MULTIPLEXED MODEL PREDICTIVE CONTROL [1]

Keck-Voon LING * **Jan MACIEJOWSKI** **
Bing-Fang WU *

* *Nanyang Technological University, Singapore*
** *Cambridge University, United Kingdom*

Abstract: Most academic control schemes for MIMO systems assume all the control variables are updated simultaneously. MPC outperforms other control strategies through its ability to deal with constraints. This requires on-line optimization, hence computational complexity can become an issue when applying MPC to complex systems with fast response times. The multiplexed MPC scheme described in this paper solves the MPC problem for each subsystem sequentially, and updates subsystem controls as soon as the solution is available, thus distributing the control moves over a complete update cycle. The resulting computational speed-up allows faster response to disturbances, and hence improved performance, despite finding sub-optimal solutions to the original problem. The multiplexed MPC scheme is also closer to industrial practice in many cases. This paper presents initial stability results for two variants of multiplexed MPC, and illustrates the performance benefit by an example.

Keywords: predictive control, decentralised control, multivariable control, periodic systems, constrained optimization.

1. INTRODUCTION

Model Predictive Control (MPC) has become an established control technology in the petrochemical industry, and its use is currently being pioneered in an increasingly wide range of process industries. It is also being proposed for a range of higher bandwidth applications, such as ships (Perez *et al.*, 2000), aerospace (Murray *et al.*, 2003) (Richards and How., 2003), and road vehicles (Morari *et al.*, 2003).

This paper is concerned with facilitating applications of MPC in which computational complexity is likely to be an issue. One can foresee that applications to embedded systems, with the MPC algorithm implemented in a chip or an FPGA, are likely to have this characteristic.

MPC operates by solving an optimisation problem on-line, in real time, in order to decide how to update the control inputs (manipulated variables) at the next update instant. All MPC theory to date, and as far as we know all implementations, assume that all the control inputs are updated at the same instant (Maciejowski, 2002). Suppose that a given MPC control problem can be solved in not less than T seconds, so that the smallest possible update interval is T. The computational complexity of typical MPC problems, including time requirements, tend to vary as $O(m^3)$, where m is the number of control inputs. We propose to use MPC to update only one control variable at a time, but to exploit the reduced complexity to update successive inputs at intervals smaller

[1] This work was supported by A*STAR project "Model Predictive Control on a Chip" (Ref: 022-106-0044)

than T, typically T/m. After m updates a fresh cycle of updates begins, so that each whole cycle of updates repeats with cycle time T. We call this scheme *multiplexed MPC*. We assume that fresh measurements of the plant state are available at these reduced update intervals T/m. The main motivation for this scheme is the belief that in many cases the approximation involved in updating only one input at a time will be outweighed — as regards performance benefits — by the more rapid response to disturbances, which this scheme makes possible. It is often the case that "do something sooner" leads to better control than "do the optimal thing later". Figure 1 shows the pattern of input moves in the multiplexed MPC scheme with $m = 2$, compared with the conventional scheme in which the two input moves are synchronised. The

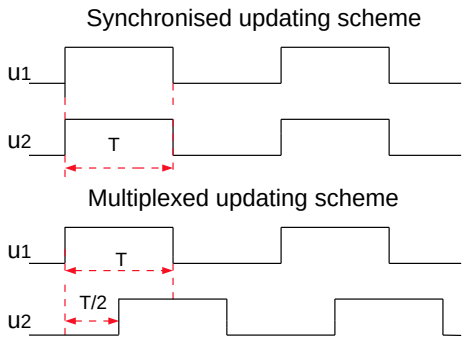

Fig. 1. Patterns of input moves for conventional 'synchronised' MPC, and for the Multiplexed MPC introduced in this paper.

scheme which we investigate here is close to common industrial practice in complex plants, where it is often impossible to update all the control inputs simultaneously, because of their sheer number, and the limitations of the communications channels between the controller and the actuators.

