Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Model Predictive Control using Hybrid Feedback

Mathieu Gerard [**], **Michel Verhaegen**

Delft Center for Systems and Control, Delft University of Technology,
Mekelweg 2, 2628 CD Delft, The Netherlands.
[**] *e-mail: m.p.gerard@tudelft.nl*

Abstract: Traditional Model Predictive Controllers make use of computation expensive optimization methods. The challenge of this research is to take advantage of properties of MPC and use a hybrid gradient descent method to replace the on-line optimization by a simple set of differential equations. Continuous- and discrete-time controllers are presented. Promising results are provided for the control of linear systems with smooth convex constraints with different controller tunings. This technique, even if slightly suboptimal, has a clear interpretation, is efficient and is suitable for implementation on limited embedded microcontrollers.

1. INTRODUCTION

A very interesting modern control method is Model Predictive Control (MPC), see Camacho and Bordons [1995], Bemporad [2006] and Qina and Badgwell [2003]. Thanks to an internal model of the plant, the controller is able to predict its expected future outputs and therefore optimize the current input accordingly. The main advantages of MPC are the following:

- It can very easily handle multi-input multi-output processes, processes with large time-delay, non-minimum phase processes, and unstable processes;
- It is easy to tune;
- It can take constraints into account in a natural manner: input constraints as well as output constraints or state constraints;
- It can be easily reconfigured by changing the internal model, for example in case of fault detection.

MPC is formulated as an optimization problem on a receding horizon. At all time, a cost function penalizing the difference between the desired and expected outputs should be minimized under the constraints. A huge amount of publications already studied properties and tuning of MPC cost functions and constraints, see for example Maciejowski [2002], Garcia et al [1989], Rossiter [2003] and references therein. This paper will instead focus on a new way to implement the controller based on given MPC characteristics.

Typically, most implementations of MPC are done using discrete-time controllers and a discrete-time model of the plant is used as an internal model. Continuous-time models could also be used like in Receding Horizon Control, see Primbs [1999]. This paper focuses on MPC based on discrete-time internal models. A large number of papers in the literature rely on on-line optimization, where a traditional optimization method is used and the exact value of the optimum is computed at each time step, see Maciejowski [2002]. Other approaches compute a multiparametric optimal solution beforehand (off-line) and store the results for on-line use, see Bemporad et al [2000]. In the first case, such an on-line optimization can be very complex and demanding, especially on small embedded microcontrollers. In the second case, the amount of data to be stored can become extremely large if many parameters or constraints appear in the problem formulation. Thus, real-time implementation of MPC is not an easy task and this is probably one reason why it is cautiously used in embedded systems with fast dynamics.

In control theory, most of the systems and controllers are described as dynamical systems, i.e. vector differential equations on the form

$$\dot{x} = f(x(t))$$

The objective of the research presented here is to investigate the possibility to also implement a MPC-like controller using a traditional dynamical system.

Moreover, the challenge is to compete with traditional techniques while reducing the complexity and the computation cost. Obviously the proposed solution will have limitations, but it will aim at exploiting characteristics of MPC to design the feedback controller.

The basis of this new method relies on the fact that MPC can be seen as a time-varying cost function that should be minimized over time by updating the optimizations variables, i.e. the state of the controller. In this paper, the focus will be on the control of linear systems that leads to convex cost functions. The Reader is referred to Boyd and Vandenberghe [2004] for more details about the notions optimization and convexity. Moreover, the state of the controller should stay at all time in a prescribed smooth convex set. Therefore, a constrained gradient method can be used and the idea proposed by the authors in Gerard [2007] will be exploited. If rate constraints are present in the problem, which is often the case when actuators are involved, they can directly be used to define the speed of the descent method.

A more detailed description of MPC is given in Section 2. The proposed controller, in its continuous-time version, is presented and analysed in Section 3. Section 4 discusses a possible discretization of the controller to allow discrete-

time implementation. Finally Section 5 shows simulation results for a high-order linear system.

