x \in X_j^i. \qquad (11)$$

### B. Technical results

Technical details of stage A1.4, which is the key point of Algorithm 1, are now discussed.

*Satisfaction of Property **P**1.2*

Checking if the controlled system satisfies Property **P**1.2 requires the solution to the following infinite-dimensional feasibility problem:

*Problem 1:* $\forall x(k) \in X_0$, the set of the one-step ahead predicted state $f(x(k), u_0^c)$ belongs to $\mathcal{X}$. ■

Problem 1 is equivalent to the following finite-dimensional feasibility problem:

*Problem 2:* The set $\{x \in X_0 : f(x, u_0^c) \notin \mathcal{X}\}$ is not empty, that is $g_j^* < 0$ for some $j = 1, \ldots, n_g$, where

$$g_j^* = \min_{x \in X_0} g_j(f(x, u_0^c)). \qquad (12)$$

■

Indeed, Problem 1 is feasible if and only if no solution exists to Problem 2. Then, in order to guarantee that the approximate control law $\tilde{\mathbf{k}}$ satisfies Property **P**1.2 for all $x \in X_0$, the global minimum $g_j^*$ of problem (12) has

to be computed, and then inequality $g_j^* \geq 0$ has to be checked for all $j = 1, \ldots, n_g$. Unfortunately, problem (12) is not convex since the function $g \circ f = g(f(x, u_0^c))$ is a polynomial function of the decision variable $x$. Therefore, standard nonlinear optimization methods (e.g. Gauss-Newton algorithm or nonlinear conjugate gradient method) can not be exploited to solve problem (12) since they can trap in a local minimum $g_j^l$. As a consequence, the solutions to problems (12) obtained through such nonlinear optimization algorithms cannot be used to infer that the approximate control law $\tilde{\mathbf{k}}$ satisfies Property **P**1.2 for all $x \in X_0$. In fact, from condition $g_j^l \geq 0$, it is not possible to infer that $g_j^* \geq 0$. On the other hand, by relaxing (12) into convex optimization problems, lower bounds $\underline{g}_j$ of the global minimum $g_j^*$ can be numerically computed. If $\underline{g}_j \geq 0$ for all $j = 1, \ldots, n_g$, then $g_j^* \geq 0$ for all $j = 1, \ldots, n_g$. This means that the approximate control law $\tilde{\mathbf{k}}$ is guaranteed to satisfy Property **P**1.2 for all $x \in X_0$. Indeed, relaxing problem (12) may lead to conservativeness in partitioning the domain of the approximate control law $\tilde{\mathbf{k}}$. In fact, $g_j^*$ can be greater or equal than zero even if $\underline{g}_j < 0$ and, as a consequence, $X_0$ can be either splitted into $3^n$ boxes or marked as infeasible at stage A1.4 of Algorithm 1 even if the control law $\tilde{\mathbf{k}}$ satisfies Property **P**1.2 for all $x \in X_0$. Anyway, the approximate control law $\tilde{\mathbf{k}}$ is guaranteed to satisfy Property **P**1.2 for all $x$ belonging to the domain $\mathcal{F}_{in}$ of $\tilde{\mathbf{k}}$.

Since (12) is a polynomial optimization problem, a lower bound $\underline{g}_j$ of the global minimum $g_j^*$ can be computed through the relaxation technique proposed by Lasserre in [18], which is based on the idea of constructing an increasing sequence of convex semidefinite programming (SDP) problems, whose optima are guaranteed to converge monotonically to the global optimum $g_j^*$ of the original nonconvex polynomial problem (12). Although the method is guaranteed to converge as far as the length of the number of successive SDP relaxations (relaxation order) goes to infinity, exact global optima have been obtained for a number of small and medium size problems with a low relaxation order (see [19] for a collection of test problems solved with relaxation order less or equal than 4). An efficient Matlab implementation of such a relaxation technique has been recently developed by Henrion and Lasserre in the open source freeware software Gloptipoly [20], which exploits the solver SeDuMi [21] to solve SDP problems in polynomial time.

*Fulfillment of Property **P**1.3*

In order to check if the controlled system satisfies Property **P**1.3, feasibility of the following infinite-dimensional problem has to be evaluated.

