Constrained Predictive Control Methods
A State Space Formulation

Mihai Huzmezan and Dr. Jan Maciejowski
Control Group, Department of Engineering
University of Cambridge
Cambridge, CB2 1PZ, U.K.
{mh2 , jmm}@eng.cam.ac.uk
tel.+44-1223-339222 fax.+44-1223-32662

Abstract

Specifically, this paper applies a state space formulation of Model Based Predictive Control to flight control systems. It is a case study of an experimental transport aircraft controlled by a MBP controller. A brief theoretical description of MBPC algorithm is given followed by a Matlab-Simulink test framework and simulation results.

Keywords: MBPC, predictive control, flight control.

1 Introduction

This paper addresses a *multi-model state space formulation* of Model Based Predictive Control. The reason for such an approach is that in the alternative *input/output formulation* states are not directly involved in the control algorithm so constraining them is difficult. The multi-model formulation opens the path for reconfigurable control in the case of failures. The main attributes of MBPC are: *Simplicity* – the basic idea of MBPC does not require complex mathematics and is fairly intuitive, *Richness* – all basic components of MBPC like models, horizons, objective function can be tailored to the current problem to be solved and *Practicality* – it deals naturally with all constraints and in practice already works routinely and profitably in process control. It is interesting to exploit these features in the context of flight control. This present paper presents a case study of an experimental transport aircraft controlled by a Model Based Predictive Controller. The first section gives a brief theoretical description of an MBPC algorithm. A Matlab-Simulink test framework is then described and simulation results presented.

2 Description of Model Based Predictive Control technique

The defining feature of MPBC is the repeated optimization of a performance objective over a finite horizon extending from the current or some future time up to a prediction horizon. The use of an explicit model is a requirement of MBPC.

Inherent structures of most of MBPC schemes, see figure 1, are identical in their main format but differ in details. Generally these details are critical for the success of the algorithm.

THE OPTIMIZER contains all constraints and the main attribute of this system component is that it must work iteratively in real time. The main task of the optimizer is to compute the present and future manipulated variable moves such that the predicted outputs follow the reference in a desirable manner.

THE OBSERVER provides state estimates, which can be use in the optimizer.

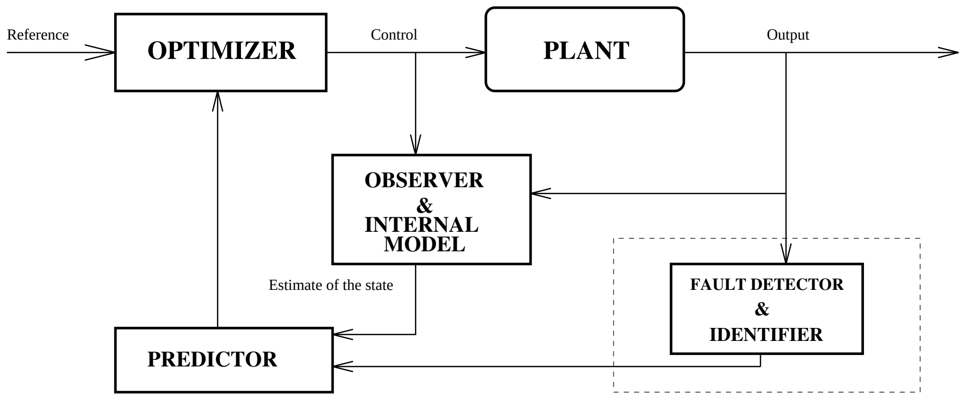

Figure 1: General structure of MPC schemes

THE INTERNAL MODEL represents the plant

THE PREDICTOR is the engine of the predictive control algorithms being able to provide the optimizer with future predicted values of states and outputs.

