19th World Congress of the International Federation of Automatic Control, 24-29 August 2015, Cape Town, South Africa.
IFAC Proceedings Volumes, Volume 47, Issue 3, 2014, Pages 2939-2944

Robust explicit MPC design under finite precision arithmetic

Andrea Suardi [*] **Stefano Longo** [**] **Eric C. Kerrigan** [***] **George A. Constantinides** [****]

[*] *Department of Electrical and Electronic Engineering, Imperial College London, London, SW7 2AZ, UK(e-mail: a.suardi@imperial.ac.uk).*
[**] *Department of Automotive Engineering, Cranfield University, Cranfield, MK43 0AL, UK (e-mail: s.longo@cranfield.ac.uk)*
[***] *Department of Electrical and Electronic Engineering and Department of Aeronautics, Imperial College London, London, SW7 2AZ, UK(e-mail: e.kerrigan@imperial.ac.uk).*
[****] *Department of Electrical and Electronic Engineering, Imperial College London, London, SW7 2AZ, UK(e-mail: g.constantinides@imperial.ac.uk).*

Abstract: We propose a design methodology for explicit Model Predictive Control (MPC) that guarantees hard constraint satisfaction in the presence of finite precision arithmetic errors. The implementation of complex digital control techniques, like MPC, is becoming increasingly adopted in embedded systems, where reduced precision computation techniques are embraced to achieve fast execution and low power consumption. However, in a low precision implementation, constraint satisfaction is not guaranteed if infinite precision is assumed during the algorithm design. To enforce constraint satisfaction under numerical errors, we use forward error analysis to compute an error bound on the output of the embedded controller. We treat this error as a state disturbance and use this to inform the design of a constraint-tightening robust controller. Benchmarks with a classical control problem, namely an inverted pendulum, show how it is possible to guarantee, by design, constraint satisfaction for embedded systems featuring low precision, fixed-point computations.

Keywords: Hardware/software co-design, Identification and control methods, Cyber-physical systems

1. INTRODUCTION

Since the widespread use of single- and double-precision floating-point arithmetic in computer architectures, control system designers routinely take the assumption that computation is performed with infinite numerical precision. The consequence is that the two activities of control system design and its implementation are often decoupled. This is safe for simple and well-understood algorithms. The control engineer worries about high-level issues, such as closed-loop performance, while the software engineer worries about implementation issues, such as code efficiency and timing.

In addition to high numerical precision, other factors such as high clock speed and small packaging have become standard features of modern embedded systems processors. Such advances in digital electronics (together with the development of sophisticated algorithms) have allowed the implementation of computationally-heavy control schemes in low-cost applications with fast dynamics. This has had the effect to reduce costs even further and to allow for fast computations (Jerez et al., 2013; Constantinides, 2009). This includes, for example, implementations with low number precision or fixed-point arithmetic. It is well-known that low precision, especially if implemented in fixed point, allows for much simpler circuits and greater computational speeds (Patterson and Hennessy, 1990). All of the above is at the expense of increased numerical errors that cannot and should not be ignored. There is a surprisingly small amount of theory for the design of such computer-based control systems. These issues could be considered as part of the emerging science called cyber-physical systems theory (Wolf, 2009). Cyber-physical Systems are integrations of computation with physical processes and therefore would also embrace the problem of control algorithm performance under numerical errors.

Model Predictive Control (MPC) is a powerful control scheme that, due to the necessity of solving an optimization problem every sampling instant, has only recently found application outside the process industry. One of the often ignored drawbacks of MPC, however, is its sensitivity to numerical errors (Hasan et al., 2013). The use of different discretization methods has been proven to be an advantage when working with low precision (Longo et al., 2014). Methods to avoid variable overflow have been proposed by constraining their ranges with carefully selected scaling methods (Jerez et al., 2013b). However, for these approaches, stability and constraint satisfaction are not guaranteed and, in practice, the only solution to this problem is extensive simulation analysis.

In this paper, we propose a method to guarantee hard constraint satisfaction of an explicit MPC scheme (Bemporad et al., 2002; Kvasnica and Fikar, 2010) when the algorithm is implemented on a platform using low precision arithmetic. The idea is to quantify the maximum error made by the processor when evaluating the control policy. This is done by applying forward error analysis (Higham, 2002) to the explicit MPC controller. Considering the error as an additive disturbance to the plant dynamics, a controller that is robust to such a disturbance is designed. The resulting controller will therefore be robust against its own finite-precision implementation in a true cyber-physical sense.

