Katholieke Universiteit Leuven

Departement Elektrotechniek

ESAT-SISTA/TR 04-231

Interpolation Based MPC with Exact Constraint Handling : the Nominal Case [1]

J.A. Rossiter[2] , Y. Ding[2], L. Xi[2] ,
B. Pluymers[3] , J. Suykens[3] and B. De Moor[3]

March 2005

Internal Report 04-231

[1]This report is available by anonymous ftp from *ftp.esat.kuleuven.ac.be* in the directory *pub/sista/pluymers/reports/ECC-CDC05b-inter-nominal.pdf*

[2]University of Sheffield, Department of Automatic Control and Systems Engineering, Mappin Street, Sheffield S1 3JD, United Kingdom, E-mail: *j.a.rossiter@sheffield.ac.uk*

[3]K.U.Leuven, Dept. of Electrical Engineering (ESAT), Research group SCD-SISTA, Kasteelpark 10, 3001 Leuven, Belgium, Tel. 32/16/32 10 35, Fax 32/16/32 19 70, WWW: *http://www.esat.kuleuven.ac.be/scd*. E-mail: {*bert.pluymers,johan.suykens,bart.demoor*}*@esat.kuleuven.ac.be*.

Research supported by Research Council KUL: GOA-Mefisto 666, GOA-Ambiorics, several PhD/postdoc & fellow grants; Flemish Government: FWO: PhD/postdoc grants, projects, G.0240.99 (multilinear algebra), G.0407.02 (support vector machines), G.0197.02 (power islands), G.0141.03 (Identification and cryptography), G.0491.03 (control for intensive care glycemia), G.0120.03 (QIT), G.0452.04 (QC), G.0499.04 (robust SVM), research communities (ICCoS, ANMMM, MLDM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants, GBOU (McKnow) Belgian Federal Government: Belgian Federal Science Policy Office: IUAP V-22 (Dynamical Systems and Control: Computation, Identification and Modelling, 2002-2006), PODO-II (CP/01/40: TMS and Sustainibility); EU: FP5-Quprodis; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; Contract Research/agreements: ISMC/IPCOS, Data4s, TML, Elia, LMS, IPCOS, Mastercard; Bert Pluymers is a research assistant with the I.W.T. (Flemish Institute for Scientific and Technological Research in Industry). Dr. Johan Suykens is an associate professor at the Katholieke Universiteit Leuven. Dr. Bart De Moor is a full professor at the Katholieke Universiteit Leuven, Belgium.

Abstract

Interpolation techniques are known to reduce computational complexity of MPC algorithms (Rossiter *et al.*, 2004, Bacic *et al.*, 2003). However, despite giving good feasible regions, there is nevertheless often some conservatism. This paper presents a new insight on general interpolation which reduces this conservatism and hence allows a substantial increase in feasible regions and therefore the potential of the approach. In particular it shows that the feasible region may be far larger than the convex hull of the underlying regions. Rigorous proofs of the results are also provided.

Interpolation based MPC with exact constraint handling: the nominal case

J.A. Rossiter‡, Y. Ding‡, L. Xi‡, B. Pluymers†, J.A.K. Suykens†, B. De Moor†

‡University of Sheffield
Department of Automatic Control and Systems Engineering
E-Mail : j.a.rossiter@sheffield.ac.uk
Internet : http://www.shef.ac.uk/acse/
†Katholieke Universiteit Leuven
Department of Electrical Engineering, ESAT-SCD-SISTA
E-Mail : {bert.pluymers, johan.suykens, bart.demoor}@esat.kuleuven.ac.be
Internet : http://www.esat.kuleuven.ac.be/scd/

***Abstract*— Interpolation techniques are known to reduce computational complexity of MPC algorithms (Rossiter *et al.*, 2004, Bacic *et al.*, 2003). However, despite giving good feasible regions, there is nevertheless often some conservatism. This paper presents a new insight on general interpolation which reduces this conservatism and hence allows a substantial increase in feasible regions and therefore the potential of the approach. In particular it shows that the feasible region may be far larger than the convex hull of the underlying regions. Rigorous proofs of the results are also provided.**

I. Introduction

This paper is set in the context of predictive control (MPC) [10], [4] whereby one has a system, possibly multivariable, that is subject to constraints. In order to ensure consistent and reliable behaviour, MPC algorithms build the constraints into the control law formulation from the outset, rather than using ad hoc rules at a later stage. However, there a some major obstacles to the implementation of MPC.

1) Constraint handling usually requires an online optimiser which may imply significant computation.
2) The feasible region within which the control law is well defined, may be small unless the algorithm uses large numbers of degrees of freedom (d.o.f.).
3) One can sometimes enlarge the feasible region by detuning, but this could be undesirable.

