This document is downloaded from DR-NTU, Nanyang Technological University Library, Singapore.

Title	Robust multiplexed model predictive control
Author(s)	Richards, Arthur; Ling, Keck-Voon; Maciejowski, Jan
Citation	Richards, A., Ling, K.-V., & Maciejowski, J. (2007). Robust multiplexed model predictive control. Proceedings of the European Control Conference 2007, 441-446. doi:10.23919/ECC.2007.7068779
Date	2015
URL	http://hdl.handle.net/10220/47444
Rights	© 2007 EUCA. All rights reserved. This paper was published in Proceedings of the European Control Conference 2007 and is made available with permission of EUCA.

Proceedings of the European Control Conference 2007
Kos, Greece, July 2-5, 2007

Robust Multiplexed Model Predictive Control[†]

Arthur Richards*
Department of Aerospace Engineering
University of Bristol
United Kingdom

Keck-Voon Ling
School of Electrical and
Electronics Engineering
Nanyang Technological University
Singapore

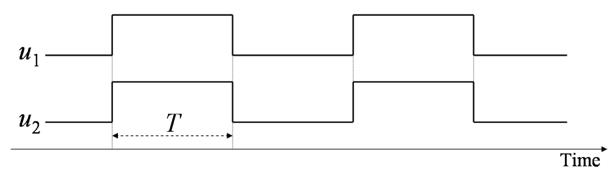
Jan Maciejowski
Department of Engineering
University of Cambridge
United Kingdom

***Abstract*— This paper extends the recently developed *multiplexed model predictive control* (MMPC) concept to ensure satisfaction of hard constraints despite the action of persistent, unknown but bounded disturbances. MMPC uses asynchronous control moves on each input channel instead of synchronised moves on all channels. It offers reduced computation, by dividing the online optimisation into a smaller problem for each channel, and potential performance improvements, as the response to a disturbance is quicker, albeit via only one channel. Robustness to disturbances is introduced using the *constraint tightening* approach, tailored to suit the asynchronous updates of MMPC and the resulting time-varying optimisations. Numerical results are presented, involving a simple mechanical example and an aircraft control example, showing the potential computational and performance benefits of the new robust MMPC.**

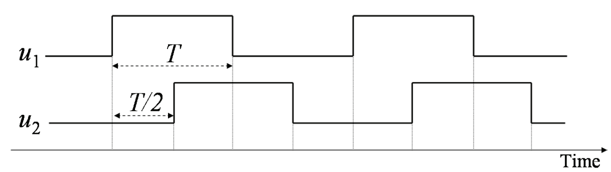
I. Introduction

Model Predictive Control (MPC) has become a popular control technology in the process industry [1] and has recently been proposed for systems with faster dynamics [2], [3]. MPC operates by solving an optimisation problem online, in real time, to determine a plan for future operation. Only an initial portion of that plan is implemented, and the process is repeated, re-planning when new information becomes available. Since numerical optimisation naturally handles hard constraints, MPC offers good performance while operating close to constraint boundaries [4].

Solving a numerical optimisation can be a complex problem, and for situations in which computation is limited, the time to find the solution can be the limiting factor in the choice of the update interval. This can impact upon controller performance, especially for systems with fast dynamics. Recent work [5], [6] has proposed the concept of *multiplexed MPC* (MMPC), in which control moves are applied to each channel asynchronously, as shown in Fig. 1, instead of applying synchronous moves on all channels. Furthermore, the computation is divided into a sequence of optimisations, one for each channel. While the total number of control moves in a given period remains the same, two key benefits result. First, the total computational complexity decreases. The time to solve an optimisation typically grows as $O(m^3)$, where m is the number of inputs. Therefore, solving m problems with one input is typically faster than solving one problem with m inputs. Second, there is a reduction in the delay between the onset of a disturbance and a response through the control. The immediate reaction under MMPC has restricted authority, as it operates through only one channel, but performance benefits can be achieved through this faster response.



(a) Conventional "synchronous" MPC



(b) Multiplexed MPC

Fig. 1. Example Timing Diagrams for Two-Input System

The contribution of this paper is the extension of MMPC to guarantee robust constraint satisfaction and feasibility of all optimisations despite the action of unknown but bounded disturbances. These are key issues in MPC: performance benefits are achieved by operating close to constraint boundaries, but when the state evolution no longer matches the predictions, constraint violation and infeasibility can result. Many methods have been developed to endow conventional synchronous MPC with robustness [7], [8]. For use with MMPC, we have adopted the *constraint tightening* approach [9]–[12], in which the constraints of the optimisation are modified to retain a margin for future feedback action. Since only the constraint limits are modified, the computational complexity remains the same as for the equivalent nominal MPC. Constraint tightening is therefore well-suited to MMPC, which is aimed at computation-limited applications.

