Implementation of Predictive Control in a Hybrid Diesel-Electric Marine Powertrain

Apostolos Serraos

Diploma Thesis

School of Naval Architecture and Marine Engineering
National Technical University of Athens

Supervisor: Assistant Prof. George Papalambrou

Committee Member : Assist.Prof. Ch. Papadopoulos

Committee Member : Prof. K. Kyriakopoulos

September 2017

Aknowledgements

This work has been carried out at the Laboratory of Marine Engineering (LME) at the School of Naval Architecture and Marine Engineering (SNAME) of the National Technical University of Athens (NTUA), under the supervision of Assistant Professor George Papalambrou.

I would like thank Professor Nikolaos Kyrtatos for providing the opportunity to work with the full-scale hybrid diesel-electric marine propulsion powertrain of LME. Access to LME experimental facilities was essential in order to verify theoretical concepts from a practical perspective and evaluate experimentally the designed control system.

I owe my greatest appreciation to my thesis supervisor Assistant Professor George Papalambrou, for giving me the chance and motivation to work on this thesis. I would also like to thank him for his patience, continuous support and immense knowledge. His guidance helped me in all the time working on this thesis.

I would also like to thank Professor Konstantinos Kyriakopoulos (Mech Eng, NTUA) and Assistant Professor Christos Papadopoulos (SNAME, NTUA) for evaluating my work and being a member of my supervisors committee.

I would like to express my sincere gratitude to Mr. Nikolaos Planakis, PhD student of LME for his feedback and held during the development phase of the control system as well as for his help during the experimental setup and evaluation of the control system performance. His practical insight and experience on the matter at hand was really precious.

I take this opportunity to express gratitude to all LME fellow members for their help and support.

I am also sincerely grateful to my family, for the unceasing encouragement, support and motivation throughout my studies.

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture, provided with feedback and support. This accomplishment would not have been possible without them.

Abstract

In this thesis, the feasibility of using model predictive control in a hybrid diesel-electric marine powerplant is investigated. The effort was concentrated in designing a Model Predictive Controller (MPC) without using any commercially available MPC software, like the Model Predictive Control Toolbox (MatLAB) in order to create a fully customizable, fully parametric controller which could be used in the experimental facility of LME, NTUA. The work consists of two main parts. The first part is the design of the controller, as this is mathematically described in the available literature and the second part is the experimental evaluation of the controller in the experimental testbed of LME, NTUA.

Model Predictive Controller is a model-based controller which adopts a more natural approach than other classic controllers. With the help of its internal model it tries to predict the future behaviour of the system and compute the optimal sequence of control actions during a finite time horizon. This approach is called "natural" because this is the way a human would operate when asked to handle a problem. Before acting it would consider a number of possible future outcomes during a period of time and choose the control action which would lead to the best one of them. In this work the MPC is used to control a Hybrid Diesel- Electric marine powertrain by using the electric motor to control the excaust gases quality. The experiments conducted included variable load conditions with constant reference tracking of two variants of the same controller. The results were compared with previously done work.

Contents

Chapter 1

Introduction

As fuel prices keep rising and environmental regulations regarding emissions become even stricter and enforced in constantly more areas of the world [SECAs- ECAs] (see Figure 1.1) the need for an alternative form of propulsion has risen. The solutions vary in the industry with the more widely applied until now being Dual Fuel engines, multiple-stage turbocharging, variable geometry turbochargers (VGT) as well as excaust gases' processing devices such as scrubbers, excaust gas recirculation (EGR) systems and others.

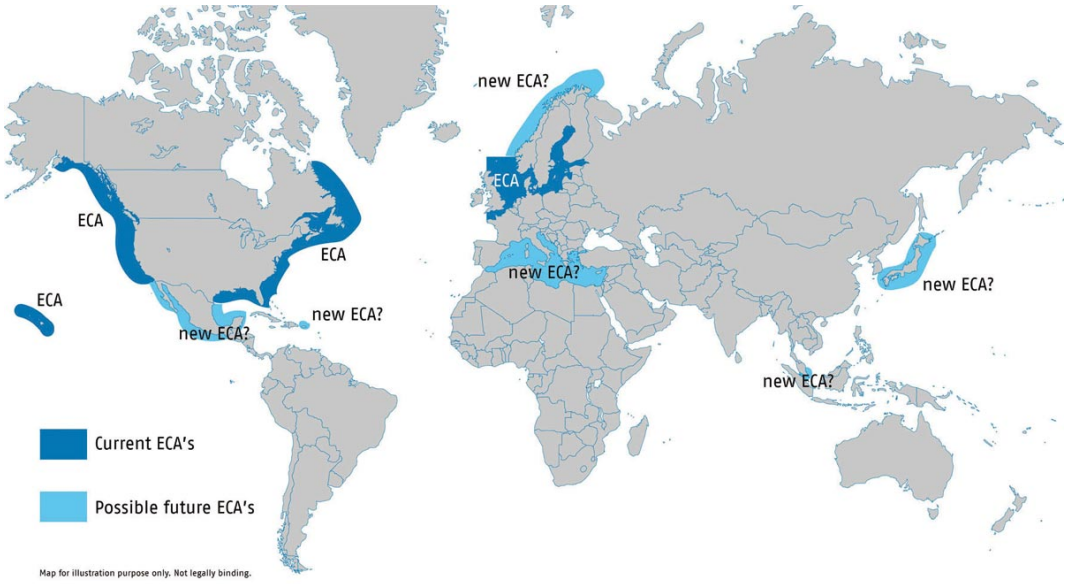


Figure 1.1: Emission Controlled Areas (ECAs) around the world.

Nowadays, coming mainly from the automotive industry, another alternative seems promising, this of the Hybrid Diesel- Electric powerplants. In this approach the diesel internal combustion engine (ICE) is assisted when needed by an electric motor. In the applications tested by the industry there are different operation modes when even the electric

motor can operate alone for limited periods of time, powered by batteries. A hybrid plant offers great flexibility, instant load taking capability, reduced visible smoke under all conditions as well as built-in redundancy, reduced maintenance due to less cylinder-hours and reduced stress to the components. The overall efficiency is increased, the system is easily upgradable, for example with an increased battery capacity as well as easily optimized and tuned for a wide range of applications. Another great advantage, is that the plant can operate without issues at low propeller loads and speeds, operation points when internal combustion engines, especially the larger ones suffer. Moreover the engine and energy storage system can supply power at the same time, thus enabling an instant power boost in the output if needed. Finally any load fluctuations are absorbed by the energy storage system, allowing stable operation of the machinery, as well providing energy recovery options during slowing-down for instance. A proposed installation of a hybrid system can be seen in Figure 1.2 and Figure 1.3.

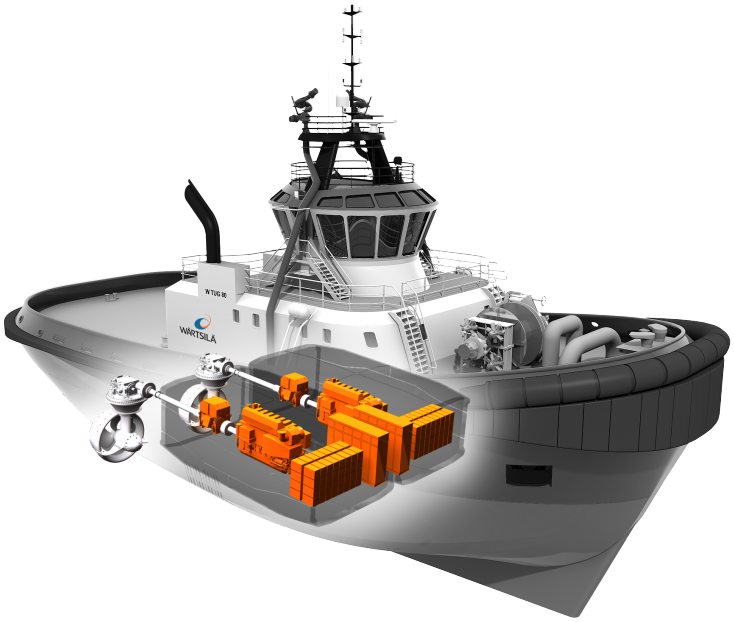


Figure 1.2: Installation of Hybrid propulsion system.

Vessels with variable operation profiles have very large potential for fuel savings as well as operation optimization regarding emissions and are therefore good candidates to install a hybrid system on. The key to the hybrid power system is that it allows the engines, dual-fuel or diesel to run at optimal load. A key element, therefore, is the control algorithms that determine how to share the load between power sources. When the vessel has peaks in power requirement, especially in bad weather the electric motor, powered by a battery pack, can respond to these loads, resulting in a much more stable load on the engines. In that way the engines become more efficient and consumption and emissions can be reduced.

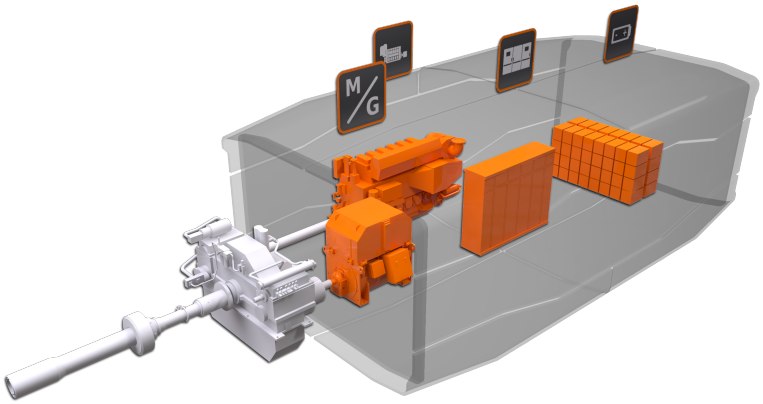


Figure 1.3: A Diesel- Electric Hybrid system.

In applications tested in practice by Wärtsilä and DNV, on the platform supply vessel "Viking Lady" the results have been promising. A reduction in fuel consumption by 15% has been measured with projection going up to 20%. Moreover, the nitrogen oxide (NOx) emissions were down by 25%. In 2015 the car ferry MF Folgefonn was retrofitted into a full scale hybrid and plug-in hybrid ferry. The ferry services the connection between the islands of Stord, Tysnes and Huglo in Norway. MF Folgefonn is now unique in terms of having all types of electrical power solutions in one vessel; it can be run as conventional diesel electric, as hybrid electric and plug-in hybrid. In hybrid operation the savings in fuel consumption in optimised mode is 10-20%. Emissions will be reduced by 30%, as a result of both the reduced fuel consumption and the improved operational profile for the combustion engines on board. In plug-in hybrid operation the fuel savings will be 20-30%, and in pure plug-in operation the potential is 100%. [20]

As mentioned above in a complex system like a hybrid one the need for a sophisticated control system arises in order to manage the power split between the different units, as well as optimize the performance in terms of various, sometimes contradicting, optimization objectives. In the current study we will focus on one of the most aggressive operational states of a diesel engine, in terms of emissions and fuel consumption, that of the transient load operation. During these load transients the engine operates far from its optimal point, stressing all its components and affecting drastically the exhaust gas quality and fuel consumption.

Based on the above content, in this work we will try to control these parameters by eventually controlling the combustion quality. As a result of the lean combustion, λ value, which represents the air-to-fuel ratio in the cylinder to stoichiometric air-to-fuel ratio drops

Figure 1.4: The Hybrid powered vessel "Viking Lady"

with a consequent rise of the pollutant emissions of the engine[1]. A hybrid diesel-electric configuration could be used in order to assist the engine operation during transient loading conditions and enhance the total performance index of the powerplant. We will treat the diesel engine as a "black box" only controlled by its speed governor, trying to keep constant engine speed. By observing the λ values (primarily) as well as other parameters (fuel consumption, NOx, exhaust gas opacity) we can use the electric motor and decrease the load of the diesel engine.

Diesel engines have relatively low engine-out emissions. In particular, hydrocarbon and carbon monoxide can usually be neglected. The main pollutant species in the exhaust gas are nitrogen oxide NOx and particulate matter (PM), see Figure 1.5. The air/fuel ratio is one primary factor that affects the formation of pollutants. Since Diesel engines are load- controlled by variations of the air/fuel ratio, this parameter plays an even more important role here than for homogeneous-charge (Spark Ignition) engines. Other important parameters are injection timing and pressure, as well as the EGR rate but we will not deal with these issues in this study as mentioned above. [17]

The system is Single Input-Multiple Output, with some of the outputs being just measured and others controlled, the single input being the control command on the electric motor. In order to cope with such a system a Model Predictive Controller (MPC) was chosen to be designed and implemented. The success of MPC in industrial applications is due to its ability to handle processes with many manipulated and controlled variables and con-

[1]
$$\lambda = \frac{FAR}{FAR_{stoich}}, \quad FAR = \frac{m_{air}}{m_{fuel}}$$

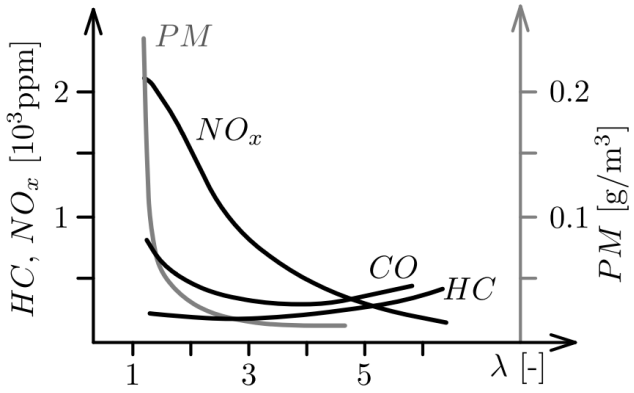


Figure 1.5: Engine-out emission of Nitrogen oxide (NOx) , hydrocarbon (HC), and particulate matter (P.M) of a direct-injection Diesel engine as a function of air/fuel ratio.

straints in a rather systematic manner. Furthermore, MPC allows an objective function to be optimized by the controller. Other advantageous MPC features are the capability of dealing with time delays, of taking advantage from future information, and of rejecting measured and unmeasured disturbances. It is noteworthy that MPC embodies both (receding horizon) optimization and feedback adjustment. Model predictive control has been applied,amongst others, to diesel engines control, catalyst control, transmission control, and Hybrid Electric Vehicles (HEV) / Plug-in Hybrid Elecric Vehicles (PHEV) power management [12].

Thesis Structure

This thesis consists of three main parts.

- Theoretical investigation of the way a Model Predictive Controller can be constructed in order to account for the internal model of the plant, constraints of various types on a number of outputs or inputs, measured or unknown or un-modelled disturbances on either the inputs or the outputs as well as state estimation. Since it is a known weakness of the MPC that its capabilities can be limited due to the computational effort required for solving the on-line optimization problem, special care has been taken so that the program is light as possible and yet fully parametric.

- Simulations were done, firstly on a known model/system from the literature in order to test the initial performance of the controller and then on the model of the HIPPO-1 experimental facility of the Laboratory of Marine Engineering / NTUA in order to validate the controller, tune it and move to the next phase of experiments

- Full scale experiments were conducted at the experimental test-bed HIPPO-1, following the simulations of the model on a variety of load patterns. The results were analysed, evaluated and compared to previous works done.

Chapter 2

Theoretical Backround

2.1 SISO Model

2.1.1 Concept

Model Predictive Controller is a model-based controller which attempts to compute the optimal sequence of the control moves in order to achieve an optimal control performance of a plant over a finite prediction horizon. As MPC can predict the future behavior of a system and plan an optimal control strategy, at the same time it can be aware of the limits and the constraints of the plant, hence reacting very differently to a disturbance which pushes the output towards the constraint compared to what it would do in response to a disturbance which pushes he plant away from it. Consequently, it is possible by adopting MPC strategy to operate a plant very close to its limits of operation[1].
The basic idea of MPC regarding a SISO plant is presented in Fig. 2.1 [7].