Various generalizations of our scheme are possible. For example, subsets of control inputs might be updated simultaneously, perhaps all the inputs in each subset being associated with one subunit. Alternatively, sensor outputs might become available one at a time (or one subset at a time), as in the common practice of "polling" sensors. A further generalization would be not to update each control input in a fixed sequence, but to decide in real time which input (if any) needs updating most urgently — one could call this *just-in-time MPC*; note that this would then resemble *statistical process control (SPC)*, which is used widely in manufacturing processes (Box and Luceno, 1997).

Several works have been published which propose 'decentralized MPC' in the sense that subsets of control inputs are updated by means of an MPC algorithm. But these usually assume that several sets of such computations are performed in parallel, on the basis of local sensor outputs only, and that all the control inputs are then updated simultaneously. In some applications, such as formation flying of unmanned vehicles, it is assumed that the state vectors of subunits (vehicles) are distinct, and that coupling between subunits occurs only through constraints and performance measures. In (Venkat *et al.*, 2004) five different MPC-based schemes are proposed, of which four are decentralized MPC schemes of some kind. Their schemes 4 and 5 are the closest to our multiplexed scheme. In these schemes an MPC solution is solved iteratively for each control input, but it is assumed that no new sensor information arrives during the iteration, and that all the control inputs are updated simultaneously when the iterations have been completed.

As far as we are aware, the original feature of the scheme proposed in this paper is that the inputs are updated sequentially, and that each control update takes account of all the information available at that time, namely knowledge of all updates already performed, and of the latest sensor outputs.

The distinction between previous proposals for decentralised MPC and our proposal for *multiplexed MPC* is completely analogous to the distinction between the Jacobi and the Gauss-Seidel iterative algorithms for inverting a matrix (Barrett *et al.*, 1994).

An alternative strategy for speeding up the computations involved in MPC is off-line precomputation of the 'pieces' of the piecewise-affine controller which is the optimal solution (Morari *et al.*, 2003). But that is not feasible if the number of 'pieces' required is excessively large, or if the constraints or the plant model change relatively frequently.

The rest of this paper is organized as follows. In Section 2, two possible schemes for Multiplexed MPC are formulated. Sections 3 and 4 establish the stability of Schemes 1 and 2, respectively. Section 5 presents a simulation example to demonstrate the performance benefits of the proposed scheme. Finally, concluding remarks are given in Section 6.

2. PROBLEM FORMULATION

Consider the following discrete-time linear plant model in state-space form, with state vector $x_k \in \mathbb{R}^n$ and m (scalar) inputs $u_{1,k}, \ldots, u_{m,k}$:

$$x_{k+1} = Ax_k + \sum_{j=1}^{m} B_j \Delta u_{j,k} \qquad (1)$$

where each B_j is a column vector and $\Delta u_{j,k} = u_{j,k} - u_{j,k-1}$. (This could be generalised to the case

where $B_j \in \mathbb{R}^{n\times p_j}$ and $\Delta u_{j,k} \in \mathbb{R}^{p_j}$, with $\sum_j p_j$ inputs.) We wish to devise a control strategy based on MPC which, at discrete-time index k, changes only plant input $(k \bmod m) + 1$. We consider two alternative schemes for determining the appropriate plant inputs. In both schemes, an increase of k by 1 corresponds to a time duration of T/m, where T is the complete update cycle duration — see section 1.

In both schemes an infinite prediction horizon has been chosen, because that is one way of ensuring closed-loop stability with MPC (Maciejowski, 2002). Alternative ways of obtaining stability exist, such as combining a finite horizon with a terminal constraint or a suitable terminal weight in the cost function — see (Mayne *et al.*, 2000) for a comprehensive survey.

We assume in both schemes that at time step k the complete state vector x_k is known exactly from measurements. We will consider only the regulation problem in detail, but tracking problems, especially those with non-zero constant references, can be easily transformed into equivalent regulation problems.