2. MODEL PREDICTIVE CONTROL

In MPC, the future evolution of the system on a certain time interval in considered. An internal model of the process to be controlled is used to compute the predictions. To limit the number of variables and simplify the computation, a discrete-time internal model is used with a sampling time ΔT. Therefore, if the noise-free process has the state-space model

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{1}$$

then the equivalent internal model takes the form

$$\begin{cases} x_{k+1} = \tilde{A}x_k + \tilde{B}u_k \\ y_k = \tilde{C}x_k \end{cases} \tag{2}$$

Using a prediction horizon N_p, the prediction covers the time interval $[t, t + N_p\Delta T]$; and the cost function will be based on the value of the output at the N_p next sampling time instants. The variables to be optimized in the procedure are the input u at the following N_u sampling time instants. Through this paper, t is the measurement time; therefore time before t has already been measured and is known, while time after t is expected by the controller. Using the following variables:

$$\tilde{u}(t) = \begin{pmatrix} u(t) \\ u(t+\Delta T) \\ \vdots \\ u(t+(N_u-1)\Delta T) \end{pmatrix} \tag{3}$$

$$\tilde{x}(t) = \begin{pmatrix} x(t+\Delta T) \\ x(t+2\Delta T) \\ \vdots \\ x(t+N_u\Delta T) \end{pmatrix} = \begin{pmatrix} \tilde{A}x(t) + \tilde{B}u(t) \\ \tilde{A}x(t+\Delta T) + \tilde{B}u(t+\Delta T) \\ \vdots \\ \tilde{A}x(t+(N_u-1)\Delta T) + \tilde{B}u(t+(N_u-1)\Delta T) \end{pmatrix} \tag{4}$$

$$\tilde{y}(t) = \begin{pmatrix} y(t+\Delta T) \\ y(t+2\Delta T) \\ \vdots \\ y(t+N_u\Delta T) \end{pmatrix} = \begin{pmatrix} \tilde{C}x(t+\Delta T) \\ \tilde{C}x(t+2\Delta T) \\ \vdots \\ \tilde{C}x(t+N_u\Delta T) \end{pmatrix} \tag{5}$$

the traditional MPC cost function q is

$$q(\tilde{u}(t), x(t), \tilde{y}_{ref}(t)) = ||\tilde{y} - \tilde{y}_{ref}||^2_{Q_y} + ||\Delta\tilde{u}||^2_{Q_u} \tag{6}$$

where $\tilde{y}_{ref}$ is the desired output at the N_p next sampling time instants, $\Delta\tilde{u}$ contains the value of the signal $u(t) - u(t-\Delta T)$ at the next N_u sampling time instants and Q_y and Q_u are weighting matrices.

This cost function can be rewritten in a standard quadratic form

$$q(\tilde{u}(t), x(t), \tilde{y}_{ref}(t)) = \frac{1}{2}\tilde{u}^T Q\tilde{u} + x^T R^T\tilde{u} + \tilde{y}^T_{ref} S^T\tilde{u} + c(x, \tilde{y}_{ref}) \tag{7}$$

where c does not depend on $\tilde{u}$ and Q, R and S are time-independent matrices. Note that the cost function depends on the future values of u and y_{ref}, which are embedded in $\tilde{u}$ and $\tilde{y}_{ref}$ respectively; while it is parametrized by the value of x only at the current time t which is the "initial state" of the current horizon. For concision, the explicit dependency on t is omitted.

The gradient is therefore

$$\nabla_{\tilde{u}} q = \frac{\partial q}{\partial \tilde{u}} = Q\tilde{u} + Rx + S\tilde{y}_{ref} \tag{8}$$

Then a set of constraints is defined for the variables. Constraints on the control input can directly be expressed while constraints on state or outputs require the use of the internal model. In this paper, the constraint set, also called feasible set, is restricted to the smooth convex case described by

$$g_i(\tilde{u}, x) \leq 0 \quad i = 1...m \tag{9}$$

where $g_i : \Re^n \to \Re$ is a differentiable convex function. Obviously, if the internal model is used to express constraints, the functions g_i will also depend on the state x.

Furthermore, rate constraints can be expressed on the control inputs. The general form of those constraints are the following:

$$|\dot{u}(t)| < \text{umaxrate} \quad \forall t \tag{10}$$

In traditional implementation, to give time for the optimization to be performed, the MPC controller is implemented in discrete time. In this paper, we will first have a look at the continuous-time case.