Hence a typical conflict is between computational load which is linked to the number of d.o.f., the volume of the feasible region and the performance.

For processes with large throughput and/or slow sample times, these obstacles are of less consequence. However, for systems where either the sampling time is fast [7] or the implied optimiser must be simple [9], then there is a need to formulate algorithms which tackle the necessary compromises systematically.

In this paper we will consider interpolation approaches; these aim to reduce the computational load by formulating classes of predictions with small numbers of d.o.f., hence reducing the optimiser complexity. The difficulty is that as one is restricted to interpolating between fixed predictions, one may find that the feasible region is small [1], [8] compared to a more conventional approach [11] with large d.o.f. Here we present a new insight to interpolation which can allow substantial increases to the feasibility of the algorithms given in [1], [8] and perhaps surprisingly, achieves this by removing one of the optimisation variables and hence reducing the computational load! Moreover, the resulting algorithm is surprisingly simple.

It is noted that this paper focuses on the certain case. Due to the need to tackle some detailed technicalities, a parallel paper considers extensions to linear parameter varying (LPV) systems. Section 2 will give some background to modelling assumptions and previous work. Section 3 develops the new insights, section 4 formulates these into an algorithm and section 5 gives some simulation examples.

II. Background

A. Model and objective

This paper considers linear systems of the form

$$x(k+1) = Ax(k) + Bu(k), \qquad k = 0, \dots, \infty \quad (1)$$

The system is subject to constraints

$$u(k) \in \mathcal{U} \equiv \{u : \underline{u} \leq u \leq \overline{u}\}, \qquad k = 0, \dots, \infty, \quad (2a)$$
$$x(k) \in \mathcal{X} \equiv \{x : \underline{x} \leq x \leq \overline{x}\}, \qquad k = 0, \dots, \infty. \quad (2b)$$

$x(k) \in \mathbb{R}^{n_x}$ and $u(k) \in \mathbb{R}^{n_u}$ denote state and input vectors at discrete time k with n_x and n_u respectively denoting the number of states and inputs of the system. More general linear state, input and mixed state/input constraints can also be considered without significantly complicating further sections. In this paper an new algorithm is proposed

that stabilizes system (1) and guarantees satisfaction of constraints (2). The algorithm aims to minimise

$$J = \sum_{k=0}^{\infty} (x(k)^{\mathrm{T}} Q x(k) + u(k)^{\mathrm{T}} R u(k)) \tag{3}$$

as a cost objective with $Q = Q^{\mathrm{T}} \in \mathbb{R}^{n_x \times n_x}$ and $R = R^{\mathrm{T}} \in \mathbb{R}^{n_u \times n_u}$ positive definite state and input cost weighting matrices.

B. Invariant Sets - the nominal case

Invariant sets [3] are key to the developments discussed in this paper and hence are overviewed next.

Definition 1 (Feasibility): Given system (1), an asymptoticaly stabilizing feedback $u(k) = -Kx(k)$ and constraints (2), then a set $\mathcal{S} \subset \mathbb{R}^{n_x}$ is feasible iff $\mathcal{S} \subset \{x | x \in \mathcal{X}, -Kx \in \mathcal{U}\}$.

Definition 2 (Invariance): Given system (1), a stabilizing feedback $u(k) = -Kx(k)$ and constraints (2), then a set $\mathcal{S} \subset \mathbb{R}^{n_x}$ is positive invariant iff

$$x \in \mathcal{S} \quad \Rightarrow \quad (A - BK)x \in \mathcal{S}, \quad \forall [A \ B] \in \Omega. \tag{4}$$

This paper makes use of sets which are both feasible and invariant. The largest possible feasible invariant set (in the sense that no other feasible invariant set can contain points outside this set) is uniquely defined and is called the Maximal Admissable Set (MAS, [5]). Under certain convergence conditions, the MAS for an LTI system (1) is given by $\mathcal{S}_i = \{x | M_i x \leq d\}$ with $\Phi_i = A - BK_i$ and

$$M_i = \begin{bmatrix} F\Phi_i \\ F\Phi_i^2 \\ \vdots \\ F\Phi_i^n \end{bmatrix}; \ F = \begin{bmatrix} I \\ -I \\ -K \\ K \end{bmatrix}; \ d = \begin{bmatrix} \tilde{d} \\ \tilde{d} \\ \vdots \end{bmatrix} \tag{5}$$

$\tilde{d} = [\overline{x}^T, -\underline{x}^T, \overline{u}^T, -\underline{u}^T]^T$. The requirement for a finite number of inequalities indicates that $\mathcal{S}_i$ is polyhedral.