2. PROBLEM SETUP

Consider the discrete-time feedback control law

$$u_k := \kappa(x_k), \tag{1}$$

where $\kappa : \mathbb{R}^n \to \mathbb{R}^m$ is designed to stabilize and guarantee some performance for the discretized plant

$$x_{k+1} = Ax_k + Bu_k, \tag{2}$$

where n is the number of states, m the number of inputs inputs, and $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are the discretized plant matrices. At sample instant k the state vector $x_k \in \mathbb{R}^n$ is either measured or estimated.

When n and m are small but high sampling rates, good closed-loop performance and polyhedral constraint satisfaction are required, we can compute an MPC feedback controller κ explicitly by solving a multi-parametric optimization problem. This could be done, for example, by using the MATLAB Multi Parametric Toolbox (MPT) (Herceg et al., 2013). The resulting κ is a continuous piecewise affine (PWA) function defined over a polyhedral partition of the state space. Therefore, computing (1) requires: i) the solution of a point location problem to determine in which polytope – defined by a soft linear inequality ($Hx_k \leq k$) – the current state x_k belongs; ii) the evaluation of a control law of the form

$$u_k = Fx_k + g \tag{3}$$

associated with the selected region in i). A variety of algorithms have been proposed to solve the point location problem, since this is the most time-consuming task (Tøndel et al., 2002; Jones et al., 2006; Storace and Poggi, 2011; Monnigmann and Kastsian, 2011). Such algorithms range from simple ones (a *sequential* search through the regions of the partition) to more complex ones where the region is found via a binary *search tree.*

If infinite-precision arithmetic is available, the control action u_k can be computed exactly without introducing any numerical errors. However, computing u_k in a processor that works with finite precision (i.e. any standard processor) results in the introduction of an error. Such an error is the combination of two factors: first, the selection of the wrong region due to the the point location algorithm and second, numerical errors due to the computation of the control law in (3).

It should be noticed that this is not a particular feature of fixed-point arithmetic, since errors are also introduced when computations are performed in other (but finite) arithmetics, such as IEEE floating-point double-precision. However, in high precision, this error can often be safely ignored. Hence, in the sequel, when we refer to 'infinite precision' we are in practice performing computations in a desktop PC using floating-point double-precision (high enough not to cause noticeable failures in the practical problems we have studied).

The computational error cannot be computed exactly, because the error depends on the actual value of the current state x_k. However, as will be shown in Section 4, the error can be bounded. Finally, if we consider this error as a bounded additive disturbance w_k to the plant dynamic, such as

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{4}$$

a robust controller can be designed for which constraint satisfaction is guaranteed (Bemporad et al., 2003; Baotic et al., 2008; Kerrigan and Maciejowski, 2003; Kerrigan and Mayne, 2002). The interesting point here is that the newly designed robust controller will result in a new control scheme with possibly different error bounds. Hence, the proposed error analysis must be re-applied and the controller design process is repeated iteratively until convergence (Section 3). In practice, only a few design iterations are required. Constraint satisfaction for the resulting explicit MPC scheme is guaranteed for arbitrary low numerical precision, if a controller realization exists. We will carry out the analysis for a fixed-point implementation because this is more suitable for inexpensive applications with fast dynamics (Jerez et al., 2013). We will assume that enough bits are used for the integer part to avoid overflow. However, a similar procedure could be applied to other number representations, including floating point.

