Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Process Control 15 (2005) 149–158

JOURNAL OF PROCESS CONTROL

www.elsevier.com/locate/jprocont

Efficient implementation of constrained min–max model predictive control with bounded uncertainties: a vertex rejection approach

T. Alamo, D.R. Ramírez *, E.F. Camacho

Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, Sevilla 41092, Spain

Received 6 December 2003; revised 17 May 2004; accepted 4 June 2004

Abstract

Min–Max Model Predictive Control (MMMPC) is one of the strategies used to control plants subject to bounded additive uncertainties. The implementation of MMMPC suffers a large computational burden due to the NP-hard optimization problem that has to be solved at every sampling time. This paper shows how to overcome this by transforming the original problem into a reduced min–max problem in which the number of extreme uncertainty realizations to be considered is significantly lowered. Thus, the solution is much simpler. In this way, the range of processes to which MMMPC can be applied is considerably broadened. A simulation example is given in the paper.

Keywords: Predictive control; Minimax techniques; Uncertain linear systems; Process control

1. Introduction

Mathematical models, especially control models which have to be kept simple, can only describe the dynamics of a process in an approximate way. There are different approaches for modelling uncertainties mainly depending on the type of technique used for designing the controllers. The approach considered here is that of global uncertainties [5]. In this way, uncertainties will be considered to affect the 1-step ahead prediction equation, i.e. the uncertainties will affect the prediction capability of the model.

Min–max control techniques have a great computational burden in common [21,10,19] which limits the range of processes to which they can be applied. Furthermore, the computational burden is even greater when hard constraints are taken into account. Few applications of Min–Max MPC can be found in the literature, even for the unconstrained case [9,4]. For fast dynamics the min–max problem cannot be solved numerically, and approximate solutions have to be used [14]. However, these techniques impose great rigidity in the controller parameters, as well as a certain degree of approximation error.

Recently, the MMMPC control law, traditionally regarded as highly nonlinear, has proven to be piecewise affine when a quadratic [15] or 1-norm based criterion [3,8] is used as the cost function. With these results, together with those obtained when multiparametric mathematical programming is applied [3], explicit forms of the control law can be built. However, the number of regions in which the state space has to be partitioned grows with the prediction horizon in a combinatorial explosion. Thus, storage requirements and searching time for the appropiate region can be very high for practical values of the prediction and control horizons.

[*] Corresponding author. Tel.: +34 954 487 347; fax: +34 954 487 340.

E-mail addresses: alamo@cartuja.us.es (T. Alamo), danirr@cartuja.us.es (D.R. Ramírez), eduardo@cartuja.us.es (E.F. Camacho).

0959-1524/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jprocont.2004.06.003

A search tree strategy has been proposed to reduce the searching time in the MPC context [6,20]. If the process model or the controller tuning parameters change, however, the computation of the regions has to be done again. This field continues evolving and new and more elaborate robust predictive controllers based on multiparametric programming have appeared in [17,18].

This paper shows a way of implementing constrained MMMPC that requires only a fraction of the time required by usual min–max solvers. The method is based on transforming the original min–max problem into a reduced min–max problem whose solution is much simpler. Thus, for many processes in which time constants are measured in seconds or minutes, the reduced min–max problem can be solved on line using standard numerical algorithms. In this paper it is presented a way to obtain a reduced min–max problem from a candidate solution (e.g. the solution of the min–max problem solved in the previous sampling time). This reduced problem can be solved in a fraction of the time required by the original problem.

The paper is organized as follows: Section 2 presents the standard constrained Min–Max MPC with bounded global uncertainties algorithm. The efficient implementation strategy is introduced in Section 3. The procedure to get a reduced min–max problem equivalent to the original one is presented in Section 4 and illustrated in Section 5. Finally, Section 6 presents conclusions and questions to be addressed in future works.

2. Min–Max MPC with bounded global uncertainties

The objective of MPC control is to compute the future control sequence $u(t), u(t+1), \ldots, u(t+N_u-1)$ in such a way that the optimal j-step ahead predictions of the process output $y(t+j|t)$ are driven close to the set point sequence $r(t), r(t+1), \ldots, r(t+N-1)$ for the prediction horizon. A cost function J is used to indicate how well the process follows the desired trajectory. That cost function depends on the setpoint, process state, present and future control signals and uncertainties.

When bounded uncertainties are considered explicitly, it would seem that more robust control is obtained if the controller has tried to minimize the objective function for the worst situation. Furthermore, if hard constraints have to be taken into account the control sequence will be computed solving the following min–max problem:

$$\mathbf{u}(x) = \arg \min_{\mathbf{u}\in\Gamma(x)} \max_{\mathbf{w}\in W} J(\mathbf{w},\mathbf{u},x) \tag{1}$$

where $\mathbf{w}$ represents the sequence of future uncertainties, x the process state and $\Gamma(x)$ usually is a convex set which can be described as $\{\mathbf{u}\colon H\mathbf{u} \leqslant Sx + d\}$. On the other hand, $W = \{\mathbf{w} \in \mathbb{R}^L : \underline{\theta} \leqslant \theta \leqslant \overline{\theta}\}$, where L is usually $N \times \dim(y)$ in an input–output description or $N \times \dim(x)$ in a state-space approach. Note that a polytopic terminal constraint devised to provide robust stability can also be included within the constraints $H\mathbf{u} \leqslant Sx + d$ (see [7] and references therein). Furthermore, as it is shown in Section 5, the effect of the uncertainties can be easily taken into account in the output constraints. Thus, it is ensured that for all uncertainties values within the bounds, the constraints would be satisfied. The cost function of the resulting min–max problem can be written as

$$J^s(x) = \min_{\mathbf{u}\in\Gamma(x)} \max_{\mathbf{w}\in W} J(\mathbf{w},\mathbf{u},x) = \min_{\mathbf{u}\in\Gamma(x)} J^*(\mathbf{u},x)$$

with

$$J^*(\mathbf{u},x) = \max_{\mathbf{w}\in W} J(\mathbf{w},\mathbf{u},x) \tag{2}$$

The properties of the min–max problem depend on the structure and properties of the cost function $J(\mathbf{w}, \mathbf{u}, x)$ which in turn is closely related to the model structure used to predict the future evolution of the process state. MPC and Min–Max MPC based on linear prediction models can be formulated either in input–output description, state space or convolution models (i.e., FIR and finite step response models) [5]. The results presented in this paper can be applied to that descriptions provided that two conditions are fulfilled:

- The prediction equation is an affine function of process uncertainties, inputs and state, i.e.

$$\xi = G_u\mathbf{u} + G_w\mathbf{w} + F_x x(t) \tag{3}$$

  where ξ can be either the predictions of process state or output over the prediction horizon.
- The cost function is a quadratic function of the predictions and $\mathbf{u}$.

