Preamble. This is a reprint of the article:

M. Schulze Darup and M. Mönnigmann. Approximate explicit NMPC with guaranteed stability ensured by a simple auxiliary controller. In *Proc. of the 2012 IEEE Multi-Conference on Systems and Control*, pp. 270–275, 2012.

The digital object identifier (DOI) of the original article is:

`10.1109/ISIC.2012.6398279`

Approximate explicit NMPC with guaranteed stability ensured by a simple auxiliary controller

Moritz Schulze Darup† *and* M. Mönnigmann†

Abstract

We investigate two methods for the calculation of suboptimal explicit solutions to nonlinear MPC problems and show that these two methods can be combined for guaranteed stability and good performance. The first method calculates an explicit piecewise constant (PWC) control law and a corresponding positively invariant set that is represented by a hyperrectangular partition in the state space. The explicit PWC law provides a suboptimal solution to the nonlinear MPC problem, but asymptotical stability of the closed-loop system can be guaranteed. A second explicit controller is constructed by solving the nonlinear MPC problem for a representative set of initial conditions and interpolating these pointwise solutions nonlinearly. Note that the PWC law provides feasible initial solutions to the nonlinear MPC problem and therefore can be used to speed up the construction of the second controller significantly. The PWC control law and the explicit nonlinear control law can be combined for guaranteed asymptotical stability (by virtue of the PWC control law) and good performance (from the nonlinear control law). We claim the hybrid controller is an interesting alternative, because its domain of attraction is typically larger than that of the nonlinear controller alone.

---

1 Introduction

Explicit model predictive control [1,2,10] may be an alternative to model predictive control if receding horizon optimization problems cannot be solved online. Exact explicit solutions can in general only be calculated for linear or piecewise linear systems, linear constraints and linear or quadratic cost functions. Approximate, or more specifically, suboptimal explicit control laws can also be calculated, however, for nonlinear systems and linear or quadratic cost functions with several approaches. For one, it is an option to approximate the nonlinear system by a piecewise linear system and to apply the exact methods that exist for this system class. Secondly, one can attempt to extend approaches for solving

† M. Schulze Darup and M. Mönnigmann are with Automatic Control and Systems Theory, Department of Mechanical Engineering, Ruhr-Universität Bochum, 44801 Bochum, Germany. E-mail: `moritz.schulzedarup@rub.de`.

multiparametric linear and quadratic programs to multiparametric nonlinear systems [5]. Thirdly, the calculation of explicit control laws is related to the calculation of positive and control invariant sets. In the present paper we explore a method of the third kind, which is an extension of an earlier approach presented by the authors [9]. See [8, 11] for other contributions of this type. We point out the differences between our previous work and the approach presented here in Sect. 3 (see last paragraph before 3.1). The relation to [8] and [11] is explained at the beginning of Sect. 5.

The paper is organized as follows. After an introduction of the system and problem class in Sect. 2, reachable sets are used in Sect. 3 to construct a piecewise constant (PWC) control law over a hyperrectangular partition of the state space. The resulting control law guarantees asymptotic stability of an equilibrium of the closed loop system for a finite subset of the state space around this equilibrium. Moreover, as stated in Sect. 4, the PWC control law provides a feasible solution for the nonlinear MPC problem for any initial condition in the constructed positively invariant set. This is exploited in Sec. 5, where the nonlinear MPC problem is solved for initial conditions chosen on a state space grid, and a nonlinear control law is constructed by interpolation of the optimal solutions for the grid points. This nonlinear control law typically provides a better performance than the PWC control law from Sect. 3, but asymptotic stability of the closed loop system can no longer be guaranteed without further measures. In Sect. 6 we show that the two control laws can be combined, and, loosely speaking, we can decide at runtime of the controller which one to select in order to retain both, the good performance of the nonlinear control law, and the asymptotical stability guarantee of the PWC one. Since this decision is made at runtime, we need not establish stability and attractivity of the nonlinear control law before runtime, but the hybrid control law inherits the domain of attraction from the PWC control law. Sections 7 and 8 present an example and state conclusions, respectively.

2 Problem statement and preliminaries

Consider a nonlinear discrete-time system of the form

$$x(k+1) = f(x(k), u(k)) \tag{1}$$

that is subject to state constraints $x \in \mathcal{X} \subseteq \mathbb{R}^n$ and control constraints $u \in \mathcal{U} \subseteq \mathbb{R}^m$.

Assumption 1: *The function f may be nonlinear and is assumed to be defined on at least $\mathcal{X} \times \mathcal{U}$, twice continuously differentiable, and maps into $\mathbb{R}^n$. The tuple $(0, 0) \in \operatorname{int}(\mathcal{X} \times \mathcal{U})$ is an equilibrium, i.e. $f(0, 0) = 0$. $\mathcal{X}$ and $\mathcal{U}$ are assumed to the compact. $\mathcal{U}$ is assumed to be convex.*

We seek an explicit control law $g(x)$ according to Def. 1, which results in the closed loop system

$$x(k+1) = f(x(k), g(x(k))). \tag{2}$$

Definition 1: *We refer to a control law $u = g(x)$ on a domain $\mathcal{T}$ as a feasible attractive stable (FAS) controller of system (2), if*

(a) $g(0) = 0$, $x \in \mathcal{T}$ implies $g(x) \in \mathcal{U}$ (feasibility w.r.t. $\mathcal{U}$),

(b) $0 \in \operatorname{int}(\mathcal{T})$, $\mathcal{T} \subseteq \mathcal{X}$ and $\mathcal{T}$ positive invariant (p.i.), i.e. $x \in \mathcal{T}$ implies $f(x, g(x)) \in \mathcal{T}$ (feasibility w.r.t. $\mathcal{X}$),

(c) $x(0) \in \mathcal{T}$ *implies* $\lim_{k\to\infty} \|x(k)\| = 0$ *(attractivity, [12]),*

(d) there exists a $\delta(\epsilon) > 0$*, such that for each* $\epsilon > 0$ *and for all* $x(0)$ *with* $\|x(0)\| < \delta(\epsilon)$ *we have* $\|x(k)\| < \epsilon$ *for all* $k \geq 0$ *(stability, [12]).*

We use the cost function (3) of the following finite horizon ($h \in \mathbb{N}$) optimal control problem (FHOCP) to measure performance.