3. ITERATIVE CONTROLLER DESIGN

Let the chosen finite-precision implementation of the feedback control law (1), at time k, produce an error q_k. We can define the finite precision control action $\hat{u}_k$ as

$$\hat{u}_k := u_k + q_k, \tag{5}$$

where u_k is the control action obtained if computations were performed in infinite precision. By applying (5) to the discrete plant dynamic we get

$$\begin{aligned} x_{k+1} &= Ax_k + B\hat{u}_k = Ax_k + Bu_k + Bq_k \\ &= Ax_k + Bu_k + w_k, \end{aligned} \tag{6}$$

where $w_k \in \mathbb{R}^n$ and $w_k := Bq_k$ is an additive state disturbance to the plant states. Equation 6 is in the same form as (4) and therefore, if upper and lower bound vectors on w_k are known ($\overline{w}$ and $\underline{w}$, respectively) a robust MPC law can be designed using a minimax approach (Bemporad et al., 2003). However, such a controller may produce numerical errors with different bounds. To design an MPC law that is robust to the numerical errors introduced by the algorithm itself, we have devised an iterative design procedure that: i) designs a nominal controller for a plant without disturbances; ii) evaluates the error introduced by the finite-precision implementation of the controller; iii) re-designs a controller that is robust against the calculated error; iv) re-evaluates the error introduced by the finite-precision implementation of this new controller; v) repeats the process iteratively until the controller-design process has converged. This procedure is detailed in Algorithm 1, where *tol* is the algorithm exit condition. Although a mathematical proof of the algorithm's convergence is not provided (and difficult to formulate, since every iteration involves the solution of a multi-parametric optimization problem), experience shows that convergence is achieved

Algorithm 1

```
Set state disturbance bounds: $\overline{w} := 0$ and $\underline{w} := 0$
while (1) do
    DESIGN A ROBUST CONTROLLER with
    disturbance bounds $\overline{w}$ and $\underline{w}$
    COMPUTE ERROR BOUND $\underline{w}_{new}$ and $\overline{w}_{new}$
    (Section 4)
    if ($\|\overline{w}_{new} - \overline{w}\|_2 \leq tol$) and
        ($\|\underline{w}_{new} - \underline{w}\|_2 \leq tol$) then break
    else
        $\overline{w} := \overline{w}_{new}$ and $\underline{w} := \underline{w}_{new}$
    end if
end while
```

in only a few iterations (typically less than 5). This can be explained by the fact that two robust explicit MPC formulations do not differ much from each other if their disturbances are similar.

4. ERROR BOUND COMPUTATION

Computing the bounds $\underline{w}$ and $\overline{w}$ (in Algorithm 1) means investigating all the possible sources of the control law error q_k. As mentioned before, this error is given by: i) the selection of a wrong region due to quantization of the (measured or estimated) current state x_k and/or quantization of the polyhedral partition; ii) numerical errors introduced by the computation of the control law (3) using finite precision.

The analysis of these two sources of error will be considered in Sections 4.1 and 4.2, respectively. The quantification of the error bounds, achieved by solving an optimization problem, will be given in Section 4.3.

4.1 Point Location Algorithm Analysis

Using a finite-precision arithmetic implementation of a *sequential* search through the polyhedral partition might result in an error when selecting a region. This happens because overlaps and/or uncovered areas might occur. Figure 1-b shows that the state space for a finite-precision implementation might not be covered as it would be in the infinite precision case (Figure 1-a).

On the other hand, when using a *search tree* point location algorithm, it is possible to avoid the above limitations. This is because a *search tree* has one and only one leaf that is reachable. Each leaf is defined as a unique convex hull, made of the hyperplanes encountered while traversing the tree. The state space is uniquely and fully covered by all leaves. This is shown graphically in Figures 1-c (infinite precision) and 1-d (finite precision). In Figure 1-d (finite precision) overlaps and uncovered areas do not occur. An explanation for this follows.

Compared to the case of infinite precision, in finite precision some leaves might not be reachable due to numerical errors. This becomes more likely to happen as the arithmetic precision is reduced. Nevertheless, this is not a problem. In finite precision, the reachable convex hulls are able to cover the areas that were associated with the unreachable leaves. However, even if the *search tree* point location algorithm can guarantee to locate a unique region (this is not true for the *sequential* search), a control law error q_k might still occur. This is because a different region