These conditions hold when the bounded additive uncertainties approach is used. In this approach the way to model the uncertainties is to assume that all modelling errors are globalized in a vector of parameters, such that the plant can be described by the following family of models:

$$\hat{x}(t+1) = \hat{f}(x(t), u(t)) + w(t) \tag{4}$$

In [5] it is shown that global uncertainties can be related to other types of uncertainties. The conditions hold for input–output transfer matrices models (CARIMA or CARMA), state-space and finite step or impulse response [5]. In the case of transfer matrices models, the state can be considered to be formed by the present values of process outputs and finite series of the input and output signals. For finite step or impulse models, the state can be considered to be formed by the present value of process output and finite series (normally much longer than in transfer matrices models) of past inputs.

Note that the system to be controlled can be either a linear open-loop system or a linear closed-loop system (e.g. a linear system with an inner linear feedback gain). The latter case is preferred whenever the conservativeness of the open-loop formulation is a concern.

In the case of transfer function models it is common to use a linear prediction model like

$$\widetilde{A}(z^{-1})y(t) = z^{-d}B(z^{-1})\Delta u(t-1) + w(t) \quad \underline{w} \leqslant w(t) \leqslant \overline{w} \tag{5}$$

with $\widetilde{A}(z^{-1}) = \Delta A(z^{-1})$, $\Delta = 1 - z^{-1}$ being $y(t)$ and $u(t)$ the output and control sequence of the plant. In this case, the vector of decision variables in problem (1) would be composed of the future increments of $u(t)$ over the control horizon. Note that in this prediction model the error concept present in CARIMA models (commonly used in GPC) is extended to incorporate the effect of modelling uncertainties and disturbances.

State-space models using the bounded additive uncertainties approach can be formulated as

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + Dw(t+1) \\ y(t) &= Cx(t) \end{aligned} \tag{6}$$

with $w(t) \in W$.

An usual form of $J(\mathbf{w}, \mathbf{u}, x)$ is a quadratic criterion given by

$$\begin{aligned} J(\mathbf{w}, \mathbf{u}, x) = {} & \sum_{j=1}^{N}(y(t+j \mid t) - r(t+j))^2 \\ & + \lambda \sum_{j=1}^{N_u}(\Delta u(t+j-1))^2 \end{aligned} \tag{7}$$

where N denotes the cost horizon, N_u is the control horizon, $\lambda > 0$ is the control effort penalty factor, $r(t+j)$ is the set point value for time $t+j$ and $y(t+j|t)$ is the prediction of the output made at time t for $t+j$ taking into account the disturbance $\mathbf{w}$. On the other hand, for the state-space approach:

$$\begin{aligned} J(\mathbf{w}, \mathbf{u}, x) = {} & \sum_{j=1}^{N} x(t+j \mid t)^{\mathrm{T}} Q_j x(t+j \mid t) \\ & + \sum_{j=1}^{N_u} u(t+j-1)^{\mathrm{T}} R_j u(t+j-1) \end{aligned} \tag{8}$$

where $x(t+j|t)$ is the prediction of the state for $t+j$ taking into account $\mathbf{w}$ and $Q_j = Q_j^{\mathrm{T}} \geqslant 0$ and $R_j = R_j^{\mathrm{T}} > 0$ are used as weighting parameters. Note that this cost function allows us the inclusion of a quadratic terminal cost $x(t+N)^{\mathrm{T}}Px(t+N)$ by making $Q_N = P$. This terminal cost along with a terminal region constraint are the ingredients of the stabilizing strategies used in most predictive controllers [12,1].

Without loss of generality suppose a zero reference. It can be seen that both cost functions can be rewritten as

$$\begin{aligned} J(\mathbf{w}, \mathbf{u}, x) = {} & \mathbf{u}^{\mathrm{T}}M\mathbf{u} + 2(Nx + n(\mathbf{w}))^{\mathrm{T}}\mathbf{u} + x^{\mathrm{T}}Cx \\ & + 2d(\mathbf{w})^{\mathrm{T}}x + h(\mathbf{w}) \end{aligned} \tag{9}$$

Recall the prediction equation (3). In the case of (7) the matrices of (9) are $M = G_u^{\mathrm{T}}G_u + \lambda I$, $C = F_x^{\mathrm{T}}F_x$, $N = G_u^{\mathrm{T}}F_x$, $n(\mathbf{w}) = G_u^{\mathrm{T}}G_w\mathbf{w}$, $d(\mathbf{w}) = F_x^{\mathrm{T}}G_w\mathbf{w}$, $h(\mathbf{w}) = \mathbf{w}^{\mathrm{T}}G_w^{\mathrm{T}}G_w\mathbf{w}$. On the other hand, for the state-space model: $M = G_u^{\mathrm{T}}\mathbf{Q}G_u + \mathbf{R}$, $C = F_x^{\mathrm{T}}\mathbf{Q}F_x$, $N = G_u^{\mathrm{T}}\mathbf{Q}F_x$, $n(\mathbf{w}) = G_u^{\mathrm{T}}\mathbf{Q}G_w\mathbf{w}$, $d(\mathbf{w}) = F_x^{\mathrm{T}}\mathbf{Q}G_w\mathbf{w}$, $h(\mathbf{w}) = \mathbf{w}^{\mathrm{T}}G_w^{\mathrm{T}}\mathbf{Q}G_w\mathbf{w}$, where $\mathbf{Q}$ and $\mathbf{R}$ are diagonal matrices

$$\mathbf{Q} = \begin{bmatrix} Q_1 & & \\ & \ddots & \\ & & Q_N \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} R_1 & & \\ & \ddots & \\ & & R_{N_u} \end{bmatrix}$$

When the conditions stated above hold, then the objective function J is convex on $\mathbf{w}$ and $\mathbf{u}$. This implies that the maximum of J will be attained at one of the $2^{N \times \dim(\mathbf{w})}$ vertices of the polytope W [2]. Also, when $\Gamma(x)$ is convex, any local minimizer of J^* will be the global minimizer [2]. In the following, $J_i(\mathbf{u}, x)$ denotes $J(\mathbf{w_i}, \mathbf{u}, x)$, where $\mathbf{w_i}$ is the uncertainty extreme realization related to vertex i. Thus,

$$J_i(\mathbf{u}, x) = \mathbf{u}^{\mathrm{T}}M\mathbf{u} + 2(Nx + n_i)^{\mathrm{T}}\mathbf{u} + x^{\mathrm{T}}Cx + 2d_i^{\mathrm{T}}x + h_i$$

where M, N and C are the same matrices as in (9) and n_i, d_i, h_i are equal to $n(\mathbf{w_i})$, $d(\mathbf{w_i})$, $h(\mathbf{w_i})$.