Multiplexed MPC is related to distributed MPC (DMPC) [13], both dividing the optimisation into smaller sub-problems. Various approaches to robust

[†] This work was supported in part by A*STAR Project "Model Predictive Control on a Chip" (Ref: 052-118-0059)

*Corresponding author. Email: arthur.richards@bristol.ac.uk

ISBN 978-3-9524173-8-6

DMPC have been investigated, including worst-case predictions [14], retention of "emergency" plans [15], [16], invariant "tube" predictions [17] and constraint tightening [18]. Work on DMPC has typically focussed on spatially distributed systems with some structure in the system, *e.g.* teams of vehicles with decoupled dynamics. In contrast, the new robust multiplexed MPC makes no assumptions on the overall system structure, and considers temporal distribution, breaking the optimisation down into a sequence of smaller problems on the same processor.

The paper begins with the problem statement in Section II, including the requirements for multiplexed control moves. Then the controller formulation for MMPC is developed in Section III in two stages, first dealing with the multiplexed moves and then dividing into smaller sub-problems. Finally, Section IV presents the results of two sets of numerical simulations. Examples involving a simple spring-mass system demonstrate the scalability of MMPC in terms of computation, compared to conventional MPC. Examples involving the longitudinal control of an aircraft illustrate the potential performance benefits of MMPC when applied to systems with fast dynamics.

II. Problem Statement

Consider the discrete-time linear system in state space form with m scalar control inputs:

$$x_{k+1} = Ax_k + \sum_{j=1}^{m} B_j \Delta u_{j,k} + Ew_k. \tag{1}$$

where $x_k \in \Re^n$ is the state at time step k and $\Delta u_{j,k} \in \Re$ is the control move applied to channel j at time step k, *i.e.* $\Delta u_{j,k} = u_{j,k} - u_{j,k-1}$. The disturbance w_k is unknown but obeys the following bound

$$w_k \in \mathcal{W} \ \forall k \tag{2}$$

where $\mathcal{W}$ is a known, bounded set. The system is required to obey the following state constraints

$$x_k \in \mathcal{X} \ \forall k. \tag{3}$$

The following indexing function identifies the input channel to be moved at each step

$$\sigma(k) = (k \bmod m) + 1 \tag{4}$$

so that the asynchronous nature of the multiplexed control moves, as illustrated in Fig. 1(b), is captured by the constraint

$$\Delta u_{j,k} = 0 \text{ if } j \neq \sigma(k). \tag{5}$$

It is then possible to rewrite the system dynamics (1) as a periodic time-varying system

$$x_{k+1} = Ax_k + B_{\sigma(k)} \Delta \tilde{u}_k + Ew_k \tag{6}$$

where $\Delta \tilde{u}_k = \Delta u_{\sigma(k),k}$. This form will be used for predictions in the optimisations and illustrates how MMPC can draw on results for periodic time-varying systems.

III. Controller Formulations

This section develops the new robust MMPC formulation in two stages. The first, referred to as Scheme 1, uses the multiplexed control moves seen in Fig. 1(b) but optimises simultaneously for moves on all channels. The constraint modifications for robust feasibility are described. Then in Scheme 2, the optimisation is divided such that the problem at each step solves only for moves on the next channel to be updated. This may be regarded as a specialisation of Scheme 1 and inherits the property of robust feasibility.

Stability can be established using the results of Refs. [5], [6] to evaluate the cost function for the nominal case, together with those of Ref. [11] to take account of constraint tightening. We omit the details for the sake of brevity.

The main contribution of this paper is Scheme 2, which offers both performance and computational benefits over conventional MPC. However, Scheme 1 is still required to initialise the Scheme 2 controller.

A. Scheme 1

The controller for Scheme 1 is based on an optimisation of the control moves of all channels over a horizon of N steps. The decision variable is therefore $\mathbf{\Delta U}_k = (\Delta \tilde{u}_{k|k} \ \Delta \tilde{u}_{k+1|k} \ \Delta \tilde{u}_{k+2|k} \ \ldots \ \Delta \tilde{u}_{k+N-1|k})^T$ where $\Delta \tilde{u}_{k+i|k}$ denotes the prediction made at time k of a control move to be executed at time $k+i$. The optimization solved at every step is

$$\min_{\mathbf{\Delta U}_k} V_{\sigma(k+N)}(x_{k+N|k}) + \sum_{i=0}^{N-1} ||x_{k+i|k}||_q^2 + ||\Delta \tilde{u}_{k+i|k}||_r^2 \tag{7}$$