3. THE HYBRID FEEDBACK CONTROLLER

It has been shown that the cost function of the MPC only depends over time on $\tilde{u}(t)$, $x(t)$ and $\tilde{y}_{ref}(t)$. Therefore the objective of the research is to design a feedback controller on the form

$$\dot{\tilde{u}}(t) = f(\tilde{u}(t), x(t), \tilde{y}_{ref}(t)) \tag{11}$$

where $f : \Re^n \to \Re^n$ is a vector function easily defined using the cost function and the constraints of the MPC formulation, that would perform, to a certain extent, close to a traditional MPC. A graphical representation of the control scheme is presented in figure 3.

In practice, control inputs are always linked to actuators which have limited bandwidth, i.e. limited rate of change. Therefore it is interesting to take it into account in the controller design. Then the controller should not try to change instantaneously the value of the control input but should rather continuously push each actuator in the best direction at maximum rate. Therefore it seems logical to imagine the controller modeled as a vector differential equation where f is the best direction. Furthermore, in this framework, the norm of f can be scaled to precisely match the rate limit so that the control inputs evolves at maximum rate. This is done using a saturation function.

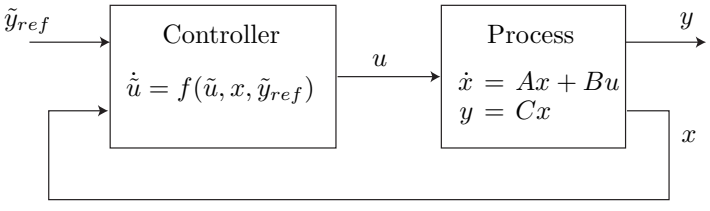

Figure 1. Controller scheme

In general, the best direction f will be close to the gradient of the cost function. However, special care should be taken with respect to the constraints. Therefore the proposed controller is based on the hybrid constrained steepest descent method developed by the authors in Gerard [2007].

The MPC cost function depends on the future values of the output reference. In case of a time-varying reference, the cost function will vary accordingly and can therefore conveniently be seen as time-varying. Also it might be interesting to consider time-varying constraints which could depend on the operating point. The proposed method is perfectly suited in case of time-varying cost function and constraints. Firstly because the direction f will be adapted at all time to take the variations of the cost function into account, therefore $\tilde{u}$ is always going toward the instantaneous optimal point. And secondly because when a constraint is changed and becomes not satisfied, $\tilde{u}$ will be instantaneously pushed toward the feasible set.

This leads us to the hybrid feedback controller described by the following equations:

$$\dot{\tilde{u}}(t) = \bar{f}(\tilde{u}(t), x(t), \tilde{y}_{ref}) \tag{12}$$

with, where only the dependency in $\tilde{u}$ is emphasized,

$$\bar{f}(\tilde{u}) = \text{sat}\left[\alpha f(\tilde{u}),\ \text{umaxrate}\right] \tag{13}$$

where sat is the traditional saturation function that limits the value of $\alpha f(\tilde{u})_i$ to + or - umaxrate$_i$; and

$$f(\tilde{u}) = \begin{cases} -\nabla_{\tilde{u}} q & \text{if } g_i(\tilde{u}) \leq 0 \ \ \forall i \\ -\sum\limits_{i \in L(\tilde{u})} \nabla_{\tilde{u}} g_i & \text{otherwise} \end{cases} \tag{14}$$

with $L(\tilde{u}) = \{l : g_l(\tilde{u}) \geq 0\}$.

α is a tuning parameter that scales the descent direction. To fully exploit the capabilities of the actuators in terms of rate of change, a large enough α should be used. However, a too large α will lead to a too reactive controller that could be difficult to discretize, as will be seen later on.

This control system has been proved to have the following properties in Gerard [2007]:

- for $\tilde{u}(t)$ outside the feasible set, the trajectory $\tilde{u}(t)$ converges to the feasible set, i.e. $\exists t_f$ s.t. $g(\tilde{u}(t_f)) \leq 0$.
- the trajectory $\tilde{u}(t)$ remains in the feasible set as soon as $\tilde{u}(t_f)$ is in the set, i.e. $g(\tilde{u}(t)) \leq 0 \ \ \forall t > t_f$ s.t. $g(\tilde{u}(t_f)) \leq 0$;
- for $\tilde{u}(t_f)$ in the feasible set, the trajectory $\tilde{u}(t)$ decreases the cost function $q(\tilde{u}(t))$ at all time until $q(\tilde{u}) = q^*$, with q^* the optimal value of q under the constraints, i.e. $q(\tilde{u}(t_1)) > q(\tilde{u}(t_2))\ \forall (t_1, t_2)$ with $t_f \leq t_1 < t_2$ s.t. $q(\tilde{u}(t_1)) > q^*$, and $\lim_{t\to\infty} q(\tilde{u}(t)) = q^*$.