3. Efficient implementation strategy

In this section the main ideas of the efficient implementation strategy are introduced. This strategy is based on the reduction of the number of vertices to be explored in the maximization part of the min–max problem. Solving such a reduced min–max problem would involve a much lower computational burden than that of the original problem. Thus, the range of processes to which MMMPC can be applied would be considerably broadened.

A main concept is that of active vertices. Note that the function $J^*(\mathbf{u}, x) = \max_{\mathbf{w} \in W} J(\mathbf{w}, \mathbf{u}, x)$ is a piecewise quadratic function of $\mathbf{u}$. Therefore, the $\mathbf{u}$ domain can be divided into different regions $\Gamma(x)_p$ so that $\mathbf{u} \in \Gamma(x)_p$ if the maximum of J is attained for the polytope vertex $\mathbf{w_p}$ [5]. As a consequence of this, the minimum of J^* for a given x will be attained in a value $\mathbf{u}(x)$ which can be inside of a region $\Gamma\ (x)_p$ or in the shared boundary of several regions. This is illustrated in Fig. 1 for a simplified min–max problem with $N_u = N = 1$ and two constraints. Those vertices related to the regions on which the minimum is attained are called the active vertices.

At any of the active vertices the cost function will be equal to the optimal cost $J^s(x)$. Thus, the following definition can be given:

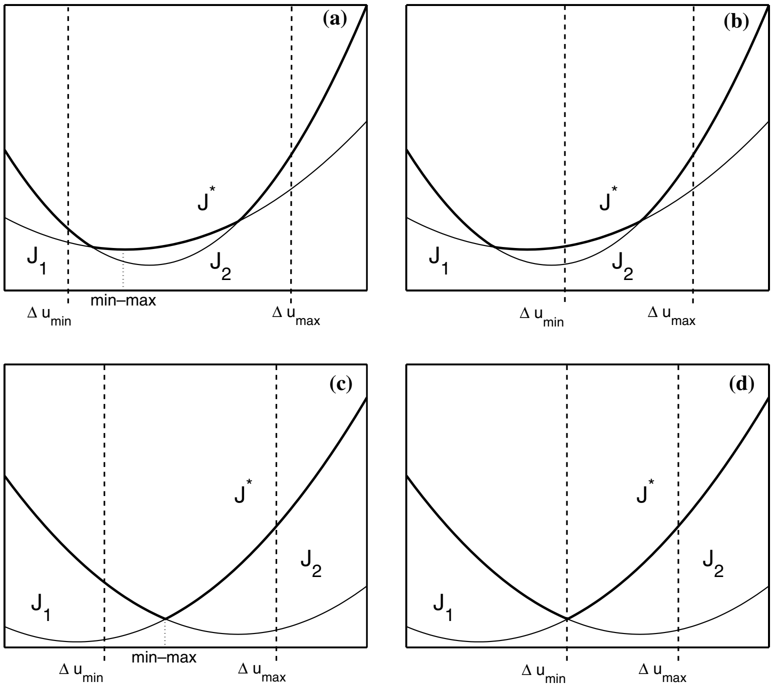

Fig. 1. Possible locations of the solution of constrained min–max problem for two quadratic functions: (a) inside a region (b) as in (a) but attained on an active constraint (c) in the boundary of two regions (d) as in (c) but attained on an active constraint.

Definition 1. $I(x)$, the set of active vertices is given by

$$I(x) = \{i : J_i(\mathbf{u}(x), x) = J^s(x)\} \tag{10}$$

It is easy to see that the solution of (1) is also the solution of a reduced min–max problem given by

$$\mathbf{u}(x) = \arg \min_{\mathbf{u}\in\Gamma(x)} \max_{i\in I(x)} J_i(\mathbf{u}, x) \tag{11}$$

This situation is illustrated in Fig. 2. As the number of active vertices is generally much lower than the total number of vertices [16], problem (11) can be solved with little computational burden. If the process state changes from x to $x + \Delta x$ the active set might also change so that $I(x + \Delta x) \neq I(x)$. Although $I(x + \Delta x)$ cannot be obtained without solving the min–max problem for $x + \Delta x$, it will be shown in this paper that a conservative estimation of $I(x + \Delta x)$ can be efficiently computed using $I(x)$ and Δx. This estimation, which will be denoted as $I_e(x + \Delta x)$, satisfies that $I(x + \Delta x) \subseteq I_e(x + \Delta x)$. Thus, the reduced min–max problem (11) for $x + \Delta x$ can be substituted by the following reduced problem:

$$\mathbf{u}(x + \Delta x) = \arg \min_{\mathbf{u}\in\Gamma(x)} \max_{i\in I_e(x+\Delta x)} J_i(\mathbf{u}, x + \Delta x) \tag{12}$$

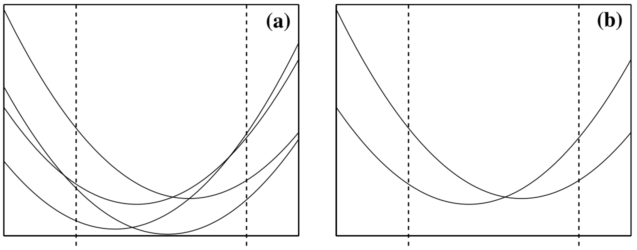

Fig. 2. Two min–max problems with the same solution: (a) full min–max with all curves; (b) reduced min–max with only curves related to active vertex intersection.

which is also equivalent to the original min–max problem. As the number of vertices included in I_e will be a small fraction of the whole $2^{N\times\dim(\mathbf{w})}$ vertices, the computational burden will be accordingly much lower.