subject to $\forall i \in \{0, \ldots, N-1\}$

$$x_{k+i+1|k} = Ax_{k+i|k} + B_{\sigma(k+i)} \Delta \tilde{u}_{k+i|k} \tag{8a}$$

$$x_{k+N|k} \in \mathcal{T}_{\sigma(k)} \tag{8b}$$

$$x_{k|k} = x(k) \tag{8c}$$

$$x_{k+i|k} \in \mathcal{X}_{i,\sigma(k)} \tag{8d}$$

The notations $\| \cdot \|_q$ and $\| \cdot \|_r$ represent typical weighted quadratic costs. The terminal penalty $V_{\sigma(k+N)}(x_{k+N|k})$ represents the cost to complete the problem from the predicted state at the horizon $x_{k+N|k}$ and a suitable cost-to-go function was derived in Ref. [5].

The constraint sets $\mathcal{X}_{i,\sigma(k)}$ and $\mathcal{T}_{\sigma(k)}$ are constructed to ensure robust feasibility, such that if some solution

$$\mathbf{\Delta U}^*_{k_0} = \left(\Delta \tilde{u}^*_{k_0|k_0}, \ \Delta \tilde{u}^*_{k_0+1|k_0}, \ldots, \ \Delta \tilde{u}^*_{k_0+N-1|k_0} \right)^T \tag{9}$$

is feasible at some time k_0 then a candidate solution

$$\widehat{\mathbf{\Delta U}}_{k_0+1} = \begin{pmatrix} \Delta \tilde{u}^*_{k_0+1|k_0} + F_{0,\sigma(k_0+1)} w(k_0) \\ \vdots \\ \Delta \tilde{u}^*_{k_0+N-1|k_0} + F_{N-2,\sigma(k_0+1)} w(k_0) \\ K_{\sigma(k_0+1)} x^*_{k_0+N|k_0} + F_{N-1,\sigma(k_0+1)} w(k_0) \end{pmatrix} \tag{10}$$

is feasible at k_0+1 for all $w(k_0) \in \mathcal{W}$. The designer chooses the feedback parameters $F_{i,\sigma(k)}$ and $K_{\sigma(k)}$ offline [11].

To achieve this robust feasibility property, the state constraints are tightened using a recursion

$$\mathcal{X}_{0,\sigma(k)} = \mathcal{X} \tag{11a}$$

$$\mathcal{X}_{i+1,\sigma(k)} = \mathcal{X}_{i,\sigma(k+1)} \sim L_{i,\sigma(k+1)}\mathcal{W} \tag{11b}$$

where

$$L_{0,\sigma(k)} = I \tag{12a}$$

$$L_{i+1,\sigma(k)} = AL_{i,\sigma(k)} + B_{\sigma(k+i)}F_{i,\sigma(k)} \tag{12b}$$

for the chosen feedback policy $F_{i,\sigma(k)}$ and the "$\sim$" operator denotes the Pontryagin difference [19] between two sets:

$$\mathcal{A} \sim \mathcal{B} = \{a \mid a + b \in \mathcal{A}\ \forall b \in \mathcal{B}\} \tag{13}$$

The terminal sets $\mathcal{T}_{\sigma(k)}$ have the robust invariance properties

$$\begin{aligned}&\left(A + B_{\sigma(k+N)}K_{\sigma(k+1)}\right)x + \\ &\left(AL_{N,\sigma(k+1)} + B_{\sigma(k+N)}F_{N-1,\sigma(k+1)}\right)w \in \mathcal{T}_{\sigma(k+1)} \\ &\forall x \in \mathcal{T}_{\sigma(k)},\ w \in \mathcal{W},\end{aligned} \tag{14a}$$

$$\mathcal{T}_{\sigma(k)} \subseteq \mathcal{X}_{N,\sigma(k)}. \tag{14b}$$

The reader is directed to Ref. [11] for a more thorough explanation of how these constraint modifications imply feasibility of the solution in (10).

The paramaters $F_{i,\sigma(k)}$ and $K_{\sigma(k)}$ are chosen by the designer. The parameters $L_{i,\sigma(k)}$, which relate the control perturbations in (10) to the corresponding changes in the state predictions, are then fixed by (12). These settings determine the amount of constraint tightening applied in (11). Typically, to achieve a large feasible region, the control policy chosen should minimise the quantities limited by the constraints.