Remark 1: In the following discussions, we will assume that the matrices M_i defining $\mathcal{S}_i$ are defined in a mutually consistent way as in (5). Hence the jth row of each, for all j, must refer to the same constraint/prediction. $\mathcal{S}_i$ may take a non-minimal form as redundant constraints are only removed if they are redundant for every $\mathcal{S}_i$.

Definition 3 (MAS and predictions): For convenience hereafter we adopt the following notation.

1) $\mathcal{S}_i$ is the MAS associated to feedback $u = -K_i x$. Assume the origin is strictly inside $\mathcal{S}_i$ and hence normalise M_i in (5) so that $d = [1, 1, ..., 1]^T$.
2) We will use the shorthand notation:
$$\lambda \mathcal{S}_i \equiv \{x : M_i x \leq \lambda d\} \tag{6}$$
3) The closed-loop predictions for a given K are:
$$\left.\begin{array}{c} x(k) = \Phi_i^k x(0) \\ u(k) = -K_i \Phi_i^{k-1} x(0); \end{array}\right\} \quad \Phi_i = A - BK_i \tag{7}$$

C. General Interpolation [1], [8]

Given a system (1), constraints (2), a set of asymptotically stabilizing feedback controllers $u(k) = -K_i x(k), i = 1, \ldots, n$ and corresponding invariant sets $\mathcal{S}_i$, consider the following decomposition:

$$x(0) = \sum_{i=1}^{n} x_i, \quad \text{with} \quad \left\{ \begin{array}{l} \sum_{i=1}^{n} \lambda_i = 1, \ \lambda_i \geq 0, \\ x_i \in \lambda_i \mathcal{S}_i \end{array} \right. \tag{8}$$

This decomposition can be performed iff $x \in \overline{\mathcal{S}}$,

$$\overline{\mathcal{S}} \triangleq \mathrm{Co}\{\mathcal{S}_1, \ldots, \mathcal{S}_n\} \tag{9}$$

Furthermore given (8) holds [1], the following control law ensures that x remains in $\overline{\mathcal{S}}$:

$$u(k) = -\sum_{i=1}^{n} K_i x_i, \tag{10}$$

More generally, define the input and state predictions as:

$$u(k) = -\sum_{i=1}^{n} K_i \Phi_i^k x_i; \quad x(k) = \sum_{i=1}^{n} \Phi_i^k x_i. \tag{11}$$

Lyapunov theory is used to compute on the infinite-horizon cost (3) as

$$J = \tilde{x}^{\mathrm{T}} P \tilde{x} = \sum_{k=0}^{\infty} x(k+1)^{\mathrm{T}} Q x(k+1) + u(k)^{\mathrm{T}} R u(k) \tag{12}$$

where $\tilde{x} = [x_1^{\mathrm{T}} \ \ldots \ x_n^{\mathrm{T}}]^{\mathrm{T}}$ and

$$P = \Gamma_u^{\mathrm{T}} R \Gamma_u + \Psi^T \Gamma_x^{\mathrm{T}} Q \Gamma_x \Psi + \Psi^{\mathrm{T}} P \Psi \tag{13}$$

with $\Psi = [(A - BK_1)^{\mathrm{T}} \ \ldots \ (A - BK_n)^{\mathrm{T}}]^{\mathrm{T}}$, $\Gamma_x = [I, \ldots, I]$, $\Gamma_u = [K_1, \ldots, K_n]$.

Algorithm 1 (GIMPC: MPC using general interpolation): Take a system (1), constraints (2), cost weighting matrices Q, R, controllers K_i and invariant sets $\mathcal{S}_i$ and compute a suitable P. Then, at each time instant, given the current state $x(0)$, solve the following QP optimisation:

$$\min_{x_i, \lambda_i} \ \tilde{x}^{\mathrm{T}} P \tilde{x}, \quad \text{subject to (8)}, \tag{14}$$

and implement the input $u = -\sum_{i=1}^{n} K_i x_i$.

Algorithm 1 guarantees recursive feasibility, constraint satisfaction and asymptotic stability and comprises algorithm 2.1 from [8] when the sets are defined as polyhedrals.

D. Weaknesses of GIMPC

The most obvious limitation is the restriction of feasibility to $\overline{\mathcal{S}}$. This paper shows that despite using polyhedral MAS, constraints (8) still imply conservative constraint handling and hence unnecessarily suboptimal performance and/or feasibility. Furthermore, it then goes on to show how one can replace the conservative constraint handling of general interpolation by non-conservative constraint handling, while using an almost identical interpolation and with two immediate benefits: (i) a reduction in optimisation complexity and (ii) an increase in feasible regions.