2.1 Scheme 1

In our first scheme, the input move $\Delta u_{(k \bmod m)+1,k}$ is to be determined by finding the sequence of predicted input moves $\{\Delta u_{k+i} : i = 0, 1, \ldots\}$ which minimises

$$J_k = \sum_{i=0}^{\infty} \left(\|x_{k+i+1}\|_q^2 + \|\Delta u_{k+i}\|_r^2 \right) \qquad (2)$$

subject to the constraint

$$\Delta u_{j,k+i} = 0 \text{ if } j \neq ([k+i] \bmod m) + 1 \qquad (3)$$

(and possibly other constraints) and applying the first element of the optimal sequence, namely Δu_k, to the plant.

Here we have used the notational conventions that x_{k+i} denotes a prediction of the state at time step $k+i$, based on information available at time k, and that Δu_{k+i} denotes the predicted vector of control moves at time step $k+i$, whose jth element is $\Delta u_{j,k+i}$. $q \geq 0$ and $r > 0$ are matrices; only diagonal matrices r are needed.

An alternative representation of the problem we pose is as a periodic linear system with one input:

$$x_{k+1} = Ax_k + B_{(k \bmod m)+1}\Delta\tilde{u}_k \qquad (4)$$

where $\Delta\tilde{u}_k = \Delta u_{(k \bmod m)+1,k}$, if $\|\Delta u_{k+i}\|_r^2$ is replaced by $|\Delta\tilde{u}_{k+i}|^2_{r([k+i] \bmod m)+1}$ in J_k.

Scheme 1, if applied over a finite horizon, has a lower computational complexity than conventional multivariable MPC, providing that the number of steps in the horizon remains the same — which means, since the step duration has been reduced to T/m, that a shorter horizon is used.

2.2 Scheme 2

In our second scheme the plant model assumed by the controller remains the same as (1), and constraint (3) still holds. But now the future trajectory of only one input is optimised, and we make further assumptions about the future behaviour of the other inputs, in order to further reduce computational complexity — the other inputs are treated like measured disturbances, in effect. The simplest possibility would be for controller j to assume that

$$\Delta u_{\ell,k+i} = 0 \text{ for all } i \geq 0 \text{ if } \ell \neq j \qquad (5)$$

but we will make other assumptions in section 4. In Scheme 2 there are essentially m MPC controllers. They share information, however, in the sense that the complete plant state is available to each controller — although not at the same times, and the currently planned future moves of each controller are also available to all the others. If the length of the planning horizon (N in the sequel) is the shortest possible, namely $N = 1$, then Schemes 1 and 2 are equivalent.

Scheme 2 has a lower computational complexity than conventional multivariable MPC, even if the same horizon length is retained, that is even if the number of steps is increased by a factor m, since the complexity per step is reduced by $O(m^3)$.

3. STABILITY OF SCHEME 1

The solution of the MPC problem posed in section 2.1 when there are no constraints is straightforward. It is essentially an infinite-time linear quadratic regulator problem, for a single-input, periodically time-varying plant. From (4) it can be seen that the solution will be a time-varying — in fact periodic — state feedback gain, which can be obtained from the solution of a Riccati difference equation (Anderson and Moore, 1990; Kwakernaak and Sivan, 1972). If we assume that N is a multiple of m then the indexing is slightly simplified, and P_k should be a solution of

$$P_k = A^T P_{k-1} A - A^T P_{k-1} B_\ell (B_\ell^T P_{k-1} B_\ell + r)^{-1} B_\ell^T P_{k-1} A + q \qquad (6)$$

where $\ell = (k \bmod m) + 1$. It follows by duality from (Bittanti *et al.*, 1988) that if P_k is a symmetric periodic positive semidefinite solution of (6) then the periodic state-feedback gain

$$K_k = -(B_\ell^T P_k B_\ell + r)^{-1} B_\ell^T P_k A \qquad (7)$$

asymptotically stabilises the plant. (Bittanti *et al.*, 1988) discuss (mild) conditions for the existence and uniqueness of such a solution of (6), and provide algorithms for finding it; note that this solution can be pre-computed off-line.