A restrictive but convenient choice of candidate policy is to select $F_{i,\sigma(k)},\ i = 0, \ldots, N-2$ such that $L_{N,\sigma(k)} = 0\ \forall k$ and then set $F_{N-1,\sigma(k)} = 0$, $K_{\sigma(k)} = 0$ and $\mathcal{T}_{\sigma(k)} = \{0\}\ \forall k$.

Algorithm 1 (MMPC Scheme 1):

1) Solve (7) subject to (8)
2) Apply control move $\Delta u_{\sigma(k),k} = \Delta\tilde{u}_k$
3) Pause for one time step, increment k and go to Step 1

Theorem 1: If the system (1) is controlled using Algorithm 1 and the initial optimisation at time $k = 0$ can be solved, then the optimisation remains feasible and the constraints (3) are satisfied for all disturbances satisfying (2).

Proof: By construction of the constraints in (11), feasibility at any time k_0 implies feasibility at time $k_0 + 1$. Therefore, feasibility at time $k = 0$ implies feasibility at all future times $k > 0$. This also implies satisfaction of the constraint (3), as satisfying the optimisation constraints $x_{k|k} = x_k$ and $x_{k|k} \in \mathcal{X}_{0,\sigma(k)}$ implies $x_k \in \mathcal{X}$. ∎

B. Scheme 2

In Scheme 2, the large time-varying optimisation of Scheme 1 is divided into m sub-problems, one for each input channel. Each sub-problem solves only for the moves on the corresponding channel. Therefore the reduced decision variable is $\overline{\mathbf{\Delta U}}_k = (\Delta\tilde{u}_k,\ \Delta\tilde{u}_{k+m},\ \Delta\tilde{u}_{k+2m},\ \ldots,\ \Delta\tilde{u}_{k+N-1})^T$ and since housekeeping implies $k + N - 1 = k + pm$ for some p we require $N = pm + 1$. After each sub-problem is solved and the immediate move performed, the local controller stores its planned moves for use in the sub-problems for the other channels. Each sub-problem plans on the assumption that the future actions of others will be based on their last communicated plans. This may be considered as having a separate controller for each channel, all sharing knowledge of their intentions.

The Scheme 2 sub-problem optimisation is

$$\min_{\overline{\mathbf{\Delta U}}_k} V_{\sigma(k+N)}(x_{k+N|k}) + \sum_{i=0}^{N-1} ||x_{k+i|k}||_q^2 + \sum_{j=0}^{p} ||\Delta\tilde{u}_{k+jm|k}||_r^2 \tag{15}$$

subject to (8) and

$$\Delta\tilde{u}_{k+i|k} = \Delta\tilde{u}_{k+i|k-1} + F_{i,\sigma(k)}w_{k-1},\ \forall i \neq jm \tag{16}$$

encoding the assumption that the moves of other channels use the predetermined candidate policy (10). This constraint implies that unlike Scheme 1, any Scheme 2 sub-problem requires knowledge of the solution of the preceding optimisation. Therefore the Scheme 2 controller must be initialised with a solution to the Scheme 1 optimisation. Since Scheme 2 also guarantees robust feasibility, this initialisation is only needed once, at the beginning of Scheme 2 operation.

The additional constraint (16) in the Scheme 2 sub-problem can be written as a linear relationship between the reduced Scheme 2 decision variable $\overline{\mathbf{\Delta U}}_k$ and the planned moves for all channels, *i.e.* the corresponding Scheme 1 solution $\mathbf{\Delta U}_k$:

$$\mathbf{\Delta U}_k = M_1\overline{\mathbf{\Delta U}}_k + M_2\mathbf{\Delta U}^*_{k-1} + M_3 w_k \tag{17}$$

with the structures

$$M_1 = \left[\begin{array}{c|c|c} \left.\begin{matrix}1\\0\\\vdots\\0\end{matrix}\right\}m & & \\ \hline & \left.\begin{matrix}1\\0\\\vdots\\0\end{matrix}\right\}m & \\ \hline & & \ddots \\ \hline & & 1 \end{array}\right],\quad M_2 = \left[\begin{array}{c|ccc|ccc|c} 0 & 0 & & & & & & \\ \vdots & & 1 & & & & & \\ & & & \ddots & & & & \\ & & & 1 & & & & \\ \hline \vdots & & & & 0 & & & \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & 1 & \\ \hline \vdots & & & & & & & \ddots \\ \hline 0 & \cdots & & & \cdots & & & \cdots 0 \end{array}\right],$$

$$M_3 = \left[\begin{array}{c} 0 \\ F_{1,\sigma(k)} \\ \vdots \\ F_{m-1,\sigma(k)} \\ \hline 0 \\ F_{m+1,\sigma(k)} \\ \vdots \\ F_{2m-1,\sigma(k)} \\ \hline \vdots \\ \hline 0 \end{array}\right].$$

This linear mapping is used to reconstruct the full solution $\mathbf{\Delta U}_k$ from each sub-problem solution $\overline{\mathbf{\Delta U}}_k$.