III. Insights into interpolation based MPC

This section gives new insights to existing interpolation algorithms [1], [8] which are then used to propose a modification to general interpolation with far wider feasibility.

A. New observations on [1]

General interpolation as first proposed was set in the robust case. As such there was a reliance on invariant ellipsoids. However the use of ellipsoids means that constraints are checked implicitly and not explicitly. The constraint of (8) in GIMPC ensures that the worst case combination of predictions does not meet a constraint, even though this worst case combination may not be possible. Hence, although $x \in \overline{\mathcal{S}}$ implies that constraints are satisfied, the contrary does not follow so $x \notin \overline{\mathcal{S}}$ does not imply constraints are violated. We will illustrate this conservatism with a simple example using only K_1, K_2 and state constraints:

Illustration: For convenience first normalise values by constraints as being $\pm 100\%$ of allowable ranges. Now consider the different trajectories hinted at in (11). One could conjecture that:

1) Maximum over k of $x_1(k)$ of $\lambda_1\%$ occurs at $k = 1$.
2) Maximum over k of $x_2(k)$ of $\lambda_2\%$ occurs at $k = 5$.

GIMPC would ensure that $\lambda_1 + \lambda_2 \leq 100\%$, even though these maxima occur at differ sampling instants. As such, the worst case of $x(k) = x_1(k) + x_2(k)$ may be far less than 100%.

Summary: General interpolation of [1] does implicit and not explicit constraint handling. This is a necessary consequence of using invariant ellipsoids. Hence even when $\sum_i \lambda_i = 1$, the actual trajectories may go no where near constraints. In fact, conservatism is introduced at two levels:

1) by the use of invariant ellipsoids so even $x \notin \mathcal{S}_i$ does not imply $u = -K_i x$ is infeasible,
2) by the use of condition (8) which sums worst case scenarios rather than doing explicit constraint handling.

B. General interpolation of [8]

One cause of conservatism was the use of ellipsoids, but for the nominal LTI case, one can replace ellipsoidal sets [8] by polyhedral invariant sets. One would expect that such an algorithm does explicit constraint handling though the inequalities implicit in the MAS. However, the authors used condition (8) as a means of establishing a proof for constraints satisfaction and therefore still used a sum of worst cases and not explicit constraint handling. As such the GIMPC algorithm 1 is only applicable to $\overline{\mathcal{S}}$, that is the convex hull of the underlying regions $\mathcal{S}_i$.

In order to extend feasibility beyond $\overline{\mathcal{S}}$, it is necessary to relax the condition (8) while still guaranteeing constraint satisfaction. The next section gives some evidence that this is possible.

C. One degree of freedom interpolations and insights

The simplest interpolation[1] uses just one d.o.f. [8] and uses just two possible control laws $K_1,\ K_2$. Here we give a brief summary of two algorithms presented in [8] because one uses a condition equivalent to (8); the other does not and has a larger feasible region.

1) The algorithms: Both deploy co-linear interpolation which means the current state is decomposed as follows:

$$x = x_1 + x_2; \quad x_1 = (1-\alpha)x; \quad x_2 = \alpha x; \quad 0 \leq \alpha \leq 1 \tag{15}$$

The control law is

$$u = -[(1-\alpha)K_1 + \alpha K_2]x. \tag{16}$$

The two algorithms differ only in how α is computed.

Algorithm 2 (ONEDOFa): α is determined from:

$$\min_{\alpha} \quad \alpha \quad \text{s.t.} \quad [M_1(1-\alpha) + M_2\alpha]x \leq d \tag{17}$$

Algorithm 3 (ONEDOFb): α is determined from an equivalent condition to (8):

$$\min_{\alpha,\beta} \quad \alpha \quad \text{s.t.} \quad \begin{cases} M_1(1-\alpha)x \leq (1-\beta)d \\ M_2\alpha x \leq \beta d \\ 0 \leq \beta \leq 1 \end{cases} \tag{18}$$

This is known to be solved by $\alpha = (\mu - 1)/(\mu - \lambda)$ where $\mu = \max(M_1 x), \quad \lambda = \max(M_2 x)$.

2) Differences: ONEDOFb is similar to GIMPC in that the following condition is analogous to (8):

$$x_1 \in (1-\beta)\mathcal{S}_1; \quad x_2 \in \beta\mathcal{S}_2; \quad 0 \leq \beta \leq 1 \tag{19}$$

ONEDOFa on the other hand checks the predictions explicitly as the rows of the inequality in (17) correspond exactly to constraint checks on predictions (11). The consequences, some alluded to in [8], are that:

1) ONEDOFa, at times, produces a smaller value for the optimum α and hence gives better performance.
2) The feasible region for ONEDOFb is a subset of that for ONEDOFa.
3) The limitations of known gaurantees of recursive feasibility with co-linear interpolation require condition (19).