2.1 Plant model requirements

Considering a possible multi-model context, let index i represent the plant in use:

$$\begin{aligned} x_i(k+1) &= A_i x_i(k) + B_{u_i} u_i(k) + B_{d_i} d_i(k) \\ y_i(k) &= C_i x_i(k) + \nu(k) \end{aligned} \tag{1}$$

where $x_i(k)\in\Re^n$ is the system state vector, $u_i(k)\in\Re^m$ is the system control input vector and $y_i(k)\in\Re^p$ is the vector of outputs, all variables being measured at time k. It is important to note that in the event of a complete actuator failure, the system and respective model state dimension may change such that $x_i(k)\in\Re^{n_i}$.The reader should note that input, state, and output (measurement) vectors are indexed using the variable i in order to indicate the model in use. Here $d_i(k)$ is considered to describe unmeasured state disturbance input and $\nu(k)$ is an unmeasured output disturbance vector. It is assumed that each pair $(A_i, (B_{u_i}, B_{d_i}))$ with controllable modes of (A_i, B_{d_i}) being stable is stabilizable and (A_i, C_i) detectable. There is no particular requirement for the plant model.

2.2 State space formulation

We express the state vector in terms of the change in the manipulated variable (at time k): $\Delta u_i(k) = u_i(k) - u_i(k-1)$, the state disturbance (at same time k): $\Delta d_i(k) = d_i(k) - d_i(k-1)$ and the output disturbance (at sampling time k): $\Delta w_i(k) = w_i(k) - w_i(k-1)$. In such a case, the state space model becomes:

$$\begin{pmatrix} \Delta x_i(k+1) \\ y_{x_i}(k+1) \end{pmatrix} = \begin{pmatrix} A_i & 0 \\ C_i A_i & I \end{pmatrix} \begin{pmatrix} \Delta x_i(k) \\ y_{x_i}(k) \end{pmatrix} + \begin{pmatrix} B_{u_i} \\ C_i B_{u_i} \end{pmatrix} \Delta u_i(k) + \begin{pmatrix} B_{d_i} & 0 \\ C_i B_{d_i} & I \end{pmatrix} \begin{pmatrix} \Delta d_i(k) \\ \Delta w_i(k) \end{pmatrix}$$

$$y_i(k) = \begin{pmatrix} 0 & I \end{pmatrix} \begin{pmatrix} \Delta x_i(k) \\ y_{x_i}(k) \end{pmatrix} + \nu(k)$$

where $y_{x_i}(k+1)$ corresponds to a vector of outputs free from measurement noise. The augmented system will be referred to by the realization $\hat{A}$, $\hat{B}$, $\hat{C}$:

$$\begin{pmatrix} \Delta x_i(k+1) \\ y_{x_i}(k+1) \end{pmatrix} = \hat{A}_i \begin{pmatrix} \Delta x_i(k) \\ y_{x_i}(k) \end{pmatrix} + \hat{B}_{u_i} \Delta u_i(k) + \begin{pmatrix} \hat{B}_{d_i} & \hat{B}_{w_i} \end{pmatrix} \begin{pmatrix} \Delta d_i(k) \\ \Delta w_i(k) \end{pmatrix}$$

$$y_i(k) = \hat{C}_i \begin{pmatrix} \Delta x_i(k) \\ y_{x_i}(k) \end{pmatrix} + \nu(k)$$

Certain elements of the state vector may not be available for measurement at time k so a Kalman observer will be employed.

2.3 State prediction

As mentioned, since certain elements of the state vector may not be available for measurement at time k we will determine a set of state estimates for each model of the plant in use. The estimated states will be either the states of the main system or states of that system augmented by the disturbance variables, as shown in the previous section.

The state estimates $\left(\Delta x_i(k) \quad y_{x_i}(k)\right)$ will be denoted as $\hat{x}_i(k)\in\Re^{(n+\hat{p})}$ (where $\hat{p}$ represents the number of outputs uncorrupted by measurement noise) and they will rely upon selecting a particular observer through the weighting matrices: Q_d, Q_w, Q_v.

So, in general, state estimation will take the following form at time $k+j$ ($j = 0...N_2-1$) from the prediction horizon N_2:

$$\hat{x}(k+j+1) = F_i(j)\hat{x}(k+j) + G_i\Delta u_i(k+j) + H_i(j)y_i(k+j) \tag{2}$$

where $\Delta u_i(k+j)$ is a vector of future control changes.