$$J^*(x_0) = \min_{\hat{u}(0),\ldots,\hat{u}(h-1)} \varphi_0(\hat{x}(h)) + \sum_{i=0}^{h-1} l(\hat{x}(i), \hat{u}(i)) \tag{3}$$

$$\begin{aligned} \text{s.t.} \qquad \hat{x}(0) &= x_0, \\ \hat{x}(i+1) &= f(\hat{x}(i), \hat{u}(i)), \ \forall\, i = 0, \ldots, h-1, \\ (\hat{x}(i), \hat{u}(i)) &\in \mathcal{X} \times \mathcal{U}, \qquad \forall\, i = 0, \ldots, h-1, \\ \hat{x}(h) &\in \mathcal{T}_0, \end{aligned} \tag{4}$$

where $\hat{x}(i)$ refers to the predicted state at time i that results for the control sequence $\hat{u}(0), \ldots, \hat{u}(i-1)$ and $l : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ and $\varphi_0 : \mathcal{X} \to \mathbb{R}$ are positiv definite (p.d.).

Assumption 2: *There exist a control law* g_0*, a domain* $\mathcal{T}_0$ *and a cost* φ_0 *such that* g_0 *is a FAS controller on* $\mathcal{T}_0$ *and* $\varphi_0(x) - \varphi_0(f(x, g_0(x))) \geq l(x, g_0(x))$ *for all* $x \in \mathcal{T}_0$.

According to [7], assumption 2 guarantees asymptotic stability of (2) with the optimal control law

$$g^*(x) = \hat{u}^*(0), \tag{5}$$

where $\hat{u}^*(0)$ is the first element from the optimal control sequence $\hat{u}^*(0), \ldots, \hat{u}^*(h-1)$ that results from solving (3), (4) for a feasible initial condition $x_0 = x$. We denote the set of feasible states, i.e. the set of $x_0 \in \mathcal{X}$ that satisfy condition (4), by $\mathcal{F}$. It is in general not possible to calculate an exact explicit formula for $g^*(x_0)$ for all $x_0 \in \mathcal{F} \subseteq \mathcal{X}$. We present a simple but efficient method for calculating an approximation to (5) that is suboptimal but guarantees asymptotic stability of (2) on a domain $\hat{\mathcal{T}} \subseteq \mathcal{F}$.

3 Piecewise constant controller with guaranteed stability

We first ignore performance and try to find a stabilizing controller $\hat{g}(x)$ for (2) on a large domain $\hat{\mathcal{T}} = \mathcal{T}_{i^*}$. Assume a controller has been found for a certain domain, say $\mathcal{T}_{i-1}$, in step $i-1$ of our controller construction. In step i we then attempt to enlarge $\mathcal{T}_{i-1}$ by identifying a control law $\hat{g}_i(x)$ on a set $\Delta\mathcal{T}_i \not\subseteq \mathcal{T}_{i-1}$ such that $\hat{g}_i(x) \in \mathcal{U}$ and

$$f(x, \hat{g}_i(x)) \in \mathrm{int}(\mathcal{T}_{i-1}) \text{ for all } x \in \Delta\mathcal{T}_i. \tag{6}$$

This procedure can be repeated until no nonempty set $\Delta\mathcal{T}_{i^*+1}$ can be found in some step i^*+1. It remains to be answered how to choose $\Delta\mathcal{T}_i$ and $\hat{g}_i(x)$. As for $\hat{g}_i(x)$, we restrict ourselves to piecewise constant (PWC) functions, i.e. $\hat{g}_i(x) = c_i$ for all $x \in \mathcal{T}_i \setminus \mathcal{T}_{i-1}$. Subsets $\Delta\mathcal{T}_i$ will be represented by hyperrectangles.

The presented approach is related to but different from the algorithms described in [3] and [9]. In contrast to [9] and to methods for the construction of invariant sets [3], we do not use step sets here. Hence, there is no need to specify the number of steps for a recursive step set construction, rendering this algorithmic tuning parameter of the earlier approaches obsolete. Moreover, the complexity of the state space partition is reduced by avoiding step sets. As a further extension of [9], we give a concise proof of the stability of the proposed controller. Stability of the controller was already claimed but not proved in [9].

3.1 PWC control law defined on a hyperrectangle-tree

The search for the PWC control law relies on bisecting $\mathcal{X}$ into a set of hyperrectangles, which can conveniently be represented by a binary tree (cf. Figs. 1 and 2).

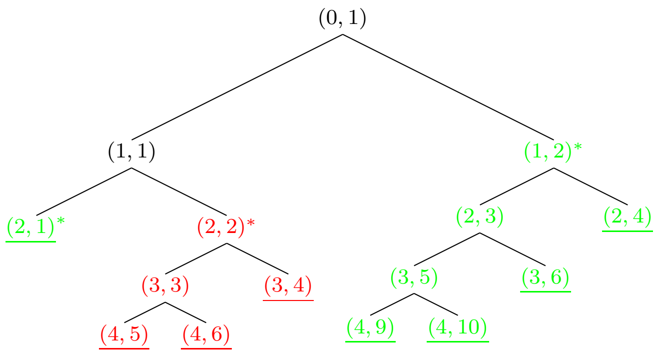

Figure 1: Sample binary tree of depth $d = 4$ with nodes (δ, β). Underlined green leaf nodes refer to hyperrectangles that are members of a sample target set $\mathcal{T}_i$ (cf. set $\hat{\mathcal{M}}$ in Sect. 3). Green nodes labeled with an asterisk constitute the most compact representation of $\mathcal{T}_i$ (cf. set $\mathcal{M}^*$ in Sect. 4). Red nodes refer to the complementary set $\mathcal{X} \setminus \mathcal{T}_i$.

Consider an arbitrary closed hyperrectangle $\mathcal{B} = [\underline{b}_1, \overline{b}_1] \times \cdots \times [\underline{b}_n, \overline{b}_n] \subset \mathbb{R}^n$. By bisection we refer to the operation that divides $\mathcal{B}$ into two closed hyperrectangles by splitting the interval of largest width $\overline{b}_i - \underline{b}_i$ of $\mathcal{B}$ into two intervals of equal length. Recursive bisection results in a hierarchy of hyperrectangles that partition $\mathcal{B}$. Figures 1 and 2 illustrate that every hyperrectangle $\boxplus_{\delta\beta}\mathcal{B}$ can uniquely be identified with the depth δ of its node in the tree and the position β of the node in its level.