The procedure to obtain I_e will be given in the next section. A preliminary proposition that characterizes the change in a quadratic function $J_i(\mathbf{u},x)$ when its parameters are perturbed is stated in the following:

Property 1. *For a given process state x and* $\mathbf{u_c} + \Delta\mathbf{u_c}$, *and for any quadratic function* J_i

$$J_i(\mathbf{u}, x) = \mathbf{u}^{\mathrm{T}} M \mathbf{u} + 2(Nx + n_i)^{\mathrm{T}} \mathbf{u} + x^{\mathrm{T}} Cx + 2d_i^{\mathrm{T}} x + h_i$$

it is true that

$$J_i(\mathbf{u_c} + \Delta\mathbf{u_c}, x) = J_i(\mathbf{u_c}, x) + \Delta\mathbf{u_c}^{\mathrm{T}} M \Delta\mathbf{u_c} + 2b_i^{\mathrm{T}} \Delta\mathbf{u_c}$$

with $b_i = M\mathbf{u_c} + Nx + n_i$.

4. Online estimation of $I(x)$

This section shows how to build an estimation of the set of active vertices for a given process state by means of a suboptimal candidate solution. With this estimation of $I(x)$, which is denoted by $I_e(x)$, the min–max problem to be solved at sampling time t could be replaced by an equivalent reduced min–max problem. The necessary steps to compute such an estimation are given by

- First, it is necessary to obtain a candidate solution $\mathbf{u_c}$, which should be close to the optimal solution. In order to fulfill this requirement a number of procedures could be followed. Here it is suggested to use the optimal solution found in the previous sampling time, $\mathbf{u}(x(t-1))$. Many of the strategies used to preserve stability in constrained MPC, which can be used within the min–max strategy, yield control signals $\mathbf{u}(x(t-1))$ that can be used to obtain a feasible control action for $x(t)$ [7,12]. Nevertheless, if $\mathbf{u}(x(t-1))$ is unfeasible at sampling time t this requirement can be fulfilled solving the following QP problem:

$$\begin{aligned} \hat{\mathbf{u}} &= \arg\min \|\xi\|_2 \\ &\quad \text{s.t.} \\ &\quad H\xi \leqslant Sx + d - H\mathbf{u}(x(t-1)) \end{aligned} \tag{13}$$

  where H, S and d are the constraints matrices and vector of the min–max problem. Then, the candidate solution would be $\mathbf{u_c} = \mathbf{u}(x(t-1)) + \xi$. Alternatively, in (13) a shifted version of $\mathbf{u}(x(t-1))$ can be used, as it could be a better approximation for $\mathbf{u}(x(t))$ because of the receding horizon strategy.
- Compute a tight ellipsoidal bounding for $\Delta\mathbf{u_c}$, where $\mathbf{u_c} + \Delta\mathbf{u_c}$ is the optimal solution. A procedure to compute such ellipsoidal bounding is given in Section 4.1.
- Taking into account the ellipsoidal bounding, reject as many vertices as possible using the criterion given in Section 4.2. Use the remaining vertices to form $I_e(x)$, which is a conservative estimation of $I(x)$.

4.1. Ellipsoidal bounding

In the following, a procedure to find an ellipsoidal bounding of $\Delta\mathbf{u_c}$ is given. First, for a given process state $x = x(t)$ choose the following set:

$$I^* = I(x(t-1)) \cup \{n \text{ vertices with higher } J_i(\mathbf{u_c}, x)\} \tag{14}$$

where n is a design parameter closely related to the degree of conservatism of the bounding of $\Delta\mathbf{u_c}$. A very small n ($n \leqslant 2$) leads to a rather conservative bounding of $\Delta\mathbf{u_c}$, which ultimately results in a too conservative estimation of the active vertex set. On the other hand, the authors have found that it is enough to make n equal to a small fraction of the total number of vertices (about 0.005% or less if the total number of vertices is very high). Furthermore, no improvement has been observed by making n much greater.

Due to the optimality of $\mathbf{u_c} + \Delta\mathbf{u_c}$ it is true that

$$J_i(\mathbf{u_c} + \Delta\mathbf{u_c}, x) \leqslant J^*(\mathbf{u_c} + \Delta\mathbf{u_c}, x) \leqslant J^*(\mathbf{u_c}, x) \quad \forall i \in I^* \tag{15}$$

Let $\lambda_i \geqslant 0$, $\sum_{i\in I^*}\lambda_i = 1$. Then it holds that

$$\sum_{i\in I^*} \lambda_i J_i(\mathbf{u_c} + \Delta\mathbf{u_c}, x) \leqslant J^*(\mathbf{u_c}, x) \tag{16}$$

Applying Property 1 on the left side of (16):

$$\Delta\mathbf{u_c}^{\mathrm{T}} M \Delta\mathbf{u_c} + 2\sum_{i\in I^*} \lambda_i b_i^{\mathrm{T}} \Delta\mathbf{u_c} + \sum_{i\in I^*} \lambda_i J_i(\mathbf{u_c}, x) - J^*(\mathbf{u_c}, x) \leqslant 0 \tag{17}$$

On the other hand, the constraints $u \in \Gamma(x)$ can be rewritten as

$$a_m^{\mathrm{T}}\mathbf{u} \leqslant b_m^r, \quad m = 1, \ldots, n_r$$

where n_r is the number of constraints in problem (1). The optimal solution is a feasible one, thus it is true that

$$a_m^{\mathrm{T}}(\mathbf{u_c} + \Delta\mathbf{u_c}) - b_m^r \leqslant 0$$

Let $\beta_m \geqslant 0$, $m = 1, \ldots, n_r$ be scalars. It can easily be seen that

$$2\sum_{m=1}^{n_r} \beta_m (a_m^{\mathrm{T}}(\mathbf{u_c} + \varDelta\mathbf{u_c}) - b_m^r) \leqslant 0 \tag{18}$$

Furthermore, adding inequalities (17) and (18), it is also true that

$$\begin{aligned} &\Delta\mathbf{u_c}^{\mathrm{T}} M \Delta\mathbf{u_c} + 2\sum_{i\in I^*} \lambda_i b_i^{\mathrm{T}} \Delta\mathbf{u_c} + \sum_{i\in I^*} \lambda_i J_i(\mathbf{u_c}, x) \\ &\quad - J^*(\mathbf{u_c}, x) + 2\sum_{m=1}^{n_r} \beta_m (a_m^{\mathrm{T}}(\mathbf{u_c} + \Delta\mathbf{u_c}) - b_m^r) \leqslant 0 \end{aligned} \tag{19}$$

which can be expressed as

$$\begin{aligned} &\Delta\mathbf{u_c}^{\mathrm{T}} M \Delta\mathbf{u_c} + 2\left(\sum_{i\in I^*} \lambda_i b_i + \sum_{m=1}^{n_r} \beta_m a_m\right)^{\mathrm{T}} \Delta\mathbf{u_c} \\ &\quad + \sum_{i\in I^*} \lambda_i J_i(\mathbf{u_c}, x) - J^*(\mathbf{u_c}, x) + 2\sum_{m=1}^{n_r} \beta_m (a_m^{\mathrm{T}}\mathbf{u_c} - b_m^r) \leqslant 0 \end{aligned} \tag{20}$$