Algorithm 2 (MMPC Scheme 2):

1) Solve (7) subject to (8)
2) Apply control move $\Delta u_{\sigma(k),k} = \Delta\tilde{u}_k$
3) Store planned moves $\mathbf{\Delta U}_k^*$
4) Pause for one time step, increment k
5) Solve (15) subject to (8) and (16)
6) Construct and store complete plan $\mathbf{\Delta U}_k^*$ using (17)
7) Go to Step 2

Theorem 2: If the system (1) is controlled using Algorithm 2 and the Scheme 1 initialisation at time $k = 0$ can be solved, then all subsequent optimisations are feasible and the constraints (3) are satisfied for all disturbance realisations satisfying (2).

Proof: Since the candidate solution (10) also satisfies the additional constraint (16) of Scheme 2, then knowledge of a feasible Scheme 1 solution at some time step k_0 implies feasibility of the Scheme 2 sub-problem at time step k_0+1. Furthermore, since the constraints of the Scheme 2 sub-problem include those of Scheme 1, then any solution to the Scheme 2 sub-problem must, when built back into a full solution using (17), form a feasible solution to the Scheme 1 problem at time step $k_0 + 1$. Therefore, by recursion, knowledge of an initial Scheme 1 solution at time step $k = 0$ implies feasibility and constraint satisfaction of Scheme 2 at all subsequent steps $k > 0$. ■

IV. RESULTS

This section demonstrates the potential benefits of MMPC by employing it in simulation for the control of two different example systems. In all cases, comparisons are made between the two MMPC schemes and normal "synchronous" MPC (SMPC). All simulations were performed on the same PC with a 3.2GHz Intel Pentium 4 processor and 1GB RAM. Matlab version 7.1 (R14, Service Pack 3) was employed, using Simulink to simulate the system dynamics and the "quadprog" optimisation function to solve the necessary quadratic programming (QP) problems. Computation times were measured using the Matlab profiler.

A. Spring-Mass Example

This section considers the control of the simple mechanical system shown in Fig. 2. The system comprises four point masses moving in one dimension. Each has mass of five units and is connected to the adjacent masses by a spring of stiffness one unit.

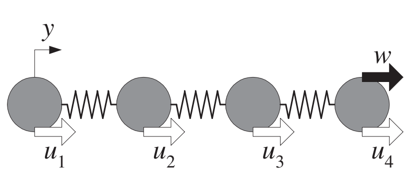


Fig. 2. Spring-Mass Example System

Each controller minimizes control energy subject to a constraint on the the position of mass 1, shown as output y in Fig. 2. Control energy is taken as $\int u(t)^T u(t)dt$ over a 400s simulation. The inputs are the control moves Δu_k applied to forces u_i acting on each mass, and therefore the control force levels $u(t)$ are elements in an augmented state vector. All controllers were made robust to a disturbance force of up to 0.01 unit acting on mass 4. In the simulations, a disturbance pulse was applied to that mass of magnitude 0.01 from 50s to 200s.

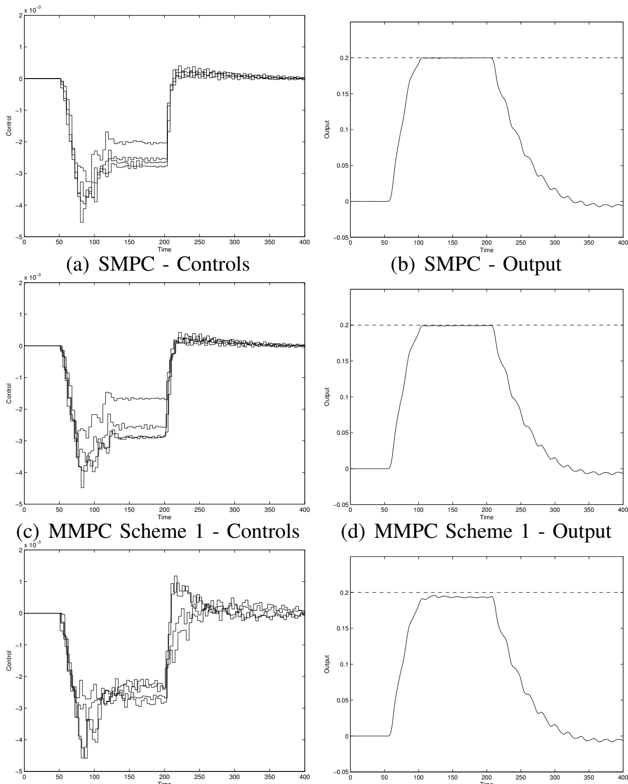
In the MMPC schemes, control moves were applied at intervals of one second, *i.e.* channel 1 moved at $t = t_1$ seconds, then channel 2 at $t = t_1 + 1$ seconds, and so on. In the comparison SMPC simulations, moves were made on all channels every four seconds, but to ensure fair comparison, the constraints were enforced at intervals of one second as in MMPC. Computation time is taken as the time spent in the "quadprog" function, totalled over all calls during the simulation.