*Problem 3:* For all $\tilde{x}(k) \in X_0$ and for all $x(k)$ such that $\|\tilde{x}(k) - x(k)\|_\infty \leq \varepsilon$, the distance between the one step-ahead predicted nominal state $x(k + 1) = f(x(k), \mathbf{k}(x))$

and the one-step predicted approximate state $\tilde{x}(k + 1) = f(\tilde{x}(k), \mathbf{k}(\tilde{x}))$ is bounded by $\varepsilon$, i.e.

$$\|f(\tilde{x}, u_0^c) - f(x, \mathbf{k}(x))\|_\infty \leq \varepsilon. \tag{13}$$

∎

In view of assumption (5), the optimal control move $\mathbf{k}(x)$ is such that

$$\|\mathbf{k}(x_0^c) - \mathbf{k}(x)\|_2 \leq \gamma_L \|x_0^c - x\|_2, \quad \forall x_0^c, x \in \mathcal{F}. \tag{14}$$

Let us define $\delta u = \mathbf{k}(x_0^c) - \mathbf{k}(x)$. Inequality (14) can be rewritten as

$$\|\delta u\|_2 \leq \gamma_L \|x_0^c - x\|_2, \quad \forall x_0^c, x \in \mathcal{F}. \tag{15}$$

From equation (10), condition (15) and definition of $\delta u$, it follows:

$$\mathbf{k}(x) = u_0^c - \delta u, \quad \text{with } \delta u : \|\delta u\|_2 \leq \gamma_L \|x_0^c - x\|_2. \tag{16}$$

Then, from conditions (16), statement of the problem can be reformulated as follows: for all $\tilde{x} \in X_0$, for all $x : \|\tilde{x} - x\|_\infty \leq \varepsilon$, and for all $\delta$ s.t. $\|\delta u\|_2 \leq \gamma_L \|x_0^c - x\|_2$, the set $\mathcal{H}$, defined as

$$\mathcal{H} = \left\{ \tilde{x}, x, \delta u : \|f(\tilde{x}, u_0^c) - f(x, u_0^c - \delta u)\|_\infty \leq \varepsilon \right\}, \tag{17}$$

is not empty. Let us rewrite the set $\mathcal{H}$ as:

$$\mathcal{H} = \left\{ \tilde{x}, x, \delta u : h_i(\tilde{x}, x, \delta u) \geq 0, \quad i = 1, \ldots, 2n \right\}, \tag{18}$$

where $h_i(\tilde{x}, x, \delta u) \geq 0$, with $i = 1, \ldots, 2n$, are the constraints defining $\mathcal{H}$ in (17), i.e.

$$h_i(\tilde{x}, x, \delta u) = \begin{cases} f_s(\tilde{x}, u_0^c) - f_s(x, u_0^c - \delta u) + \varepsilon \\ \quad \text{when } i = 1, \ldots, n; \ s = i; \\ \\ -f_s(\tilde{x}, u_0^c) + f_s(x, u_0^c - \delta u) + \varepsilon \\ \quad \text{when } i = n+1, \ldots, 2n; \ s = i - n; \end{cases} \tag{19}$$

with $f_s$ denoting the $s$-th component of the vector function $f$.

Then, Problem 3 can be evaluated by solving the following finite-dimensional feasibility problem.

*Problem 4:* There exists at least one value of $\tilde{x} \in X_0$, $x \in \mathbb{R}^n : \|\tilde{x} - x\|_\infty \leq \varepsilon$ and $\delta u : \|\delta u\|_2^2 \leq \gamma_L \|x_0^c - x\|_2^2$, such that $h_i(\tilde{x}, x, \delta u) < 0$ for some $i = 1, \ldots, 2n$, i.e. $h_i^* < 0$ for some $i = 1, \ldots, 2n$, where

$$h_i^* = \min_{\tilde{x}, x \in \mathbb{R}^n, \delta u \in \mathbb{R}} h_i(\tilde{x}, x, \delta u)$$

$$\text{s.t} \tag{20}$$

$$\tilde{x} \in X_0; \|\tilde{x} - x\|_\infty \leq \varepsilon, \|\delta u\|_2^2 \leq \gamma_L^2 \|x_0^c - x\|_2^2.$$

∎

Indeed, Problem 3 is feasible if and only if no solution exists to Problem 4. Comments on Problem 2 also hold for Problem 4, and a lower bound $\underline{h}_i$ of the global optimum of problem (20) can be efficiently computed through the same SDP-relaxation technique proposed to relax (12).