D. Connections to general interpolation

The GIMPC algorithm was designed to overcome two weaknesses of simple ONEDOF interpolation.

1) Include the tail in the class of predictions, that is allow that $x_1(k+1) = \Phi_1 x_1(k)$, $x_2(k+1) = \Phi_2 x_2(k)$, meaning that an infinite-horizon input sequence is

[1]Recursive feasibility and convergence can be hard to establish a priori, but this topic is not relevant here.

implicitly used. This had the double benefit of allowing a simple proof of convergence and recursive feasibility.
2) Extend the feasible region from $\mathcal{S}_1 \bigcup \mathcal{S}_2$ to $\overline{\mathcal{S}}$.

However, GIMPC still insisted on constraints of the form of (19) and hence could give worse performance or even feasibility than ONEDOFa [8]. The objective of this paper is to formulate an algorithm similar to ONEDOFa, but using general interpolation, by relaxing constraint (8) and allowing explicit constraint handling. The expectation is larger feasible regions and hence a more competitive GIMPC algorithm.

IV. Improved general interpolation

This section introduces the proposed algorithm and establishes proofs of recursive feasibility and convergence. For simplicity just two controllers are used although extensions to more are obvious.

A. Explicit constraint handling with general interpolation

The feasible region for GIMPC is given from

$$\overline{\mathcal{S}} = \{x : \exists x_1, x_2, \text{ s.t. } \left\{ \begin{array}{l} M_1 x_1 \leq (1-\lambda) d \\ M_2 x_2 \leq \lambda d \\ x = x_1 + x_2 \\ 0 \leq \lambda \leq 1 \end{array} \right\}\} \quad (20)$$

By analogy with ONEDOFa and making note of remark 1, explicit (exact) constraint handling of predictions (11) could intuitively be performed using:

$$S_{G2} = \{x : \exists x_1, x_2, \text{ s.t. } \left\{ \begin{array}{l} M_1 x_1 + M_2 x_2 \leq d \\ x = x_1 + x_2 \end{array} \right\}\} \quad (21)$$

However, for recursive feasibility one needs to impose (19), which would again decrease the size of the feasible region. Therefore, a novel method for explicit constraint handling, leading to recursive feasibility, is introduced in the next section.

B. The proposed algorithm

We aim to construct a set of constraints, imposed on $x(k), x_1(k), x_2(k)$, with $x(k) = x_1(k) + x_2(k)$ such that $\forall i \geq k$ it is guaranteed that $x(i) \in \mathcal{X}, u(i) \equiv (-K_1 x_1(i) - K_2 x_2(i)) \in \mathcal{U}$, where $x_1(i), x_2(i)$ are recursively defined in an analog way to GIMPC as $x_1(i+1) = (A - BK_1)x_1(i), x_2(i+1) = (A - BK_2)x_2(i)$. We do this by constructing an augmented system $\mathbf{x}(k+1) = \Phi' \mathbf{x}(k)$, with augmented state $\mathbf{x}$ constructed as

$$\mathbf{x} = \left[\begin{array}{c} x \\ x_1 \end{array} \right], \quad (22)$$

and Φ' defined as

$$\Phi' = \left[\begin{array}{cc} A - BK_2 & B(K_2 - K_1) \\ 0 & A - BK_1 \end{array} \right]. \quad (23)$$

By imposing constraints

$$\left[\begin{array}{c} \underline{x} \\ \underline{u} \end{array} \right] \leq \left[\begin{array}{cc} I & 0 \\ -K_2 & K_2 - K_1 \end{array} \right] \mathbf{x} \leq \left[\begin{array}{c} \overline{x} \\ \overline{u} \end{array} \right], \quad (24)$$

and then calculating the corresponding MAS $\overline{M}\mathbf{x} \leq \overline{d}$ for the augmented system (23) using results from [5], an improved GIMPC algorithm using explicit constraint handling is obtained :

Algorithm 4 (GIMPC2: Extended general interpolation): Using the same notation as for GIMPC, at each time instant, given the current state x, solve the following optimisation problem:

$$\min_{x_i} \ \tilde{x}^{\mathrm{T}} P \tilde{x}, \quad \text{subject to} \quad \overline{M} \left[\begin{array}{c} x \\ x_1 \end{array} \right] \leq \overline{d}, \quad (25)$$

and implement the input $u = -K_1 x_1 - K_2 x_2$.