Inequality (20) describes an ellipsoid which can be rewritten as

$$(\Delta\mathbf{u_c} - a)^{\mathrm{T}} A (\Delta\mathbf{u_c} - a) \leqslant 1 \tag{21}$$

where

$$\begin{aligned} A^{-1} = &\left(\left(\sum_{i\in I^*} \lambda_i b_i + \sum_{m=1}^{n_r} \beta_m a_m\right)^{\mathrm{T}} M^{-1} \left(\sum_{i\in I^*} \lambda_i b_i + \sum_{m=1}^{n_r} \beta_m a_m\right)\right. \\ &\quad \left. + J^*(\mathbf{u_c}, x) - \sum_{i\in I^*} \lambda_i J_i(\mathbf{u_c}, x) - 2\sum_{m=1}^{n_r} \beta_m (a_m^{\mathrm{T}}\mathbf{u_c} - b_m^r)\right) M^{-1} \end{aligned} \tag{22}$$

$$a = -M^{-1}\left(\sum_{i\in I^*}\lambda_i b_i + \sum_{m=1}^{n_r}\beta_m a_m\right)$$

Ellipsoid (21) bounds $\Delta\mathbf{u_c}$, and its size is related to some extent to λ_i, β_m. Thus, it makes sense to find such λ_i, β_m which makes (21) as small as possible. This can be achieved solving the following QP problem:

$$\begin{aligned}\min\quad & \left(\sum_{i\in I^*}\lambda_i b_i + \sum_{m=1}^{n_r}\beta_m a_m\right)^{\mathrm{T}} M^{-1}\left(\sum_{i\in I^*}\lambda_i b_i + \sum_{m=1}^{n_r}\beta_m a_m\right)\\ & -\sum_{i\in I^*}\lambda_i J_i(\mathbf{u_c},x) - 2\sum_{m=1}^{n_r}\beta_m(a_m^{\mathrm{T}}\mathbf{u_c} - b_m^r)\\ \text{s.a.}\quad & \lambda_i \geqslant 0,\quad i\in I^*\\ & \beta_m \geqslant 0,\quad m = 1,\ldots,n_r\\ & \sum_{i\in I^*}\lambda_i = 1\end{aligned}$$

The optimal values of λ_i, β_m allow us to define the least conservative ellipsoidal bounding of $\Delta\mathbf{u_c}$, which will be used in the exclusion criterion given in Section 4.2.

Finally it is noteworthy that the ellipsoid may not completely lay within the feasible space of problem (1). However, this is not a problem, it just leads to greater conservatism in the estimation of $I(x)$.

4.2. Exclusion criterion

In this section an exclusion criterion devised to reject the majority of vertices not in $I(x)$ is presented. Suppose that $k\in I(x)$, then from the min–max concept:

$$J_i(\mathbf{u_c}+\Delta\mathbf{u_c},x) \leqslant J_k(\mathbf{u_c}+\Delta\mathbf{u_c},x)\quad \forall i \tag{23}$$

Let γ_i be scalars satisfying $\gamma_i \geqslant 0$ and $\sum\gamma_i = 1$, where $i\in I^*$. Then it is true that

$$\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c}+\Delta\mathbf{u_c},x) \leqslant J_k(\mathbf{u_c}+\Delta\mathbf{u_c},x) \tag{24}$$

Applying Property 1,

$$\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c},x) - J_k(\mathbf{u_c},x) \leqslant 2\left(b_k - \sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}}\Delta\mathbf{u_c} \tag{25}$$

Furthermore, it is true that

$$\begin{aligned}\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c},x) - J_k(\mathbf{u_c},x) &\leqslant 2b_k^{\mathrm{T}}\Delta\mathbf{u_c}\\ &+ \max_{\Delta\mathbf{u_c}\in\mathscr{E}}\sum_{i\in I^*} -2\gamma_i b_i^{\mathrm{T}}\Delta\mathbf{u_c}\end{aligned} \tag{26}$$

where $\mathscr{E}$ is the ellipsoidal bounding of $\Delta\mathbf{u_c}$, computed as described in Section 4.1. This bounding can be expressed as

$$\mathscr{E} = \{\Delta\mathbf{u_c} : (\Delta\mathbf{u_c} - a)^{\mathrm{T}}A(\Delta\mathbf{u_c} - a) \leqslant 1\} \tag{27}$$

The maximization problem in (26) can be solved analytically, thus (26) can be rewritten as

$$\begin{aligned}&\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c},x) - J_k(\mathbf{u_c},x)\\ &\quad\leqslant 2b_k^{\mathrm{T}}\Delta\mathbf{u_c} + 2\sqrt{\left(\sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}} A^{-1}\left(\sum_{i\in I^*}\gamma_i b_i\right)}\\ &\quad - 2\sum_{i\in I^*}\gamma_i b_i^{\mathrm{T}} a\end{aligned} \tag{28}$$

Furthermore, grouping terms it yields

$$\begin{aligned}&\sum_{i\in I^*}\gamma_i(J_i(\mathbf{u_c},x) + 2b_i^{\mathrm{T}}a)\\ &\quad - 2\sqrt{\left(\sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}} A^{-1}\left(\sum_{i\in I^*}\gamma_i b_i\right)}\\ &\quad\leqslant J_k(\mathbf{u_c},x) + 2b_k^{\mathrm{T}}\Delta\mathbf{u_c}\end{aligned} \tag{29}$$

Inequality (29) describes a condition that must be fulfilled by all vertices $k\in I(x)$ and any valid γ_i, including those γ_i that maximize the left side of (29). On the other hand, it can easily be seen that if a given vertex k does not satisfy (29) then $k\notin I(x)$. Thus, this fact can be exploited to reject many of the vertices which are not active for x. These values of γ_i that maximize the left side of (29) can be found solving:

$$\begin{aligned}\min_{\gamma_i}\quad & 2\sqrt{\left(\sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}} A^{-1}\left(\sum_{i\in I^*}\gamma_i b_i\right)}\\ & -\sum_{i\in I^*}\gamma_i(J_i(\mathbf{u_c},x) + 2b_i^{\mathrm{T}}a)\\ \text{s.a.}\quad & \gamma_i \geqslant 0,\quad i\in I^*\\ & \sum_{i\in I^*}\gamma_i = 1\end{aligned} \tag{30}$$

This is clearly a convex problem which can be solved using standard algorithms.