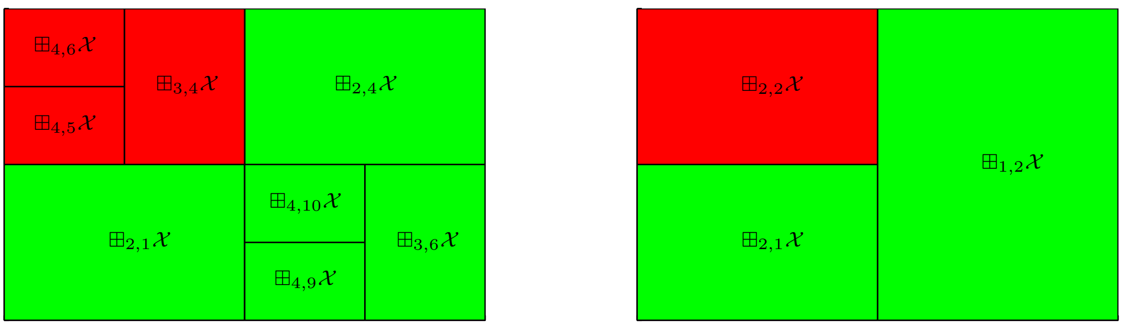

Figure 2: Hyperrectangular state space partitions that correspond to the binary tree in Fig. 1. Every hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$ uniquely corresponds to its node (δ, β). The partition on the left belongs to the leaf node representation (underlined nodes in Fig. 1). The more compact partition on the right corresponds to the nodes labeled with an asterisk in Fig. 1.

Here, we assume $\mathcal{X}$ is a hyperrectangle, i.e. $\mathcal{X} = [x_1] \times \cdots \times [x_n]$ with $[x_i] = [\underline{x}_i, \overline{x}_i] \subset \mathbb{R}$. This assumption is merely added for convenience. It is straight forward to extend our approach to compact $\mathcal{X}$ that can be represented or approximated by a union of pairwise disjoint hyperrectangles.

Starting with $\boxplus_{0,1}\mathcal{X} = \mathcal{X}$, the hyperrectangles $\boxplus_{\delta\beta}\mathcal{X}$ serve as candidates for the enlargement $\Delta\mathcal{T}_i$ of the current controller domain. Similarly, candidate control actions are selected from a finite set $\{\boxdot_1\mathcal{U}, \dots, \boxdot_r\mathcal{U}\}$ of grid points $\boxdot_j\mathcal{U}$, which we assume to be sorted in ascending order of $l(0, \boxdot_j\mathcal{U})$. While $\mathcal{X}$ has to be partitioned into hyperrectangles, it suffices to discretize $\mathcal{U}$ with a grid of points. This technical detail is important for the performance of the method; see Remark 1 below.

To check whether a hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$ belongs to the controller domain, we need to analyse sets of the form

$$\{f(x,u) \,|\, x \in \boxplus_{\delta\beta}\mathcal{X},\ u = \boxdot_j\mathcal{U}\}\,, \tag{7}$$

which we denote by $f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U})$ for short. It is in general difficult, if not impossible, to calculate sets of type (7). It suffices, however, to calculate supersets

$$\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U})) \supseteq f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U}) \tag{8}$$

instead, where the operator $\mathcal{P}$ is only required to fulfill the monotonicity property

$$\mathcal{B}_1 \subseteq \mathcal{B}_2,\ \mathcal{B}_3 \subseteq \mathcal{B}_4 \ \Rightarrow\ \mathcal{P}(f(\mathcal{B}_1, \mathcal{B}_3)) \subseteq \mathcal{P}(f(\mathcal{B}_2, \mathcal{B}_4)) \tag{9}$$

for all admissible hyperrectangles $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ and $\mathcal{B}_4$. The operator $\mathcal{P}$ can be implemented with, for example, interval arithmetic (IA) [3,6] or DC-programming [8]. We use interval arithmetic in the present paper.

Assuming that $\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U}))$ is available, the current controller domain $\mathcal{T}_{i-1}$ can be enlarged by identifying a hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$ which is not yet subset of $\mathcal{T}_{i-1}$ and for which there exists a control action $\hat{g}_{\delta\beta} = \boxdot_j\mathcal{U}$ such that

$$\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \hat{g}_{\delta\beta})) \subseteq \operatorname{int}(\mathcal{T}_{i-1}). \tag{10}$$

Algorithm 1: *Computation of PWC controller. Figure 3 illustrates the resulting PWC control law for the example discussed in Sect. 7.*

`Input`: box $\mathcal{X} = [\underline{x}, \overline{x}]$, input candidates $\boxdot_1\mathcal{U}, \dots, \boxdot_r\mathcal{U}$, terminal set $\mathcal{T}_0$ and maximal depth d

`Output`: final step $i^* = i - 1$, final domain $\hat{\mathcal{T}} = \mathcal{T}_{i^*}$, member nodes $\hat{\mathcal{M}}$ and PWC controller values $\hat{g}_{\delta\beta}$

1 set step $i = 1$, member nodes $\hat{\mathcal{M}} = \emptyset$, candidate nodes $\mathcal{C} = \{(0,1)\}$ (root node) and next candidates $\mathcal{C}^+ = \emptyset$.
2 pick (δ, β) out of candidates $\mathcal{C}$ and remove (δ, β) from $\mathcal{C}$.
3 **if** $\boxplus_{\delta\beta}\mathcal{X} \subseteq \mathcal{T}_0$ **then** add (δ, β) to members $\hat{\mathcal{M}}$.
4 **else**
5 | **for** $j = 1$ **to** r **do**
6 | | **if** $\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U})) \subseteq \operatorname{int}(\mathcal{T}_{i-1})$ **then**
7 | | | set $\hat{g}_{\delta\beta} = \boxdot_j\mathcal{U}$ and $\mathcal{T}_i = \mathcal{T}_{i-1} \cup \boxplus_{\delta\beta}\mathcal{X}$ and increase i by 1.
8 | | | add (δ, β) to $\hat{\mathcal{M}}$, add $\mathcal{C}^+$ to $\mathcal{C}$, set $\mathcal{C}^+ = \emptyset$ and **goto** line 2.
9 | **if** $\delta < d$ **then**
10 | | bisect, i.e. add nodes $(\delta + 1, 2\beta)$ and $(\delta + 1, 2\beta + 1)$ to $\mathcal{C}$.
11 | **else** add (δ, β) to next candidates $\mathcal{C}^+$.
12 **if** $\mathcal{C}$ is empty **then** terminate algorithm
else goto line 2.