Theorem 1: The GIMPC2 Algorithm has guaranteed stability and recursive feasibility.

Proof: This follows standard arguments. Since $\overline{M}\mathbf{x} \leq \overline{d}$ is invariant by construction, one can always choose $x_1(k+1) = \Phi_1 x_1(k),\ x_2(k+1) = \Phi_2 x_2(k)$, which then again leads to satisfaction of the constraints at time $k+1$. Due to the construction of the constraints (24), it is guaranteed that if $\overline{M}\mathbf{x} \leq \overline{d}$ is satisfied, constraints (2) are also satisfied, which then proves recursive feasibility. Due to the fact that P satisfies (13), the optimal values of the objective function are guaranteed to be non-increasing, which then trivially proves asymptotic stability by means of a Lyapunov argument. □

C. Further observations

Earlier work relied on the assumption that when interpolating between two control laws, it was not unreasonable to expect the resulting feasible region the convex hull of the underlying MAS. More recently [8] a simple numerical example using ONEDOFa contradicted this, but the resulting feasible region was non-convex and it was difficult to draw any general conclusions.

This paper has sought to understand this conflict and give insight into what is the maximum feasible region when interpolating. What is clear is that this may be far larger than the convex hull of the underlying MAS. This observation is novel and potentially very useful as it implies one may be able to achieve larger feasible regions than originally anticipated from interpolation techniques which hence is a significant advance. The numerical examples following will demonstrate this potential.

Remark 2: The reader may still be puzzled as to how a simple mix of controllers could give such benefits. The reason for this is that the class of controllers is far larger than simply $K = (1-\alpha)K_1 + \alpha K_2$, because of the flexiblity in the state decomposition.

V. Numerical examples

In this section we use two examples to demonstrate the advantages of the proposed GIMPC2 algorithm. With each example we will compare GIMPC and GIMPC2 by way of: (i) feasible regions; (ii) closed-loop performance and (iii)

computational load. Some comparison is also made with the generic optimal MPC (OMPC) algorithm of [11]. Due to its reliance on ellipsoids the algorithm of [1] is not illustrated as it will obviously give smaller regions, even for symmetric constraints. The examples here deploy non-symmetric constraints.

A. Example 1

The model and symetrical constraints are given by :

$$A = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0.0787 \end{bmatrix} \tag{26}$$

$$\overline{u} = 1, \quad \underline{u} = -1, \tag{27}$$

$$-2 \leq [1,\ 1]x \leq 2 \tag{28}$$

The LQR-optimal controller is $K_1 = [2.828 \quad 2.826]^{\mathrm{T}}$ with $Q = \mathrm{diag}(1, 0),\ R = 0.1$; the second detuned controller $K_2 = [0.5543.015]^{\mathrm{T}}$. Both controllers are stabilizing.

1) Feasible Regions: Figure 1 gives the underlying MAS $\mathcal{S}_1,\ \mathcal{S}_2$, the convex hull $\overline{\mathcal{S}}$ in dark shading and the feasible region S_{G2} for GIMPC2 in light shading. It is clear that GIMPC2, as expected, has better feasibility than GIMPC.

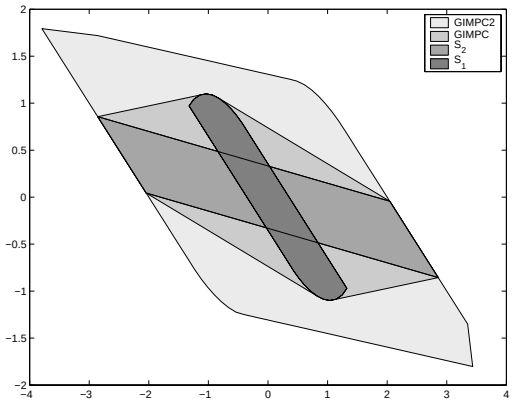


Fig. 1. Feasible region comparison of GIMPC and GIMPC2.

A comparison, in figure 2, with OMPC for a variety of numbers of d.o.f. $n_c = 1, 5, 10, 20$ shows that for $n_c < 20$, GIMPC2 has a larger feasible region.

2) Control Performance: Illustrations are meaningful only if the initial condition is within the feasible region of all algorithms to be compared. Here we take several initial points on the boundary of $\overline{\mathcal{S}}$; outside this region GIMPC2 is the only option and hence better. To give an overall picture the cost function J is evaluated for each closed-loop trajectory and compared to the global optimum (OMPC with large n_c). These costs are summed over all trajectories and normalised with respect to the smallest; the relative cost values are given in table 1.