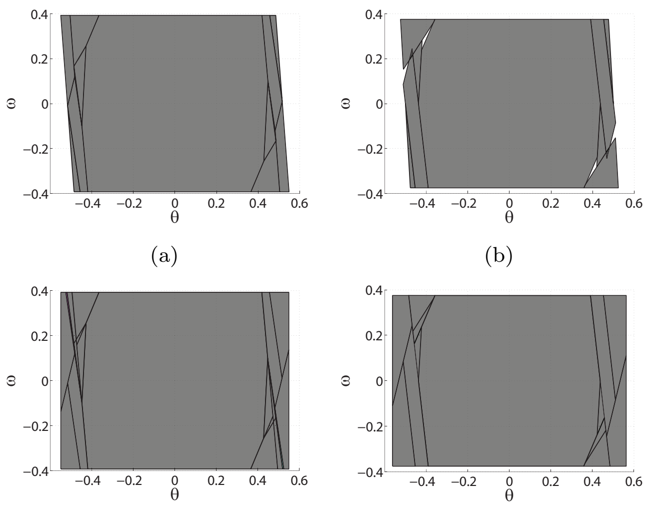


Fig. 1. Explicit MPC polyhedral partition of the inverted pendulum system (14) generated with MPT: (a) *sequential* search for point location with *infinite* precision (double floating-point); (b) *sequential* search for point location with *finite* precision (fixed point, 4 bits fraction length); (c) *binary search* tree for point location with *infinite* precision (double floating-point); (d) *search tree* for point location with *finite* precision (fixed point, 4 bits fraction length).

would have been selected compared to the infinite precision case, due to numerical errors introduced when representing the convex hulls associated with the leaves.

Let us define $\mathbb{P} := \{x|Mx \leq s\}$ to be the convex hull associated with a leaf of the *search tree* implemented in infinite precision and $\hat{\mathbb{P}} := \{\hat{x}|\hat{M}\hat{x} \leq \hat{s}\}$ to be the convex hull associated with a leaf of the *search tree* in finite precision. An error in the region identification can occur if the convex hull, resulting from the intersection $\mathbb{P} \cap \hat{\mathbb{P}}$, contains at least one finite precision value of the state vector x_k. However, this is only true when $\mathbb{P}$ and $\hat{\mathbb{P}}$ do not represent the same convex hull.

4.2 Function Evaluation Analysis

Once a region is selected, the associated control law is computed. Here, errors are introduced because of the finite-precision arithmetic used to perform the algebraic operations.

Let us define the finite-precision representation $\hat{\alpha}$ of a real number $\alpha \in \mathbb{R}$ as

$$\hat{\alpha} := \alpha + \epsilon_{\hat{\alpha}} \tag{7}$$

where $\epsilon_{\hat{\alpha}} \in \mathbb{R}$ is the quantization error. If a fixed-point representation is used, the quantization error due to truncation is $\epsilon_{\hat{\alpha}} \in \left(-2^{-l}, 0\right]$ and $l \in \mathbb{N}^+$ is an integer that defines the fraction length, in terms of the number of bits. The assumption here is that we use enough bits for the integer part so that overflow does not occur.

By applying the finite-precision representation (7) to the control law (3) we define the finite-precision control action $\hat{u}_k$ associated with the convex hull $\hat{\mathbb{P}}$ as

$$\hat{u}_k = \hat{F}\hat{x}_k + \hat{g}, \tag{8}$$

where

$$\hat{x}_k = x_k + e_{\hat{x}_k}, \tag{9}$$

and $\hat{F}$ and $\hat{g}$ are the finite-precision representation of F and g, respectively. The values in $\hat{F}$ and $\hat{g}$ can be computed exactly, since their infinite-precision values are known and fixed. The state vector x_k, however, is unknown and thus is its quantization error vector $e_{\hat{x}} \in \mathbb{R}^n$.

Considering the infinite-precision control action (3) associated with the convex hull $\mathbb{P}$ and substituting (8) into (5), we can compute the control law error q_k and express the additive state disturbance w_k to the plant as

$$\begin{aligned} w_k = Bq_k &= B\left(\hat{u}_k - u_k\right) \\ &= B\left[\left(\hat{F} - F\right)\hat{x}_k + Fe_{\hat{x}_k} + \left(\hat{g} - g\right)\right]. \end{aligned} \tag{10}$$

The state disturbance w_k cannot be computed exactly, because the value of $\hat{x}_k$ and $e_{\hat{x}_k}$, as well as the error in the point location algorithm, are unknown. Hence, we propose to compute the disturbance bounds $\overline{w}$ and $\underline{w}$ based on the the worst case scenario. This can be done by computing the maximum $\overline{w}(n_i)$ and minimum $\underline{w}(n_i)$ for each element $n_i \in \{1, 2, \ldots n\}$ of the disturbance vector w_k (Section 4.3) among all possible errors.