The current time interval is labeled as k. At the current time the measured plant output is $y(k)$; the previous history of the output trajectory is also shown. $s(k)$ denotes the set-point trajectory, which should ideally be followed by the output. The reference trajectory r defines the ideal trajectory that the output should follow in order to return to the set-point trajectory. This reference trajectory is a function of the controller tuning parameters. The current error value is defined as $e(k) = s(k) - y(k)$ and is formed over the prediction horizon H_p as

$$e(k+i|k) = s(k) - \hat{y}(k+i|k), \quad i = 1 : H_p \tag{2.1}$$

The notation $(k+i|k)$ indicates that the future value of a signal is depended on the conditions at time k. The trajectory $\hat{y}(k+i|k)$ is the controller prediction of the output value according to its internal model responding to the future sequence of the inputs $\hat{u}(k+i|k)$, $i = 1 : H_u$

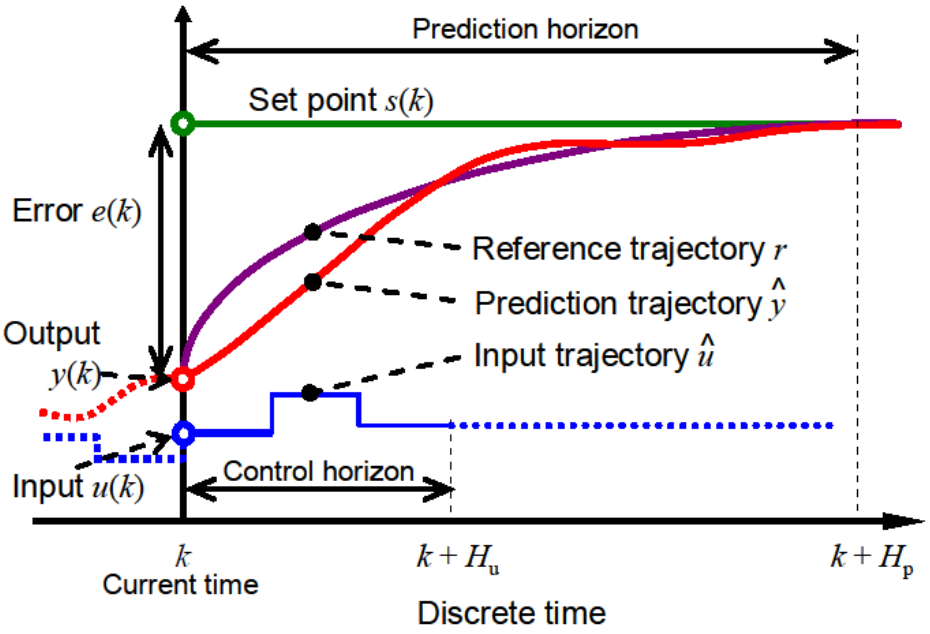


Figure 2.1: Concept of Model Predictive Controller, from [7]

of the controller, where $H_u \leq H_p$ is the control horizon and defines the acceptable control moves within the prediction horizon. The indication $\hat{u}$ means that the estimated value of $u(k+i|k)$ may be different from the actual input value $u(k+i)$ that will be applied at time interval $k+i$.

The concept of the MPC is to find the most promising control strategy of the input trajectory, so that fits as good as possible the output trajectory to its reference, according to the conditions at time k.

Once the optimal input trajectory has been selected, the first control move $u(k) = u(k|k)$ is applied to the plant, until the new measurement $y(k+i)$ of the output is available in order to devise the new optimal control strategy at time $(k+1)$ over the new horizon $i = 2 : (H_p + 1)$. This strategy, where the H_p-length horizon slides by one sample interval at each step, is called *receding horizon strategy* [1].

2.1.2 Design and Implementation

The first step into designing and implementing Model Predictive Control, is to design and control a Single Input- Single Output system of low order since it is the simplest system to control. In this case a SISO system is assumed and it is introduced in Transfer Function form, either continuous or discrete.

In the beginning the plant in transfer function form is introduced. It can have either continuous or discrete form. In the first case the discrete plant is created and then the

corresponding polynomials are extracted. In the second case the discrete transfer function polynomials are explicitly introduced without any need for further process. It is important to be consistent with the sampling interval 'ts' introduced in the beginning of the algorithm.

Next, the unit step input response needs to be calculated. In oder to do that the transfer function polynomials need to be transformed into a difference equation form. More specifically we have

$$H(z) \triangleq \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + ... + b_M z^{-M}}{1 + a_1 z^{-1} + ... + a_N z^{-N}} \tag{2.2}$$

By using the 2 properties of the z-transform, linearity and the time-shift theorem [1] we can write down the z-transform in difference equation form

$$y(n) = b_0 u(n) + b_1 u(n-1) + ... + b_M u(n-M) - a_1 y(n-1) - ... - a_N y(n-N) \tag{2.3}$$

By observing the equation we can clearly see the dependence of the order of the plant as well as the relationship between the orders of B(z) and A(z) with the response delay to new inputs. For instance for the simple case of the unit step input response, assuming initial conditions=0, for the following plant there is a lag of 2 time steps before the system reacts to the input.

$$H(z) = \frac{1}{z^2 + z + 1} = \frac{z^{-2}}{z^{-2} + z^{-1} + 1} \tag{2.4}$$

which in difference equation form becomes

$$y(n) = u(n-2) - 1 - y(n-1) - y(n-2) \tag{2.5}$$

Special care has to be taken when defining the input vector u(t). In order to reduce its dimensions we set as [M+1x1], where 'M' are the steps to the past needed to compute each new response state, u(1) is the newest input and u(M+1) is the oldest. So, with each new time step, all elements have to be transposed to one position to the past ie. $u(i) \longrightarrow u(i+1)$. For example in the case discussed above the unput vector will be:

[1] Theorem: For any $x \in \mathbb{C}^N$ and any integer Δ ,

$DFT_k[Shift_\Delta(x)] \triangleq \sum_{n=0}^{N-1} x(n-\Delta) e^{-j2\pi nk/N} = e^{-j\omega_k \Delta} X(k)$.

The shift theorem is often expressed in shorthand as

$x(n-\Delta) \longleftrightarrow e^{-j\omega_k \Delta} X(\omega_k)$.

The shift theorem suggests that a delay in the time domain corresponds to a linear phase term in the frequency domain. More specifically, a delay of Δ samples in the time waveform corresponds to the linear phase term $e^{-j\omega_k \Delta}$ multiplying the spectrum, where $\omega_k \triangleq 2\pi k/N$ .Note that spectral magnitude is unaffected by a linear phase term. That is, $\left|e^{-j\omega_k \Delta} X(k)\right| = |X(k)|$.

$$\begin{bmatrix}0\\0\\0\end{bmatrix} \xrightarrow{t=0} \begin{bmatrix}0\\0\\1\end{bmatrix} \xrightarrow{t=1} \begin{bmatrix}0\\1\\1\end{bmatrix} \xrightarrow{t=2} \begin{bmatrix}1\\1\\1\end{bmatrix} \xrightarrow{t=3} .... \xrightarrow{t=H_p} \begin{bmatrix}1\\1\\1\end{bmatrix} \tag{2.6}$$

Where H_p is the prediction horizon used for the controller.
What is essentially needed are the values of the step input response at the predetermined coincidence points, defined by the user into the vector:

$$p = \Big[p_1, p_1, ..., p_c\Big] \tag{2.7}$$

These points are then used for the matrix Θ which is later used for the computation of the optimal sequence of inputs. Matrix Θ is defined as follows

$$\Theta = \begin{bmatrix} s(p_1) & s(p_1-1) & \dots & s(1) & 0 & \dots & \dots & \dots & 0 \\ s(p_2) & s(p_2-1) & \dots & \dots & \dots & s(1) & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s(p_c) & s(p_c-1) & \dots & \dots & \dots & \dots & \dots & \dots & s(p_c-H_u+1) \end{bmatrix} \tag{2.8}$$

Where $s(p_k)$ is the unit step input response at the corresponding coincidence point p_k and H_u is the control horizon ie. the time steps within the prediction horizon in which the control input is allowed to change. As it is clear from this formulation is that the prediction horizon H_p does not in itself affect the calculation of the control inputs since it usually is $H_u < H_p$. What actually plays the most significant role is the number of the specified **coincidence points** and their position in time.

Next the prediction loop and the plant simulation are implemented. More specifically first we calculate the free response trajectory. That is the trajectory that the plant would take, if the control input remained unchanged since the last time step. The reference trajectory is also calculated. This represents the desired trajectory in order to reach the set-point trajectory. Both are also calculated at the coincidence points exclusively and are defined as follows

$$Y_f = \begin{bmatrix} \hat{y_f}(k+p_1|k) \\ \hat{y_f}(k+p_2|k) \\ \vdots \\ \hat{y_f}(k+p_c|k) \end{bmatrix}, T = \begin{bmatrix} \hat{r}(k+p_1|k) \\ \hat{r}(k+p_2|k) \\ \vdots \\ \hat{r}(k+p_c|k) \end{bmatrix} \tag{2.9}$$

where

$$\hat{r}(k+p_i|k) = s(k+i) - e^{\frac{-iT_s}{T_{ref}}}\epsilon(k) \tag{2.10}$$

The reference trajectory in this case approaches the setpoint trajectory exponentially from the current output value. The constants T_s and T_{ref} represent the sampling time and speed of response respectively and is essentially the 'time constante' of the exponential.

The predicted output would be $Y = Y_f + \Theta\Delta U$, where ΔU is the sequence of inputs applied to the plant inside the prediction horizon with regard to the last applied input $u(k|k)$. Since we want to achieve $Y = T$ but we don't have enough variables to do so exactly we solve the system in a 'least squares' sense and hence we get.

$$\Delta U = \Theta \setminus [T - Y_f] \tag{2.11}$$

It is important to note that the way we choose the coincidence points affects greatly the stability of the final solution. If the coincidence points are chosen to be at an early stage of the response of the system then the matrix Θ is **sparse**. This leads to a rank deficient matrix leading to problems regarding inverting it and finding a solution. For example, for a given system we have the following options of coincidence point vectors with their resulting Θ matrices

$$p = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \Longrightarrow \Theta = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1.3695 & 0 & 0 & 0 & 0 \\ 2.5812 & 1.3695 & 0 & 0 & 0 \end{bmatrix}$$

$$p = \begin{bmatrix} 3 \\ 5 \\ 7 \end{bmatrix} \Longrightarrow \Theta = \begin{bmatrix} 2.5812 & 1.3695 & 0 & 0 & 0 \\ 4.6017 & 3.6532 & 2.5812 & 1.3695 & 0 \\ 6.1833 & 5.4408 & 4.6017 & 3.6532 & 2.5812 \end{bmatrix} \tag{2.12}$$

In the end, the first element of the matrix ΔU is applied to the plant and the cycle repeats itself for the next time step.

2.2 The Model

In the controller designed and implemented in the current thesis, a discrete-time, linearized state-space model of the plant is assumed in the following form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= C_y x(k) \\ z(k) &= C_z x(k) \end{aligned} \tag{2.13}$$

where $\mathbf{x}(\mathbf{k})$ in the n-dimensional state vector,$\mathbf{u}(\mathbf{k})$ is the l-dimensional input vector, $\mathbf{y}(\mathbf{k})$ is the m_y-diamensional vector of the measured outputs and $\mathbf{z}(\mathbf{k})$ is the m_z-diamensional vector of the outputs to be controlled. In this work we assume that all outputs are to be controlled in some way, either directly by the control input $u(k)$ or by keeping them within certain boundaries (see Ch. 2.6). We shall then assume that $C_z = C_y = C$.

It has to be noted that the state-space model used in Eq. 2.13 is derived from the linearised continuous state-space model

$$\begin{aligned} \dot{x} &= A_c x + B_c u \\ y &= Cx \end{aligned} \tag{2.14}$$

The discretization of the LTI model presented above is performed by assuming using zero-order hold on the inputs $\mathbf{u}$ and a sample time of T_s. Then we have

$$\begin{aligned} A &= e^{A_c\ T_s} = \mathcal{L}^{-1}\{(sI - A_c)^{-1}\}_{t=T_s} \\ B &= \left(\int_{\tau=0}^{T_s} e^{A_c\ \tau} d\tau \right) B_c = A_c^{-1}(A - I)B_c, \quad Det(A_c) \neq 0 \end{aligned} \tag{2.15}$$

In order to account for any time delays of the model, a Padé approximation is used of appropriate order.

2.3 The Unconstrained Controller

The cost function to be minimized, subject to a QP or least squares solution is the following

$$V(k) = \sum_{i=H_w}^{H_p} ||\hat{z}(k+i|k) - r(k+i)||^2_{Q(i)} + \sum_{i=0}^{H_u-1} ||\Delta\hat{u}(k+i|k)||^2_{R(i)} \tag{2.16}$$

Where $\hat{z}$ **are the outputs to be controlled** and $r(k+i)$ **is the reference trajectory**. Equation 2.16 can be written in a more compact form

$$V(k) = ||Z(k) - T(k)||^2_Q + ||\Delta U(k)||^2_R \tag{2.17}$$

The weighting matrices Q, R consist of the symmetric matrices Q(i), R(i) which correspond to each time step i. However it is common practice (for simplicity reasons) that Q(i), R(i) are constant throughout the prediction cycle and hence take the form

$$\mathcal{Q} = \begin{bmatrix} Q(H_w) & 0 & \dots & 0 \\ 0 & Q(H_w+1) & \dots & 0 \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & \dots & Q(H_p) \end{bmatrix}, \quad Q(i) = Q(j),\ i,j = H_w, \dots, H_p \tag{2.18}$$

$$\mathcal{R} = \begin{bmatrix} R(0) & 0 & \dots & 0 \\ 0 & R(1) & \dots & 0 \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & \dots & R(H_u-1) \end{bmatrix}, \quad R(i) = R(j),\ i,j = 0, \dots, H_u - 1 \tag{2.19}$$

Matrix Z can also be written in an expanded form by taking into account the following properties

$$\begin{aligned} \hat{x}(k+i|k) &= A^i x(k) + \sum_{j=0}^{i-1} A^j B\hat{u}(k+(i-1-j)|k), \quad i = 1, \dots H_p \\ \hat{u}(k+i|k) &= u(k-1) + \sum_{j=0}^{i} \Delta\hat{u}(k+i|k), \quad i = 1, \dots H_u - 1 \end{aligned} \tag{2.20}$$

Matrix Z takes then the following form:

$$Z = \Psi x(k) + \Upsilon u(k-1) + \Theta\Delta U(k) \tag{2.21}$$

Where matrices Ψ, Υ, Θ can be proven that have the following form by taking into consideration Equation 2.20

$$\Psi = \begin{bmatrix} C_Z & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & C_Z & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & C_Z \end{bmatrix} \begin{bmatrix} A \\ \vdots \\ A^{H_u} \\ \vdots \\ A^{H_p} \end{bmatrix}, \ \Upsilon = \begin{bmatrix} C_Z & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & C_Z & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & C_Z \end{bmatrix} \begin{bmatrix} B \\ \vdots \\ \vdots \\ \vdots \\ \sum_{i=0}^{H_p-1} A^i B \end{bmatrix}$$

$$\Theta = \begin{bmatrix} C_Z & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & C_Z & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & C_Z \end{bmatrix} \begin{bmatrix} B & 0 & \dots & 0 \\ AB+B & B & \dots & \vdots \\ \vdots & \dots & \dots & \vdots \\ \vdots & \dots & \dots & \vdots \\ \sum_{i=0}^{H_p-1} A^i B & \dots & \dots & \sum_{i=0}^{H_p-H_u} A^i B \end{bmatrix} \tag{2.22}$$

This formulation in theory looks clear, however the elements of the matrices above are matrices themselves which complicates the algorithmic implementation. By using Equation 2.21 into Equation 2.17 we get

$$V(k) = ||\Theta \Delta U(k) - \mathcal{E}(k)||_Q^2 + ||\Delta U(k)||_R^2 \tag{2.23}$$

Where:

$$\mathcal{E}(k) = T(k) - \Psi x(k) - \Upsilon u(k-1) \tag{2.24}$$

The last element that needs to be calculated in order to be able to solve Equation 2.23 with respect to ΔU is the tracking error $\mathcal{E}(k)$ from Equation 2.24 . The only unknown is the matrix $T(k)$ that has to be specified by the user. The state vector at time 'k' and the last control input at time 'k-1' are considered to be known. Hence we have

$$T(k) = \begin{bmatrix} \hat{r}(k+H_w|k) \\ \vdots \\ \hat{r}(k+H_p|k) \end{bmatrix} \tag{2.25}$$

where

$$\hat{r}(k+i|k) = s(k+i) - e^{\frac{-iT_s}{T_{ref}}} \epsilon(k) \tag{2.26}$$

As discussed in the previous section the constants T_s and T_{ref} represent the sampling time and speed of response respectively. The set-point trajectory $s(k+i)$ is usually constant throughout the prediction horizon. However the case in which it is variable should be also investigated. The term $\epsilon(k)$ is the error at the beginning of each prediction cycle.

Another approach is to set $\hat{r}(k+i|k) = s(k+i)$ and then define the behaviour ot the reference trajectory through the weighting matrices Q, R. This is the preferred approach in the controlled designed in this thesis since it allows for more flexibility regarding the tuning of the controller. The elements $[k+i]$ of the reference trajectory vector $T(k)$ should have the corresponding dimensions of the output vector $y(k)$ which means that

$$y \to m \times 1 \Rightarrow \hat{r}(k+i|k) \to m \times 1 \tag{2.27}$$

The matrix $T(k)$ has dimensions $m(H_p - H_w + 1) \times 1$ with

$$T(k,i) = T(k, i + \lambda m), \quad \lambda = 1, \ldots, H_p - H_w, \quad i = 1, \ldots, m \tag{2.28}$$

In order to achieve the initialization the set-point vector $\hat{r}(k|k)$ at time 'k' we then project it in the prediction horizon as follows

$$T(k) = \hat{r}(k|k) \begin{bmatrix} I_m \\ I_m \\ \vdots \\ I_m \end{bmatrix} \tag{2.29}$$

It must be noted that in practice the state vector $x(k)$ may be unknown and must be therefore be estimated. This issue is discussed in a following chapter section 2.8

2.4 Matrix Formulation

In this section before we proceed with the computation of the optimum input some Matrix Properties and Linear algebra concepts used above are explained.