Figure 3 shows the control input signals and the output signals for each of the three controllers considered, using a horizon of 120s in all cases. The asynchronous control moves can be seen in the control signal plots from the two MMPC simulations. In all three cases, the output signal runs tightly against the constraint (shown dashed) for the duration of the disturbance pulse. This is as expected, since the objective is to minimize control energy and therefore the controller makes use of all available flexibility in the output constraint. The output under MMPC Scheme 2 is slightly further from the limit than under the other two controllers, possibly because that controller effectively solves a more constrained problem due to the reduced decision variable set. However, the effect is not significant.

To further illustrate the ability of the new robust MMPC to satisfy hard constraints despite disturbances, the simulation using MMPC Scheme 2 was repeated using different constraint levels. The resulting output signals are shown in Figure 4. In every case, the signal goes right to its limit, but never beyond, and the optimisations remain feasible. These results illustrate that the constraints are active in these simulations and that the robust MMPC method does not introduce undue conservatism.

Table I compares detailed statistics from the results in Fig. 3. Observe that the performance, in terms of the control energy, is roughly the same across all three controllers. However, the computation times vary considerably. Predictably, MMPC Scheme 1 is much slower, in terms of computation, than SMPC, since the two problems are (roughly) the same size, but MMPC Scheme 1 solves the problem four times as often. However, MMPC Scheme 2 is slightly faster than SMPC, since its sub-problems have only a quarter as many decision variables as SMPC. This illustrates the underlying premise of MMPC: it is faster to solve four problems of 31 variables than one problem of 124.

To further explore the issue of scalability, the simulations from Fig. 3 using SMPC and MMPC Scheme 2 were repeated with various horizon lengths. Figure 5 shows the variation of total computation time with horizon length for both controllers. With a very short horizon, SMPC is faster than MMPC. We hypothesize that this is due to



(a) SMPC - Controls (b) SMPC - Output

(c) MMPC Scheme 1 - Controls (d) MMPC Scheme 1 - Output

(e) MMPC Scheme 2 - Controls (f) MMPC Scheme 2 - Output

Fig. 3. Spring-Mass Example: Responses to Disturbance Pulse

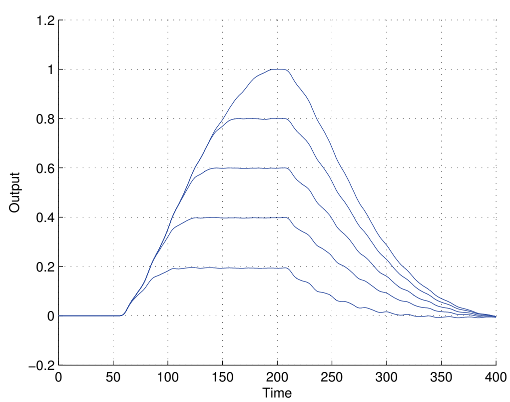


Fig. 4. Spring-Mass Example: Outputs from MMPC Scheme 2 for Constraint Settings 0.2, 0.4, 0.6, 0.8, and 1.0

overheads in the QP solver, such as set-up time, which dominate the solution time for small problems and therefore penalise the more frequent optimisation calls of MMPC. However, as the horizon length increases, the computation time becomes dominated by the actual solution process and MMPC Scheme 2 scales more favorably than SMPC.

B. Flight Dynamics Example

This section considers longitudinal control of an A-7A Corsair II aircraft. The dynamics model was taken from Example 6.1 in Ref. [20] and augmented to include a thrust input as well as the elevator input. Both inputs are

TABLE I

SPRING-MASS EXAMPLE: RESULTS FOR EACH CONTROLLER REJECTING DISTURBANCE PULSE.