## IV. A SIMULATED EXAMPLE

In order to show the effectiveness of the proposed approach, a simulated example is here reported. The following nonlinear system is considered:

$$\begin{cases} x_1(k+1) = \frac{8}{10}x_1(k) + \frac{1}{20}\left(-x_2(k) - \frac{3}{2}x_1^2(k) - \frac{1}{2}x_1^3(k)\right) \\ x_2(k+1) = \frac{1}{2}x_2(k) + \frac{1}{20}(3x_1(k) - x_2(k)) + \frac{1}{2}u(k) \end{cases}$$

$$(21)$$

State $x \in \mathbb{R}^2$ is constrained to belong to the set $\mathcal{X}$ defined as

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 : \|x\|_\infty \leq 1 \right\}, \qquad (22)$$

while the input $u$ is assumed to take values in the interval $[-0.5, \ 0.5]$, i.e. $u \in \mathcal{U}$ with

$$\mathcal{U} = \{ u \in \mathbb{R} : |u| \leq 0.5 \}. \qquad (23)$$

A model predictive controller is designed by solving problem (4) using the following values of matrixes $Q, P, R$:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad R = 0.01. \qquad (24)$$

The prediction horizon $N_p$ is chosen equal to 10, while the control horizon $N_c$ is set to 5. All the control moves $u(k + N_c|k), u(k + N_c + 1|k), \ldots, u(k + N_p - 1|k)$ are set equal to $u(k + N_c - 1|k)$. The approximate control law $\tilde{\mathbf{k}}$ is computed through Algorithm 1 by imposing that the distance between the nominal and the approximate state trajectory is smaller than $\varepsilon = 0.05$, i.e. $\|\tilde{x}(k) - x(k)\|_\infty \leq 0.05$, for all $k = 1, 2, \ldots$. An upper-bound of the Lipschitz $\gamma_L$ in (5) has been numerically evaluated with the algorithm proposed in [15] and it is equal to 1.1. The algorithm is initialized at stage A1.1 with the set $X_0 = \left\{ x \in \mathbb{R}^2 : \|x\|_\infty \leq 2 \right\}$. The obtained equal-sized boxes partition of the feasibility set $\mathcal{F}_{in}$ for function $\tilde{\mathbf{k}}$ is reported in Fig. 1. The edge of each box is equal to 0.0494. A comparison between the nominal state trajectory and the approximate state trajectory is shown in Fig. 2, starting from initial conditions $x(0) = [1.5; 1.5]$. It is worth remarking that, although the initial state $x(0)$ does not belong to the state-constraint set $\mathcal{X}$,
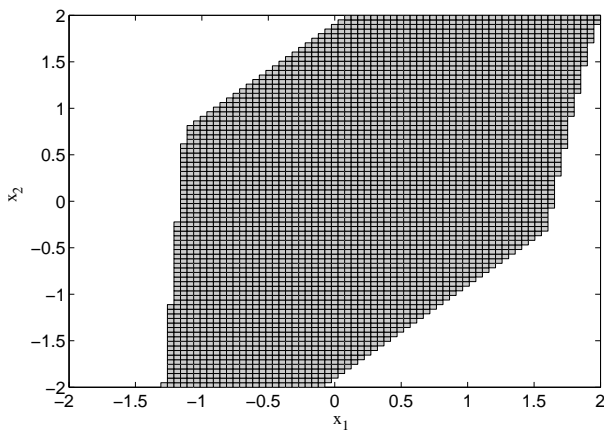
the evolution of both the nominal $x(k)$ and approximate state $\tilde{x}(k)$ lies in $\mathcal{X}$ for all $k = 1, 2, \ldots$. This means that the approximate control law $\tilde{\mathbf{k}}$ satisfies Property **P** 1.2. The error $\varepsilon_1(k) = x_1(k) - \tilde{x}_1(k)$ between the nominal state variable $x_1(k)$ and the approximate state variable $\tilde{x}_1(k)$ at each time $k = 0, 1, \ldots$ is reported in Fig. 3, while the error $\varepsilon_2(k) = x_2(k) - \tilde{x}_2(k)$ between the nominal variable $x_2(k)$ and the approximate state variable $\tilde{x}_2(k)$ is reported in Fig. 4. As can be seen from Fig. 3 and Fig. 4, such errors are lower than $\varepsilon = 0.05$ for each time $k = 0, 1, \ldots$. Therefore, the approximate control law $\tilde{\mathbf{k}}$ satisfies also requirement P1.3, as expected. Fig. 5 shows that both the nominal and the approximate control moves satisfy the input constraints, in fact the input signal $u(k)$ belongs to the interval $[-0.5; 0.5]$ for all $k = 0, 1, \ldots$