It can be shown that every linear least squares problem is in fact a QP problem as follows

$$\frac{1}{2}\|Qx-c\|^2 = \frac{1}{2}(Qx-c)^T(Qx-c) = \frac{1}{2}\left(x^TQ^TQx - x^TQ^Tc - c^TQx + c^Tc\right)$$

Since c^Tc is a fixed quantity it is sufficient to solve the QP problem

$$f(x) = \frac{1}{2}x^TAx + q^Tx$$

where $A = Q^TQ$ and $q = -Q^Tc$

In this case, as explained above the cost function to be minimized is

$$V(k) = ||Z(k) - T(k)||_Q^2 + ||\Delta U(k)||_R^2$$

The notation $||Z(k) - T(k)||_Q^2$ represents the quadratic form, subject to Q which can be calculated as follows

$$||Z(k) - T(k)||_Q^2 = [Z(k) - T(k)]^TQ[Z(k) - T(k)] \tag{2.30}$$

The cost function can be then written into the following form

$$V(k) = [Z(k) - T(k)]^TQ[Z(k) - T(k)] + \Delta U(k)^TR\Delta U(k) \tag{2.31}$$

We can then find matrices S_Q and S_R such that $S_Q^TS_Q = Q$ and $S_R^TS_R = R$. Then Equation 2.31 becomes

$$V(k) = [Z(k) - T(k)]^TS_Q^TS_Q[Z(k) - T(k)] + \Delta U(k)^TS_R^TS_R\Delta U(k) \tag{2.32}$$

The following known matrix properties have to be reminded.

$$||AB||^2 = (AB)^T(AB), \quad (AB)^T = B^TA^T \tag{2.33}$$

By using the properties introduced in eq.2.33 we have

$$\left\|\begin{bmatrix} S_Q[Z(k) - T(k)] \\ S_R\Delta U(k) \end{bmatrix}\right\|^2 = \left\|\begin{bmatrix} S_Q[\Theta\Delta U(k) - \mathcal{E}(k)] \\ S_R\Delta U(k) \end{bmatrix}\right\|^2 \tag{2.34}$$

The optimum solution $\Delta U(k)_{opt}$ is therefore the one that in the 'least squares' sense satisfies the equation

$$\begin{bmatrix} S_Q[\Theta\Delta U(k) - \mathcal{E}(k)] \\ S_R\Delta U(k) \end{bmatrix} = 0 \tag{2.35}$$

Which in turn leads to the solution, using Matlab notation

$$\Delta U(k)_{opt} = \begin{bmatrix} S_Q\Theta \\ S_R \end{bmatrix} \setminus \begin{bmatrix} S_Q\mathcal{E}(k) \\ 0 \end{bmatrix} \tag{2.36}$$

For the next time step we use the part of the solution above corresponding to the first step. If the number of plant inputs is $'l'$ then we just use the first $'l'$ rows of the vector $\Delta U(k)_{opt}$ which is

$$\Delta u(k)_{opt} = \Big[I_l, 0_l, \ldots, 0_l\Big]\, \Delta U(k)_{opt} \tag{2.37}$$

From the formulation above it is clear that the only variable entity is the matrix $\mathcal{E}(k)$. All others are constant and are dependant only on the dynamic characteristics of the plant. It would therefore be useful and computationally more efficient to evaluate them only once off-line. In order to solve Equation 2.34 it is required that both matrices have the same number of rows. We can therefore omit the term $\mathcal{E}(k)$ for the time being since it only affects the number of columns and then pick the first $'l'$ rows of the solution, multiply it by $\mathcal{E}(k)$ and compute the next control input. One way to implement this is by evaluating a constant 'Gain' K_{MPC} defined as follows

$$K_{MPC} = K(1:l,:), \quad K = \begin{bmatrix} S_Q\Theta \\ S_R \end{bmatrix} \setminus \begin{bmatrix} S_Q \\ 0 \end{bmatrix} \tag{2.38}$$

And

$$\Delta u(k)_{opt} = K_{MPC}\ \mathcal{E}(k) \tag{2.39}$$

All that remains is to apply the next control input to the plant which will be

$$u(k)_{opt} = \Delta u(k)_{opt} + u(k-1) \tag{2.40}$$

2.5 The Constrained Controller

As discussed in the chapter above the function to be minimised remains the same and is given by Eq. 2.34. As far as the constraints are concerned we can assume constraints on the $\Delta U(k), U(k)$ and $Z(k)$ since these have to do with actuator rates, actuator limits and plant limits respectively.

The constraints, initially of the form $x_{min} < X < x_{max}$ can take the following form

$$E \begin{bmatrix} \Delta U(k) \\ 1 \end{bmatrix} \leq 0 \tag{2.41}$$

$$F \begin{bmatrix} U(k) \\ 1 \end{bmatrix} \leq 0 \tag{2.42}$$

$$G \begin{bmatrix} Z(k) \\ 1 \end{bmatrix} \leq 0 \tag{2.43}$$

Since the cost function is minimised with respect to $\Delta U(k)$ it is convenient to express all the constrain inequalities ie. Equation 2.41, Equation 2.42 and Equation 2.43 as functions of $\Delta U(k)$. The detailed steps can be found in the literature [1] and the end result is the following

$$\begin{bmatrix} \overline{F} \\ \Gamma\Theta \\ W \end{bmatrix} \Delta U(k) \leq \begin{bmatrix} -\overline{F}_1 u(k-1) - f \\ -\Gamma[\Psi x(k) + \Upsilon u(k-1)] - g \\ w \end{bmatrix} \tag{2.44}$$

Where

$$\overline{F} = \left[\overline{F}_1, \ldots, \overline{F}_{H_u}\right], \quad \overline{F_i} = \sum_{j=1}^{H_u} F_j \tag{2.45}$$

While Γ is defined so that $G = [\Gamma, g]$ and W so that $E = [W, -w]$. As far as the optimization problem is concerned we can recall Eq. 2.34 and rewrite it in the QP problem formulation

$$\min_{\Delta U(k)} \left\| \begin{bmatrix} S_Q[Z(k) - T(k)] \\ S_R \Delta U(k) \end{bmatrix} \right\|^2 = \min_{\Delta U(k)} \left\| \begin{bmatrix} S_Q[\Theta \Delta U(k) - \mathcal{E}(k)] \\ S_R \Delta U(k) \end{bmatrix} \right\|^2 s.t\ Eq.2.44 \tag{2.46}$$

The second expression in Eq. 2.46, if expanded gives

$$\begin{aligned} &\min_{\Delta U(k)} [\Delta U(k)^T \Theta^T - \mathcal{E}(k)^T] Q [\Theta \Delta U(k) - \mathcal{E}(k)] + \Delta U(k)^T R \Delta U(k) \Rightarrow \\ &\min_{\Delta U(k)} \mathcal{E}(k)^T Q \mathcal{E}(k) - 2\Delta U(k)^T \Theta^T Q \mathcal{E}(k) + \Delta U(k)^T [\Theta^T Q \Theta + R] \Delta U(k) \end{aligned} \tag{2.47}$$

The term $\mathcal{E}(k)^T Q \mathcal{E}(k)$ is independent of $\Delta U(k)$ (see Equation 2.24) and can therefore be considered as constant, not interfering with the position of the minimum of Equation 2.47.

Table 2.1: Matrices

Matrix	**Dimensions**	**Other Comments**
U	$lH_u \times 1$	control inputs
ΔU	$lH_u \times 1$	Δ of control inputs
Z	$m(H_p - H_w + 1) \times 1$	controlled outputs
G	$lH_u \times 1$	$2\Theta^T Q\mathcal{E}(k)$
Θ	$m(H_p - H_w + 1) \times lH_u$	prediction matrix
Q	$m(H_p - H_w + 1) \times m(H_p - H_w + 1)$	weighting matrix $\rightarrow$ output
$\mathcal{E}(k)$	$m(H_p - H_w + 1) \times 1$	tracking error
H	$lH_u \times lH_u$	$\Theta^T Q\Theta + R$

Then Equation 2.47 can be brought in the following form, being consistent with the formulation of a QP problem

$$\min_{\Delta U(k)} \Delta U(k)^T H \Delta U(k) - \Delta U(k)^T G + const \; s.t \; Equation \; 2.44 \tag{2.48}$$

Here $G = 2\Theta^T Q\mathcal{E}(k)$ and $H = \Theta^T Q\Theta + R$

To sum up the matrices used so far can be seen in Table 2.1

As we can see from Tbl. 2.1 some interesting properties arise due to the special form of the matrices used. Since G and ΔU are actually vectors it holds that

$$\begin{aligned} \Delta U(k)^T G &= G^T \Delta U(k) \\ \Delta U(k)^T \Theta^T Q\mathcal{E}(k) &= \mathcal{E}(k)^T Q\Theta\Delta U(k) \; (see \; Equation \; 2.47) \end{aligned} \tag{2.49}$$

So Equation 2.47 can be re-written in the following form

$$\min_{\Delta U(k)} \Delta U(k)^T H \Delta U(k) - G^T \Delta U(k) \; s.t \; Equation \; 2.44 \tag{2.50}$$

By using a standard QP optimisation routine such as the 'quadprog' provided by Matlab we can find the minimum for each time step and compute the optimal solution. It has to be noted that the formulation above describes and solves the 'hard constrained' problem. In that case there is always the risk of the solution being 'infeasible'. In order to counteract that, various methods are proposed, as discussed in a following chapter.

2.6 Constraints

2.6.1 Hard Constraints

In order to get the final form of Inequalities 2.41, 2.42 and 2.43 we start from a more basic form witch corresponds directly to the constraints of the problem. For constraints on $\Delta U(k)$,for example, given that there are l inputs that is:

$$\begin{bmatrix} min_1 \\ min_2 \\ \vdots \\ \vdots \\ min_l \end{bmatrix} \leq \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \vdots \\ \vdots \\ \Delta u_l \end{bmatrix} \leq \begin{bmatrix} max_1 \\ max_2 \\ \vdots \\ \vdots \\ max_l \end{bmatrix} \tag{2.51}$$

In order for Equation 2.52 to make sense it is safe to assume that $min_i < 0$ and $max_i > 0$. It would also make sense that $min_i = -max_i$ but we will not take that into consideration in the case discussed here. We then work by splitting Ineq. 2.52 in its 2 parts. Hence we have

$$\begin{bmatrix} min_1 \\ min_2 \\ \vdots \\ \vdots \\ min_l \end{bmatrix} \leq \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \vdots \\ \vdots \\ \Delta u_l \end{bmatrix}, \quad \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \vdots \\ \vdots \\ \Delta u_l \end{bmatrix} \leq \begin{bmatrix} max_1 \\ max_2 \\ \vdots \\ \vdots \\ max_l \end{bmatrix} \tag{2.52}$$

By rearranging terms we get

$$E_1 = \begin{bmatrix} \frac{1}{min_1} & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & \frac{1}{min_2} & 0 & \dots & \dots & \dots & 0 \\ 0 & 0 & \frac{1}{min_3} & 0 & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & \dots & 0 & \frac{1}{min_l} \end{bmatrix} \tag{2.53}$$

And

$$E_2 = \begin{bmatrix} \frac{1}{max_1} & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & \frac{1}{max_2} & 0 & \dots & \dots & \dots & 0 \\ 0 & 0 & \frac{1}{max_3} & 0 & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & \dots & 0 & \frac{1}{max_l} \end{bmatrix} \tag{2.54}$$

We then have that

$$E = \begin{bmatrix} E_1 & \mathbf{-1} \\ E_2 & \mathbf{-1} \end{bmatrix} : \quad E \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \Delta u_3 \\ \vdots \\ \Delta u_l \\ 1 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}, \ \mathbf{-1} = \begin{bmatrix} -1 \\ \vdots \\ \vdots \\ -1 \end{bmatrix} \tag{2.55}$$

The approach presented above holds for control horizon $H_u = 1$ which is almost always not the case. In the general case we have to define vector $\Delta\hat{u}(k+i|k)$ at time step step 'i' in a similar manner to Eq. 2.25 such that:

$$\Delta\hat{u}(k+i|k) = \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \Delta u_3 \\ \vdots \\ \Delta u_l \end{bmatrix}, \quad 0 \leq i \leq H_u - 1$$
$$and\ \Delta U = \begin{bmatrix} \Delta\hat{u}(k|k) \\ \Delta\hat{u}(k+1|k) \\ \vdots \\ \Delta\hat{u}(k+Hu-1|k) \end{bmatrix} \tag{2.56}$$

For $H_u > 1$ the matrix E

$$E = \begin{bmatrix} E_1 & 0 & 0 & \dots & \dots & 0 & \mathbf{-1} \\ E_2 & 0 & 0 & \dots & \dots & \vdots & \mathbf{-1} \\ 0 & E_1 & 0 & \dots & \dots & \vdots & \vdots \\ 0 & E_2 & 0 & \dots & \ddots & \vdots & \vdots \\ \vdots & 0 & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & E_1 & \mathbf{-1} \\ 0 & \dots & \dots & \dots & 0 & E_2 & \mathbf{-1} \end{bmatrix} \tag{2.57}$$

We can then get Ineq. 2.42 as introduced above.

Now we have to work accordingly for the constraints regarding **the control input u**. The only difference is that in this case there is no 'logical' limitation to the values of the control input like in the case of Δu. We therefore have to consider all possible cases regarding the values of the constraints. So, we define the following matrices as we did in equations

2.53- 2.57.

$$F_1 = \begin{bmatrix} f_1(min_1) & 0 & \dots & \dots & \dots & 0 & k_{1,1} \\ 0 & f_1(min_2) & 0 & \dots & \dots & 0 & k_{1,2} \\ 0 & 0 & f_1(min_3) & 0 & \dots & 0 & k_{1,3} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & 0 & f_1(min_l) & k_{1,l} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_l \\ 1 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.58}$$

$$where\ f_1(min_i) = \begin{cases} 1/min_i, & if\ min_i < 0 \\ -1/min_i, & if\ min_i > 0 \Rightarrow \\ -1, & if\ min_i = 0 \end{cases}$$

$$f_1(min_i) = \begin{cases} -1/|min_i|, & if\ min_i \neq 0 \\ -1, & if\ min_i = 0 \end{cases}$$

The same logic applies also for the right part of the inequality (matrix F_2). We then have

$$f_2(max_i) = \begin{cases} -1/max_i, & if\ max_i < 0 \\ 1/max_i, & if\ max_i > 0 \\ 1, & if\ max_i = 0 \end{cases} \tag{2.59}$$

We then have that:

$$F = \begin{bmatrix} F_1 & \mathbf{k_1} \\ F_2 & \mathbf{k_2} \end{bmatrix}, \quad \mathbf{k_1} = \begin{bmatrix} k_{1,1} \\ k_{1,1} \\ \vdots \\ k_{1,l} \end{bmatrix}, \quad and\ \mathbf{k_2} = \begin{bmatrix} k_{2,1} \\ k_{2,1} \\ \vdots \\ k_{2,l} \end{bmatrix} \tag{2.60}$$

Where

$$k_{1,i} = \begin{cases} -1, & if\ min_i < 0 \\ 1, & if\ min_i > 0 \\ 0, & if\ min_i = 0 \end{cases} \quad and\ k_{2,i} = \begin{cases} 1, & if\ max_i < 0 \\ -1, & if\ max_i > 0 \\ 0, & if\ max_i = 0 \end{cases} \tag{2.61}$$

Equation 2.62 can be written in the following alternative, computationally more convenient form

$$k_{1,i} = \begin{cases} min_i/|min_i|, & if\ min_i \neq 0 \\ 0, & if\ min_i = 0 \end{cases} \quad and$$
$$k_{2,i} = \begin{cases} -max_i/|max_i|, & if\ max_i \neq 0 \\ 0, & if\ max_i = 0 \end{cases} \tag{2.62}$$

Similarly with Equation 2.57 for $H_u > 1$ we get

$$F = \begin{bmatrix} F_1 & 0 & 0 & \dots & \dots & 0 & \mathbf{k_1} \\ F_2 & 0 & 0 & \dots & \dots & \vdots & \mathbf{k_2} \\ 0 & F_1 & 0 & \dots & \dots & \vdots & \vdots \\ 0 & F_2 & 0 & \dots & \ddots & \vdots & \vdots \\ \vdots & 0 & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & F_1 & \mathbf{k_1} \\ 0 & \dots & \dots & \dots & 0 & F_2 & \mathbf{k_2} \end{bmatrix} \tag{2.63}$$

We can then quite easily get Equation 2.42.

Finally we can work accordingly for the last kind of constraints on the controlled outputs $Z(k)$. Since $Z(k)$ has as well no logical limits we could definitely know beforehand, matrix G would have the same form as F. It is worth noting that matrices E, F, G can be computed off-line and only once at the beginning of the program, contributing to the overall efficiency of the controller.