We should emphasize that $F_i(j)$, G_i, $H_i(j)$ matrices are derived, from the open-loop model itself, in a way that depends on the estimation/prediction scheme employed. This depends on the type of observer chosen and its weighting matrices as well as on the disturbances, that affect the plant, and on noise upon measurements. We assume that $F_i(j)$, G_i, $H_i(j)$ depend only on i (the model set) and j (time in the future) but not on k (current time) in order to exclude certain possibilities, such as non-stationary Kalman filters.

Some simplifications, such as: $H_i(j) = 0$, $F_i(0)\neq F_i(1)$ and $F_i(j) = F_i(j+1)$ for $j > 0$ may occur. In this case the state estimator is constructed in the following way:

$$\hat{x}_i(k+1) = F_i(0)\hat{x}_i(k) + G_i\Delta u_i(k) + H_i(0)y_i(k) \tag{3}$$

in which $H(0) = K$, $F(0) = \hat{A} - H(0)\hat{C}$, $F(1) = F(2) = \ldots = F(N_2-1) = \hat{A}$ and $G = \hat{B}$. For the future computation it is not necessary to assume the last two simplifications.

From the earlier definition of the state space description (1) and using (2) the vector of future state estimates may be written as:

$$\hat{X}_i(k) = \begin{pmatrix} \hat{x}_i(k+1) \\ \vdots \\ \hat{x}_i(k+N_2) \end{pmatrix} = \mathcal{F}_i(k)\hat{x}_i(k) + \mathcal{G}_i\Delta U_i(k) + \mathcal{H}_i(k)y_i(k) \tag{4}$$

We should note here that the vector of the change of state estimates denoted by:
$\hat{X}_i(k) = \left[\hat{x}_i(k+1)^T, \ldots, \hat{x}_i(k+N_2)^T\right]^T$ is dependent only upon the state estimate at time $j = 0$ which is $\hat{x}_i(k)$. Therefore it is necessary to have a state estimator that operates outside of model based controller to determine current state estimate $\hat{x}_i(k)$. In the following, we shall use the notation $\prod_{\lambda=\epsilon}^{\delta} R(\lambda)$ to denote the product:
$R(\epsilon)R(\epsilon-1)R(\epsilon-2)...R(\delta)$ for $\epsilon > \delta$.

Now we are able to define the matrices involved in equation (4):

$$\mathcal{F}_i(k) = \begin{pmatrix} F_i(0) \\ F_i(1)F_i(0) \\ \vdots \\ \prod_{j=N_2-1}^{0} F_i(j) \end{pmatrix}$$

$$\mathcal{G}_i = \begin{pmatrix} G_i & 0 & \cdots & 0 \\ F_i(1)G_i & G_i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \left[\prod_{j=N_u-1}^{1} F_i(j)\right] G_i & \left[\prod_{j=N_u-1}^{2} F_i(j)\right] G_i & \cdots & G_i \\ \vdots & \vdots & \ddots & \vdots \\ \left[\prod_{j=N_2-1}^{1} F_i(j)\right] G_i & \left[\prod_{j=N_2-1}^{2} F_i(j)\right] G_i & \cdots & \left[\prod_{j=N_2-1}^{N_u} F_i(j)\right] G_i \end{pmatrix}$$

$$\mathcal{H}_i(k) = \begin{pmatrix} H_i(0) \\ F_i(1)H_i(0) \\ \vdots \\ \prod_{j=N_2-1}^{1} H_i(0) \end{pmatrix}$$

Here the significance of N_u and N_2, $N_u \leq N_2$ is as defined in previous sections of this chapter (N_2 is considered to be the maximum optimization horizon).

The optimization assumes that $\Delta u_i(k+j) = 0$ for $j \geq N_u$. The vector of future manipulated variable changes will be denoted by:
$\Delta U_i(k) = \left[\Delta u_i(k)^T, \ldots, \Delta u_i(k+N_u-1)^T\right]^T$.