The unique advantage of MPC, compared with other control strategies, is its capacity to take account of constraints in a systematic manner. As usual in MPC, we will suppose that constraints may exist on the input amplitudes, $\|u_k\|_\infty \leq U$, on the input moves, $\|\Delta u_k\|_\infty \leq D$, and on states, $Mx_k \leq v$. (These can all be generalised substantially, so long as linear inequalities are retained.) Following (Rawlings and Muske, 1993), a finite-dimensional optimisation problem can be obtained by assuming that none of the constraints are active beyond the end of a prediction horizon of length N, for some sufficiently large N. The minimisation of (2) is replaced by the minimisation of

$$J_k^N = \sum_{i=0}^{N-1} \left(\|x_{k+i+1}\|_q^2 + \|\Delta u_{k+i}\|_r^2 \right) + x_{k+N+1}^T P_{k+N+1} x_{k+N+1} \quad (8)$$

where $P_{k+N+1} = P_{k+N+1}^T > 0$ is a suitably chosen terminal cost coefficient.

For time-invariant systems P_{k+N+1} can be chosen to be the positive semidefinite solution P of an algebraic Riccati equation, as proposed in (Chmielewski and Manousiouthakis, 1996), which will result in the optimal LQ state feedback law being assumed after the end of the finite horizon (ie beyond N steps into the future), and will recover the LQ solution exactly if none of the constraints are active. With this choice of terminal cost, (8) is the infinite-horizon cost-to-go of the LQ optimal control; hence it is a Lyapunov function for the closed-loop system, and stability follows providing that the minimisation of (8), subject to constraints, is feasible.

We apply the same idea, but we choose the sequence P_k to be the positive semidefinite solution of (6), as in the unconstrained case. Closed-loop stability then follows in the same way, subject as usual to feasibility of the minimisation at each step.

4. STABILITY OF SCHEME 2

The stability of Scheme 2 cannot be proved in exactly the same way, since the computation of a future input trajectory is done for only one input at a time. We retain the idea from the previous section, that each controller makes N decisions, but each decision now relates only to one input. Each input remains unchanged for m steps, so that controller 1 plans to change input 1 at steps $k, k+m, \ldots, k+m(N-1)$, if $(k \bmod m)+1 = 1$. Controller 2 plans to change input 2 at steps $k+1, k+m+1, \ldots, k+m(N-1)+1$, and so on.

We assume that, after the end of the prediction horizon, stabilising periodic state feedback gain will be applied, so that

$$\Delta u_{\ell,k+i} = K_\ell x_{k+i} \quad (9)$$

for $i > mN - \ell + 1$, where $\ell = (k \bmod m) + 1$, and the gains $\{K_j : j = 1, \ldots, m\}$ are chosen such that the monodromy matrix

$$\Psi_1 = \Phi_m \Phi_{m-1} \ldots \Phi_2 \Phi_1 \quad (10)$$

has all its eigenvalues inside the unit circle, where

$$\Phi_j = A + B_j K_j \quad (11)$$

It is a standard fact that this is a necessary and sufficient condition for closed-loop stability of a linear periodic system, and that it is invariant under cyclic permutations of the Φ_j matrices. We shall later need the monodromy matrices obtained by such cyclic permutations:

$$\Psi_2 = \Phi_1 \Phi_m \ldots \Phi_3 \Phi_2 \quad (12)$$

$$\vdots$$

$$\Psi_m = \Phi_{m-1} \Phi_{m-2} \ldots \Phi_1 \Phi_m \quad (13)$$

So for example, in the case $m = 2$, at time indices k and $k+1$ the assumed future control vectors $\Delta \boldsymbol{U}_k$ and $\Delta \boldsymbol{U}_{k+1}$ have the form (assuming $(k \bmod m)+1 = 1$):