2.6.2 Soft Constraints

The hard-constrained controller, as it occurs in practice, is relatively easy to become infeasible. Thus in the literature [1], [4] it is recommended to implement a soft constrained approach, especially on the controlled output constraints since they may incorporate unmodelled dynamics or interdependent outputs/ states which may be make the system unable to satisfy all constraints at the same time. In order to achieve this we introduce the non negative slack variable ϵ_k and the corresponding tuning factors V_{min}, V_{max} and modify the constraints introduced above in the following way (For output $z(k+i|k)$ for example)

$$\begin{bmatrix} min_1 - \epsilon V_{min_1} \\ min_2 - \epsilon V_{min_2} \\ \vdots \\ \vdots \\ min_m - \epsilon V_{min_m} \end{bmatrix} \leq \begin{bmatrix} u_1(k+i|k) \\ u_2(k+i|k) \\ \vdots \\ \vdots \\ u_m(k+i|k) \end{bmatrix} \leq \begin{bmatrix} max_1 + \epsilon V_{max_1} \\ max_2 + \epsilon V_{max_2} \\ \vdots \\ \vdots \\ max_m + \epsilon V_{max_m} \end{bmatrix} \quad , \epsilon \geq 0 \tag{2.64}$$

By following the same methodology as above, we modify Equation 2.42 such that it incorporates the slack variable ϵ by introducing the following matrix

$$V_F = \begin{bmatrix} V_{F_1} \\ V_{F_2} \end{bmatrix}, \; V_{F_1} = \begin{bmatrix} V_{min_1} f_1(min_1) \\ V_{min_2} f_1(min_2) \\ \vdots \\ V_{min_l} f_1(min_l) \end{bmatrix}, \; V_{F_2} = - \begin{bmatrix} V_{max_1} f_2(max_1) \\ V_{max_2} f_2(max_2) \\ \vdots \\ V_{max_l} f_2(max_l) \end{bmatrix} \tag{2.65}$$

So for Hu=1 the inequality condition regarding the control input u becomes

$$\begin{bmatrix} F_1 & \mathbf{k_1} \\ F_2 & \mathbf{k_2} \end{bmatrix} \begin{bmatrix} u(k+i|k) \\ 1 \end{bmatrix} + \begin{bmatrix} V_{F_1} \\ V_{F_2} \end{bmatrix} \epsilon \leq 0 \tag{2.66}$$

We then, as in the previous section generalise for $H_u > 1$ and formulate accordingly in order to get the variables to-be-optimised in a manageable form.

$$FU + V_F \; \epsilon \leq 0 \quad , V_F = \begin{bmatrix} V_{F_1} \\ V_{F_2} \\ \vdots \\ V_{F_1} \\ V_{F_2} \end{bmatrix} \tag{2.67}$$

By applying the same steps to all the other constraints we eventualy reach the following inequality, similar to the one introduced in Equation 2.44.

$$\begin{bmatrix} \overline{F} & V_F \\ \Gamma\Theta & V_G \\ W & V_E \\ 0(H_u \; times) & -1 \end{bmatrix} \begin{bmatrix} \Delta U(k) \\ \epsilon \end{bmatrix} \leq \begin{bmatrix} -\overline{F}_1 u(k-1) - f \\ -\Gamma[\Psi x(k) + \Upsilon u(k-1)] - g \\ w \\ 0 \end{bmatrix} \tag{2.68}$$

The QP problem can be then accordingly formulated as follows

$$\underset{\Delta U(k),\epsilon}{min} \begin{bmatrix} \Delta U(k)^T & \epsilon \end{bmatrix} \begin{bmatrix} H & 0 \\ 0 & w_\epsilon \end{bmatrix} \begin{bmatrix} \Delta U(k) \\ \epsilon \end{bmatrix} - \begin{bmatrix} G^T & 0 \end{bmatrix} \begin{bmatrix} \Delta U(k) \\ \epsilon \end{bmatrix} \quad s.t \; Equation \; 2.68 \tag{2.69}$$

This equation gives in turn the familiar form of the penalty cost function, modified by the term regarding the constraint violations.

$$\underset{\Delta U(k),\epsilon}{min} \; \Delta U(k)^T H \Delta U(k) - G^T \Delta U(k) + w_\epsilon \epsilon^2 \quad s.t \; Equation \; 2.68 \tag{2.70}$$

Example

In this example we examine a simple case of quadratic cost function and the behaviour of the soft boundary, according to the penalising weight w_ϵ of the slack variable ϵ. The cost function examined is the following:

$$\underset{x_1,x_2}{min}\ \mathbf{x}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x} \quad ,\ s.t\ x_1 \geq 10,\ x_2 \geq 5 \tag{2.71}$$

The 3d plot of the cost function is displayed in Figure 2.2

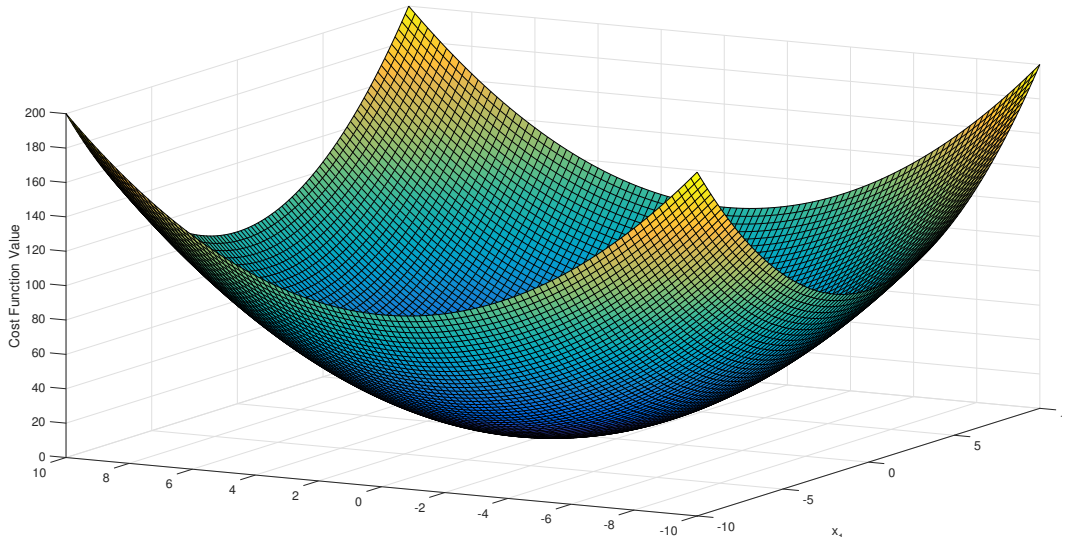


Figure 2.2: Cost Function.

In Figure 2.3 we can see the contour plot of the cost function as well as the boundaries imposed. The global minimum of the unconstrained problem is marked with the cross (+) , and the minimum of the constrained problem is marked with the circle (◯).

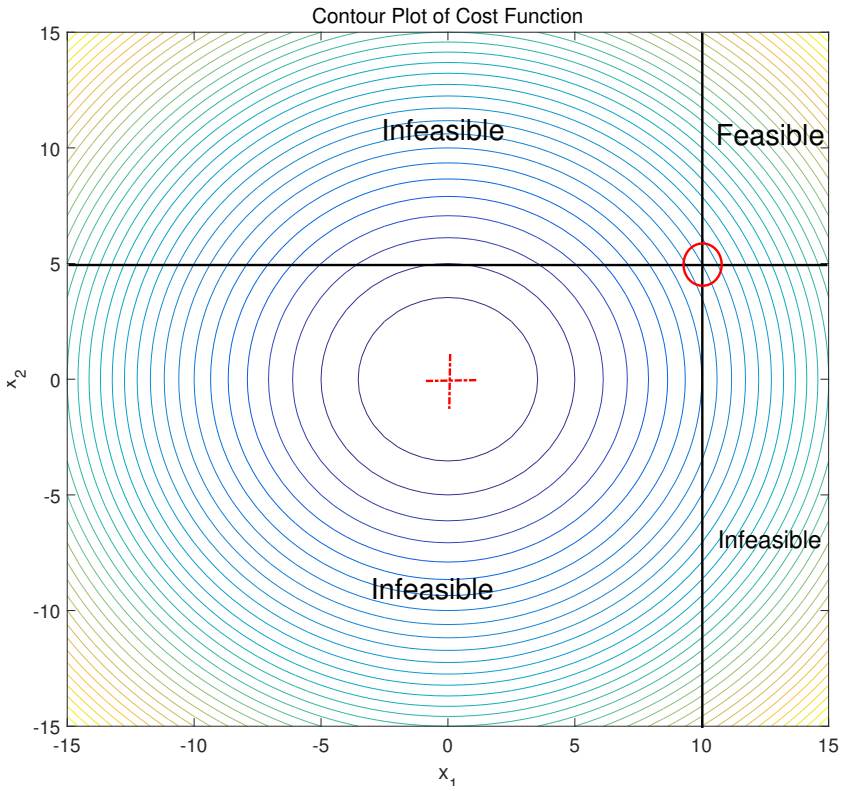


Figure 2.3: Contours of the Cost Function.

The soft constrained problem, according to the methodology introduced above would be:

$$\underset{x_1,x_2,\epsilon}{min} \begin{bmatrix} x_1 & x_2 & \epsilon \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & w_\epsilon \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \epsilon \end{bmatrix} \quad , \; s.t \; x_1 \geq 10, \; x_2 \geq 5, \; \epsilon \geq 0 \tag{2.72}$$

In Figure 2.4 one can observe the behaviour of the solution, depending on the weight w_ϵ. For $w_\epsilon = 0$ the controller is essentially unconstrained as ϵ is let free to take any value necessary in order to reach the global minimum. As the weight $w_\epsilon \to \infty$ the solution asymptotically reaches the boundary imposed and consequently $\epsilon \to 0$.

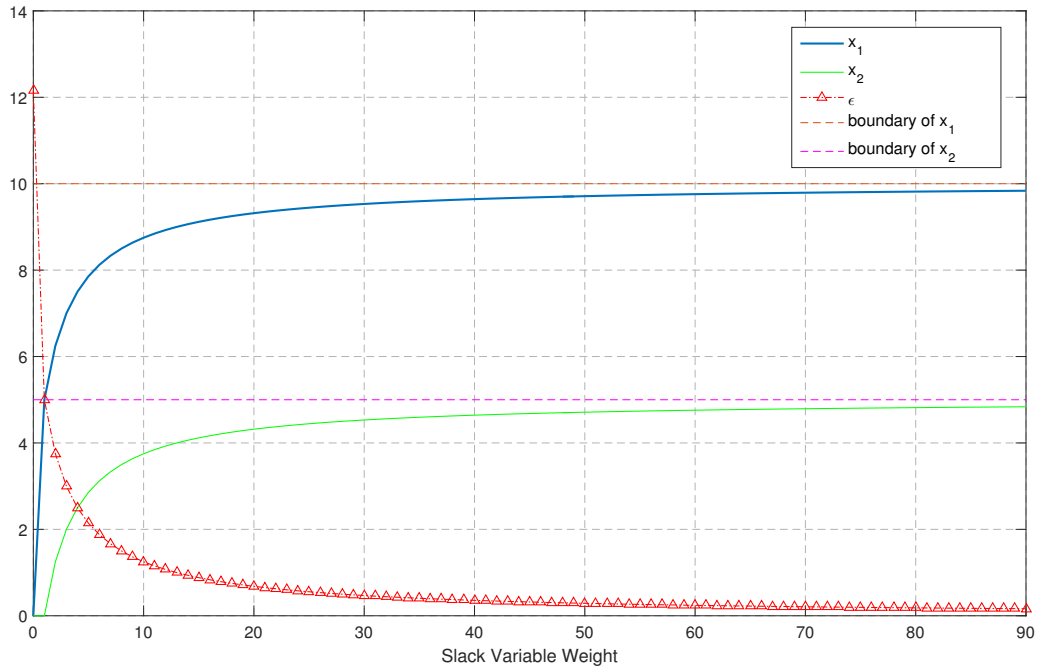


Figure 2.4: Behaviour of the solution and slack variable.

2.7 Incorporating Disturbances

2.7.1 Measured Input Disturbances

In the case where measured disturbances exist, we can consider them as inputs to the original plant, provided that we can estimate the impact they have to the plant's response. These disturbances cannot, however, be manipulated in any way and so they need to be treated differently than the ones calculated through the QP solver discussed in the chapters above. The incorporation of these disturbances in the model can result in a more accurate control and therefore a control input sequence able to compensate for their existence in the first place. We shall then modify the model as follows in order to include the disturbance vector $d_m(k)$. We can also assume that we have state estimation,hence the " ˆ " notation

$$States: \quad \hat{x}(k+i|k) = A\hat{x}(k|k) + Bu(k) + B_d d_m(k) \tag{2.73}$$

$$Measured\ Outputs: \quad \hat{y}(k|k) = C_y\hat{x}(k|k) \tag{2.74}$$

$$Controlled\ Outputs: \quad \hat{z}(k|k) = C_z\hat{x}(k|k) \tag{2.75}$$

In the formulation above we assume that the disturbances have no effect on the outputs until some time after it has been measured and for this reason $D_d = 0$.

Since the calculation of the next state vector changes we have to modify the expressions connected with the prediction and consequently with the tracking error expression introduced above in Equation 2.21 and Equation 2.24. We then have:

$$Z = \Psi x(k) + \Upsilon u(k-1) + \Theta\Delta U(k) + \Xi\ D_m(k) \tag{2.76}$$

where

$$\Xi = \begin{bmatrix} C_Z & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & C_Z & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & C_Z \end{bmatrix} \begin{bmatrix} B_d & 0 & \dots & \dots & 0 \\ AB_d & B_d & \dots & \vdots & \\ \vdots & \dots & \dots & \vdots & \\ \vdots & \dots & \dots & \vdots & \\ A^{H_p-1}B_d & A^{H_p-2}B_d & \dots & B_d & \end{bmatrix} \tag{2.77}$$

The disturbance vector $D_m(k)$ in its initial form is

$$D_m(k) = \begin{bmatrix} d_m(k) \\ \hat{d_m}(k+1|k) \\ \vdots \\ \hat{d_m}(k+H_p-1|k) \end{bmatrix} \tag{2.78}$$

Usually we cannot have beforehand knowledge of the future behaviour of the disturbances except for some specific cases. For instance if we know that the disturbances have the form of some sort of noise with known probabilistic characteristics (white noise for instance) we could then randomly evaluate some future values which, though not accurate, could be a better choice than considering the disturbances as constant throughout the prediction horizon. In the literature [1] and in the *Model Predictive Control Toolbox* [4] the second approach is adopted. However both cases should be examined in cases where it is thought to be feasible. The only remaining change required is to modify the expression for the 'tracking error' $\mathcal{E}(k)$ as follows

$$\mathcal{E}(k) = T(k) - \Psi x(k) - \Upsilon u(k-1) - \Xi\ D_m(k) \tag{2.79}$$

2.7.2 Output Disturbances

We now have to deal with a more realistic case, in which we do not have full state measurement, inaccurate plant model or disturbances acting on the output. These can all be modelled as unmeasured output disturbances. At time 'k' we cannot now what the disturbance d(k) is but we can form an estimate $\hat{d}(k|k)$ of it through the plant model that we have, allowing us to make a prediction $\hat{y}(k|k-1)$. We therefore have

$$\hat{d}(k|k) = y(k) - \hat{y}(k|k-1) = y(k) - C\hat{x}(k|k-1) \tag{2.80}$$

In order to continue we need to make the assumption that the disturbance will remain unchanged throughout the prediction horizon. For relatively small step sizes and prediction horizons this assumption can be accurate enough. We then can calculated the updated predicted outputs of the plant as follows

$$\hat{z}(k+i|k) = C\hat{x}(k+i|k) + \hat{d}(k+i|k), \quad \hat{d}(k+i|k) = \hat{d}(k|k) \tag{2.81}$$

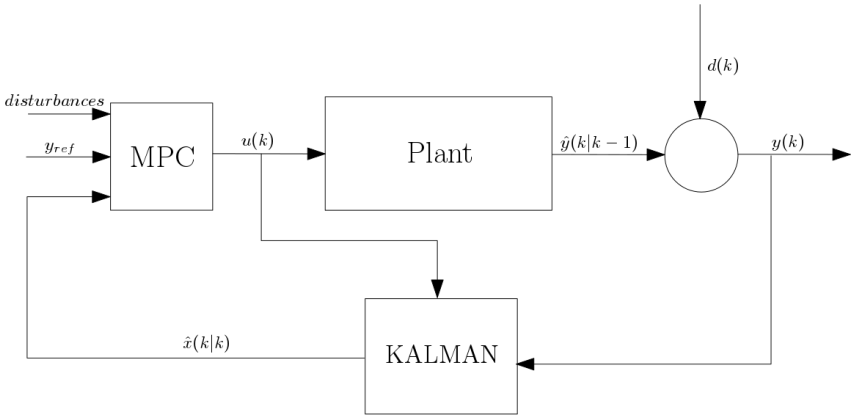
By keeping the notation introduced in Eq. 2.21 we can calculate the new output prediction matrix Z as follows, including the disturbance vector D_m introduced in Eq. 2.78

$$\begin{aligned} Z &= \Psi\hat{x}(k) + \Upsilon u(k-1) + \Theta\Delta U(k) + \overline{\Xi}\ D(k), \\ \overline{\Xi} &= [\Xi \quad \mathbf{1}], \quad D(k) = [D_m(k) \quad D_{out}(k)]^T \end{aligned} \tag{2.82}$$

We can then use the above to modify Eq. 2.79 in order to take into account the unmeasured output disturbances

$$\mathcal{E}(k) = T(k) - \Psi\hat{x}(k) - \Upsilon u(k-1) - \overline{\Xi}\ D(k) \tag{2.83}$$

We must note that since we use the actual output of the plant, we also need to know or estimate the actual state vector $\hat{x}(k)$. In order to do this we must use an observer to calculate the full state vector. In this case a Kalman Filter is implemented as shown in the figure bellow.