2.4 The model associated constraints

For each model set i we have an associated constraint set, expressed as linear, inequalities corresponding to:

- control signals (actuator limits):
  Stacking over the control horizon the inputs in the following form:
  $U_i(k) = \left[u(k)^T, \ldots, u(k+N_u)^T\right]^T$ we are able to write the linear inequalities that express the lower and the upper limit values in the form:
  $$m_{min_i}(k) \leq U_i(k) \leq m_{max_i}(k) \tag{5}$$
  where $m_{min_i}(k)$ and $m_{max_i}(k)$ contain all minimum and maximum values across the control horizon. An even more compact form of writing 5 is:
  $$M_i U_i(k) \leq m_i(k);\ m_i(k) = \begin{pmatrix} -m_{min_i}(k) \\ m_{max_i}(k) \end{pmatrix}$$
- changes of control signals (actuator rates):
  Proceeding with a similar stacking procedure over the control horizon we write the linear inequalities that express the minimum and the maximum limit values in the form:
  $$l_{min_i}(k) \leq \Delta U_i(k) \leq l_{max_i}(k) \tag{6}$$
  where $l_{min_i}(k)$ and $l_{max_i}(k)$ contains all the limit values across the control horizon.
  Following the same path as for the control signals we write the inequality (6) in the form of:
  $$L_i \Delta U_i(k) \leq l_i(k);\ l_i(k) = \begin{pmatrix} -l_{min_i}(k) \\ l_{max_i}(k) \end{pmatrix}$$
- states constraints for the augmented system (variation of the states and the outputs of the model):
  Over the prediction horizon involving the vector:
  $\hat{X}_i(k) = \left[\hat{x}_i(k+1)^T, \ldots, \hat{x}_i(k+N_2)^T\right]^T$ and the minimum and maximum values admissible for the system states across this horizon we have the following linear inequality:
  $$n_{min_i}(k) \leq \hat{X}_i(k) \leq n_{max_i}(k). \tag{7}$$
  The compact form of the inequality 7 is:
  $$N_i \hat{X}_i(k) \leq n_i(k);\ n_i(k) = \begin{pmatrix} -n_{min_i}(k) \\ n_{max_i}(k) \end{pmatrix}.$$

All the above inequalities between vectors are defined by stacking together scalar inequalities. We should note that in these inequalities $\lambda \neq \mu \neq \nu$ in general.

In order to pose the problem as a standard quadratic programming optimization we need to have all the inequalities stacked one upon another as a single linear inequality constraint on the vector of future input changes ΔU_i.

The final result of stacking constraints is:

$$\begin{gathered} \mathcal{D}_i \Delta U_i(k) \leq \mathcal{E}_i \begin{pmatrix} u_i(k-1) \\ \hat{x}_i(k) \\ y_i(k) \\ c_i(k) \end{pmatrix} \\ \mathcal{D}_i = \begin{pmatrix} L_i \\ M_i \Lambda \\ N_i \mathcal{G}_i \end{pmatrix} \\ \mathcal{E}_i = \begin{pmatrix} 0 & 0 & 0 & I_\lambda & 0 & 0 \\ -M_i \mathcal{I}_i & 0 & 0 & 0 & I_\mu & 0 \\ 0 & -N_i \mathcal{F}_i & -N_i \mathcal{H}_i & 0 & 0 & I_\nu \end{pmatrix} \end{gathered} \tag{8}$$

Where Λ and $\mathcal{I}$ are defined as follows:

$$\Lambda = \begin{pmatrix} I_m & 0 & \dots & 0 \\ I_m & I_m & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ I_m & I_m & I_m & I_m \end{pmatrix}$$

$$\mathcal{I} = [\mathcal{I}_m, \dots, \mathcal{I}_m]^T .$$

The vector $c_i(k)$ is defined as:

$$c_i(k) = \begin{pmatrix} l_i(k) \\ m_i(k) \\ n_i(k) \end{pmatrix}.$$

Note that control $u_i(k)$ is still to be determined at time k.