Let us define index $i \in \{1, 2, \ldots N_{leaf}\}$ to denote $\mathbb{P}^i$, the convex hull associated to leaf i of the *search tree* implemented in *infinite* precision, and index $j \in \{1, 2, \ldots \hat{N}_{leaf}\}$ denote $\hat{\mathbb{P}}^j$, the convex hull associated with leaf j of the *search tree* in *finite* precision. Here, N_{leaf} and $\hat{N}_{leaf}$ are the number of leaves of the *search tree* in infinite and finite precision, respectively. Therefore, every possible permutation of i and j has to be exhaustively investigated to determine when the point location algorithm makes an error when selecting the region. The procedure is outlined in Algorithm 2.

Algorithm 2

```
for $n_i = 1, 2, ..., n$ do
    for $i = 1, 2, ..., N_{leaf}$ do
        for $j = i, i + 1, ..., \hat{N}_{leaf}$ do
            if $\exists\, \hat{x}_k \subset \mathbb{P}^i \cap \hat{\mathbb{P}}^j$ then
                COMPUTE BOUNDS $\overline{w}^{ij}(n_i)$ and $\underline{w}^{ij}(n_i)$
                associated to $\mathbb{P}^i \cap \hat{\mathbb{P}}^j$
                (Section 4.3)
            end if
        end for
    end for
    $\overline{w}(n_i) := \max_{i,j} \overline{w}^{ij}(n_i)$ and $\underline{w}(n_i) := \min_{i,j} \underline{w}^{ij}(n_i)$
end for
```

4.3 Maximum and Minimum Bound Computation

Based on the above considerations, the task of computing the upper $\overline{w}^{ij}(n_i)$ and lower $\underline{w}^{ij}(n_i)$ bounds associated with the polytope intersection $\mathbb{P}^i \cap \hat{\mathbb{P}}^j$ is translated into solving two optimization problems: a maximization and a minimization, respectively.

As an example, let us consider the maximization problem (a similar approach can be used for the minimization). Because the state $\hat{x}_k$ has fixed-point values, it can be scaled by a factor of 2^l and expressed as an integer $z_k \in \mathbb{Z}^n$, i.e.

$$z_k = \hat{x}_k \cdot 2^l. \tag{11}$$

This will lead to the mixed-integer linear programming (MILP) problem

$$\begin{aligned} \max_{z_k, e_{\hat{x}_k}} \quad & b_{n_i}\left[\left(\hat{F}^j - F^i\right)2^{-l}z_k + F^i e_{\hat{x}_k} + \left(\hat{g}^j - g^i\right)\right] \\ s.t. \quad & z_k \in \mathbb{Z}^n & \text{(12a)} \\ & z_k 2^{-l} \in \mathbb{P}^{ij} & \text{(12b)} \\ & -2^{-l} < e_{\hat{x}_k} \leq 0, & \text{(12c)} \end{aligned}$$

where z_k is the integer decision variable and the vector b_{n_i} is the row n_i of matrix B. The scaling factor 2^l is also applied to the maximization function and to the left-hand-side of the inequality constraint (12b).

The solution of a MILP problem is computationally demanding (Vivek and Pistikopoulos, 2000), especially when the number of bits l used for the fraction length is large. This is because the search space area (given by the inequality constraints (12b)) increases proportionally with 2^l. Our proposed solution is to assume that the variable $\hat{x}_k$ is continuous. This will allow us to solve, instead of the NP-hard MILP in (12), the P-hard linear programming (LP) problem

$$\begin{aligned} \max_{\hat{x}_k, e_{\hat{x}_k}} \quad & b_{n_i}\left[\left(\hat{F}^j - F^i\right)\hat{x}_k + F^i e_{\hat{x}_k} + \left(\hat{g}^j - g^i\right)\right] \\ s.t.: \quad & \hat{x}_k \in \mathbb{P}^{ij} & \text{(13a)} \\ & -2^{-l} < e_{\hat{x}_k} \leq 0. & \text{(13b)} \end{aligned}$$

The solution, as shown for an example in Figure 2, will be a worst case approximation of the real solution provided by (12). This is admissible, although slightly more conservative, because we are computing a bound.