Controller	SMPC	MMPC Scheme 1	MMPC Scheme 2
$\int \mathbf{u}(t)^T \mathbf{u}(t) dt \times 1000$	4.312	4.358	4.320
Computation Time (s)	6.6	21.7	5.6
N$^{o.}$ of QP Solutions	100	400	400
N$^{o.}$ of Decision Vars. per QP	124	121	31
N$^{o.}$ of Constraints per QP	248	242	242

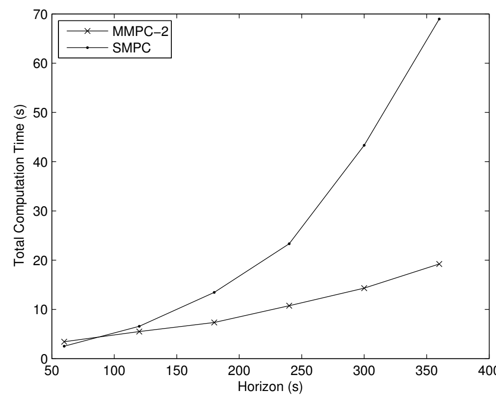
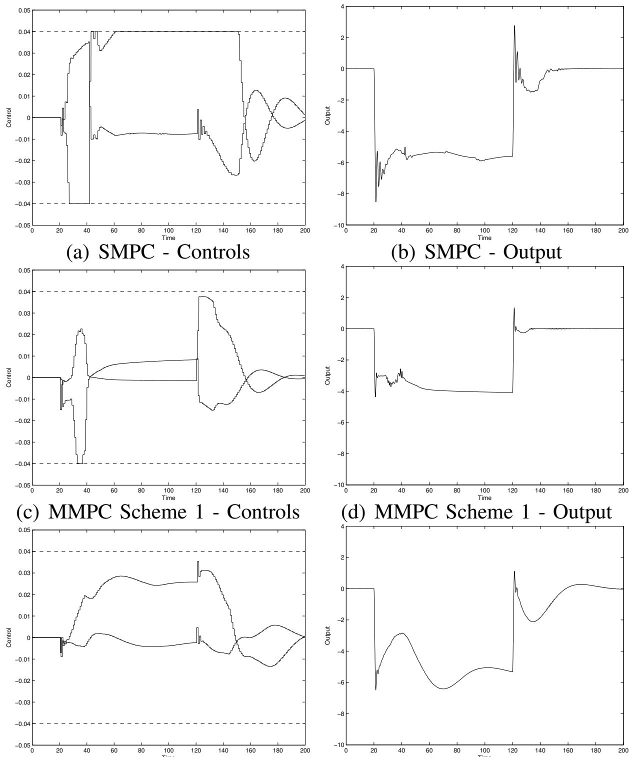


Fig. 5. Spring-Mass Example: Variation of Computation Time with Horizon Length for SMPC and MMPC Scheme 2

constrained to $[-0.04, 0.04]$ and the constraints are made robust to input disturbances in the range $[-0.01, 0.01]$ on each channel. The simulation runs for 200s and a disturbance of 0.01 is applied to both channels from 20s to 120s. The planning horizon is 80s in all cases and the objective is to minimize x_2^2 where the state element x_2 corresponds to the velocity normal to the aircraft axis in the body frame.

Figure 6 shows the control and output signals from simulations using each of the three different controllers. SMPC executes moves on both channels at intervals of one second. MMPC performs a single move on alternating channels every half a second. Thus the total number of moves on each channel in each simulation is the same. Table II compares the three results using the same metrics as in the previous section, except for the performance which is here taken as the peak value of the normal velocity $\|x_2\|_\infty$.

Unlike in the spring-mass example, there is significant variation in performance between the three controllers. The two MMPC controllers, with their faster response times, are able to mitigate the short period response more effectively than SMPC, which leaves a significant spike at the onset of the disturbance. Scheme 1 performs better than Scheme 2, to be expected as Scheme 1 solves a less constrained optimisation. However, Scheme 2 still outperforms SMPC, indicating that in this case, it is better to respond to a disturbance quickly with one channel than slowly with both. MMPC Scheme 2 also requires significantly less computational effort than SMPC for this example. It is surprising that SMPC is so slow, requiring almost as much computation as the more



(a) SMPC - Controls (b) SMPC - Output

(c) MMPC Scheme 1 - Controls (d) MMPC Scheme 1 - Output

(e) MMPC Scheme 2 - Controls (f) MMPC Scheme 2 - Output

Fig. 6. Aircraft Example: Responses to Disturbance Pulse

TABLE II

AIRCRAFT EXAMPLE: RESULTS FOR EACH CONTROLLER REJECTING DISTURBANCE PULSE.