Algorithm 1 formalizes the procedure for calculating the desired PWC control law and the associated domain. In Alg. 1, $\mathcal{T}_i$ denotes the controller domain after step i, and $\hat{g}_{\delta\beta}$

is the constant value of the control law on the hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$ associated with node (δ, β). The symbol $\hat{\mathcal{M}}$ denotes the set of all hyperrectangles for which a PWC control has been found or which are in $\mathcal{T}_0$. Finally, the sets $\mathcal{C}$ and $\mathcal{C}^+$ contain candidate nodes which will be investigated during the current and next step, respectively. Algorithm 1 results in the control law

$$\hat{g}(x) = \begin{cases} g_0(x) & \text{if } x \in \mathcal{T}_0 \\ \hat{g}_{\delta\beta} & \text{if } x \notin \mathcal{T}_0,\ x \in \boxplus_{\delta\beta}\mathcal{X},\ (\delta,\beta) \in \hat{\mathcal{M}}, \end{cases} \tag{11}$$

with the following properties.

Proposition 1: *The PWC controller (11) is a FAS controller on $\hat{\mathcal{T}}$ according to Def. 1.*

Proof. Constraint satisfaction and positive invariance hold by construction. Attractivity is guaranteed because $\lim_{k\to\infty} \|x(k)\| = 0$ for all $x(0) \in \mathcal{T}_0$ according to assumption 2, and since there exists a $i \le i^* < \infty$ such that $x(i) \in \mathcal{T}_0$ for all $x(0) \in \hat{\mathcal{T}} \setminus \mathcal{T}_0$. Finally, stability holds, because we include the stable controller $g_0(x)$ on the domain $\mathcal{T}_0$. To verify this, first note that Ass. 2 implies the existence of a function $\delta_0(\epsilon)$, which guarantees stability of g_0 on $\mathcal{T}_0$ according to Def. 1. We may choose $\delta(\epsilon) = \min(\delta_0(\epsilon), \overline{\delta}) > 0$ in order to guarantee stability of (2) using controller (11), where $\overline{\delta} > 0$ is selected such that for all $\|x\|_2 \le \overline{\delta}$ we have $x \in \mathcal{T}_0$. ■

Algorithm 1 terminates the search for an appropriate control action whenever condition (10) is satisfied for the first time (cf. lines 6–8 in Alg. 1). Performance requirements could be included by analysing *all* input candidates $\boxdot_1\mathcal{U}, \ldots, \boxdot_r\mathcal{U}$ instead of stopping after the first suitable one has been identified. We found, however, that this obvious extension considerably increases the computational effort without significantly improving the closed-loop performance (cf. [9]).

PWC functions are not an obvious choice for the construction of a controller and a feasible domain. Intuitively, one would expect a larger feasible set to result for more flexible functions. Comments on this matter are collected in Remark 1 for ease of reference.

Remark 1: *The inclusion test (10) is carried out in line 6 of Alg. 1. Together with* (8) *this yields $f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U}) \subseteq \operatorname{int}(\mathcal{T}_{i-1})$, which is the implementation of* (6) *for $\Delta\mathcal{T}_i = \boxplus_{\delta\beta}\mathcal{X}$ and the constant control on $\boxplus_{\delta\beta}\mathcal{X}$*

$$\hat{g}(x) = \boxdot_j\mathcal{U} \text{ for all } x \in \boxplus_{\delta\beta}\mathcal{X}. \tag{12}$$

If we replace the constant control law (12) *by a more flexible function $\tilde{g}$, we will in general have to replace $\hat{g}_{\delta\beta} = \boxdot_j\mathcal{U}$ in* (10) *by the overestimation $\mathcal{P}(\tilde{g}(\boxplus_{\delta\beta}\mathcal{X})) \supseteq \tilde{g}(\boxplus_{\delta\beta}\mathcal{X})$. The inclusion test* (10) *therefore becomes*

$$\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \mathcal{P}(\tilde{g}(\boxplus_{\delta\beta}\mathcal{X})))) \subseteq \operatorname{int}(\mathcal{T}_{i-1}). \tag{13}$$

Applying the monotonicity property (8) *to $\mathcal{P}(\tilde{g}(\boxplus_{\delta\beta}\mathcal{X})) \supseteq \tilde{g}(\boxplus_{\delta\beta}\mathcal{X}) \supseteq \boxdot_j\mathcal{U}$ yields*

$$\mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \mathcal{P}(\tilde{g}(\boxplus_{\delta\beta}\mathcal{X})))) \supseteq \mathcal{P}(f(\boxplus_{\delta\beta}\mathcal{X}, \boxdot_j\mathcal{U})), \tag{14}$$

where we assume $\boxdot_j\mathcal{U} \in \hat{g}(\boxplus_{\delta\beta}\mathcal{X})$. The left hand side of (14) *typically is a very conservative overestimation of the right hand side. As a consequence,* (10) *can often be shown to hold for $\boxplus_{\delta\beta}\mathcal{X}$ for which* (13) *does not hold due to its greater conservatism. For this reason, a PWC controller typically results in a greater feasible area than a more flexible controller. Note this remark also applies if a different technique than IA is used.*

Figure 3 illustrates the PWC control law for the example discussed in Sect. 7. As a preparation to Sect. 4 we state an appropriate cost function $\hat{\varphi}$ for the computed PWC controller (11):

$$\hat{\varphi}(x) = \begin{cases} \varphi_0(x) & x \in \mathcal{T}_0 \\ l(x, \hat{g}(x)) + \hat{\varphi}(f(x, \hat{g}(x))) & x \in \hat{\mathcal{T}} \setminus \mathcal{T}_0. \end{cases} \tag{15}$$

This recursion yields a finite result for all $x_0 \in \hat{\mathcal{T}}$, since the set $\mathcal{T}_0$ is reached after at most i^* steps.