We can see that the Kalman Filter has as inputs the last control command $u(k)$ and the last measurement of the plant output $y(k)$. It can then calculate the full state vector $\hat{x}(k)$. The MPC Controller uses this vector as input along with the reference vector y_{ref} and the various disturbances, in the form presented above.

2.8 State Estimation- Kalman Filter

For a discrete state-space plant of the form of Eq. 2.13 the equations of the steady-state Kalman filter are given as follows.

- Measurement update:

$$\hat{x}\left[k|k\right] = \hat{x}\left[k|k-1\right] + M(y_v\left[k\right] - C\hat{x}\left[k|k-1\right])$$

- Time update:

$$\hat{x}\left[k+1|k\right] = A\hat{x}\left[k|k\right] + Bu\left[k\right]$$

In these equations:

- $\hat{x}\left[nk|k-1\right]$ is the estimate of $x\left[k\right]$, given past measurements up to $y_v\left[k-1\right]$.
- $\hat{x}\left[k|k\right]$ is the updated estimate based on the last measurement $y_v\left[k\right]$.

Given the current estimate $\hat{x}\left[n|n\right]$, the time update predicts the state value at the next sample n + 1 (one-step-ahead predictor). The measurement update then adjusts this prediction based on the new measurement $y_v\left[n+1\right]$. The correction term is a function of the innovation, that is, the discrepancy between the measured and predicted values of $y\left[n+1\right]$. This discrepancy is given by:

$$y_v\left[k+1\right] - C\hat{x}\left[k+1\,|k\right]$$

The innovation gain M is chosen to minimize the steady-state covariance of the estimation error, given the noise covariances:

$$E\left(w\left[k\right]w[k]^T\right) = Q \quad ; \quad E\left(v\left[k\right]v[k]^T\right) = R \qquad N = E\left(w\left[k\right]v[k]^T\right) = 0$$

The time and measurement update equations are combined into one state-space model, the Kalman filter:

$$\hat{x}[k+1|k] = A(I - MC)\ \hat{x}[k|k-1] + \left[\begin{array}{cc} B & AM \end{array}\right] \left[\begin{array}{c} u\left[k\right] \\ y_v\left[k\right] \end{array}\right]$$

$$\hat{y}\left[k|\,k\right] = C\left(I - MC\right)\hat{x}[k|k-1] + CMy_v[k]$$

This filter generates an optimal estimate $\hat{y}\left[k|k\right]$ of y_k. Note that the filter state is $\hat{x}\left[k|k-1\right]$.

2.9 Controller Architecture

To implement all of the above the controller has been designed in the Matlab and Simulink® environment. The first part consist of the offline computation of the prediction matrices and the second part of the implementation along with the plant model and other simulation parameters decided by the user. The Controller in simulink is shown in figure 2.5. The inputs are marked with the blue circles, whereas the output is marked with the red circle. One can observe three basic components.

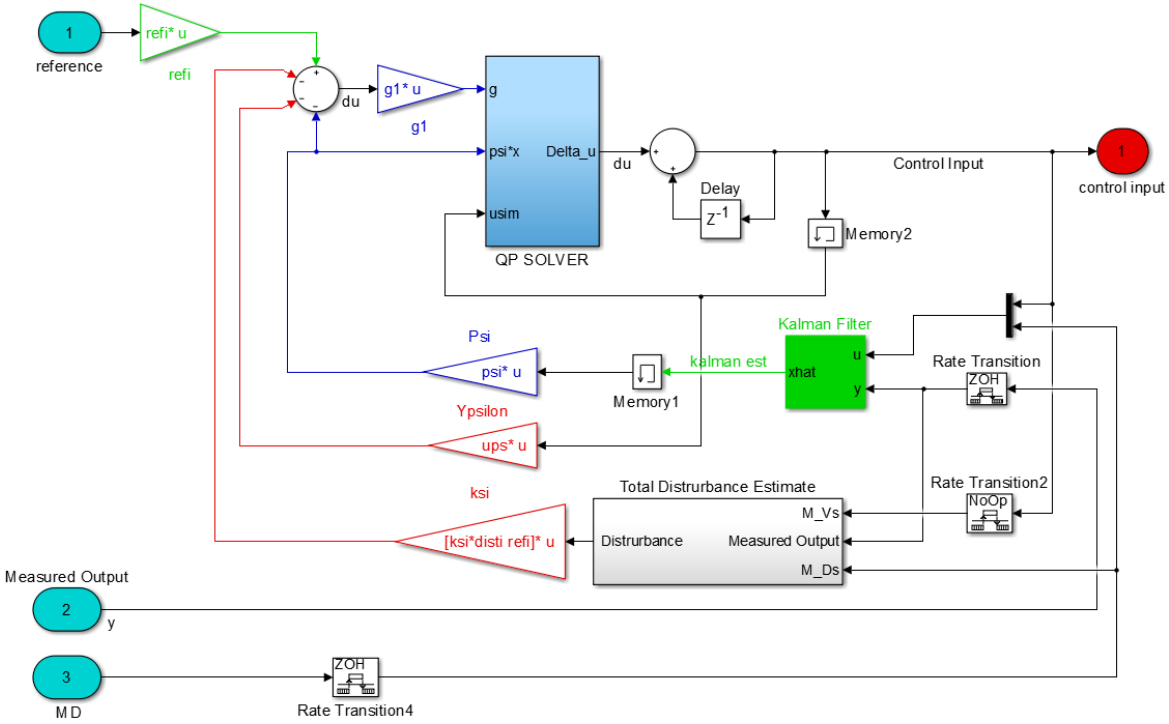


Figure 2.5: The MPC Controller in Simulink.

- The prediction

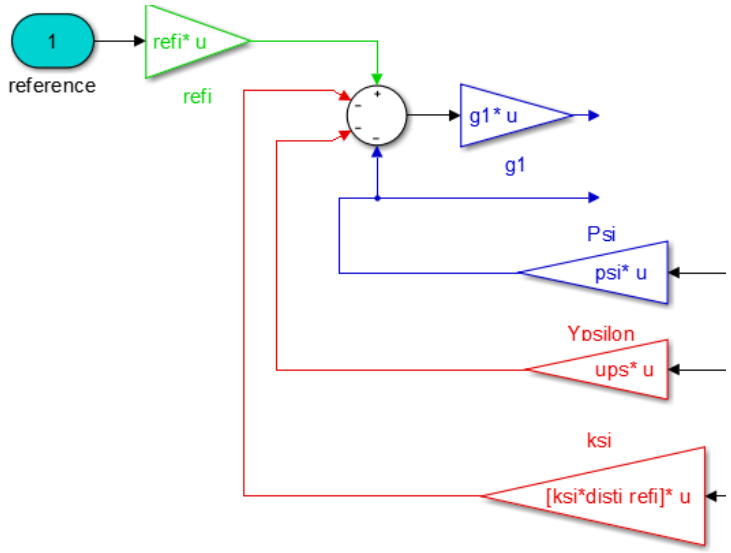


Figure 2.6: The Prediction Component.

In this part of the controller, the reference input is projected to the prediction horizon via Gain-Block "refi", which is a matrix of suitable dimensions as described in Equation 2.29. Then the similar procedure is followed for the control input *usim* as well as the state estimate *xhat* produced by the Kalman Filter. The Gain-Block "ksi" essentially implements Equation 2.77, slightly modified in order to compute at once the whole disturbance vector, comprised of the measured and estimated disturbances. Finally the tracking error is computed as well as with the second term of the cost function, as described in section 2.5 and in Equation 2.50. As discussed before, all gain matrices are computed off-line in order to simplify the on-line calculations in an effort to make the controller faster. For large simulation times, and to avoid re-running the initial setup program the tuning could be done also on-line. This, however, would be problematic for large MIMO systems which would require different tuning parameter values for each input or output. As far as the imputs of the various blocks are concerned these are either given by the user such as the reference, or calculated on-line such as the control command, the state vector and the disturbance vector. More specifically the last two have to be estimated from the measured inputs from the plant as discussed in the following paragraph.

- State and Disturbance Estimation

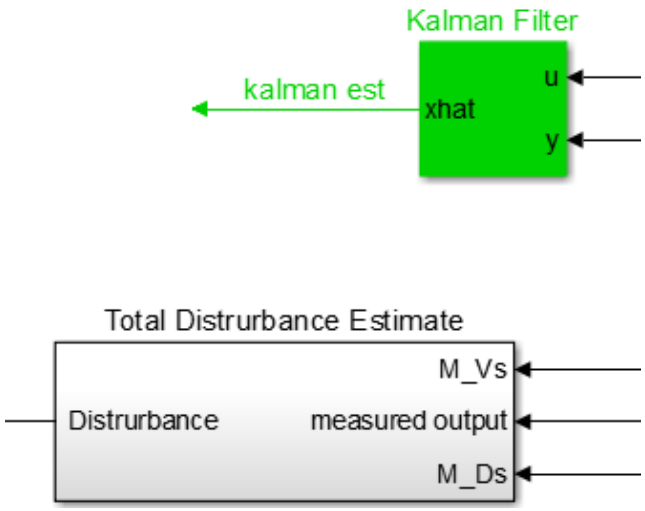


Figure 2.7: The Estimators Component.

This part of the controller is consisted of two sub-systems as illustrated in figure 2.7. The first one, illustrated in green is the Kalman Filter calculating the state estimate $xhat$ or $\hat{x}$ as was the notation used in section 2.8. The second one is the block estimating the total disturbance as is shown in the figure bellow

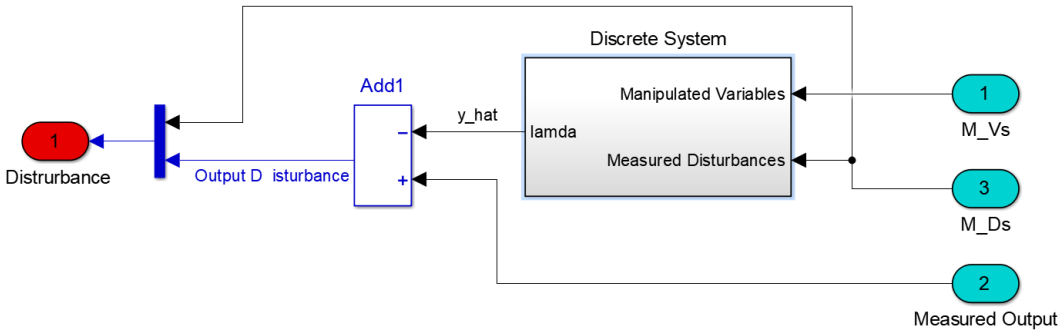


Figure 2.8: Disturbance Estimation.

In order to calculate the unmeasured disturbances, such as the disturbances on λ a discrete model of the plant is used. This can also calculate any model-plant mismatch and treat it as a disturbance, taking it into consideration in the calculation of the prediction which comes after.

- The QP Solver

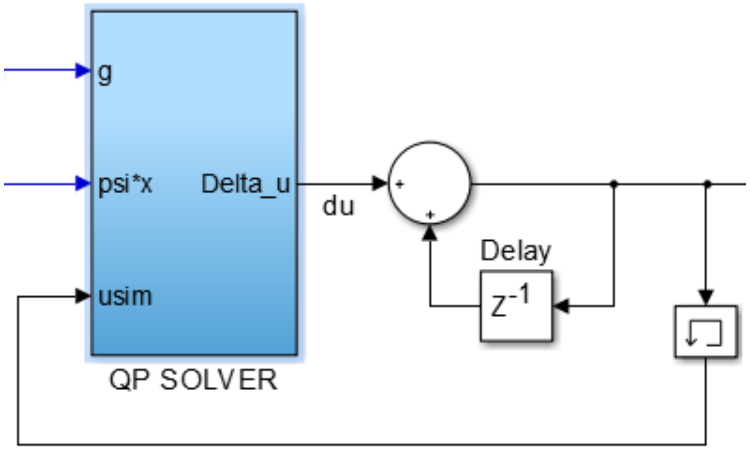


Figure 2.9: The QP Solver Component.

Finally the last component of the controller is the QP solver itself, which given all the inputs by the previous components, combined with the constraints imposed by the user calculates the optimal control command and feeds it to the plan. The essence of this component is the routine that solves the QP problem, calculating the optimal input. For this two alternatives have been used

1. Routine "quadprog"

   This routine is part of the Matlab Optimization Toolbox [8] and uses an Interior- Point-Convex algorithm to solve the Quadratic Programming problem. This routine is used in most of the simulations since it is stable and reliable. Especially in the case of the Soft- Constrained MPC, no infeasibility was recorded in any scenarios being tested for the purpose of this thesis. This routine is however not available for standalone code generation and could not therefore be compiled into $C/C\#$ which is required by the D-Space controller of the experimental testbed of LME

2. Routine "quadprog2"

   This routine is a Convex Quadratic Programming solver, written by *Michael Kleder*, featuring the freeware optimizer SOLVOPT, written by *Alexei Kuntsevich* and *Franz Kappel* and are available for free use online through the MathWorks website. This routine is exclusively written in Matlab code and after some modifications is suitable for standalone code generation into $C/C\#$ in order for the experiment to be carried out. However it has been observed that it is not as stable as the routine "quadprog"

reaching infeasibility even at specific soft-constrained cases. Nevertheless apart from the cases that the solver does not manage to yield a solution, there have not been recorded any significant differences on solution times and solution values between the two controllers.

Chapter 3

Experimental Facility

The HIPPO-1 hybrid diesel-electric power plant consists of a internal combustion engine (ICE) in parallel connection to an electric motor (EM). In this configuration the rotational speed of the ICE and the EM are identical and the supplied torque add together to meet the total torque demand applied by a hydrodynamic water brake (WB). The experimental hybrib powertrain of LME is presented in Fig. 3.1 and 3.2 .

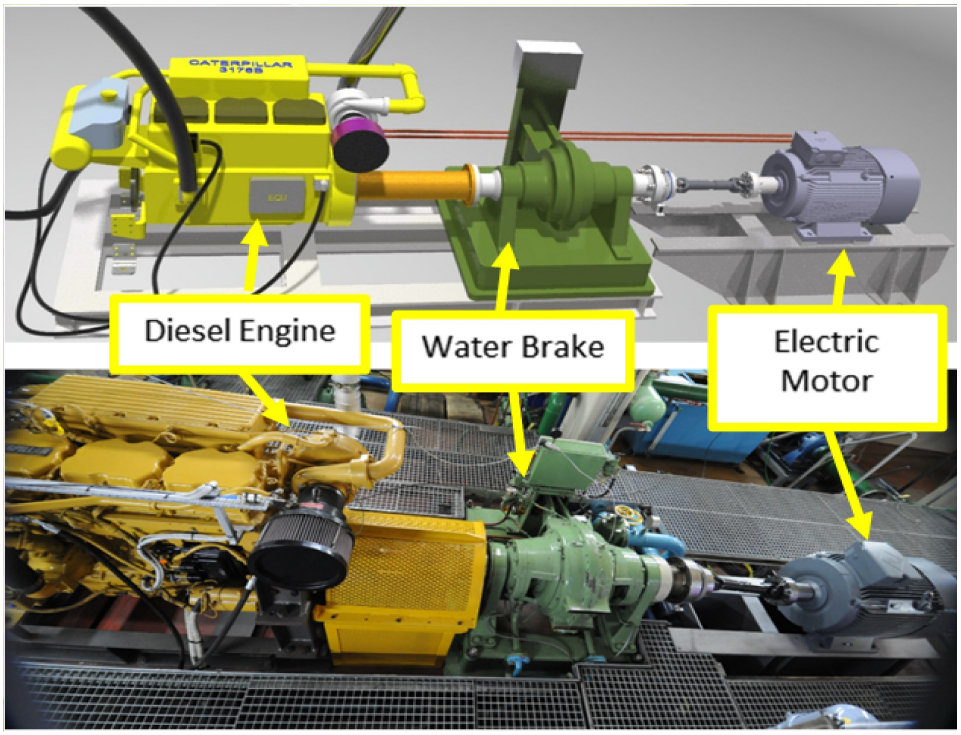


Figure 3.1: The HIPPO-1 hybrid diesel-electric testbed of LME.