The computed values of γ_i together with those λ_i and β_i found when computing the ellipsoidal bounding can be used to reject non active vertices. Taking into account (25) an online estimation of $I(x)$ can be obtained rejecting any vertex k that fulfills:

$$\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c},x) - J_k(\mathbf{u_c},x) > \max_{\Delta\mathbf{u_c}\in\mathscr{E}} 2\left(b_k - \sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}}\Delta\mathbf{u_c} \tag{31}$$

which can be expressed as

$$\begin{aligned}&\sum_{i\in I^*}\gamma_i J_i(\mathbf{u_c},x) - J_k(\mathbf{u_c},x)\\ &\quad > 2\sqrt{\left(b_k - \sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}} A^{-1}\left(b_k - \sum_{i\in I^*}\gamma_i b_i\right)}\\ &\quad + 2\left(b_k - \sum_{i\in I^*}\gamma_i b_i\right)^{\mathrm{T}} a\end{aligned} \tag{32}$$

The set $I_e(x)$, i.e. the conservative estimation of $I(x)$, is then built from all vertices not rejected with (32). Finally it is noteworthy that computing the ellipsoidal bounding as described in Section 4.1 and the values of γ_i obtained solving (30) requires solving a QP problem and a tractable convex problem only once per sampling time. The percentage of rejected vertices is so high that for the usual values of N the whole procedure (i.e. computing the ellipsoidal bounding and the optimal values of γ_i, testing all vertices and solving the reduced min–max problem) takes only a fraction of the time required to solve the complete min–max problem.

5. Simulation examples

The results obtained in Section 4 will be illustrated with simulation examples. Two examples are given, first a SISO process described by an input–output model and then a MIMO process modelled using a state-space description.

5.1. SISO system

This example uses a prediction model for a first order system, which is one of the more common choices for a wide class of industrial processes. The integrated uncertainties prediction model used is given by

$$y_{k+1} = 1.9048y_k - 0.9048y_{k-1} + 0.0129\Delta u_k + w_{k+1} \qquad (33)$$

It describes the dynamics of a heat exchanger [14]. The efficient implementation of Constrained Min–Max MPC presented here has been applied with the following controller parameters: $N_u = 10$, $N = 10$, $\lambda = 1$ and the expected uncertainty values are $-0.05 \leqslant w(t) \leqslant 0.05$. Constraints on process output, control signal and control moves have been taken into account. Their values are: $-0.5 \leqslant y \leqslant 1.5$, $0 \leqslant u \leqslant 8$ and $-0.75 \leqslant \Delta u_k \leqslant 0.75$.

The reference is set to 1, and the initial conditions are $y_k = y_{k-1} = 0$. The min–max problem has been solved using standard solver provided with the *fmincon* function of *Matlab*. Fig. 3 shows the process output, control signal and control moves. In the simulation a random disturbance has been added to the plant output to simulate more realistic conditions. On the other hand, at sampling time $t = 61$ an stationary step disturbance suddenly hits the process output, causing the great deviation from the set point seen in the output plot. This step disturbance disappears at $t = 101$. It can be seen in Fig. 3 that control signal and control moves become saturated when the output is far from the set point.

The number of vertices included in $I_e(x)$ at each sampling time is shown in Fig. 4. In this example, the parameter n in (14) is set to 5. It can be seen that most of the 1024 vertices are rejected every sampling time. In fact, on the average 99.35% of the vertices are rejected.

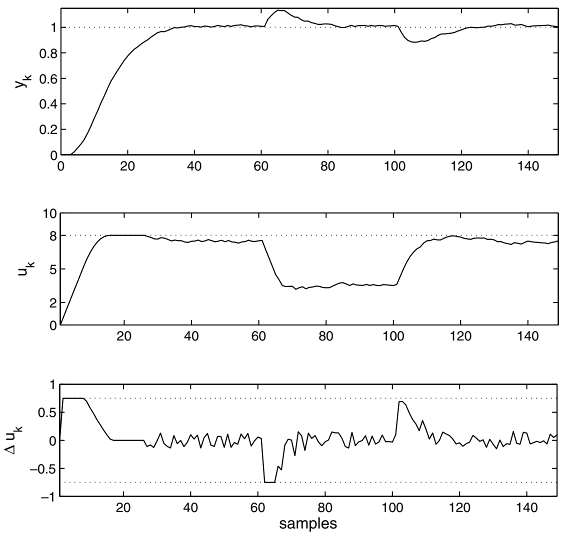

Fig. 3. Simulation example, top to bottom: process output, control signal and control moves.

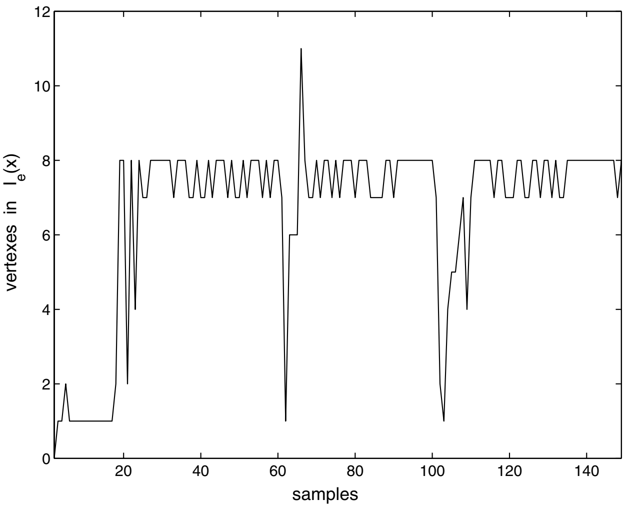

Fig. 4. Number of vertices included in $I_e(x)$.

5.2. MIMO system

As stated in section 2, the results presented in this paper are not constrained to SISO systems and may be used with different types of process descriptions. To illustrate this, consider the two-tank network shown in Fig. 5. For this process, liquid streams flow into tanks 1 and 2 at respective volumetric rates F_1 and F_2; the outflow from each tank is assumed to be proportional to the respective liquid levels h_1 and h_2 in each tank. The liquid leaving tank 2 is split into two with a fraction F, exiting, and the remainder R pumped back to the first tank. Thus, this is a two-input, two-output system, with the flow rates of the two inlet streams as the two inputs, and the liquid level in each tank as the two output variables.