Therefore, it can be concluded that if the cost function stay constant, then the control inputs will converge to the optimum that achieves the best results regarding the cost function. If the cost function does vary with time, then the control inputs will at all time go toward the instantaneous optimum. Because there are rate constraints and because the feedback system is running at maximum speed, the performances should very similar to traditional implementation.

Unfortunately, because of the hybrid characteristics and in particular the sliding mode, see Filippov [1960], such a controller can not be directly implemented. Some authors have already studied the implementation of discrete-time sliding mode control and found interesting rules to deal with them. However, as first approach, a very simple discretization is done.

4. DISCRETIZATION

For practical implementation, a good idea is probably to sample the system at a given sampling frequency. In that way, the computation can easily be scheduled on time-driven microcontrollers and the number of switches in the sliding mode is limited to one every period.

Obviously, because of the sampling, the accuracy of the system is going to be reduced. Two main drawbacks can be seen:

- during a sliding mode, the trajectory will not stay perfectly on the boundary but will meander slightly around it. The amplitudes of the oscillations will depend on the sampling period and the norms of the gradients $\nabla_{\tilde{u}} q$ and $\nabla_{\tilde{u}} g_i$.
- the trajectory will not precisely converge toward the precise optimal point but will vacillate around it.

The sampling is done by approximating the derivative by a forward difference. For a sampling time Δt, the difference equation is given by

$$\tilde{u}_{k+1} = \tilde{u}_k + \bar{f}(\tilde{u}_k, x_k, \tilde{y}_{ref_k})\Delta t \tag{15}$$

The sampling method in this paper is rather simple. Some more advanced methods exist and might lead to improved behaviours. The possibility to use such advanced methods and their benefits will be investigated in future work.

It should be noted that the sampling frequency of the controller does not need to be related to the sampling frequency of the internal model. Two different notations are used in this paper.

5. SIMULATION

Finally let us now have a look at some simulations to show that the method is working and to analyse the influence of the parameters Δt and α. The first simulations focus on input constraints while the last one introduces output constraints. A random stable non-minimum phase linear system is taken as process to be controlled. The transfer function contains 11 poles and 2 zeros in the left half plane, and one zero in the right half plane. The pole-zero map is shown in figure 2.

For use as internal model in the MPC, the model should be discretized and transformed to a state-space form. The sampling time is here taken as $\Delta T = 0.1$.

The control input is constrained by both a range and a rate limit, which apply for each component of $\tilde{u}$:

$$\begin{aligned} \text{umax} &= 0.8 \\ \text{umaxrate} &= 3 \end{aligned} \tag{16}$$

The tuning of the MPC is not the primary objective of this paper. Moreover, it might be of interest to show that

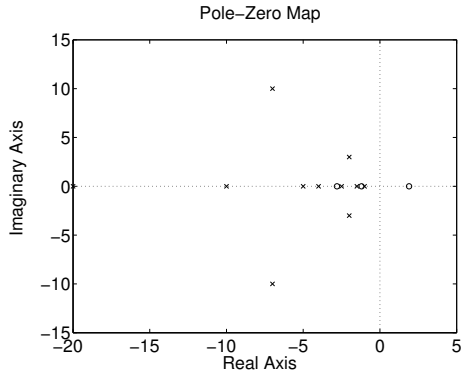

Figure 2. Pole-zero map of the process to be controlled.

the controller is able to deal with large horizons. So the following parameters are taken: control horizon $N_u = 20$, prediction horizon $N_p = 40$, and identity matrices for Q_y and Q_u.

In the simulation, the initial state of the process is $x_0 = [0, 0]$ and a step output reference is asked after 1 second. Note that in this setup, the controller is not aware of the change in set-point before it actually takes place. This precisely happen when a human operator set in real-time the reference of the controller.