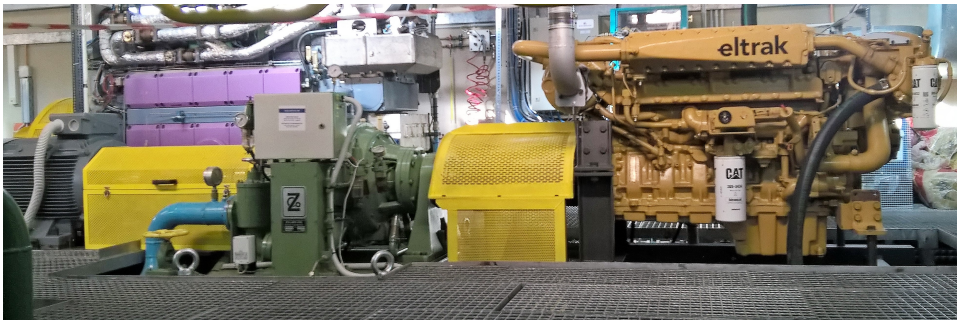


Figure 3.2: The HIPPO-1 hybrid diesel-electric testbed of LME. Between the internal combustion engine (right) and the electric motor (left) stands the water brake next to its controller board

3.1 Mechanical Componets

The ICE is a turbocharged CATERPILLAR 6-cylinder 10.3-liter 4-stroke marine diesel engine, model 3671B, producing 425 kW at 2300 rpm. According to the speed reference and the deviation of the speed measurement, the electronic control unit (ECU) of the ICE controls the fuel injection in the cylinders in closed loop control, using controller in the form of look-up tables. The installed sensors and measured variables in the diesel engine are presented in Fig. 3.3

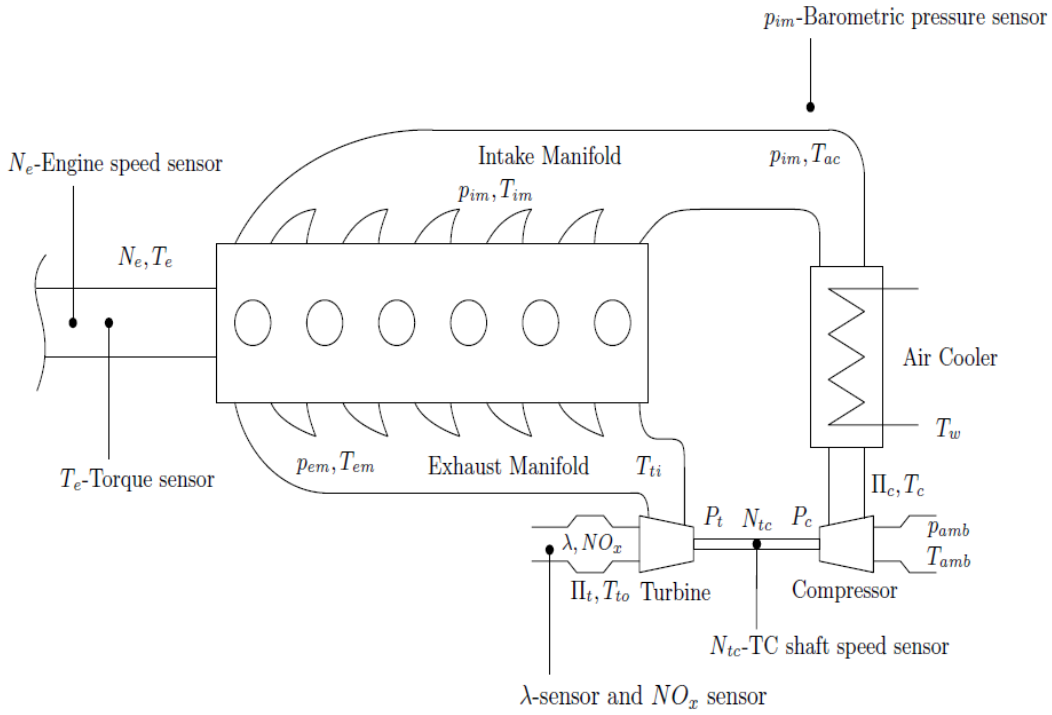


Figure 3.3: The HIPPO-1 diesel engine installed sensors and measured variables.

The EM is a standard AC asychronous-induction 3-phase motor, with a rated power of 112 kW, type IE1-K21R 315 S4, manufactured by VEM. The electric torque output is regulated by a frequency inverter (Fr Inv) under closed loop control. The electrical panel of the frequency inverter is shown in picture 3.4.

The water brake of HIPPO-1 installation is manufactured by AVL Zöllner Gmbh, type 9n 38F, with 1200 kW load capacity, operating up to 4000 rpm. The water brake consists of two parts, the stator and the rotor, which is driven by the engine shaft. Between the two WB parts, the water level is regulated in order to produce the requested torque demand. The WB is controlled by a H_∞ controller designed at LME[1].

[1]C. Gkerekos. 2015. Experimental Modeling and Robust Controller Design for the Transient Loading of a Marine Diesel Engine. *Diploma Thesis*

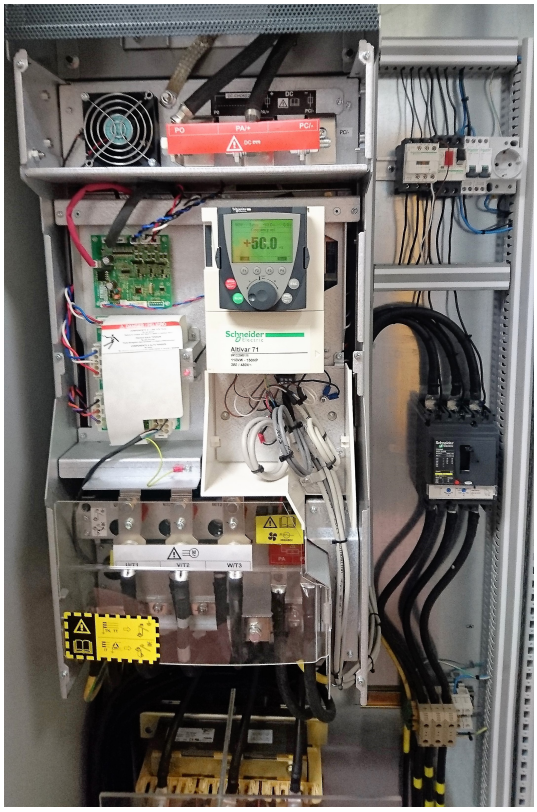


Figure 3.4: The Electric Motor frequency Inverter of HIPPO-1 at LME.

3.2 Sensors and Data Acquisition System

The installed sensors in the diesel engine are presented in Fig. 3.3. The NOx and λ values are provided by a *SmartNOx* sensor in the manifold downstream of the turbocharger (TC), manufactured by NGK. Exhaust gas opacity is measured by a AVL 439 opacimeter in the exhaust duct of the CAT engine. Fuel mass flow measurements are provided by two ABB Coriolis flow-meters, one at supply and one at return fuel lines. TC speed and intake manifold pressured are also measured.

The platform for the Data Acquisition and control of the powertrain is based on the dSpace DS1103 (Fig. 3.6) controller board, with rapid control prototyping capability, programmed under the MATLAB/Simulink environment.

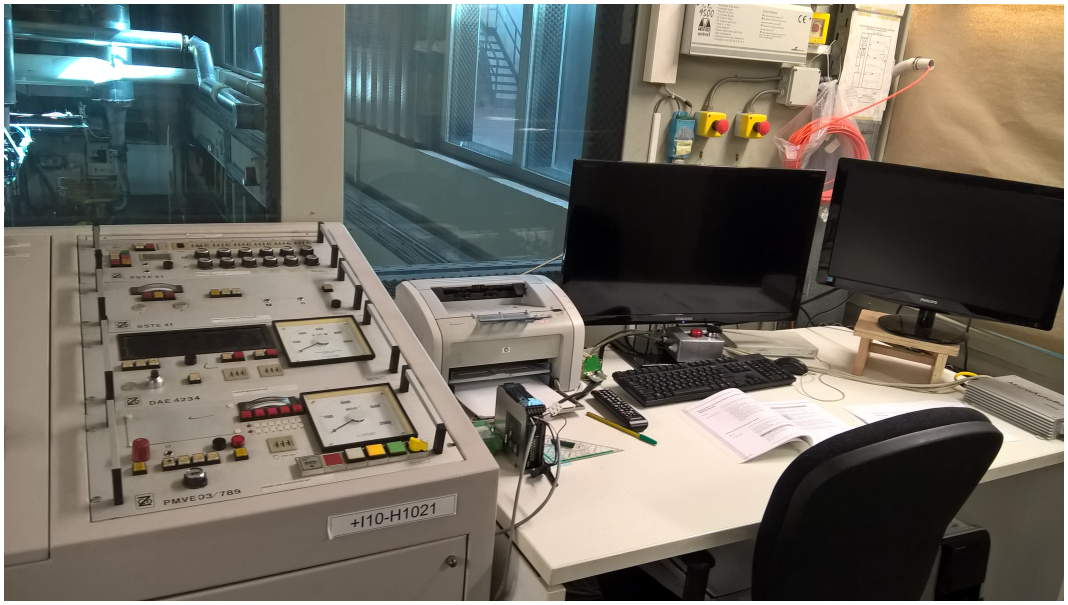


Figure 3.5: The engine control room. WB control board in front and the monitoring system of the hybrid plant on the right. Behind the safety glass, HIPPO-1 powertrain can be distinguished

A picture of the powertrain monitoring screen in the control room (Fig. 3.5) at LME is shown in Fig. 3.7, where all the utilities of the monitoring and the control board are presented.

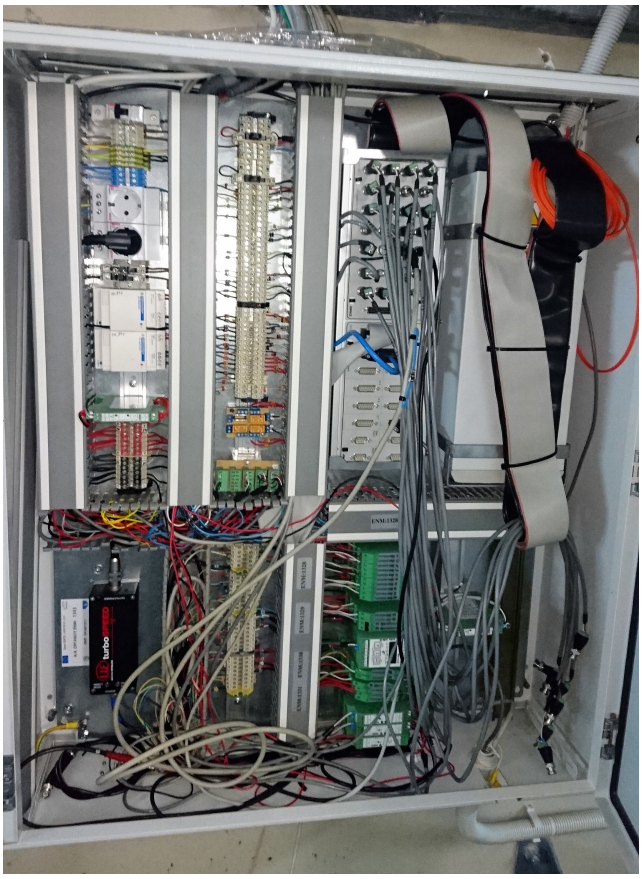


Figure 3.6: The HIPPO-1 dSpace monitoring and control board.

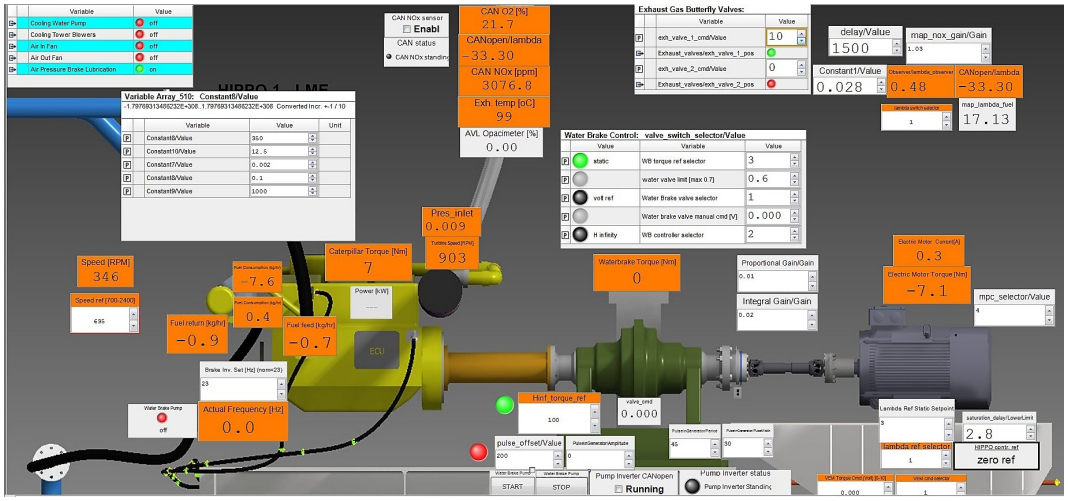


Figure 3.7: The HIPPO-1 monitoring and control screen.

Chapter 4

Simulation results

4.1 An example from Literature

In this section we will present an example from literature [1] in which the MPC controller designed above is implemented in a state-space model, describing the dynamics of a Cessna Citation aircraft.

Figure 4.1: Cessna Citation Aircraft.

The continuous State-Space model of the plant as defined in Equation 2.14 consists of

the following matrices A, B, C and D

$$A = \begin{bmatrix} -1.2992 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.4293 & 0 & -1.8366 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.1}$$

The system has as an input the elevator angle and as outputs the altitude of the aircraft as well as the pitch angle and altitude rate. In this case we will experiment with different tuning parameters and controller designs, starting by using the unconstrained controller, then the hard constrained and finally the soft constrained approach.

4.1.1 Unconstrained MPC

In this case we will use the controllers described in section 2.3.Obviously this is not a realistic case since constraints must be imposed in at least the control input, as the actuators have their own physical limits. We will use this however to determine at a first glance the performance of the controller.

The tuning parameters used in this case are:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = 10^7, \quad H_u = 1, \quad H_p = 30$$

As we can see in Figure 4.2 and Figure 4.3, displaying the results of the simulation, the altitude is one order of magnitude greater than the other outputs. This fact has to be taken into account when choosing the proper weights. Another approach would be to scale all outputs and inputs down to the same magnitude by using scale factors, as it is proposed in [4].

As we can see, the controller manages to control successfully the plant in both increasing and decreasing the altitude set-point Moreover we have to note that the altitude tracking error dominates the other 2 outputs, the pitch angle and the altitude rate errors, during most of the transient, so that the pitch angle and altitude rate depart from their set-points in order to allow the altitude error to be reduced. As the required altitude os acquired, all three outputs settle rapidly to their set-points This behaviour is entirely a result of the numerical values of the errors which arise, as mentioned above.

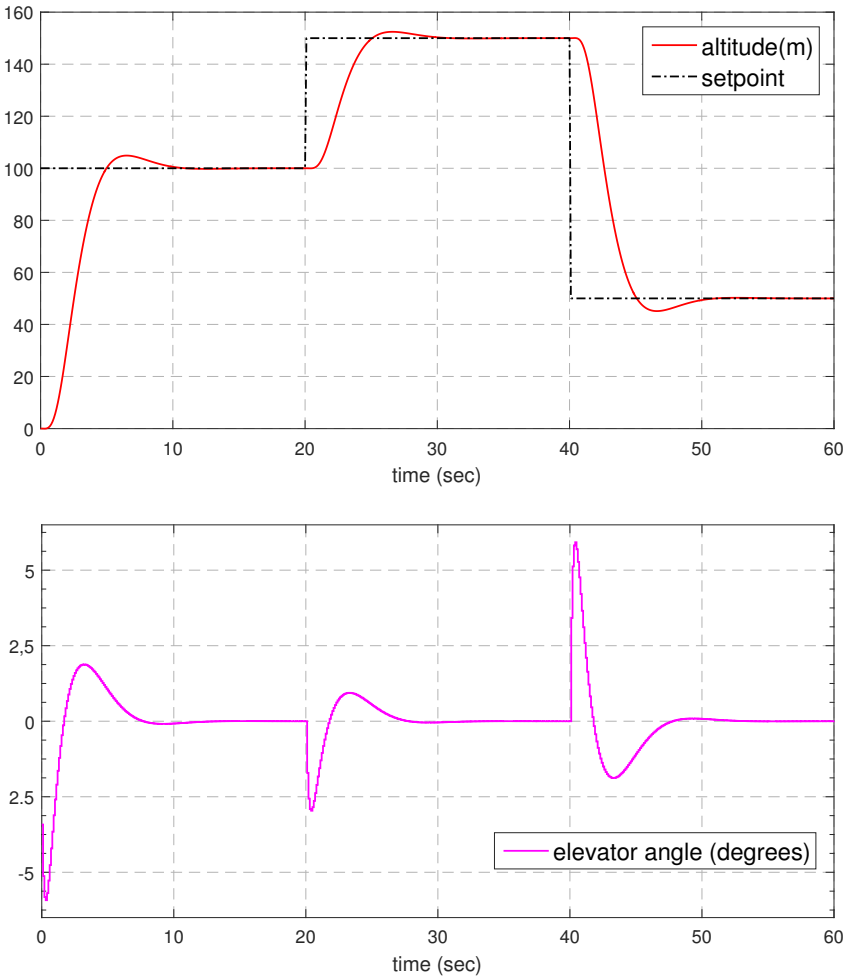


Figure 4.2: Results of the Unconstrained MPC Controller (altitude and control input).