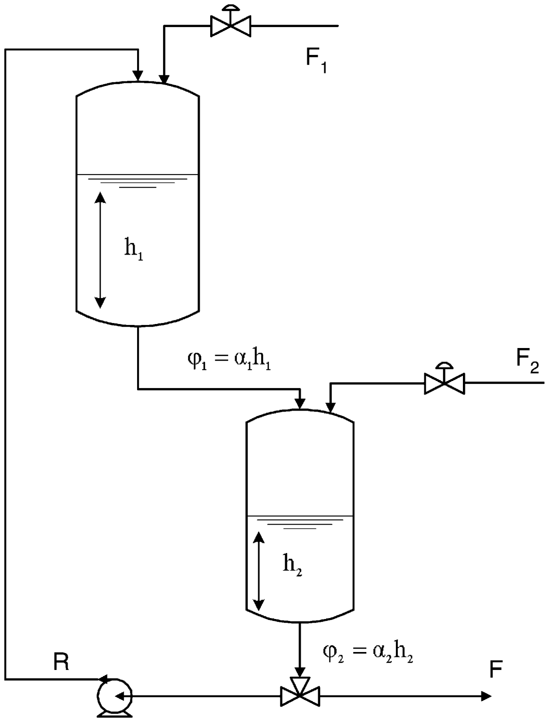

Fig. 5. A two-tank network.

Let the section for tank 1 be 3 m^2 and that of the second tank 2 m^2. Moreover, assume that the constants of proportionality are identical and given as $\alpha_1 = \alpha_2 = 0.5$ $m^2 min^{-1}$ and that 40% of the amount of liquid leaving tank 2 is recycled back to tank 1. With these assumptions the following continuous time state-space model can be obtained (see Chapter 20 of [13] for details):

$$\dot{x} = \begin{bmatrix} -\frac{0.5}{3} & \frac{0.2}{3} \\ \frac{0.5}{2} & -\frac{0.5}{2} \end{bmatrix} x + \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x \tag{34}$$

A discrete time model has been obtained from (34) sampling at 0.2 min using a zero-order holder. Fig. 6 shows the results of a min–max MPC applied to the two-tank model. The set-point for the liquid level of each tank was 0.4 and 0.5 m respectively. The prediction and control horizons were $N = 10$ and $Nu = 4$ respectively. Note that being this a two-input, two-output system the number of vertices to be considered is 2^{20} instead of 2^{10}. Furthermore, the number of decision variables in the optimization problem is doubled. The weighting matrices were

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 12 & 0 \\ 0 & 12 \end{bmatrix}$$

An uncertainty of ±0.02 m is considered to affect both liquid levels. Note that a random noise of that amplitude has been added to the liquid levels in the simulation. Finally, in the simulation the following constraints were taken into account when computing the control signal:

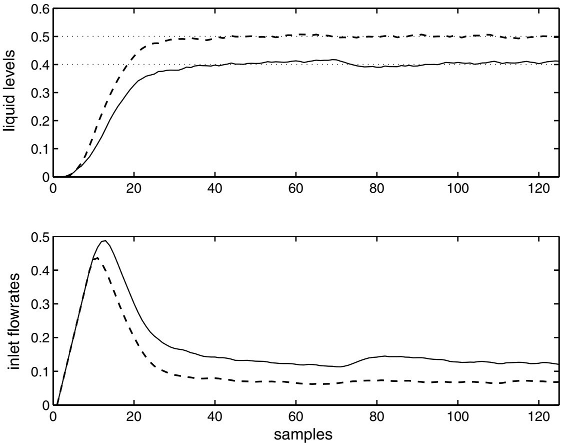

Fig. 6. Liquid levels and inlet flows for the MMMPC (tank 1 solid plot, tank 2 dashed plot).

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \leqslant x(k) \leqslant \begin{bmatrix} 0.6 \\ 0.7 \end{bmatrix} \tag{35}$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \leqslant u(k) \leqslant \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \tag{36}$$

$$\begin{bmatrix} -0.05 \\ -0.05 \end{bmatrix} \leqslant \Delta u(k) \leqslant \begin{bmatrix} 0.05 \\ 0.05 \end{bmatrix} \tag{37}$$

In this example the set I^* defined in Section 4.1 was computed by making n such that the total number of vertices in I^* was always 30.

This example will be used to discuss how the increasing horizons affect the number of rejected vertices and the relative speed-up obtained, as well as the total number of floating point operations. Table 1 shows the percentage of rejected vertices when the prediction horizon takes progressively higher values. Note that as the prediction horizon grows, the percentage also grows. This suggests that this strategy will perform better as the prediction horizon gets higher.

The overall computational burden of the proposed strategy also includes all the floating operations needed to compute the set $I_e(x)$, i.e. the online estimation of the active vertices set. Table 2 shows the number of floating operations (as counted by *Matlab*) needed to compute the full min–max problem and the proposed strategy. In the latter case all additional computations to obtain $I_e(x)$ are taking into account. [1] For both cases, the

[1] Note that all the results given in Tables 1 and 2 were obtained using a set I^* of 30 vertices.

Table 1
Percentage of rejected vertices (minimum, average and maximum) and retained vertices (maximum only) for different values of the prediction horizon (N) in the simulation example of Section 5.2

N	Total vertices	Rejected (min) %	Rejected (avg) %	Rejected (max) %	Ret. (max)
4	256	96.48	99.49	99.609	9
5	1024	98.7	99.8	99.9	13
6	4096	98.63	99.4	99.97	56
7	16,384	98.63	99.2	99.99	225
8	65,536	99.59	99.73	99.998	272
9	262,144	99.656	99.77	99.999	905
10	1,048,576	99.71	99.8	99.999	3037

Table 2
Number of floating point operations (minimum, average and maximum) for different values of the prediction horizon (N) in the simulation example of Section 5.2

N	Full (min)	Full (avg)	Full (max)	Prop. (min) %	Prop. (avg) %	Prop. (max)
4	3.09×10^6	4.29×10^7	5.87×10^7	4.55×10^7	4.61×10^7	4.66×10^7
5	2.06×10^8	2.7×10^8	4.404×10^8	5.85×10^7	5.94×10^7	6.02×10^7
6	1.82×10^8	1.803×10^9	4.456×10^9	8.261×10^7	8.904×10^7	1.009×10^8
7	2.967×10^9	9.175×10^9	1.835×10^{10}	1.483×10^8	2.272×10^8	3.593×10^8
8	2.546×10^9	4.281×10^{10}	8.198×10^{10}	4.082×10^8	4.519×10^8	5.541×10^8
9	7.545×10^{10}	2.225×10^{11}	3.458×10^{11}	1.597×10^9	1.907×10^9	2.164×10^9
10	2.374×10^{11}	1.313×10^{12}	2.1×10^{12}	7.163×10^9	8.724×10^9	9.793×10^9