For comparison purpose, the MPC is implemented both using traditional optimization techniques and the new hybrid dynamic feedback. The traditional version relies on the solver QuadProg in Matlab. The response is given by

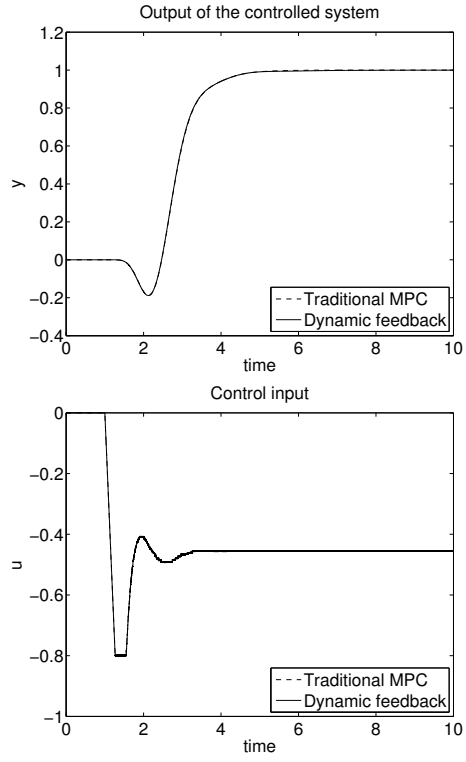

Figure 3. Simulation of the dynamic feedback Model Predictive Controller discretized with a very high sampling frequency. α is also chosen large. For comparison, an implementation of a traditional MPC is shown as dotted line.

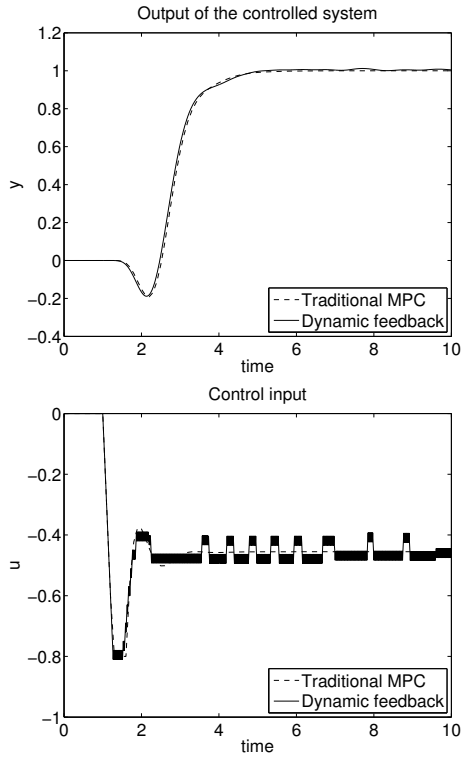

Figure 4. Simulation of the dynamic feedback Model Predictive Controller discretized with a 10 times longer sampling period. α is still large. Oscillations appears when a constraint is reached and near the steady-state.

the dotted lines. The feedback system (12) is tuned using first a large and then a small α. In the case of the large α, 3 sampling frequencies are compared.

The first simulation gives a case that should be very close to the continuous version. To this end, the sampling frequency is taken very high: $\Delta t = 0.001$. In order to make sure that the controller will be reactive enough to fully exploit the actuators, a large value is taken for α : $\alpha = 100$. The results are shown in figure 3. The results are extremely close compared to the traditional MPC. The desired steady-state is reached in the same time and the constraints are perfectly respected. This seems really promising.

Then the sampling frequency of the controller is decreased and a 10 times larger sampling period is taken $\Delta t = 0.01$. The plots are displayed in figure 4. As it was expected, oscillations appears in the control input during a sliding mode, i.e. when a constraint is reached, and when the minimum of the cost function is reached, i.e. near the steady-state. However, the global behaviour is still really good.