If the primary output to-be-controlled was another one, then proper scaling and/or modification of the weighting matrices would be the only way to get around this problem. We can also see that the pitch angle takes relatively large values, which is the reason why constraints need to be imposed as we will see in the next section.

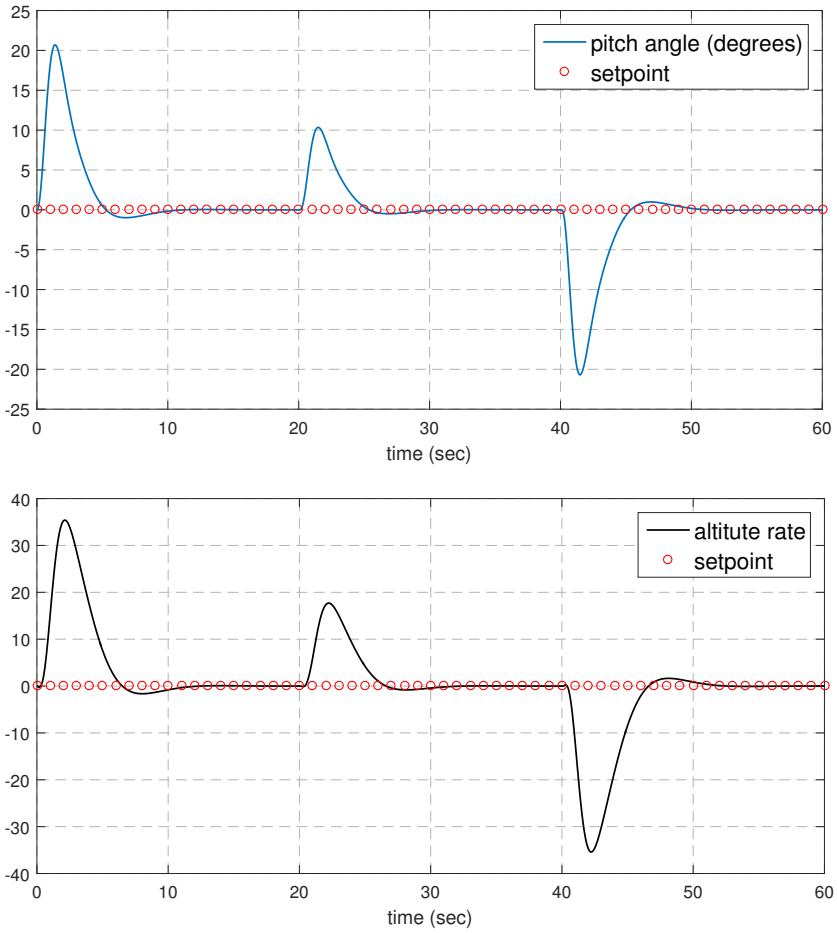


Figure 4.3: Results of the Unconstrained MPC Controller (pitch angle and rate).

4.1.2 Hard- Constrained MPC

In this case we will impose hard constraints, examine 2 cases with different control horizons H_u and observe how this difference in the tuning of the controller affects the solution. For both cases the weights were:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = 10^9$$

The boundaries imposed where:

$$\begin{bmatrix} -0.01 \\ -5^o \\ -6^o \\ -\infty \\ -15\ m/s \end{bmatrix} \leq \begin{bmatrix} \Delta u \\ u \\ pitch\ angle \\ altitude \\ altitude\ rate \end{bmatrix} \leq \begin{bmatrix} 0.01 \\ 5^o \\ 6^o \\ \infty \\ 15\ m/s \end{bmatrix}$$

There has to be noted that there is no reason to impose boundaries on the altitude since it is the primary controlled output and it is bound by the setpoint. Boundaries would be needed in case of large oscillations around the setpoint trajectory or in the case of large overshoot. This however can also be controlled through the weights Q, R. Now the results of the 2 cases mentioned above are presented in Figure 4.4 and Figure 4.5.

- **Case 1**: $H_p = 30, \quad H_u = 1$

  In this case the results of the simulation are presented in Figure 4.4 and Figure 4.5

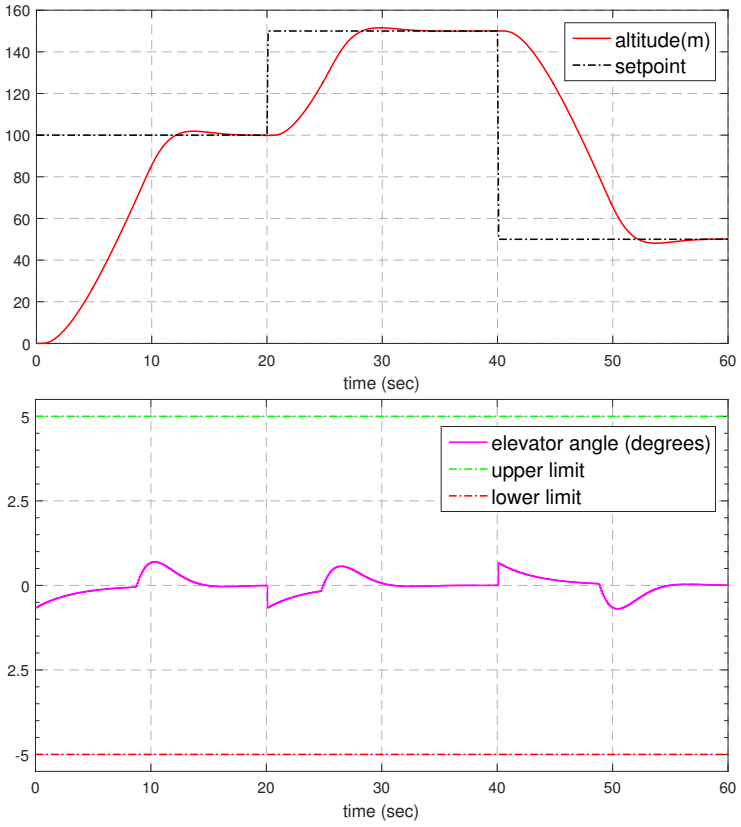


Figure 4.4: Results of the Constrained Controller, Case 1 (altitude and control input).

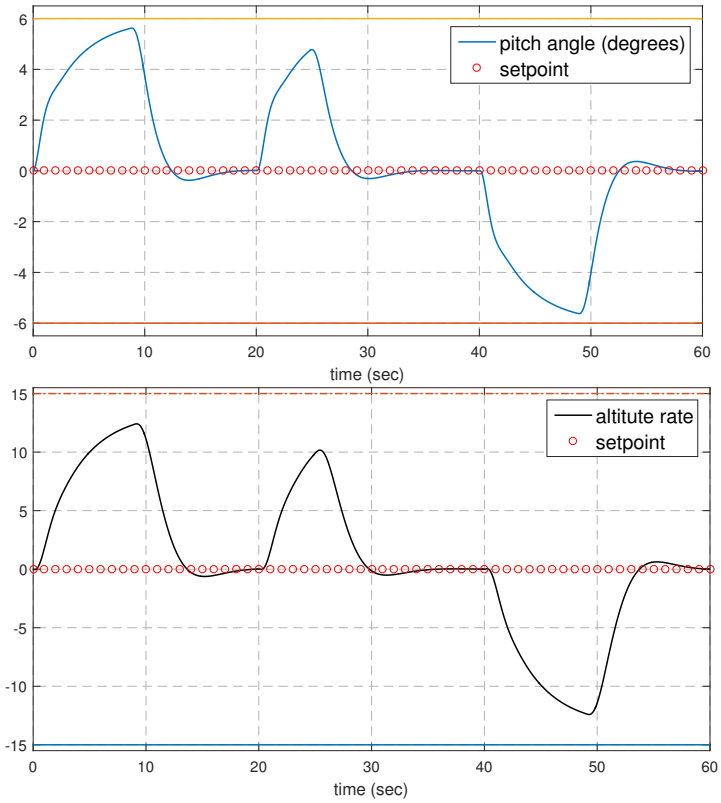


Figure 4.5: Results of the Constrained Controller, Case 1 (pitch angle and rate).

We can see that the controller respects the bounds imposed by the user on all constraints. We can also observe that due to the control horizon set to $H_u = 1$ the controller is somewhat conservative and thus approaches the boundary but never reaches it since it does not have much freedom in choosing the control input sequence. So as the boundaries tend to be violated the controller takes proactive action in order to prevent that. Due to the constraints we can also see that the plant reaches is designated set-point, as far as the altitude is concerned, much slower in comparison to the previous case (see subsection 4.1.1).

In Figure 4.6 we can see the behaviour of Δu with respect to its boundary. As we can see in the constraints regarding Δu is also respected.

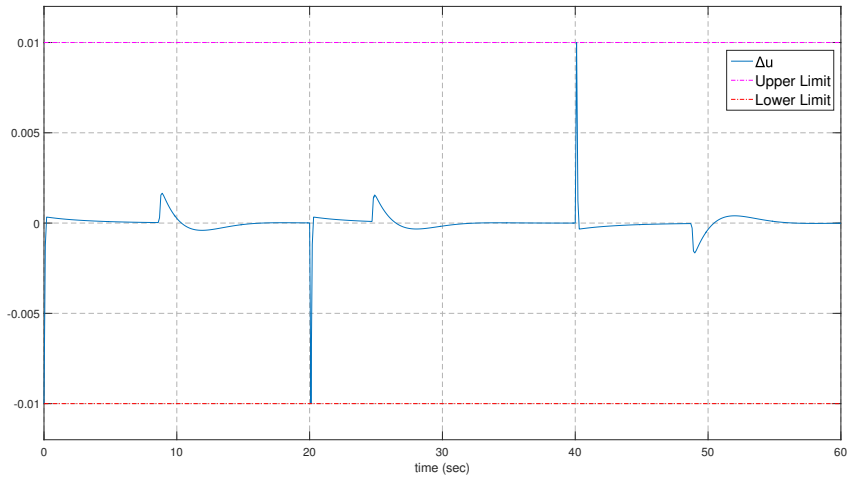


Figure 4.6: Δu, Case 1.

- **Case 2**: $H_p = 30, \quad H_u = 3$

  In this case we will investigate a more computationally demanding case, in which the control horizon is $H_u = 3$. As we will see at the end of this chapter the QP solver requires significantly more time to compute each control input at each time step. The results are presented in Figure 4.7 and Figure 4.8.

  As it is obvious from Figure 4.7 and Figure 4.8, the plant reaches its boundaries, since the controller has more freedom and can be more aggressive. The plant response is also faster as far as the altitude is concerned, since the pitch angle reaches its maximum value for large periods of time. We can also observe that the control input sequence is more complicated as a result of the larger freedom of the controller, due to the longer prediction horizon. As soon as a constraint tends to be violated, the controller takes immediate action in order to prevent the violation. This allows the controller to stay closer to the constraints and results in a better overall performance.

  The behaviour of Δu is presented in Figure 4.9

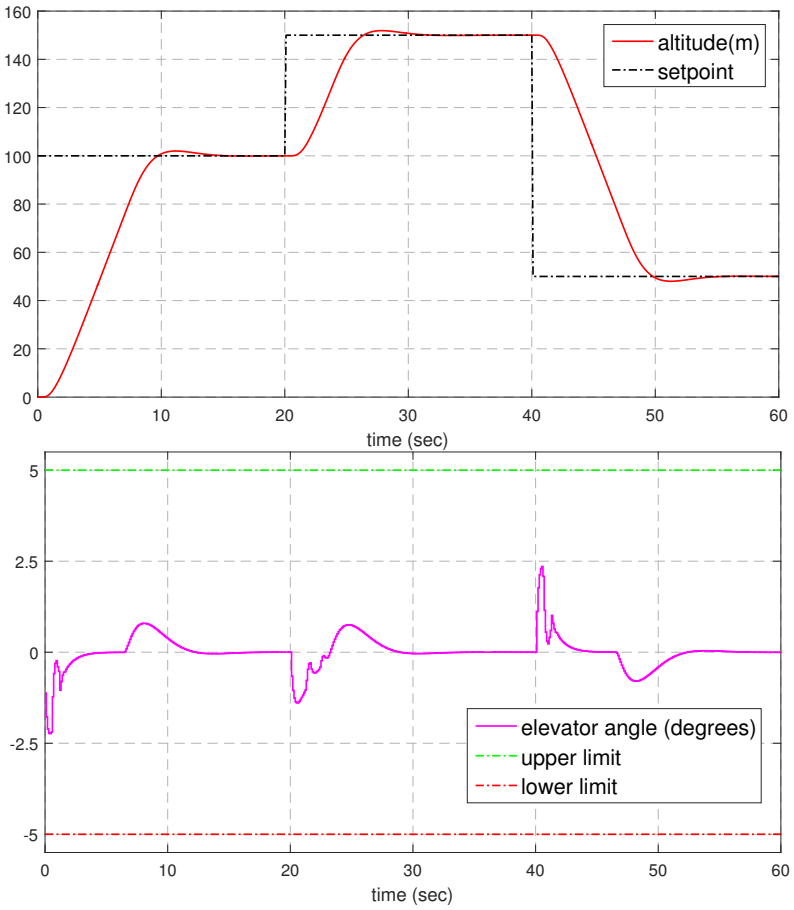


Figure 4.7: Results of the Constrained Controller, Case 2 (altitude and control input).

In this case we can see that the constraints are respected as well. Moreover, being consistent with the behaviour of the control input, this also is more aggressive than in Case 1

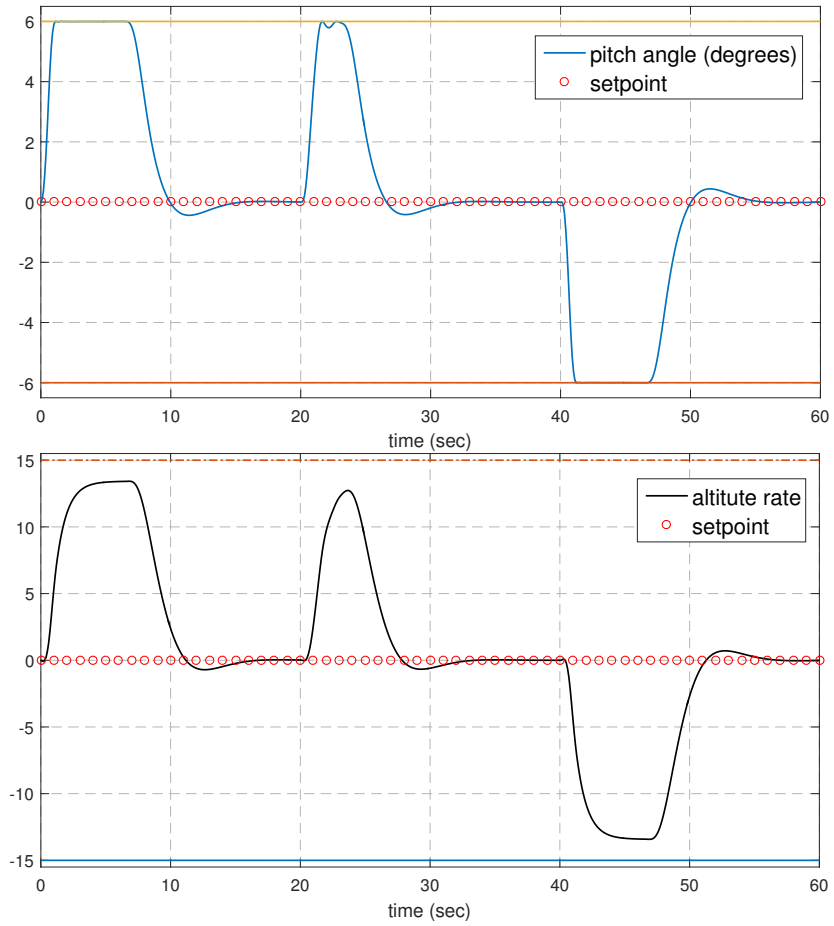


Figure 4.8: Results of the Constrained Controller, Case 2 (pitch angle and rate).