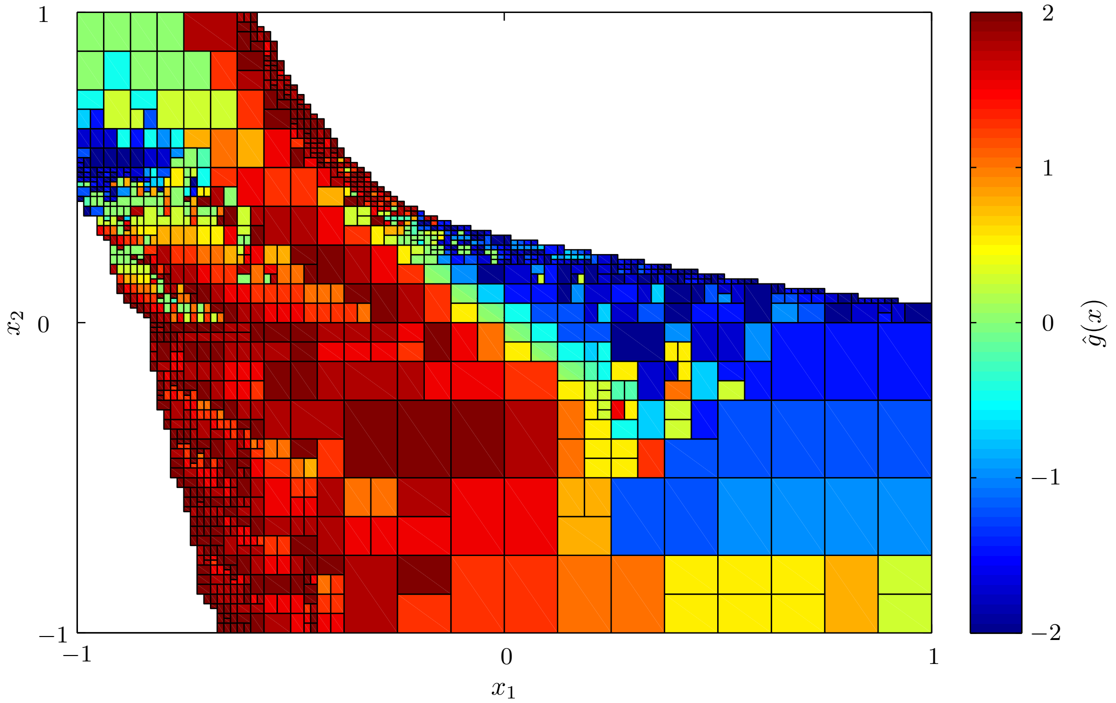

Figure 3: PWC control law $\hat{g}(x)$ defined on 1214 hyperrectangles for the example from Sect. 7. Colors indicate the value of u as shown in the bar on the right hand side.

4 Compact representation of the feasible set

We can use the PWC controller from Prop. 1 to generate feasible solutions to the FHOCP (3), (4) by substituting the results $\hat{\mathcal{T}}$, $\hat{g}$ and $\hat{\varphi}$ from the PWC controller construction for $\mathcal{T}_0$, controller g_0 and cost φ_0, respectively. Then for every $x_0 = \hat{x}(0) \in \hat{\mathcal{T}}$, the trajectory $\hat{x}(i+1) = f(\hat{x}(i), \hat{u}(i))$, $\hat{u}(i) = \hat{g}(\hat{x}(i))$ with $i = 0, \dots, h-1$ is a feasible solution to the FHOCP (3), (4) for any $h \in \mathbb{N}$. Since its construction involved the conservative operator $\mathcal{P}$, the domain $\hat{\mathcal{T}}$ of the PWC controller from Sect. 3 is an underestimation of the feasible set of problem (3), (4), i.e. $\hat{\mathcal{T}} \subseteq \mathcal{F}$.

We anticipate that a more compact representation of the feasible set is useful to construct an interpolated nonlinear control law. We can simplify the representation of $\hat{\mathcal{T}}$ by merging state space hyperrectangles based on the following observation: If the hyperrectangles associated with two sibling nodes $(\delta+1, 2\beta)$ and $(\delta+1, 2\beta+1)$ are part of $\hat{\mathcal{T}}$, then the parent hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$ is also a member of $\hat{\mathcal{T}}$. Formally, this aggregation of nodes corresponds to solving the optimization problem

$$\mathcal{M}^* = \arg\min_{\mathcal{M}} |\mathcal{M}| \quad \text{s.t.} \bigcup_{(\delta,\beta)\in\mathcal{M}} \boxplus_{\delta\beta}\mathcal{X} = \bigcup_{(\delta,\beta)\in\hat{\mathcal{M}}} \boxplus_{\delta\beta}\mathcal{X}. \tag{16}$$

The resulting equivalent representation of $\hat{\mathcal{T}}$ is more compact since $|\mathcal{M}^*| \leq |\hat{\mathcal{M}}|$. Figure 4 visualizes the aggregation that results from Eq. (16) for the example from Sect. 7.