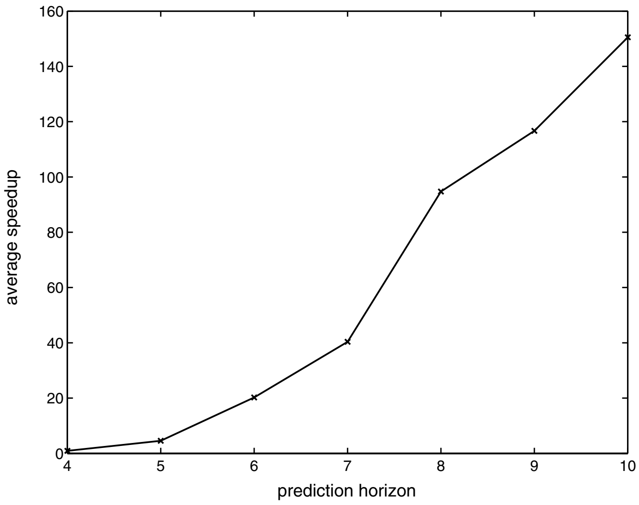

Fig. 7. Average speed-up for different values of the prediction horizon.

min–max problem was solved using the same numerical solver provided with *fmincon* function of *Matlab*. Fig. 7 shows the average speed-up for each value of the prediction horizon. It can be seen that when the number of vertices is small (i.e., N small) the speed-up is not very high. This means that the computations needed to obtain $I_e(x)$ overcome the number of operations needed to solve the full min–max problem. As N grows higher, the speed-up increases clearly. Again, this means that the proposed strategy performs better as the prediction horizon grows.

6. Conclusions

An efficient implementation of the Constrained MMMPC control law has been presented. The results presented in this paper broaden the range of processes to which, in practice, MMMPC can be applied. The strategy proposed in this work ensures much lower computational burden in most cases. This is accomplished by reducing the number of uncertainty extreme realizations to be considered, from $2^{N \times \dim(\mathbf{w})}$ to only a small fraction of them.

However, many open questions remain to be addressed. It is worth investigating how to extend this approach to other formulations of Constrained Min–Max MPC such as quasi-min–max methods [11] and closed loop formulations [19,3]. Also, techniques such as branch and bound could be applied to obtain nearly optimal solutions to the reduced min–max problem with lesser computational burden.

Acknowledgment

The authors acknowledge MCYT-Spain for funding this work (contracts DPI2001-2380-C02-01 and DPI2002-04375-C03-01).

References

[1] T. Alamo, D. Muñoz de la Peña, D. Limón Marruedo, E.F. Camacho, Constrained min max predictive control: a polynomial

time approach, in: Proccedings of the 42nd IEEE Conference on Decision and Control, Maui, Hawai, USA, 2003.

[2] M.S. Bazaraa, C.M. Shetty, Nonlinear programming. Theory and algorithms, John Wiley, 1979.

[3] A. Bemporad, F. Borrelli, M. Morari, Min–max control of constrained uncertain discrete-time linear systems, IEEE Transactions on Automatic Control 48 (9) (2003) 1600–1606.

[4] E.F. Camacho, M. Berenguel, Robust adaptive model predictive control of a solar plant with bounded uncertainties, International Journal of Adaptive and Signal Processing 11 (1997) 311–325.

[5] E.F. Camacho, C. Bordóns, Model Predictive Control, Springer-Verlag, 1999.

[6] T. Johansen, A. Grancharova, Approximate explicit constrained linear model predictive control via orthogonal search tree, IEEE Transactions on Automatic Control 48 (5) (2003) 810–815.

[7] E.C. Kerrigan, Robust constraint satisfaction: invariant sets and predictive control, PhD thesis, University of Cambridge, 2000.

[8] E.C. Kerrigan, J.M. Maciejowski, Feedback min–max model predictive control using a single linear program: robust stability and the explicit solution, International Journal of Robust Nonlinear Control 14 (4) 395–413.

[9] Y.H. Kim, W.H. Kwon, An application of min–max generalized predictive control to sintering processes, Control Engineering Practice 6 (1998) 999–1007.

[10] J.H. Lee, Z. Yu, Worst-case formulations of model predictive control for systems with bounded parameters, Automatica 33 (5) (1997) 763–781.

[11] Y. Lu, Y. Arkun, Quasi-Min–Max MPC algorithms for LPV systems, Automatica 36 (4) (2000) 527–540.

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

[13] B.O. Ogunnaike, W.H. Ray, Process Dynamics, Modeling, and Control, Oxford University Press, 1994.

[14] D.R. Ramírez, M.R. Arahal, E.F. Camacho, Min–Max Predictive Control of a heat exchanger using a neural network solver, IEEE Transactions on Control Systems Technology 12 (5).

[15] D.R. Ramírez, E.F. Camacho, On the piecewise linear nature of Min–Max Model Predictive Control with bounded uncertainties, in: Proceedings of the 40th Conference on Decision and Control, CDC'2001, December 4–7, 2001.

[16] D.R. Ramírez, E.F. Camacho, Characterization of Min–Max MPC with global uncertainties, in: Proc. American Control Conference, ACC, 2002.

[17] V. Sakizlis, N.M.P. Kakalis, J.D. Perkins, V. Dua, E.N Pistikopoulos, Design of robust model-based controllers via parametric programming, Automatica 40 (2004) 189–201.

[18] V. Sakizlis, J.D. Perkins, V. Dua, E.N. Pistikopoulos, Robust model-based tracking control using parametric programming, Computers & Chemical Engineering 28 (2004) 195–207.

[19] P.O.M. Scokaert, D.Q. Mayne, Min-max feedback model predictive control for constrained linear systems, IEEE Transactions on Automatic Control 43 (8) (1998) 1136–1142.

[20] P. Tøndel, T.A. Johansen, A. Bemporad, Evaluation of piecewise affine control via binary search tree, Automatica 39 (5) (2003) 945–950.

[21] S.M. Veres, J.P. Norton, Predictive self-tuning control by parameter bounding and worst case design, Automatica 29 (4) (1993) 911–928.