$$\Delta \boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{\Delta u_{1,k}} \\ \Delta u_{2,k+1} \\ \boldsymbol{\Delta u_{1,k+2}} \\ \vdots \\ \boldsymbol{\Delta u_{1,k+2N-4}} \\ \Delta u_{2,k+2N-3} \\ \boldsymbol{\Delta u_{1,k+2(N-1)}} \\ K_2 x_{k+2N-1} \\ K_1 x_{k+2N} \\ K_2 x_{k+2N+1} \\ K_1 x_{k+2N+2} \\ \vdots \end{bmatrix}, \Delta \boldsymbol{U}_{k+1} = \begin{bmatrix} \boldsymbol{\Delta u_{2,k+1}} \\ \Delta u_{1,k+2} \\ \boldsymbol{\Delta u_{2,k+3}} \\ \vdots \\ \boldsymbol{\Delta u_{2,k+2N-3}} \\ \Delta u_{1,k+2(N-1)} \\ \boldsymbol{\Delta u_{2,k+2N-1}} \\ K_1 x_{k+2N} \\ K_2 x_{k+2N+1} \\ K_1 x_{k+2N+2} \\ \vdots \end{bmatrix}$$

The elements which are in boldface are those which are optimised in each case. The elements x_{k+i} in $\Delta \boldsymbol{U}_k$ are predictions conditional on information available up to time k, while those in $\Delta \boldsymbol{U}_{k+1}$ are conditional on information available at time $k+1$.

Some assumptions must be made by controller j about those inputs which have already been planned by the other controllers, but which have not yet been executed. We will assume that all such planned decisions are known to the controller, and that it assumes that they will be executed as planned. (This assumption will usually be false, because new decisions will be made in the light of new measurements.) Thus, in the example

above, all those 'Δu' elements which are not in boldface are assumed to be imported from the corresponding controller.

Each controller optimises the same cost function, namely (2). However, each one evaluates it slightly differently. Since we will need to consider values $\ell, \ell+1, \ldots, \ell+m$, but modulo m, it will be convenient to introduce the indexing function

$$\sigma(k) = (k \bmod m) + 1 \tag{14}$$

Then at time k controller $\sigma(k)$ evaluates the cost function as

$$J_k = \sum_{i=0}^{m(N-1)} \left(\|x_{k+i+1}\|_q^2 + \|\Delta u_{k+i}\|_r^2 \right) + x_{k+m(N-1)+1}^T P_{\sigma(k)} x_{k+m(N-1)+1} \tag{15}$$

where $P_{\sigma(k)}$ can be obtained as follows. The second term in (15) should be the same as

$$\begin{aligned} J_{k+m(N-1)+1} &= \sum_{i=m(N-1)+1}^{\infty} \left(\|x_{k+i+1}\|_q^2 + \|\Delta u_{k+i}\|_r^2 \right) \\ &= \sum_{i=N-1}^{\infty} \left(\|\mathcal{X}_{k+mi+2}\|_Q^2 + \|\Delta \mathcal{U}_{k+mi+1}\|_R^2 \right) \end{aligned} \tag{16}$$

where

$$\mathcal{X}_k = \begin{bmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_{k+m-1} \end{bmatrix}, \; \Delta \mathcal{U}_k = \begin{bmatrix} \Delta u_{\sigma(k),k} \\ \Delta u_{\sigma(k+1),k+1} \\ \vdots \\ \Delta u_{\sigma(k+m-1),k+m-1} \end{bmatrix} \tag{17}$$

and $Q = \mathrm{diag}[q, \ldots, q]$, $R = \mathrm{diag}[r, \ldots, r]$.

The state transition equation relating successive $\mathcal{X}$ terms which occur in (16) is simply:

$$\mathcal{X}_{k+i+m} = \begin{bmatrix} \Psi_{\sigma(k+i)} & 0 & \cdots & 0 \\ 0 & \Psi_{\sigma(k+i+1)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Psi_{\sigma(k+i+m)} \end{bmatrix} \mathcal{X}_{k+i} \tag{18}$$

if $i > m(N-1)$ — that is, for predictions beyond the horizon.