5. EXPERIMENTAL RESULTS

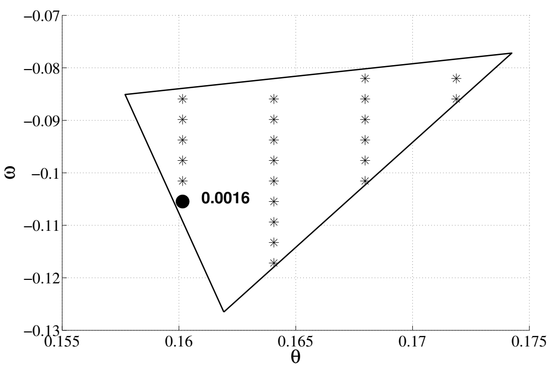
For benchmarking purposes, a classical unstable control problem has been considered: an inverted pendulum system. The design procedure presented in Sections 3 and 4 has been implemented using MATLAB. The MPT function `mpt_control` has been employed to generate the robust explicit MPC controller (Bemporad et al., 2003). The fixed-point toolbox has been used for the error analysis and closed-loop simulations. In order to verify the correctness of the proposed design methodology, extensive closed-loop simulations using 'infinite' (double precision floating-point) and finite (fixed-point) precision arithmetic from various initial conditions to the steady state have been performed. Also, various controller designs based on different precisions have been verified.

Consider an inverted pendulum system described by the continuous-time dynamics

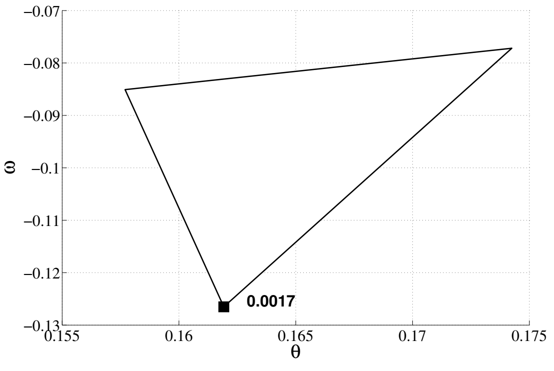
$$\begin{bmatrix} \dot{\vartheta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \dfrac{g}{L} & -\dfrac{b}{mL^2} \end{bmatrix} \begin{bmatrix} \vartheta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{mL^2} \end{bmatrix} u, \tag{14a}$$

$$y = [1\ 0] \begin{bmatrix} \vartheta \\ \omega \end{bmatrix}, \tag{14b}$$

where the states ϑ and ω are, respectively, the angular displacement measured from the equilibrium position and the angular velocity; u is the input torque, $g = 9.81\text{m/s}^2$ the gravitational force, $m = 344\text{kg}$ the mass, $b = 0.48\text{Ns/m}$ the rotation friction and $L = 1.703\text{m}$ the length of the pendulum. Given the continuous-time weight matrix



(a)



(b)

Fig. 2. Maximization optimization problem solutions: (a) MILP problem; (b) LP problem. Here, stars represent the feasible value of $\hat{x}_k$ inside the polytopes intersection $\mathbb{P}^i \cap \hat{\mathbb{P}}^j$. The circle represents the solution of the MILP problem in (a) and the square the solution of the LP in (b).

on the states $Q_c = I$ and on the inputs $R_c = 0.1$, the continuous-time plant matrices and weight matrices have been discretized with a sampler with period $T_s = 0.1$s and a zero-order-hold. An explicit MPC controller has been formulated with a prediction horizon of $T = 0.40$s. State constraints have been set to

$$\begin{bmatrix} -\pi \\ -\pi/8 \end{bmatrix} \leq \begin{bmatrix} \vartheta \\ \omega \end{bmatrix} \leq \begin{bmatrix} \pi \\ \pi/8 \end{bmatrix}. \quad (15)$$

Upper and lower bounds on the state disturbance have been computed with the proposed iterative control design approach – Algorithm 1 – for different finite precision (fixed-point) representations, varying the number of bits l of the fraction length. As an example, Figure 3 shows the norm of the lower disturbance bound $\|\underline{w}\|_2$ against the algorithm's iterations. Experimentally, it has been observed that the proposed iterative algorithm converges after few iterations and that the magnitudes of the computed bounds have the same order of the fixed-point quantization error (2^{-l}) and decrease as the computation precision increases, as expected. For this particular test case, we have experienced that, with a precision lower than 8 bits, it is not possible to generate a feasible robust controller. This is because the maximum arithmetic error introduced by such a low precision is too large, and hence also the resulting state disturbance bounds.