To investigate further the influence of the sampling frequency of the controller, the sampling time is again multiplied by 10 to reach $\Delta t = 0.1$. As can be seen in figure 5, the oscillations are even larger; which had to be expected. A small steady state error appears because the oscillations are not symmetric compared to the steady-state control input. But on the other hand, the trajectory is still going in the right direction. So it can be concluded that the

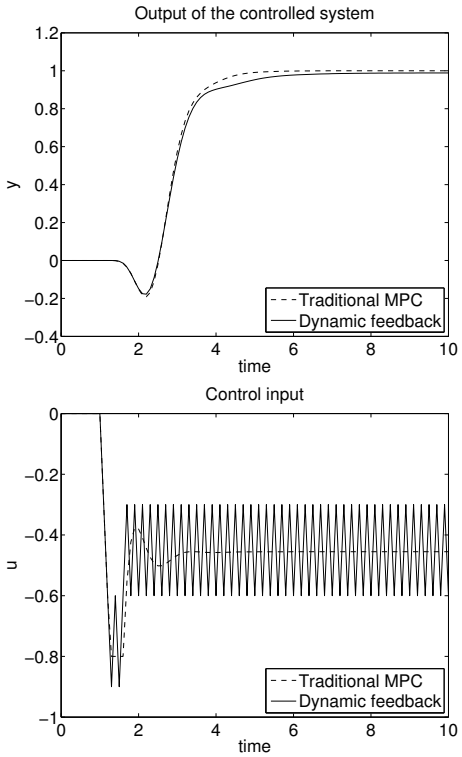

Figure 5. Simulation of the dynamic feedback Model Predictive Controller discretized with an again 10 times longer sampling period. α is still large. The oscillations becomes even larger and a steady-state error appears but the behaviour of the system is still globally correct.

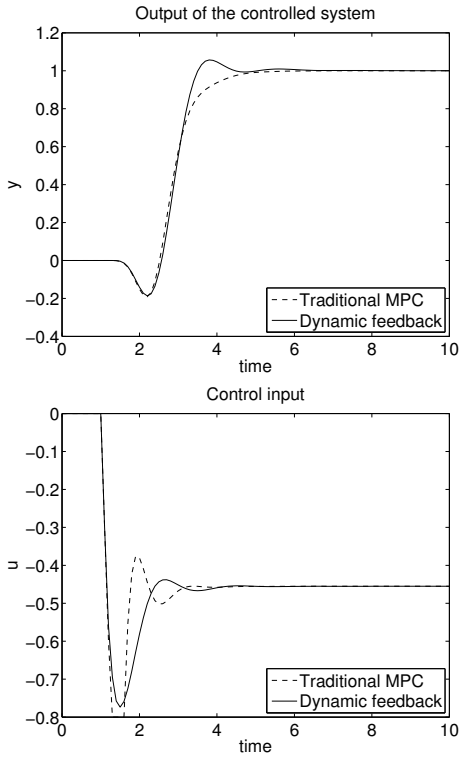

Figure 6. Simulation of the dynamic feedback Model Predictive Controller discretized with the same long sampling period; but α is now chosen very small. Oscillations and steady-state error completely disappear but the response is slower.

sampling time will influence the oscillations and the precision of the steady-state; but it also directly influences the computation complexity.

From the equations defining the controller (12)-(14), it can be understood that α will have a large influence on the size of the oscillations. To show this, the same simulation with a large sampling period $\Delta t = 0.1$ is repeated with an α much smaller: $\alpha = 0.01$. Figure 6 show the results. It can easily be noticed that the oscillations have been completely removed and that the steady-state error has completely disappeared; but at a cost of a slower response. Therefore, a good tuning of α or even an adapting value could improve largely the method.

Furthermore this method allows, similarly to traditional MPC, the expression of output constraints. This is done by including the internal model in the computation of the constraint set. For linear systems, the predicted outputs $\tilde{y}$ can be computed using two time-invariant prediction matrices P_x and P_u and the equation

$$\tilde{y}(t) = P_x x(t) + P_u \tilde{u}(t) \tag{17}$$

It is then possible to guarantee that y will stay above a minimum value y_{min} by using the state-dependent constraints

$$g(\tilde{u}, x) = -P_u \tilde{u} - P_x x + y_{min} \leq 0 \tag{18}$$

For the same system described by the pole-zero map of figure 2, the output constraint $y_{min} = -0.01$ is added to the previous input constraints (16). The results of the simulation are shown on figure 7. It can be seen that the undershoot is very well removed. With a good tuning of α, the results are really good compared to traditional MPC. However, the choice of α seems to be more critical than before. In case of a too low α, the controller will not be reactive enough to steer the system properly at the beginning to respect the constraint. On the other hand, a too large α will again induce very large oscillations near the steady-state. In this simulation, the value of α is decreased after 5 seconds so that both fast response and small oscillations are achieved. This shows the need for an adaptive scheme for α which is under development. So it has been shown that output constraints can be handled very easily by this control method.