Now it is a standard result (Anderson and Moore, 1990; Kwakernaak and Sivan, 1972) that if $x_{k+1} = \Phi x_k$ and $\Delta u_k = K x_k$, then $J_k = x_k^T P x_k$, where P is the solution of the Lyapunov equation $P = \Phi^T P \Phi + \Phi^T q \Phi + K^T r K$, and J_k is defined in (2). Applying this result to (18) gives

$$J_{k+m(N-1)+1} = \mathcal{X}_{k+m(N-1)+1}^T \times \mathrm{diag}[\Pi_{\sigma(k+1)}, \ldots, \Pi_{\sigma(k+m)}] \times \mathcal{X}_{k+m(N-1)+1} \tag{19}$$

where

$$\Pi_\ell = \Psi_\ell^T \Pi_\ell \Psi_\ell + \Psi_\ell^T q \Psi_\ell + K_\ell^T r K_\ell \quad \text{for } \ell = 1, 2, \ldots, m \tag{20}$$

(The Lyapunov equation arising from (18) contains block-diagonal matrices only, which allows its expression in terms of m separate — though related — Lyapunov equations in (20).)

But

$$x_{k+m(N-1)+2} = \Phi_{\sigma(k+1)} x_{k+m(N-1)+1} \tag{21}$$

$$x_{k+m(N-1)+3} = \Phi_{\sigma(k+2)} \Phi_{\sigma(k+1)} x_{k+m(N-1)+1} \tag{22}$$

etc

so by substituting these in (19) we obtain

$$J_{k+m(N-1)+1} = x_{k+m(N-1)+1}^T P_{\sigma(k)} x_{k+m(N-1)+1} \tag{23}$$

where

$$\begin{aligned} P_{\sigma(k)} &= \Pi_{\sigma(k+1)} + \Phi_{\sigma(k+1)}^T \Pi_{\sigma(k+2)} \Phi_{\sigma(k+1)} + \cdots \\ &+ [\Phi_{\sigma(k+1)}^T \cdots \Phi_{\sigma(k+m-1)}^T \Pi_{\sigma(k+m)} \Phi_{\sigma(k+m-1)} \cdots \Phi_{\sigma(k+1)}] \end{aligned} \tag{24}$$

Let J_k^o be the optimal value of J_k achieved by controller $\sigma(k)$. Stability follows from the fact that $J_{k+1}^o \le J_k^o$, with equality holding only if $x_k = 0$. This monotonic decreasing property of J_k^o can be established by a standard argument: suppose that at step $k+1$ controller $\sigma(k+1)$ leaves the moves $\Delta u_{\sigma(k+1),k+1}, \ldots, \Delta u_{\sigma(k+1),k+m(N-1)+1}$ unchanged from the values which were assumed by controller $\sigma(k)$ at step k, and that this achieves the value J_{k+1}. Then

$$J_{k+1} = J_k^o - \|x_{k+1}\|_q^2 - \|\Delta u_k\|_r^2 \tag{25}$$

But $J_{k+1}^o \le J_{k+1}$, from which the conclusion follows. Consequently J_k^o is a Lyapunov function for the overall controller of Scheme 2, and stability is established.

Remark: The trajectory obtained by the Scheme 2 controller will usually be different from that found by the Scheme 1 controller. It will be 'less optimal' in the sense that it will usually result in larger values of the cost (although each constituent controller will be optimising the same cost function with respect to the degrees of freedom available to it).

5. EXAMPLE

Here an example will be presented to compare the disturbance rejection performance of conventional 'synchronised' MPC control and our new Multiplexed MPC with $m = 2$ and $N = 1$. For the simulation, we have tuned both controllers to give comparable step responses, so that the comparison of the disturbance rejection performance is meaningful.

The plant has a continuous-time model

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{1}{7s+1} & \dfrac{1}{3s+1} \\ \dfrac{2}{8s+1} & \dfrac{1}{4s+1} \end{bmatrix} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \tag{26}$$

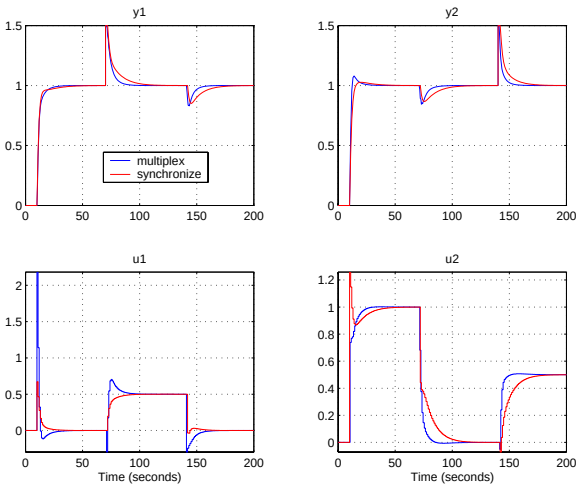

Fig. 2. Performance comparison of the two schemes.