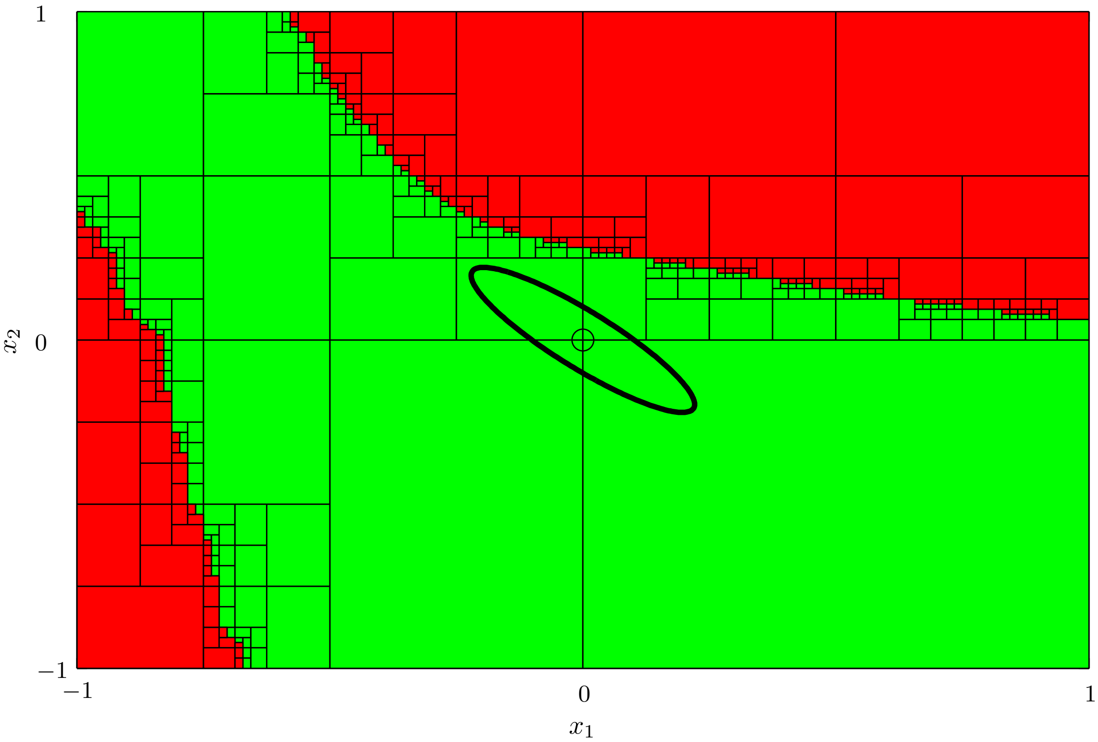

Figure 4: Underestimation of the feasible set (green) of the FHOCP (3), (4) for the example from Sect. 7. The PWC controller domain is the same one as in Fig. 3. $\mathcal{M}^*$ contains 207 nodes. The black ellipse represent the terminal set $\mathcal{T}_0$.

5 Approximation of the optimal control law

An approximation of the optimal control law $g^*(x)$ on $\hat{\mathcal{T}}$ is constructed using an established idea (see e.g. [5], [8] or [11]): We solve the FHOCP for some points $x_0 \in \hat{\mathcal{T}}$ on a grid and interpolate solutions based on the solution at the grid points. However, in contrast to [8] and [11], we benefit here from the fact that the PWC control law from Prop. 1 provides a feasible solution to the FHOCP (using $\hat{\mathcal{T}}$, $\hat{g}$ and $\hat{\varphi}$), which can be used as a feasible starting point for (3), (4). This proves to be very helpful, since (3), (4) is generally nonconvex and we may fail to solve it if no good starting guess is known.

In principle, the presented approach is not restricted with respect to the choice of supporting points and interpolation functions. We may, for example, think of vertices of simplices as supporting points and piecewise affine interpolation functions. Furthermore, nearest neighbor, natural neighbor or any other multivariate interpolation scheme could be applied. Here, we use a n-linear interpolation[1] (NLI) on the existing hyperrectangle tree, which is similar to the approach suggested in [8, 11].

The procedure is as follows. Let $\mathcal{M}^*$ be the condensed leaf node set described in Sect. 4. For any hyperrectangle $\boxplus_{\delta\beta}\mathcal{X}$, $(\delta, \beta) \in \mathcal{M}^*$, we solve the FHOCP (3), (4) for the vertices

$$\{v_1, \ldots, v_{2^n}\} = \mathrm{extr}(\boxplus_{\delta\beta}\mathcal{X}).$$

This yields the optimal controls $u^*(v_j)$ at the vertices. The bilinear interpolation function on the hyperrectangle $\boxplus_{\delta\beta}\mathcal{X} = [\underline{x}, \overline{x}]$ with $\underline{x} \in \mathbb{R}^n$, $\overline{x} \in \mathbb{R}^n$ can then be defined as

$$\tilde{g}_{\delta\beta}(x) = \sum_{j=1}^{2^n} u^*(v_j) \prod_{i=1}^{n} \frac{1}{\overline{x}_i - \underline{x}_i} \begin{cases} x_i - \underline{x}_i & \text{if } (v_j)_i = \overline{x}_i \\ \overline{x}_i - x_i & \text{if } (v_j)_i = \underline{x}_i \end{cases}.$$

We repeat this procedure for all other $(\delta, \beta) \in \mathcal{M}^*$. Note that the FHOCP associated with a vertex is only solved once for each vertex v_j. Further note that the resulting controller is not guaranteed to be asymptotically stable without further measures. However, control constraints are satisfied due to convexity of $\mathcal{U}$, since the NLI control law represents a convex combination of the optimal control actions at the vertices v_j, and since $u^*(v_j) \in \mathcal{U}$ holds by construction.

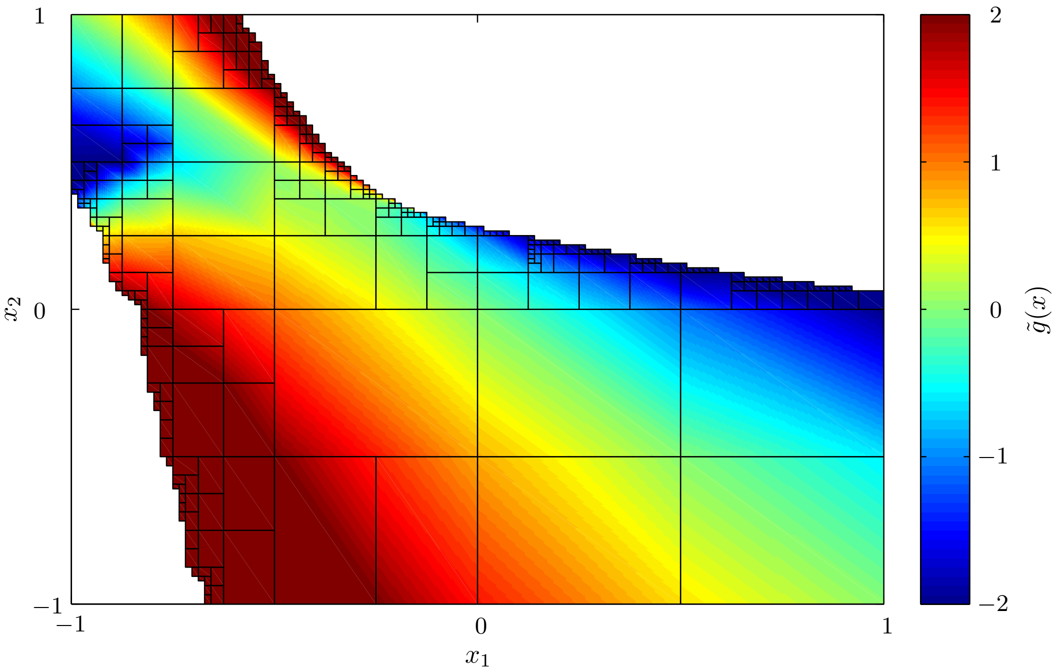

Figure 5: NLI controller $\tilde{g}(x)$ for the example from Sect. 7. The control law is defined based on the optimal control actions of the 505 vertices of the shown 282 hyperrectangles. Colors indicate the value of u as shown in the bar on the right hand side.