6. CONCLUSION

A new way to implement an MPC-like controller using traditional feedback has been presented and simulations support the interest of such a method. This technique has the advantage to be very simple and has a clear interpretation. Moreover, the computation time of each step is extremely low. Also, input constraints as well as output constraints can be handled very naturally.

Unfortunately, the method suffers from the discretization which might lead to oscillations in the control input and steady-state error, especially if α is large. Future work will include the investigation of more advanced discretization methods as well as the development of an

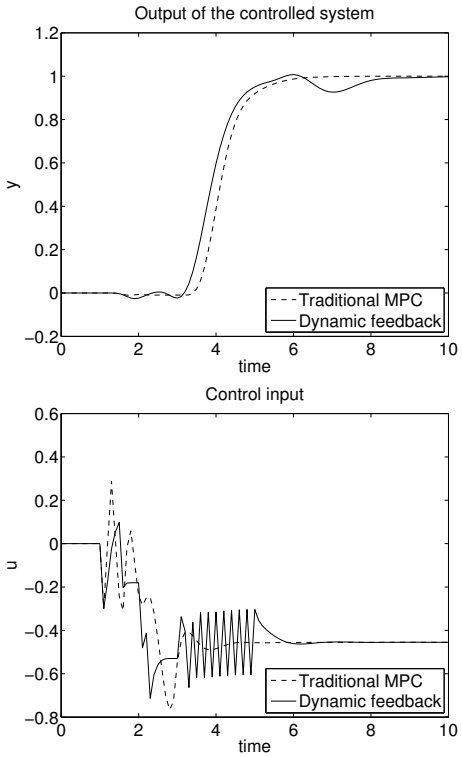

Figure 7. Simulation of the dynamic feedback Model Predictive Controller with output constraints $y_{min} = -0.01$. The controller is discretized with a long sampling period $\Delta t = 0.1$. α has been chosen large at the beginning to rapidly steer the system and respect the constraint; and is decreased after 5 seconds to cancel out the large oscillations. The constraint is well respected and the global behaviour is really good compared to a traditional MPC.

adaptive scheme for α to improve the response speed and damp the oscillations.

In order to investigate further the interest of this method, future work will also include comparisons with other MPC implementations regarding both the performances and the computation time.

Finally, it can be concluded that this method seems to give an opening to link dynamic feedback control with on-line constrained convex optimization, and to allow the implementation of MPC in an efficient way on limited embedded microcontrollers.

REFERENCES

M. Gerard, M. Verhaegen. A Hybrid Steepest Descent Method for Constrained Convex Optimization. To be published.

J.M. Maciejowski. Predictive Control with Constraints. Prentice Hall, Harlow, England, 2002.

C.E. Garcia, D.M. Prett, M. Morari. Model Predictive Control: theory and practice - A survey. Automatica, 25(3):335-348, May 1989.

E.F. Camacho, C. Bordons. Model Predictive Control in the Process Industry. Springer-Verlag, Berlin, Germany. 1995.

J.A. Rossiter. Model-Based Predictive Control: A Practical Approach. CRC Press. 2003.

A. Bemporad. Model Predictive Control Design: New Trends and Tools. 45th IEEE Conference on Decision & Control, San Diego, CA, USA. December 13-15, 2006.

S.J. Qina, T.A. Badgwell. A survey of industrial model predictive control technology. Control Engineering Practice, vol. 11, pp. 733-764, 2003.

A. Bemporad, M. Morari, V. Dua, E.N. Pistikopoulos. The Explicit Solution of Model Predictive Control via Multiparametric Quadratic Programming. American Control Conference, Chicago, USA, vol. 2, pp. 872-876. 2000.

J. Primbs. Nonlinear Optimal Control: A Receding Horizon Approach. PhD Thesis, Control and Dynamical Systems, California Institute of Technology, 1999.

S. Boyd, L. Vandenberghe. Convex Optimization. Cambridge University Press. 2004.

A.F. Filippov. Differential equations with discontinuous right-hand side. Matemat. Sbornik., 51:99-128, 1960. In Russian. English translation: Am. Math. Soc. Transl. 62, 1964.