2.5 Design objectives

The purpose of stacking inequalities in the form of (8) was to minimize of the following cost functional:

$$J_i(k) = \sum_{j=1}^{N_2} ||(C_i \hat{x}_i(k+j) - s_i(k+j))||^2_{Q_i(j)} + \sum_{j=0}^{N_u-1} ||\Delta u_i(k+j)||^2_{R_i(j)}$$

using the QP algorithm. In the cost function: $s_i(k)$ represent the vector of set-points for states and outputs (at time k). The weights $Q_i(j)$ and $R_i(j)$ are assumed to be independent of k, although they depend on i and may depend on j.

Now the cost function can be written as:

$$J_i(k) = \Delta U_i(k)^T \left[\mathcal{G}_i^T \mathcal{C}_i^T \mathcal{Q}_i \mathcal{C}_i \mathcal{G}_i + \mathcal{R}_i\right] \Delta U_i(k) + 2\left[[\mathcal{F}_i \hat{x}_i(k) + \mathcal{H}_i y_i(k)]^T \mathcal{C}_i^T - \mathcal{S}_i(k)^T\right] \mathcal{Q}_i \mathcal{C}_i \mathcal{G}_i \Delta U_i(k) + \mathcal{K}_i(k)$$

$$\mathcal{Q}_i = diag\,[Q_i(1), \dots, Q_i(N_2)]$$

$$\mathcal{R}_i = diag\,[R_i(1), \dots, R_i(N_u)]$$

$$\mathcal{C}_i = diag\left[\hat{C}_i, \dots, \hat{C}_i\right]$$

$$\mathcal{S}_i(k) = \left[s_i(k+1)^T, \dots, s_i(k+N_2)^T\right]^T .$$

The term $\mathcal{K}_i(k)$ is independent of $\Delta U_i(k)$ and therefore can be omitted from $J_i(k)$ without affecting the optimization.

In the cost function there are several types of variables:

- constants of the process model, connected with estimation and prediction: $\mathcal{G}_i$, $\mathcal{C}_i$, $\mathcal{Q}_i$, $\mathcal{R}_i$, $\mathcal{F}_i$ and $\mathcal{H}_i$.
- parameters varying with k and the model set i such as $\hat{x}(k)$, $y_i(k)$ and $\mathcal{S}_i(k)$, being initialized at each new optimization.
- parameters constant during the optimization algorithm like N_2, and N_u that define the various horizons.
- the real optimization variable vector $\Delta U_i(k)$

Now it is possible to obtain the solution of the optimization problem as a result of the following QP problem:

$$\min_{\Delta U_i(k)} \Delta U_i(k)^T \mathcal{A}_i \Delta U_i(k) + \left[\hat{x}_i(k)^T, y_i(k)^T, \mathcal{S}(k)^T\right] \mathcal{B}_i \Delta U_i(k) \tag{9}$$

subject to the constraints:

$$\mathcal{D}_i \Delta U_i(k) \le \mathcal{E}_i \begin{pmatrix} u_i(k-1) \\ \hat{x}_i(k) \\ y_i(k) \\ c_i(k) \end{pmatrix}$$

where the matrices $\mathcal{D}_i$ and $\mathcal{E}_i$, $c_i(k)$ are obtained as shown in section 2.4. The other unknown matrices are defined as:

$$\mathcal{A}_i = \mathcal{G}_i^T \mathcal{C}_i^T \mathcal{Q}_i \mathcal{C}_i \mathcal{G}_i + \mathcal{R}_i$$

$$\mathcal{B}_i = 2\begin{pmatrix} \mathcal{C}_i \mathcal{F}_i & \mathcal{C}_i \mathcal{H}_i & -I_{N_2 p} \end{pmatrix}^T \mathcal{Q}_i \mathcal{C}_i \mathcal{G}_i$$

where $N_2 p$ is the dimension of the vector of set point trajectories at time k.