We have chosen 0.5s and 1s sampling/update times for the multiplexed and synchronized schemes, respectively, so that the complete update cycle lasts 1s in both cases. There are no constraints. The simulation was carried with continuous-time plant model and discrete-time controller, with disturbances acting on the continuous-time model.

The setpoints for both outputs change from 0 to 1 at $t = 10$s. In addition, y_1 and y_2 are subjected to an output step disturbance with magnitude of 0.5 at $t = 70.1s$ and $t = 140.1s$, respectively. The disturbances occur just after input 1 is optimised. Fig.2 shows the simulation results. It can be seen that there is indeed a performance benefit from using multiplexed MPC. The response under multiplexed MPC scheme is better than under conventional MPC; smaller excursions of the outputs and faster settling responses are achieved in response to disturbances.

6. CONCLUSION

In this work, two versions of a novel control scheme known as *Multiplexed MPC* were proposed, and expected to be of benefit in those MPC applications for which computational complexity is a limiting factor. Both of our proposed multiplexed MPC schemes are nominally stable. Some performance benefit over conventional MPC can be obtained as a result of faster reactions to disturbances, despite suboptimal solutions being obtained. This has been demonstrated by an example.

It is interesting, and potentially important, to observe that the assumption of equal intervals between the updates of plant inputs is not essential to our proposal. Any pattern of update intervals can be supported, providing that it repeats in subsequent update cycles.

REFERENCES

Anderson, B.D.O. and J.B. Moore (1990). *Optimal Control, Linear Quadratic Methods*. Prentice Hall.

Barrett, R., M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H. Van der Vorst (1994). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. 2nd ed.. SIAM. Philadelphia PA.

Bittanti, S., P. Colaneri and G. De Nicolao (1988). The difference periodic Riccati equation for the periodic prediction problem. *IEEE Transactions on Automatic Control,* **33**(8), 706–712.

Box, G. and A. Luceno (1997). *Statistical Control by Monitoring and Feedback Adjustment*. Wiley. New York.

Chmielewski, D. and V. Manousiouthakis (1996). On constrained infinite-time linear quadratic optimal control. *Systems and Control Letters* **29**, 121–129.

Kwakernaak, H. and R. Sivan (1972). *Linear Optimal Control Systems*. Wiley. New York.

Maciejowski, J.M. (2002). *Predictive Control with Constraints*. Prentice-Hall. Harlow UK.

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

Morari, M., M. Baotić and F. Borrelli (2003). Hybrid systems modeling and control. *European Journal of Control* **9**(2–3), 177–189.

Murray, R.M., J. Hauser, A. Jadbabie, M.B. Milam, N. Petit, W.B. Dunbar and R. Franz (2003). Online control customization via optimization-based control. In: *Software-Enabled Control* (T. Samad and G. Balas, Eds.). IEEE Press and Wiley.

Perez, T., G.C. Goodwin and C.W. Tzeng (2000). Model predictive rudder roll stabilization control for ships. In: *Proc. 5th IFAC Conf. on Manoeuvring and Control of Marine Craft*. Aalborg, Denmark.

Rawlings, J.B. and K.R. Muske (1993). The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control,* **38**, 1512–1516.

Richards, A. and J.P. How. (2003). Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility. In: *Proc. American Control Conference*. Denver.

Venkat, A.N., J.B. Rawlings and S.J. Wright (2004). Plant-wide optimal control with decentralized MPC. In: *Proc. 7th International Symposium on Dynamics and Control of Process Systems (DYCOPS)*. Cambridge, MA.