A significant computational speed improvement is obtained with the approximate controller. In fact, in a 2.40-GHz Intel Pentium IV with 3 GB of RAM, the mean elapsed time taken by the function `fmincon` in Matlab to solve the nominal problem (4) at each step time $k$ is equal to 0.02 s, while the mean time in evaluating the approximate control move $\tilde{\mathbf{k}}(x(k))$ at each step time $k$ is 2.5 $10^{-5}$ s.

## V. CONCLUSION

A procedure to approximate the optimal predictive control law for nonlinear systems is presented. The approximate control law is a piecewise-constant function defined over an equally-sized hyper-cube partition of the space state, such that it can be online evaluated in the presence of a sampling time significantly smaller than the one required by the on-line optimization. Performance of the approximate control law, such as closed-loop stability, fulfillment of input and state constraints and boundedness of the distance between nominal and approximate state trajectory are guaranteed. Satisfaction of such performance specifications is formulated as infinite-dimensional feasibility problems. Such problems are recasted as finite-dimensional polynomial optimization problems and their approximate solution is computed by means of recently proposed semidefinite programming relaxation techniques.



Fig. 1. Hyper-cube partition of the feasibility set $\mathcal{F}_{in}$. In each box the approximate control law $\tilde{\mathbf{k}}$ is constant.
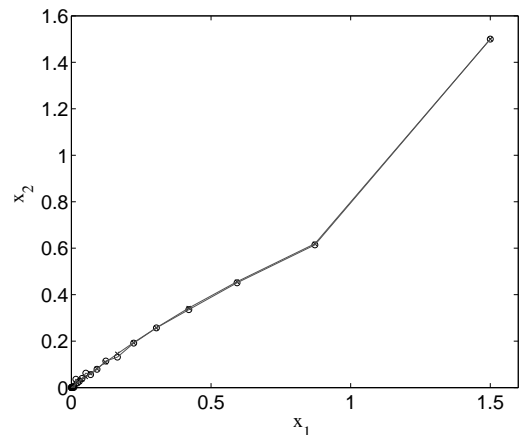


Fig. 2. Nominal state trajectory (x-marks and dashed line) and approximate state trajectory (circles 'o' and solid line).
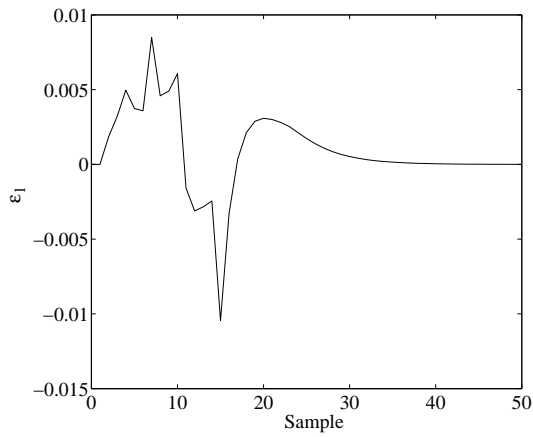
Fig. 3. Error $\varepsilon_1(k) = x_1(k) - \tilde{x}_1(k)$ between the nominal state variable $x_1(k)$ and the approximate state variable $\tilde{x}_1(k)$ at each sample time $k = 0, 1, \ldots$
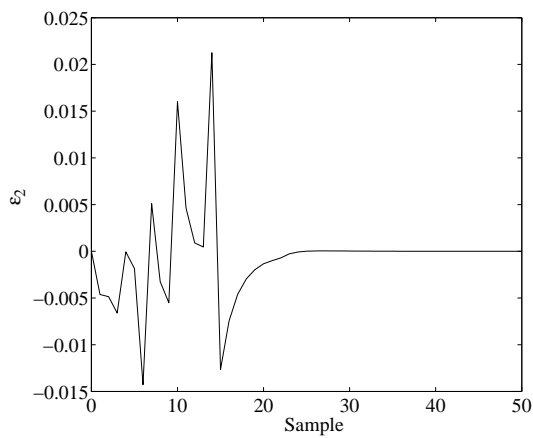