3 The model

3.1 Actuator signals

Control effectors used to control the longitudinal aircraft channel are the elevator (δ_E) and the engine commands (δ_{Th1} and δ_{Th2}). Constraints involved in the optimization algorithm are saturation of actuators and actuator rate limits (see table 1):

Limits	Constrained Variable Name	Unit
$-0.436 \leq \delta_E \leq 0.174$	saturation of elevator deflection	rad
$-0.261 \leq \dot{\delta}_E \leq 0.261$	rate limits of elevator deflection	rad/s
$0.009 \leq \delta_{Th} \leq 0.174$	engines throttle limits	-
$-0.028 \leq \dot{\delta}_{Th} \leq 0.028$	engine throttle slew rates	-

Table 1: Longitudinal channel actuator constraints

3.2 Filters and models

The decoupling into longitudinal and lateral channels is performed at the level of the linear model therefore corresponding reference model is provided to the MBPC algorithm.The longitudinal controller uses pitch rate (q), ground speed (V), vertical rate ($\dot{z}$) as feedback signals in the inner loop. Corresponding internal states of the model are pitch rate (q), pitch angle (θ), velocity in x direction of body axes (u_B) and velocity in z direction of body frame (w_B). The outer loop provides altitude tracking, and uses altitude as an output feedback signal. For the longitudinal channel the model is:

```
AS =    -0.9825     0.0000    -0.0007    -0.0161
         1.0000          0          0          0
        -2.1937    -9.7758    -0.0325     0.0743
        77.3570    -0.7675    -0.2264    -0.6684

BS =    -2.4379     0.2912     0.2912
              0          0          0
         0.1837     9.8100     9.8100
        -6.4785          0          0

CS =     1.0000          0          0          0
              0   -79.8667    -0.0283     0.9996
              0     0.0000     0.9996     0.0290

DS =     0     0     0
         0     0     0
         0     0     0

The eigen values of the AS matrix:

      real      imaginary   frequency      damping
(1)  -0.01139    -0.1264      0.126        0.0897
(2)  -0.01139     0.1264      0.126        0.0897
(3)  -0.83029    -1.1069      1.383        6.0004
(4)  -0.83029     1.1069      1.383        6.0004
```

3.3 Reference signals

Regarding the way the future set point is defined and the meaning of horizons the figure 2 is extremely relevant. Most MBPC schemes assume a priori knowledge of the future set point. In this case we employed MBPC as a SAS fact that assumed no knowledge of the future set point sequence being available. Therefore we generate the reference trajectory $s_i(k)$ by filtering the available set point ($s_i(k) = \cdots = s_i(k+j) = \ldots s_i(N_2)$). In our particular case the plant is a transport aircraft therefore such an assumption can be maintained because of the slow variation from one discrete time step to another of the command, provided either by the pilot or by the outer loop controller. The way we generate the set point assumes that it is reached within the control horizon and the performance parameters are met at the end of prediction horizon.

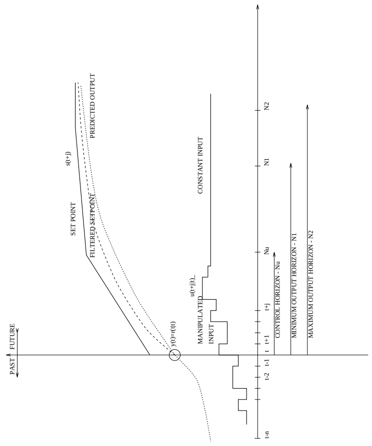

Figure 2: Prediction Strategy

4 Simulation results

In order to perform synthesis and analysis in the MBPC case, a particular test bed has been developed. The test framework is presented in figure 3. Various variables are taken out to the Matlab workspace in order to help design procedures. The integration of MBPC with the plant nonlinear model is the main functional task that the test framework performs. The major novelty in this framework is that it includes a controller that performs an on-line optimization. Therefore several problems that occur involving delays and synchronization between functional blocks were solved using built-in, but hidden, Simulink features used for multi-rate discrete time systems. This test bed will become the base of a Matlab-Simulink constrained predictive control toolbox providing in the user interface all facilities requested for the MBPC type of control analysis and synthesis.