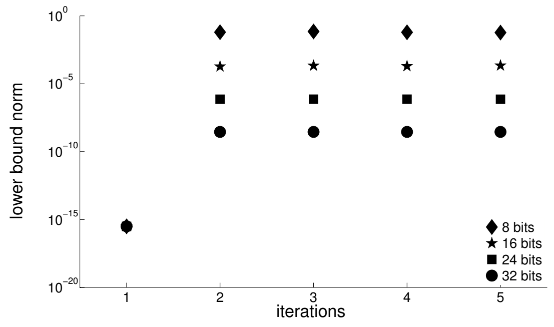
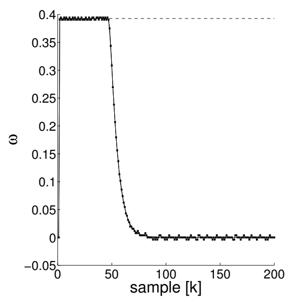
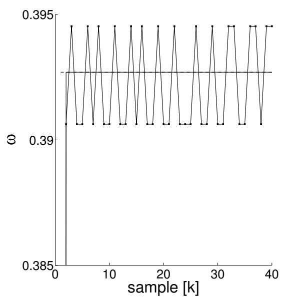


Fig. 3. Values of lower bound state disturbance norm $\|\underline{w}\|_2$ solving the (LP) problem for every algorithm's iterations using different fixed-point precisions (fraction length 8, 16, 24, 32 bits): inverted pendulum case



(a)



(b)

Fig. 4. Evolution of the state ω of the inverted pendulum system during a closed-loop simulation using floating-point double precision (continuous line) and fixed point with 8-bit fraction length (squares) when the controller have been designed robust to the error introduced by infinite precision arithmetic. The dashed line represents the state constraint. (a) full simulation; (b) detail when the state constraint is activated, shows that using fixed-point arithmetic constraints violation occurs.

Let us consider the design of an explicit MPC controller that is robust to the (very small) error introduced by infinite-precision arithmetic (in practice, floating-point double precision), which will be almost equivalent to the design of a 'non-robust' explicit MPC. If this controller is implemented on-line using finite-precision arithmetic (fixed-point), constraint violations might occur due to the numerical errors arising form the implementation. The lower the precision, the larger the errors, as shown for an example in Figure 4b.

On the other hand, if it is known that finite-precision arithmetic will be used, then the controller can be designed using Algorithm 1. Constraint satisfaction can be guaranteed by design only if the precision used in the on-line computation is the same of the one used in the design phase. Figure 5 shows this for a closed-loop simulation of two different robust controllers, designed with an 8- and a 12-bit fixed-point arithmetic, respectively. For a lower precision, the computed bound on the state disturbance (or the amount of the constraint tightening) is larger.

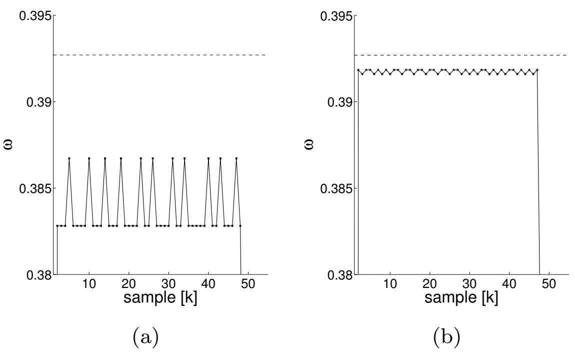


Fig. 5. Evolution of the state ω of the inverted pendulum system during a closed-loop simulation using the same precision that has been used to design the robust controller. The dashed line represents the state constraint. (a) using 8-bit fixed-point fraction length; (b) using 12-bit fixed-point fraction length. The dashed line represents the state constraint.