Fig. 4. Error $\varepsilon_2(k) = x_2(k) - \tilde{x}_2(k)$ between the nominal state variable $x_2(k)$ and the approximate state variable $\tilde{x}_2(k)$ at each sample time $k = 0, 1, \ldots$
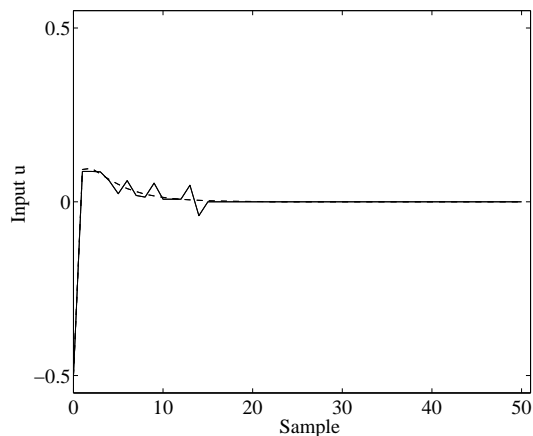


Fig. 5. Nominal control moves (dashed line) and approximate control moves (solid line)

The reported simulated example shows that the presented approximation technique can be efficiently employed to obtain a fast model predictive controller implementation.

REFERENCES

[1] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, pp. 789–814, 2000.

[2] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, pp. 3–20, 2002.

[3] M. Seron, G. Goodwin, and J. D. Doná, "Characterization of receding horizon control for constrained linear systems," *Asian Journal of Control*, vol. 5, no. 2, pp. 271–286, 2003.

[4] F. Borrelli, M. Baotić, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, no. 10, pp. 1709–1721, 2005.

[5] P. Tondel, T. Johansen, and A. Bemporad, "Computation and approximation of piecewise linear affine control laws via binary search trees," in *41$^{st}$ IEEE Conference Decision an Control*, Las Vegas, Nevada, December 2002, pp. 3144–3149.

[6] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *International Journal of Robust and Nonlinear Control*, vol. 18, pp. 816–830, 2008.

[7] R. Milman and E. J. Davison, "A fast MPC algorithm using nonfeasible active set methods," *Journal of Optimization Theory and Applications*, vol. 139, pp. 591–613, 2008.

[8] S. Richter, C. Jones, and M. Morari, "Real-time input-constrained MPC using fast gradient methods." in *Proc. of the 48th IEEE Conference on Decision and Control*, 2009, p. 7387Ű7393.

[9] J. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transaction on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[10] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[11] A. Grancharova and T. Johansen, "Explicit aprroximate model predictive control of constrained nonlinear systems with quantized input," in *L. Magni, D. M. Raimondo and F. Allgöwer, editors, Nonlinear Model Predictive Control: Towards New Challenging Applications, LNCIS*. Springer-Verlag, 2009, vol. 384, pp. 371–380.

[12] T. Johansen and A. Grancharova, "Approximate explicit constrained linear model predictive control via orthogonal search tree," *IEEE Transactions on Automatic Control*, vol. 48, no. 5, pp. 810–815, May 2003.

[13] T. Johansen, "Approximate explicit receding horizon control of constrained nonlinear systems," *Automatica*, vol. 40, pp. 293–300, 2004.

[14] A. Bemporad and C. Filippi, "An algorithm for approximate multiparametric convex programming," *Computational Optimization and Applications*, vol. 35, pp. 87–108, 2006.

[15] M. Canale, L. Fagiano, and M. Milanese, "Set membership approximation theory for fast implementation of model predictive control laws," *Automatica*, vol. 45, no. 1, pp. 45–54, 2009.

[16] ——, "Efficient model predictive control for nonlinear systems via function approximation techniques," *IEEE Transactions on Automatic Control*, vol. 55, no. 8, pp. 1911–1916, 2010.

[17] M. Vidyasagar, *Nonlinear systems analysis*. SIAM, 2002.

[18] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optimization*, vol. 11, pp. 796–817, 2001.

[19] D. Henrion and J. B. Lasserre, "Solving nonconvex optimization problems," *IEEE Control Systems Magazine*, vol. 24, no. 3, pp. 72–83, 2004.

[20] ——, "Gloptipoly: Global optimization over polynomials with Matlab and SeDuMi," *ACM Transactions Math. Soft*, vol. 29, pp. 165–194, 2003.

[21] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB Toolbox for optimization over symmetric cones," *Optim. Methods Software*, vol. 11, no. 12, pp. 625–653, 1999.