The setup programs perform the parameter definition and tuning and prepare the optimization to be carried out at any discrete time instant. The heart of the MBPC algorithm consists of an optimizer, involving the quadratic programming algorithm, that provides the control increment for the next step.

4.1 Longitudinal channel – inner loop

All the results presented were produced using the environment presented in figure 3.

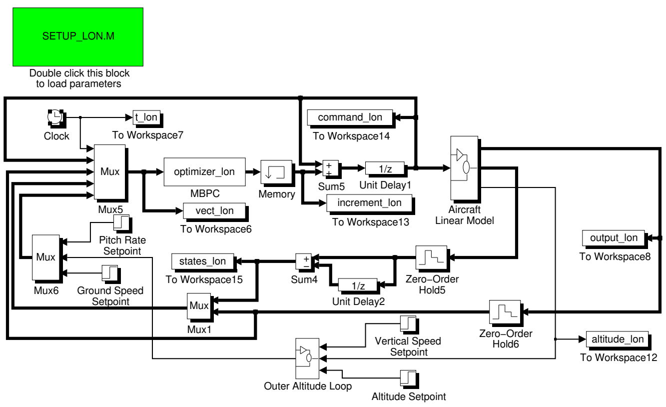

Figure 3: The Simulation Framework for the linear aircraft system

The plant used for control was the linear model of the aircraft. Within the MBPC the discrete time representation of same model was employed as a reference model. When simulating responses no pre-filters were used to attenuate transients during the response to step commands.

4.1.1 Velocity response

The system response to a step of 13 m/s in commanded velocity is shown in figure 4. No pre-filters were used when the response was plotted.

Most of the requirements were satisfied. The rise time in air speed is around the specified 12 s and the settling time is at about 45 s. The overshoot, due to no a priori information assumed to be available, is slightly higher the the specified 5 %. However, the cross coupling to flight path angle is as well a bit higher than the specified 0.5°.

As the second performance parameter the system response to a step change in airspeed of 13 m/s, caused by a wind shear, was calculated and plotted in figure 5. The flight path angle was perturbed by approximately 4°, which is higher than the specified value on cross coupling from air speed to path angle.

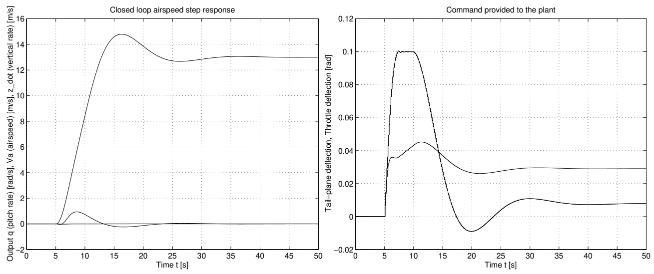

Figure 4: Linear system - Demanded air speed step response
(graph 1 - outputs [V, q, $\dot{z}$] versus time [s]; graph 2 - inputs [δ_E, $\delta_{Th1,2}$] versus time [s])

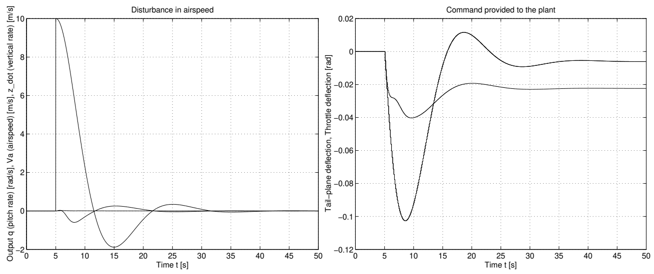

Figure 5: Linear system - Step disturbance in air speed measurement
(graph 1 - outputs [V, q, $\dot{z}$] versus time [s]; graph 2 - inputs [δ_E, $\delta_{Th1,2}$] versus time [s])

4.1.2 Flight path angle response

We subject the system to a step of 3° in commanded flight path angle. The response is shown in figure 6. The amplitude of the actuators commands is reduced at the cost of rising time. The tracking ensured is good, the overshoot is small, and the cross coupling to velocity is negligible.