6. CONCLUSIONS

We have proposed, and verified via an inverted pendulum control problem, an explicit MPC design which robustly guarantees hard constraint satisfaction in the presence of finite-precision arithmetic errors introduced by the controller's own implementation. For a given controller formulation, we have shown how to calculate the maximum computational error introduced by the finite-precision implementation, via the solution of an optimization problem. Such an error has then been used to bound a state disturbance on the plant, by calculating its effect when propagated through the plant model. The constraint-tightening robust controller was designed iteratively in order to account for the new error introduced by the previous design. Design convergence was shown experimentally.

ACKNOWLEDGEMENTS

This research has been supported by the EPSRC Impact Acceleration grant number EP/K503733/1.

REFERENCES

Baotic, M., Borrelli, F., Bemporad, A., and Morari, M. (2008). Efficient On-Line Computation of Constrained Optimal Control. *SIAM Journal on Control and Optimization*, 47(5), 2470–2489.

Bemporad, A., Borrelli, F., and Morari, M. (2003). Min-max control of constrained uncertain discrete-time linear systems. *IEEE Transactions on Automatic Control*, 48(9), 1600–1606. doi:10.1109/TAC.2003.816984.

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

Constantinides, G.A. (2009). Tutorial paper: Parallel architectures for model predictive control. In *Proc. European Control Conference 2009*, 138–143. Budapest.

Hasan, A., Kerrigan, E.C., and Constantinides, G.A. (2013). Control-theoretic forward error analysis of iterative numerical algorithms. *IEEE Transactions on Automatic Control*, 58(6), 1524–1529.

Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *Proc. of the European Control Conference*, 502–510. Zürich, Switzerland. `http://control.ee.ethz.ch/ mpt`.

Higham, N.J. (2002). *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition.

Jerez, J.L., Constantinides, G.A., and Kerrigan, E.C. (2013b). A low complexity scaling method for the lanczos kernel in fixed-point arithmetic. *IEEE Transactions on Computers*, 99(PP), 1. doi:10.1109/TC.2013.162.

Jerez, J.L., Goulart, P.J., Richter, S., Constantinides, G.A., Kerrigan, E.C., and Morari, M. (2013). Embedded online optimization for model predictive control at megahertz rates. *CoRR*, abs/1303.1090.

Jones, C., Grieder, P., and Rakovic, S. (2006). A logarithmic-time solution to the point location problem for parametric linear programming. *Automatica*, 42(12), 2215–2218. doi: http://dx.doi.org/10.1016/j.automatica.2006.07.010.

Kerrigan, E.C. and Maciejowski, J. (2003). Robustly stable feedback min-max model predictive control. In *American Control Conference, 2003. Proceedings of the 2003*, volume 4, 3490–3495 vol.4.

Kerrigan, E.C. and Mayne, D. (2002). Optimal control of constrained, piecewise affine systems with bounded disturbances. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 2, 1552–1557 vol.2.

Kvasnica, M. and Fikar, M. (2010). Performance-lossless complexity reduction in explicit MPC. In *49th IEEE Conference on Decision and Control (CDC)*, 5270–5275.

Longo, S., Kerrigan, E.C., and Constantinides, G.A. (2014). Constrained LQR for low-precision data representation. *Automatica*, 50(1), 162 – 168. doi: http://dx.doi.org/10.1016/j.automatica.2013.09.035.

Monnigmann, M. and Kastsian, M. (2011). Fast explicit MPC with multiway trees. In *Proc. of the 18th IFAC World Congress, 2011*.

Patterson, D.A. and Hennessy, J.L. (1990). *Computer architecture: a quantitative approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Storace, M. and Poggi, T. (2011). Digital architectures realizing piecewise-linear multivariate functions: Two FPGA implementations. *Int. J. Circuit Theory Appl.*, 39(1), 1–15.

Tøndel, P., Johansen, T.A., and Bemporad, A. (2002). Evaluation of piecewise affine control via binary search tree. *Automatica DOI:10.1016/S0005-1098(02)00308-4*.

Vivek, D. and Pistikopoulos, E.N. (2000). An algorithm for the solution of multiparametric mixed integer linear programming problems. *Annals of operations research*, 99(1-4), 123–139.

Wolf, W. (2009). Cyber-physical systems. *Computer*, 42(3), 88–89.