The performance of GIMPC2 is consistently good, similar to GIMPC and close to the global optimal. This is not surprising because the global constrained optimal is not in the class

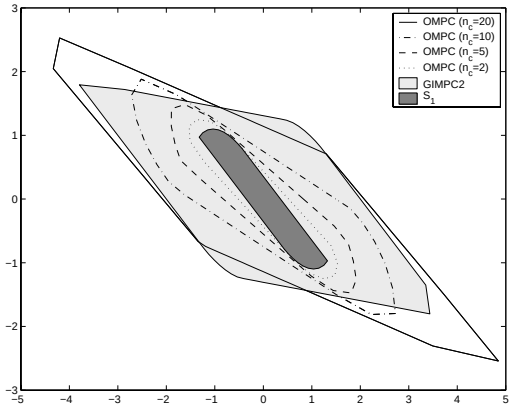


Fig. 2. Feasible region comparison of GIMPC2 and OMPC.

GIMPC	GIMPC2	OMPC ($n_c = 20$)
1.12	1.16	1

TABLE I

NORMALISED AVERAGE RUNTIME COSTS.

of predictions (the unconstrained optimal is) so one cannot make generic statements about how close the trajectories are to the optimal. As expected, all the trajectories, for all algorithms, remain feasible at all times.

3) Computational load: All three algorithms deploy a QP with similar numbers of constraints (OMPC is also based on $\mathcal{S}_1$, augmented by n_c further steps). Thus, we will discount the constraint complexity as an issue. Therefore, the only remaining comparison is of the numbers of d.o.f. utilised by each algorithm; this is summarised in table 2.

GIMPC	GIMPC2	OMPC
$n_x + 1$	n_x	$n_c \cdot n_u$

TABLE II

NUMBERS OF DEGREES OF FREEDOM ($n_x = 2$ FOR EXAMPLE 1).

GIMPC2 uses the fewest d.o.f. and interestingly gives large feasible regions with fewer d.o.f. than GIMPC and OMPC. OMPC could operate with about $n_c = 10$ to give a comparable region of attraction to GIMPC (see figure 2). The advantage over OMPC may not apply for systems with large state dimension.

B. Example 2

Example 2 has three states and one unstable mode. It is discussed briefly.

$$A = \begin{bmatrix} 0.98 & 0.2 & 0.19 \\ 0.075 & 0.607 & -0.4 \\ -0.3 & 0 & 0.607 \end{bmatrix}; \quad B = \begin{bmatrix} 0.5 \\ -0.21 \\ 0.39 \end{bmatrix} \tag{29}$$

$C = [1.69, 3.22, 0.3];\ D = 0;\ Q = Q_1 = C^{\mathrm{T}}C; C_2 = [3, 1, 2]; Q_2 = C_2^{\mathrm{T}}C_2;\ R = .1 = R_1 = R_2$.

Using the same notation as the earlier figures, the feasible regions for intersections with the principle 2D planes (with the other state set to zero) are given in figures 3a,b,c. Once again it is clear that GIMPC2 has allowed noticeable gains in feasibility and moreover has feasible regions of similar volume to OMPC with larger numbers of d.o.f. The reader may note that the GIMPC region seems larger than the convex hull of $\mathcal{S}_1,\ \mathcal{S}_2$; this is due to projection from the higher dimensional space.

To ensure that all algorithms are feasible, closed-loop simulations commence from points around the GIMPC boundary. The corresponding closed-loop simulation costs J are summed over all trajectories and normalized with respect to OMPC ($n_c = 20$); the relative cost values are given in table 2. GIMPC2 has outperformed GIMPC and performed close to the global optimum, despite using just 3 d.o.f. A similar comparison on the GIMPC2 feasibility boundary gave that GIMPC2 had a relative cost (to OMPC) of 1.0005!

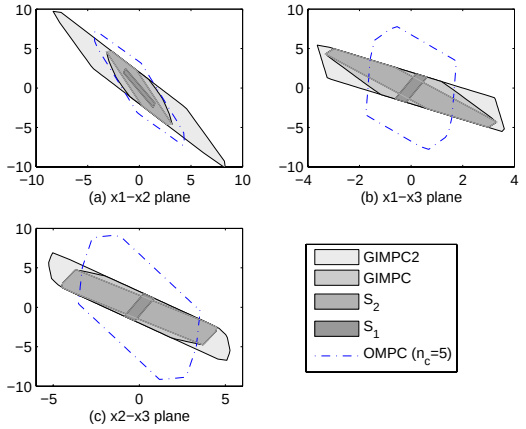


Fig. 3. Feasible region comparison.