Controller	SMPC	MMPC Scheme 1	MMPC Scheme 2
$\|x_2(t)\|_\infty$	8.53	4.37	6.49
Computation Time	42.25	44.52	9.15
N$^{o.}$ of QP Solutions	200	400	400
N$^{o.}$ of Decision Vars. per QP	80	81	41
N$^{o.}$ of Constraints per QP	160	324	324

complex MMPC Scheme 1: this is the subject of continuing investigation. These results suggest that when computation is a limiting factor and the characteristic time constants of the system are comparable to the computation times required, MMPC can offer performance benefits over SMPC.

V. CONCLUSION

Multiplexed model predictive control (MMPC) updates one input at a time, of a multi-input controlled plant. The motivation is to reduce the computational complexity of MPC, in order to allow reduced control update intervals. For some plants this leads to improved control, as a result of the controller being able to react to disturbances more quickly. MMPC scales well with increasing numbers of inputs, since the computational complexity of the 'Scheme 2' variant depends only weakly on the number of inputs.

In this paper we have extended the basic MMPC idea to obtain robust feasibility and robust constraint satisfaction in the presence of unknown but bounded disturbances.

Simulation examples have demonstrated that our scheme succeeds in maintaining constraint satisfaction and feasibility despite the presence of disturbances. Furthermore, they have shown that performance improvements can indeed be obtained in some circumstances, compared with conventional MPC, they have indicated the kind of computational speed-up that can result from adoption of the MMPC scheme, and they have illustrated that these benefits are retained in circumstances where the constraints are active.

REFERENCES

[1] S. J. Qin and T. A. Badgwell, "An overview of industrial model predictive control technology," in *Chemical Process Control-V*, J. Kantor, C. Garcia, and B. Carnahan, Eds., 1997.

[2] R. Franz, M. Milam, and J. Hauser, "Applied receding horizon control of the Caltech ducted fan," in *Proceedings of American Control Conference*, 2002, p. 3735.

[3] A. G. Richards and J. P. How, "Model predictive control of vehicles maneuvers with guaranteed completion time and robust feasibility," in *Proceedings of American Control Conference*, 2003.

[4] J. M. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[5] K.-V. Ling, J. M. Maciejowski, and B.-F. Wu, "Multiplexed model predictive control," in *Proceedings of 16th IFAC World Congress*, July 2005.

[6] K. V. Ling, J. M. Maciejowski, and B. F. Wu, "Multiplexed model predictive control," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR.561, July 2006.

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

[8] A. Bemporad and M. Morari, "Robust model predictive control: A survey," *Robustness in Identification and Control*, vol. 245, pp. 207–226, 1999.

[9] J. R. Gossner, B. Kouvaritakis, and J. A. Rossiter, "Stable generalized predictive control with constraints and bounded disturbances," *Automatica*, vol. 33(4), pp. 551–568, 1997.

[10] L. Chisci, J. A. Rossiter, and G. Zappa, "Systems with persistent disturbances: Predictive control with restrictive constraints," *Automatica*, vol. 37(7), pp. 1019–1028, 2001.

[11] A. G. Richards and J. P. How, "Robust stable model predictive control with constraint tightening," in *Proceedings of American Control Conference*, 2006.

[12] A. G. Richards, "Robust model predictive control for time-varying systems," in *Proceedings of the joint 44th IEEE Conference on Decision and Control and European Control Conference*, 2005.

[13] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar, "Distributed model predictive control," *IEEE Control Systems Magazine*, vol. 2, pp. 44–52, 2002.

[14] D. Jia and B. Krogh, "Min-max feedback model predictive control for distributed control with communication," in *Proceedings of American Control Conference*, 2002, pp. 4507–45.

[15] T. Schouwenaars, J. P. How, and E. Feron, "Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees," in *Proceedings of AIAA Guidance Navigation and Control Conference*, 2004.

[16] F. Borrelli, T. Keviczky, and G. J. Balas, "Collision-free UAV formation flight using decentralized optimization and invariant sets," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, December 2004.

[17] P. A. Trodden and A. G. Richards, "Robust distributed model predictive control using tubes," in *Proceedings of the American Control Conference*, 2006.

[18] A. G. Richards and J. P. How, "A decentralized algorithm for robust constrained model predictive control," in *Proceedings of the American Control Conference*, 2004.

[19] I. Kolmanovsky and E. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical Problems in Engineering: Theory, Methods and Applications*, vol. 4, pp. 317–367, 1998.

[20] M. V. Cook, *Flight Dynamics Principles*. Butterworth Heinemann, 1997.