5.1 Refinement of the interpolated control law

The procedure stated above provides an NLI control law $\tilde{g}(x)$, whose level of detail depends on the structure of $\mathcal{M}^*$. Obviously, it would be more reasonable to link the refinement of

[1] Special cases of n-linear interpolations are the bilinear interpolation for $n = 2$ and the trilinear interpolation for $n = 3$. Except for $n = 1$, the resulting interpolation is nonlinear.

$\tilde{g}(x)$ to features of the optimal control law $g^*(x)$ such as its slopes and curvatures. In this context, we check whether the optimal control actions at the supporting points associated with the next bisection step are in agreement with the interpolated values for these points or not. In case of disagreement, we execute the bisection; otherwise we keep the current representation (a similar heuristic is used in ([11])). Formally, we have to analyze those vertices of the hyperrectangle of either child node (e.g. the right node), that are not vertices of the current node hyperrectangle, i.e.

$$\{v'_1, \ldots, v'_{2^{n-1}}\} = \mathrm{extr}(\boxplus_{\delta+1,2\,\beta}\mathcal{X}) \setminus \{v_1, \ldots, v_{2^n}\}.$$

If the condition $\|\tilde{g}_{\delta\beta}(v'_j) - u^*(v'_j)\|_\infty > \epsilon$ holds for at least one of the 2^{n-1} bisection points v'_j, then a bisection is necessary. Note that a bisection modifies the structure of the leaf node set $\tilde{\mathcal{M}}$ associated to the approximation $\tilde{g}(x)$, i.e. the node (δ, β) is replaced by the child nodes $(\delta+1, 2\,\beta-1)$ and $(\delta+1, 2\,\beta)$. The final dual mode control law $\tilde{g}(x)$ reads

$$\tilde{g}(x) = \begin{cases} g_0(x) & \text{if } x \in \mathcal{T}_0 \\ \tilde{g}_{\delta\beta}(x) & \text{if } x \notin \mathcal{T}_0,\ x \in \boxplus_{\delta\beta}\mathcal{X},\ (\delta,\beta) \in \tilde{\mathcal{M}}. \end{cases} \tag{17}$$

See Fig. 5 for an illustration.

6 Combining performance and stability

The last three sections dealt with the calculation of a suboptimal PWC controller with guaranteed stability and the approximation of the optimal control law in terms by a NLI controller. It remains to compare and combine these two controllers for guaranteed stability and their best possible performance. While $\hat{g}(x)$ guarantees stability but was constructed without consideration of its performance, $\tilde{g}(x)$ provides better performance without guaranteed stability. Proposition 2 states an efficient combination of the two controllers.

Proposition 2: *The control law*

$$g(x) = \begin{cases} \tilde{g}(x) & \text{if } \tilde{\varphi}(x) \le \hat{\varphi}(x) \\ \hat{g}(x) & \text{otherwise} \end{cases} \tag{18}$$

with

$$\tilde{\varphi}(x) = \begin{cases} \varphi_0(x) & x \in \mathcal{T}_0 \\ l(x, \tilde{g}(x)) + \tilde{\varphi}(f(x, \tilde{g}(x))) & x \in \hat{\mathcal{T}} \setminus \mathcal{T}_0 \\ \infty & x \notin \hat{\mathcal{T}} \end{cases} \tag{19}$$

is a FAS controller on $\hat{\mathcal{T}}$ according to Def. 1.

Proof. Stability follows for the same reasons as in the proof of Prop. 1, i.e. because the asymptotically stable control law $g_0(x)$ is applied on the domain $\mathcal{T}_0$. The PWC controller $\hat{g}(x)$ is feasible and attractive according to Prop. 1. It remains to prove that $g(x)$ is feasible and attractive, if the NLI controller $\tilde{g}(x)$ is selected in (18). Selecting $\tilde{g}(x)$ implies $\tilde{\varphi}(x) \le \hat{\varphi}(x)$ by definition. Since $\hat{\varphi}(x) < \infty$ for all $x \in \hat{\mathcal{T}}$ according to the comment below (15), this implies $\tilde{\varphi}(x) < \infty$ for all $x \in \hat{\mathcal{T}}$. By its definition (19) $\tilde{\varphi}(x)$ can only be finite, however, if the trajectory that results from $\tilde{g}(x)$ remains in $\hat{\mathcal{T}}$ for all times and is driven into $\mathcal{T}_0 \subseteq \hat{\mathcal{T}}$ in a finite time. This implies feasibility and attractivity. ■

7 Numerical example

We apply the proposed approach to the nonlinear system $x(k+1) = f(x(k), u(k))$ with

$$\begin{aligned} f_1(x,u) &= x_1 + 0.1\,x_2 + 0.1\,(0.5 + 0.5\,x_1)\,u \\ f_2(x,u) &= x_2 + 0.1\,x_1 + 0.1\,(0.5 - 2.0\,x_2)\,u \end{aligned}$$

where $\mathcal{X} = [-1,1] \times [-1,1]$ and $\mathcal{U} = [-2,2]$. The system is a discrete-time variant[2] of the continuous time system treated in [3], [4] and [11]. Obviously, $(0,0)$ is an equilibrium. We claim without proof that $g_0(x) = -K\,x$ with $K = \begin{pmatrix} 1.9198 & 1.9198 \end{pmatrix}$ is a FAS control law according to Def. 1 on $\mathcal{T}_0 = \{x \in \mathbb{R}^n \,|\, \|x\|_P^2 \le \alpha^2\}$ with

$$P = \begin{pmatrix} 5.9353 & 5.2774 \\ 5.2774 & 5.9353 \end{pmatrix} \quad \text{and} \quad \alpha = 0.2462.$$

We choose the prediction horizon $h = 15$, stage cost $l(x,u) = \|x\|_Q^2 + \|u\|_R^2$ with

$$Q = \begin{pmatrix} 0.05 & 0.00 \\ 0.00 & 0.05 \end{pmatrix} \quad \text{and} \quad R = 0.1$$

and the terminal cost $\varphi_0(x) = \|x\|_P^2$. The maximal depth of the bisection tree in Alg. 1 is set to $d = 14$. Furthermore, we define $r = 17$ input candidates with $\boxdot_j\mathcal{U} = -2+(j-1)\cdot 0.25$ and $j = 1,\ldots,r$. We set the tolerance to $\epsilon = 0.1$ when computing the NLI controller.