J(GIMPC)	J(GIMPC2)	J(OMPC($n_c = 20$))
1.096	1.0014	1

TABLE III
NORMALISED AVERAGE RUNTIME COSTS ON GIMPC BOUNDARY.

VI. CONCLUSIONS AND FUTURE WORK

This paper exposes some of the weaknesses in general interpolation based MPC algorithms. The insight gained is used to propose an improved variant of general interpolation for LTI systems. Improvements are given by way of the volume of the feasible region, and this is due to elimination of the conservatism in the constraint handling.

The method is shown to have a guarantee of recursive feasibility and asymptotic stability and more significantly, the coding and set up (25) are simple. Illustrations on numerical examples demonstrate the improvements over pre-existing general interpolation algorithms. It is also worth pointing out that the new algorithm can cope efficiently with non-symmetrical state and input constraints which ellipsoidal based methods cannot.

General interpolation was originally concieved for the robust case (hence using ellipsoidal sets) and rather ironically applied to the LTI case using polyhedral sets later. The logical next step is to investigate whether the GIMPC2 algorithm, which relies on polyhedral MAS, can be applied to the robust case. A parallel paper looks at the technical details required for this.

VII. ACKNOWLEDGMENTS

Research supported by: Royal Academy of Engineering and the Royal Society. Research Council KULeuven: GOA-Mefisto 666, several PhD/postdoc & fellow grants; Flemish Government: FWO: PhD/postdoc grants, projects, G.0240.99, G.0407.02, G.0197.02, G.0141.03, G.0491.03, G.0120.03, research communities (ICCoS, ANMMM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants, Soft4s (softsensors), Belgian Federal Government: DWTC (IUAP IV-02 (1996-2001) and IUAP V-22 (2002-2006)), PODO-II (CP/40: TMS and Sustainibility); EU: CAGE; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; Contract Research/agreements: Data4s, Electrabel, Elia, LMS, IPCOS, VIB. B. Pluymers is a research assistant with the I.W.T. (Flemish Institute for Scientific and Technological Research in Industry) at the Katholieke Universiteit Leuven. J. Suykens is an associate professor and B. De Moor a full professor, both at the Katholieke Universiteit Leuven, Belgium.

REFERENCES

[1] Bacic, M. and Cannon, M. and Lee, Y. I. and Kouvaritakis, B., General Interpolation in MPC and Its Advantages, IEEE Transactions on Automatic Control, 48, 6, June, 2003, 1092–1096.
[2] Blanchini, F., Ultimate Boundedness Control for Uncertain Discrete-Time Systems via Set-Induced Lyapunov Functions, IEEE Transactions on Automatic Control, 1994, 39, 428-433
[3] Blanchini, F., Set Invariance in Control, Automatica, 1999, 35, 1747-1767
[4] Camacho, E. and C. Bordons, Model predictive Control, Springer Verlag, 1999
[5] Gilbert, E. G. and Tan. K. T., Linear Systems with State and Control Constraints : The Theory and Application of Maximal Output Admissable Sets, IEEE Transactions on Automatic Control, 1991, 36, 9, 1008-1020
[6] Kothare, M. V. and Balakrishnan, V. and Morari, M., Robust Constrained Model Predictive Control using Linear Matrix Inequalities, Automatica, 32, 1361-1379, 1996
[7] J. Richalet, A. Rault, J.L. Testud and J. Papon, Model predictive heuristic control: applications to industrial processes, Automatica, 14(5), 413-428, 1978.
[8] Rossiter, J.A. and Kouvaritakis, B. and Bacic, M., Interpolation based computationally efficient predictive control, International Journal of Control, 77, 290-301, 3, 2004
[9] J.A. Rossiter, P.W. Neal and L. Yao, Applying predictive control to fossil fired power station, Proceedings Inst. MC, 24(3), 177-194, 2002.
[10] Rossiter, J.A., Model based predictive control: a practical approach, CRC Press, 2003
[11] Scokaert, P.O.M. and Rawlings, J.B., Constrained linear quadratic regulation, IEEE Transactions on Automatic Control, 43, 8, 1163-1168
[12] Pluymers, B. and Rossiter, J.A. and Suykens, J.A.K. and De Moor, B., The Efficient Computation of Polyhedra Invariant Sets for Linear Systems with Polytopic Uncertainty, Accepted for publication in proceedings of ACC'05, Portland, USA, 2005
[13] Pluymers, B. and Roobrouck, L. and Buijs, J. and Suykens, J. A. K. and De Moor, B., Model-Predictive Control with Time-Varying Terminal Cost using Convex Combinations, Internal Report 04-28, ESAT-SISTA, K.U.Leuven (Leuven, Belgium), to appear in Automatica, 2005