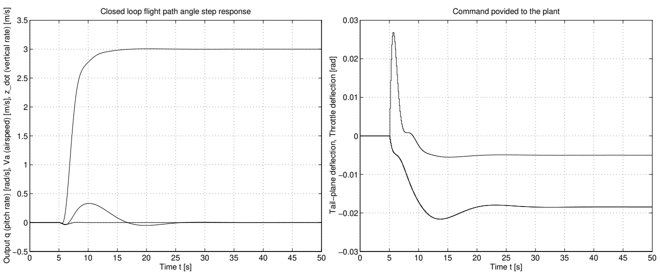

Figure 6: Linear system - Flight path angle response
(graph 1 - outputs [V, γ, $\dot{z}$] versus time [s]; graph 2 - inputs [δ_E, $\delta_{Th1,2}$] versus time [s])

4.2 Longitudinal channel – outer loop

4.2.1 Altitude response

We present the results obtained in the case of altitude tracking employing MPB controller as an inner loop controller and an H_∞ controller as an outer controller for a step of one in altitude demand. In this case the closed inner loop will be consider as a block with flight path angle and velocity commands as inputs. The outer loop added for tracking purposes proved the response provided in figure 7. The requirements on rise time (12 s), settling time (45 s) and overshoot (max 5 %) are met. As well as in the inner loop case no pre-filter was used to produce the shown response.

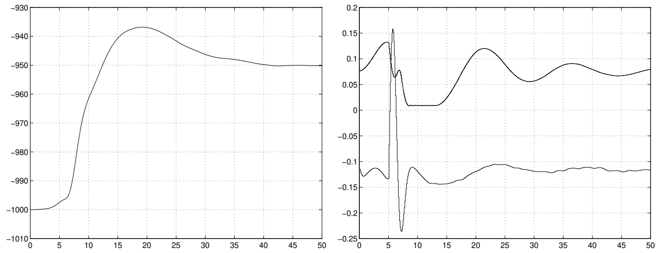

Figure 7: Altitude response
(graph 1 - output [z] versus time [s]; graph 2 - inputs [δ_E, $\delta_{Th1,2}$] versus time [s])

5 Conclusions

The main achievements to date are the implementation of a Simulink test bed and the constrained MBP controller for the nonlinear longitudinal aircraft model. Current activities involve the C language implementation of the MBPC block, improvements at the level of MBPC design such as validation of the MBPC with state estimation and the extension of design methodology in order to have a routine for more general problems (eg. lateral channel).

Long term research objectives are the use of MBPC in reconfigurable control in case of failures, a Matlab-Simulink constrained predictive toolbox, the real time implementation of MBPC algorithms (eventually using a real plant) and to investigate applicability in flight control management systems.

References

[1] J.A. Rossiter B. Kouvaritakis and A.O.T. Chang. Stable generalized predictive control: An algorithm with guaranteed stability. *IEE Procedings-D*, 139(4):349–362, 1993.

[2] R.A.J. de Vries and H.B. Verbruggen. Multivariable unified predictive control. In D. Clarke, editor, *Advances in Model-Based Predictive Control*, pages 84–102. Oxford University Press, 1993.

[3] S.A. Heise and J.M. Maciejowski. Stability constrained MBPC using a internal model control structure. In D. Clarke, editor, *Advances in Model-Based Predictive Control*, pages 230–244. Oxford University Press, 1993.

[4] J.A. Rossiter and B. Kouvaritakis. Advances in generalized and constrained predictive control. In D. Clarke, editor, *Advances in Model-Based Predictive Control*, pages 303–317. Oxford University Press, 1993.

[5] E. Zafiriou. Robust model predictive control of processes with hard constraints. *Computers Chemical Engineering*, 14(4/5):359–371, 1990.