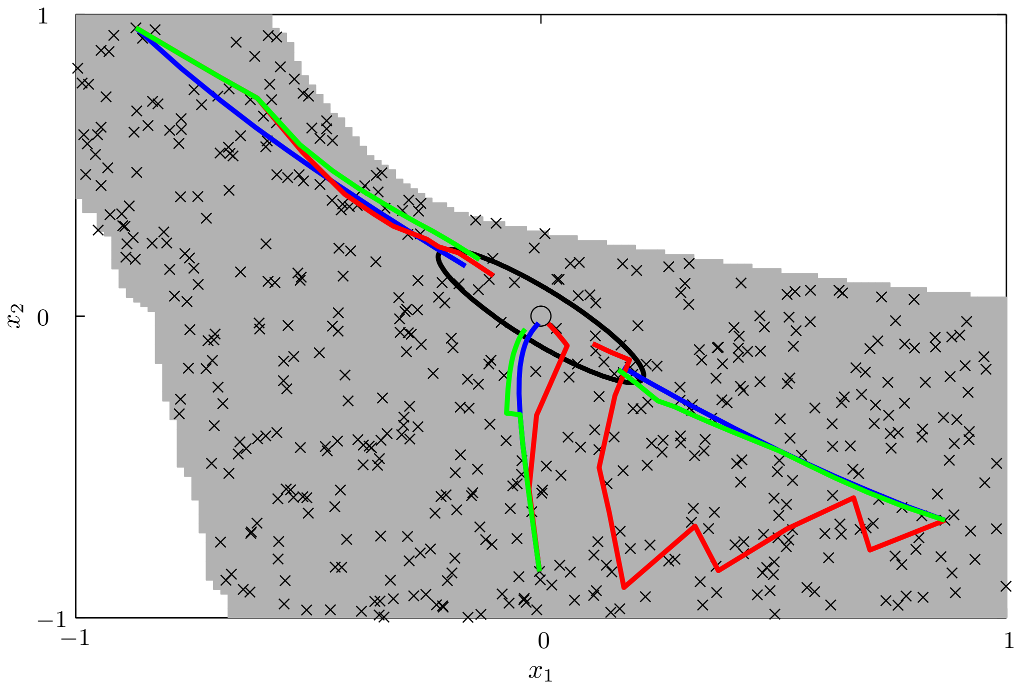

Figure 6: Crosses mark random initial states $x_0 \in \hat{\mathcal{T}}$. Red and trajectories result from the application of the PWC controller $\hat{g}(x)$ and the combined controller from Sect. 6, respectively. Blue trajectories are (locally) optimal solutions. Each trajectory consists of 15 steps. Black ellipse marks $\mathcal{T}_0$.

[2] An Euler discretization with sample time $\Delta t = 0.1\,\mathrm{s}$ was applied to the continuous system presented in [4].

Figures 3, 4 and 5 visualize the resulting PWC controller, the compact representation of the feasible set, and the NLI control law, respectively. We compare the combined controller and the PWC controller to the optimal solution that results from solving (3), (4) for 500 randomly generated initial conditions $x_0 \in \hat{\mathcal{T}}$ shown in Fig. 6. On average cost function values of the combined controller are 4% higher than those of the optimal solution. In contrast, the PWC controller results in an average increase of 103%. In total we calculated $500 \cdot h = 7500$ propagation steps. In 100 out of these 7500 cases, the PWC controller is used instead of the NLI controller to guarantee asymptotic stability.

The feasible set $\hat{\mathcal{T}}$ that results from our method is about 11% larger than the one that results with a nonlinear controller alone [11] using the same bisection depth. We need 505 coefficients in the nonlinear controller (581 in [11]).

8 Conclusions

We presented a method for the computation of suboptimal explicit piecewise constant (PWC) control laws for nonlinear discrete-time systems. While suboptimal, the PWC control law provides a feasible solution to the FHOCP, which can be used to calculate an explicit expression for a nonlinear approximation of the optimal control law. We then combine the PWC control law and the nonlinear control law to obtain both guaranteed stability with the former and good performance with the latter.

The approach was illustrated with a benchmark example [3, 4, 11]. Numerical experiments show that the combined controller results in a suboptimality of about 4% for this example. The PWC control is chosen in very few cases ($< 2\%$) by the combined controller. We were able to increase the area of the feasible set by about 11% compared to previous approaches.

Future work has to address the application of the method to systems of higher dimension and technical relevant processes. Due to the exponential cost (with respect to n) of bisection algorithms, the procedure is naturally not suitable for large-scale systems. However, the authors believe that systems up to dimension $n = 5$ can benefit from the results of the presented method.

References

[1] A. Bemporad, F. Borrelli, and M. Morari. Model predictive control based on linear programming - The explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985, 2002.

[2] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[3] J. M. Bravo, D. Limon, T. Alamo, and E. F. Camacho. On the computation of invariant sets for constrained nonlinear systems: An interval arithmetic approach. *Automatica*, 41(9):1583–1589, 2005.

[4] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[5] T. A. Johansen. Approximate explicit receding horizon control of constrained nonlinear systems. *Automatica*, 40:293–300, 2004.

[6] R. B. Kearfott. *Rigorous Global Search: Continuous Problems.* Kluwer Academic Publishers, 1996.

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

[8] D. M. Raimondo, S. Riverso, C. N. Jones, and M. Morari. A robust explicit nonlinear MPC controller with input-to-state stability guarantees. In *Proc. of 18th IFAC World Congress*, 2011.

[9] M. Schulze Darup and M. Mönnigmann. Explicit feasibil initialization for nonlinear MPC with guaranteed stability. In *Proc. of 50th Conference on Decision and Control*, 2011.

[10] M. M. Seron, G. C. Goodwin, and J. A. DeDona. Finitely parameterised implementation of receding horizon control for constrained linear systems. In *Proc. of the 2002 American Control Conference*, 2002.

[11] S. Summers, D. M. Raimondo, C. N. Jones, J. Lygeros, and M. Morari. Fast explicit nonlinear model predictive control via multiresolution function approximation with guaranteed stability. In *Proc. of 8th IFAC Symposium on Nonlinear Control Systems*, 2010.

[12] M. Vidyasagar. *Nonlinear System Analysis.* Society for Industrial Mathematics, 2002.