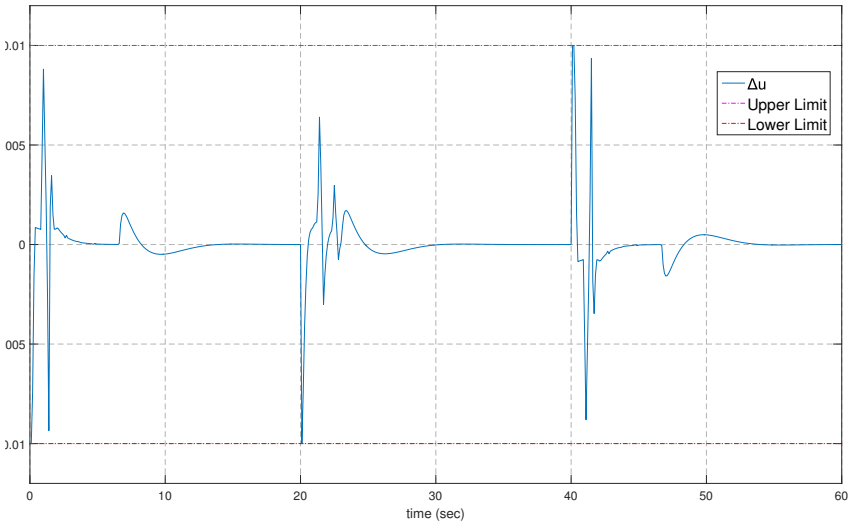


Figure 4.9: Δu, Case 2.

4.1.3 Soft - Constrained MPC

In this section we will discuss the implementation of the Soft-Constrained MPC in the plant described by Equation 4.1. As it is obvious from the figures in subsection 4.1.2 the pitch angle reaches quite easily its constraints. We could therefore relax them in order to allow the controller to slightly violate them for a brief period of time. By this we ensure that the QP problem is always feasible which in turn makes the controller reliable under any circumstances.

The additional tuning parameters in this case are the weight on the slack variable w_ϵ and the tuning factors V as described in Equation 2.64 and Equation 2.70. These were chosen so that we have slight violations, ensure feasibility in any case but keep the plant close to its boundaries at all times. The values of these tuning parameters were

$$w_\epsilon = 3\ 10^5, \quad V = \begin{bmatrix} V_f \\ V_e \\ V_g \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ [0.01;\ 0;\ 1]^T \end{bmatrix}$$

This means that the constraints on u and Δu are hard and that the constraints on the outputs are soft. More specifically the constraint on the pitch angle is set to be harder than the one on the altitude rate. However the violation on the pitch angle is larger than the one on the altitude rate because the first reaches the boundary and tends to cross it as it is demonstrated in Figure 4.7 and in Figure 4.8:

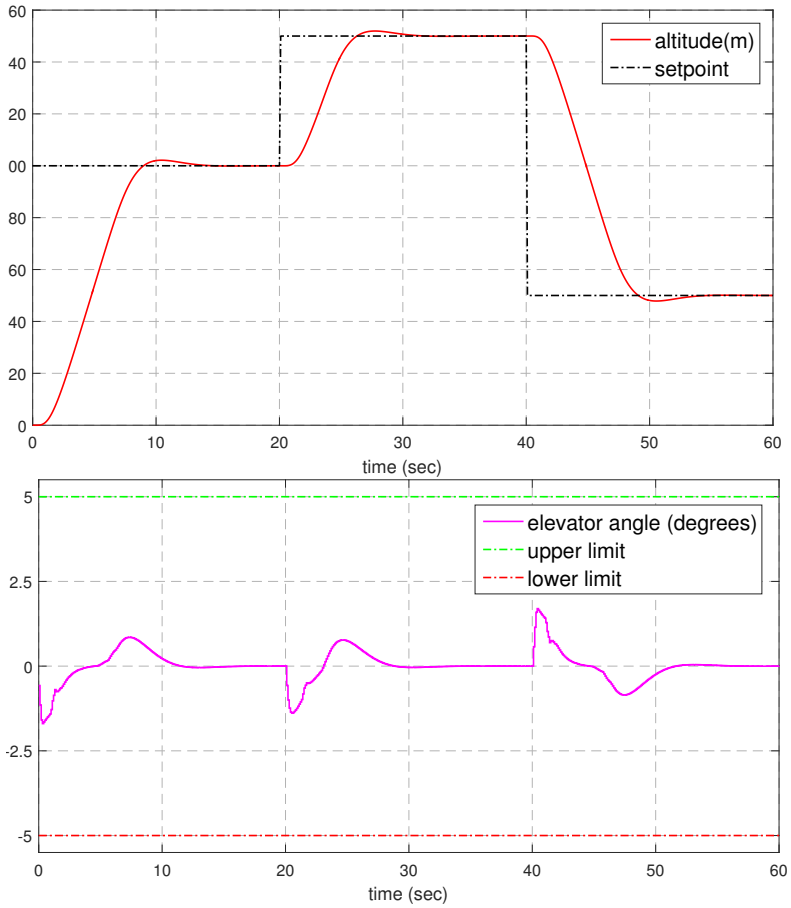


Figure 4.10: Results of the Soft - Constrained Controller (altitude and control input).

As we can see the plant is again controlled successfully, and the boundaries are being slightly violated as expected. The controller is also less aggressive than in Case , subsection 4.1.2 and the response is also faster as the aeroplane can reach its altitude at a greater rate.

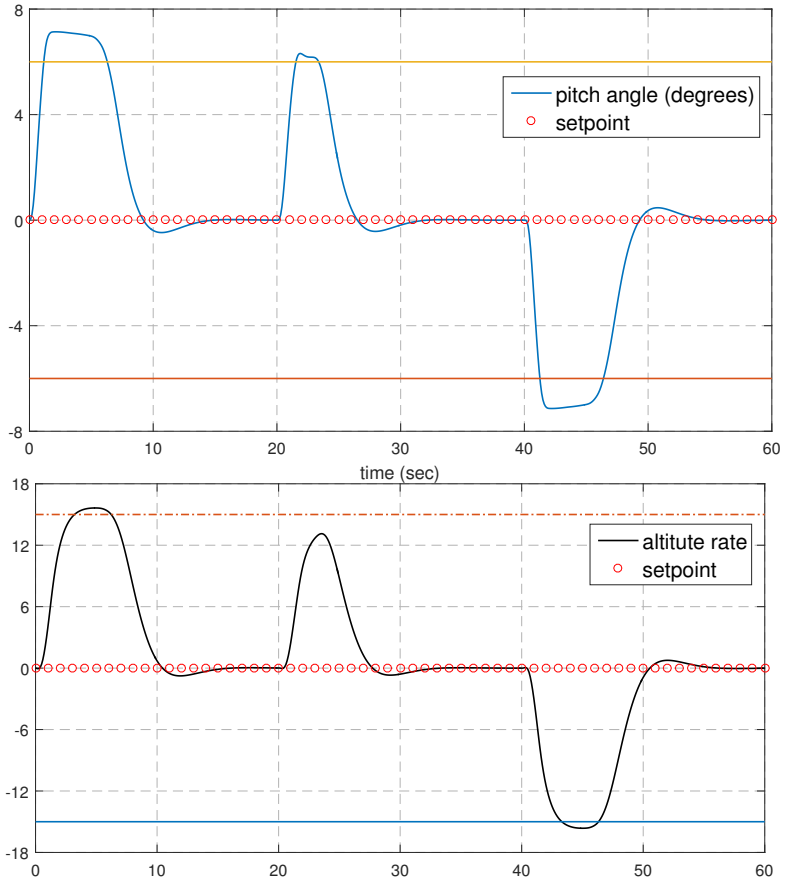


Figure 4.11: Results of the Soft - Constrained Controller (pitch angle and rate).

In this case it is also useful to observe the values of the slack variable ϵ in Figure 4.12 along with Δu as in the previous cases.

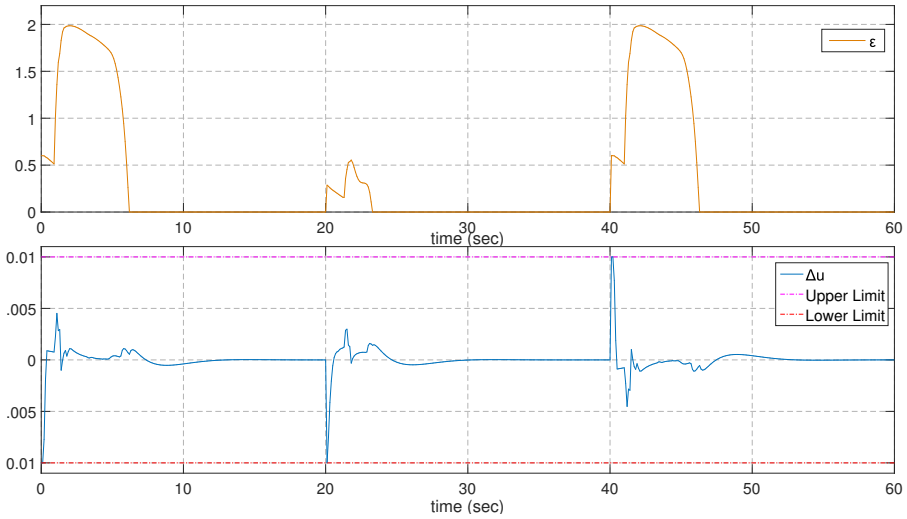


Figure 4.12: Results of the Soft - Constrained Controller for parameters ϵ and Δu

As expected the slack variable takes non-zero values during the periods of time when violations occur, with value proportional to the violation. We have to note that the slack variable is 'chosen' by the controller so that the QP problem is always feasible.

4.1.4 Calculating Performance

In this final section we will discuss briefly the computational speed of the controller. It is of vital importance for the controller to be able to estimate the next control input faster than the pre-set time step. If this is not the case, then controller does not have enough time for the control input computation and then the system will fail.

The results presented bellow correspond to a system with the following characteristics and during the same conditions regarding the general load of the system

- CPU: Intel Core i5-7200U @ 2.50 GHz, 2701 Mhz, 2-core, 4 logical processors
- RAM: 8GB DDR4, 2400 MHz
- OS: Windows 10, x64

Table 4.1: Speed Comparison

Case	Prediction Horizon H_p	Control Horizon H_u	Time/loop
Unconstrained	-	-	2.21 10^{-5} sec
Constrained (Hard)	30	1	0.0083 sec
	30	3	0.0236 sec
Constrained (Soft)	30	3	0.0218 sec

As we can see, as the control horizon increases, so does the computation time required for each control input. There is also considerable difference between the unconstrained and the constrained problem. That is because in the unconstrained case the program has to solve a simple linear algebra problem, in the constrained case, the solution of the QP problem is added. Finally we can see that the soft constrained problem is solved a little faster than the respective hard constrained. That is because in the soft constrained problem it is easier and consequently faster to find a solution that satisfies the constraint since this lies in a broader field of feasible solutions.

4.2 The Hybrid Integrated Propulsion Power-plant (HIPPO) Simulation

In this Section we will discuss the results of the simulations at the experimental facility of the Laboratory of Marine Engineering (NTUA) on the test-bed setup HIPPO 1 as presented in chapter 3

In this case we used a Multiple Input-Single Output (MI-SO) model (model: m73) with measured disturbances as inputs, unmodelled disturbances on the outputs as well as output delays, incorporated in the continuous-time State-Space model. The model was derived with system identification methods and experiments presented in previous work [11]. The continuous state-space plant is the following (Equation 4.2)

$$a = \begin{bmatrix} & x_1 & x_2 \\ x_1 & 0.07556 & -1.739 \\ x_2 & 3.093 & -1.946 \end{bmatrix}, \quad b = \begin{bmatrix} & FrInvCmd & ErrorEngSpeed \\ x_1 & 0.3254 & 0.001003 \\ x_2 & -0.7647 & 0.0005494 \end{bmatrix}$$

$$c = \begin{bmatrix} & x_1 & x_2 \\ \lambda & 30 & 1.166 \end{bmatrix}, \quad d = \begin{bmatrix} & FrInvCmd & ErrorEngSpeed \\ \lambda & 0 & 0 \end{bmatrix} \tag{4.2}$$

$$Output\ Delays\ = 0.8\ sec$$

In this model the '$FrInvCmd$' is the command in the *Frequency Inverter* of the electric motor and is the **manipulated variable** of the plant whereas '$ErrorEngSpeed$' is the $EngineSpeed - SpeedSetpoint$ error and is considered to be the **Measured Disturbance** of the plant, acting as an input as well.

The fuel injection of the ICE in the hybrid powertrain is independently controlled by the ECU. The ECU uses the speed deviation from the speed reference, as described by Eq. (4.3) in order to inject more or less fuel in the cylinders in order to maintain the engine speed, accelerate or slow down, according to the speed reference (SE_{Ref}). So the dSE measurement could be a precursor of the λ value change, which could help a model based controller predict a change in the output trajectory[1] and act preventively.

$$dSE = SE - SE_{Ref} \tag{4.3}$$

The characteristics-dynamics of the system discussed here can be observed in Figure 4.13 and Figure 4.14, consisting of the pole-map of the model as well as the Bode-plot respectively.

[1] trajectory here denotes the sequence of a signal, as this evolves in time.

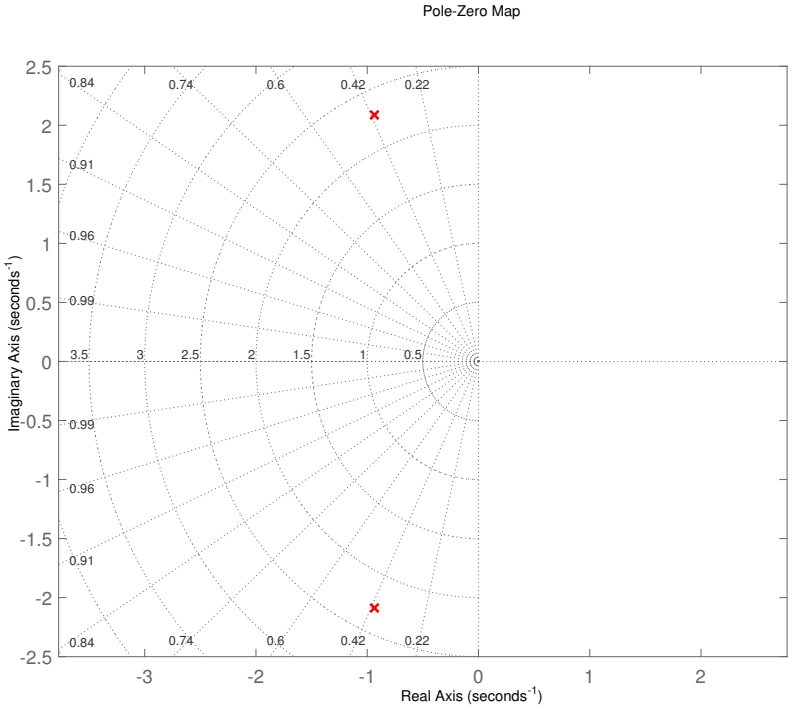


Figure 4.13: m73 Model Poles.

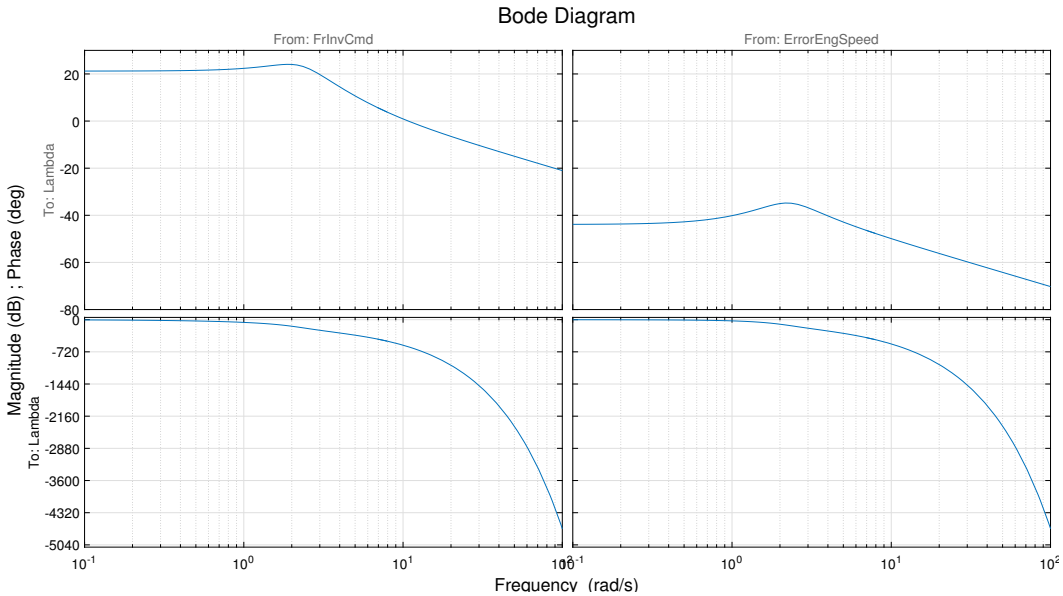


Figure 4.14: m73 Bode Plot.

As we can see both poles of the model lie in the left-hand side of the plane, making the model stable. Moreover as we can observe from the Bode-Plots there are large delays and the phase is increasing as a linear function of frequency in both inputs The output delays

are modelled with a Padé approximation of fourth (4) order. The comparison between the approximation and the model is presented in Figure 4.15:

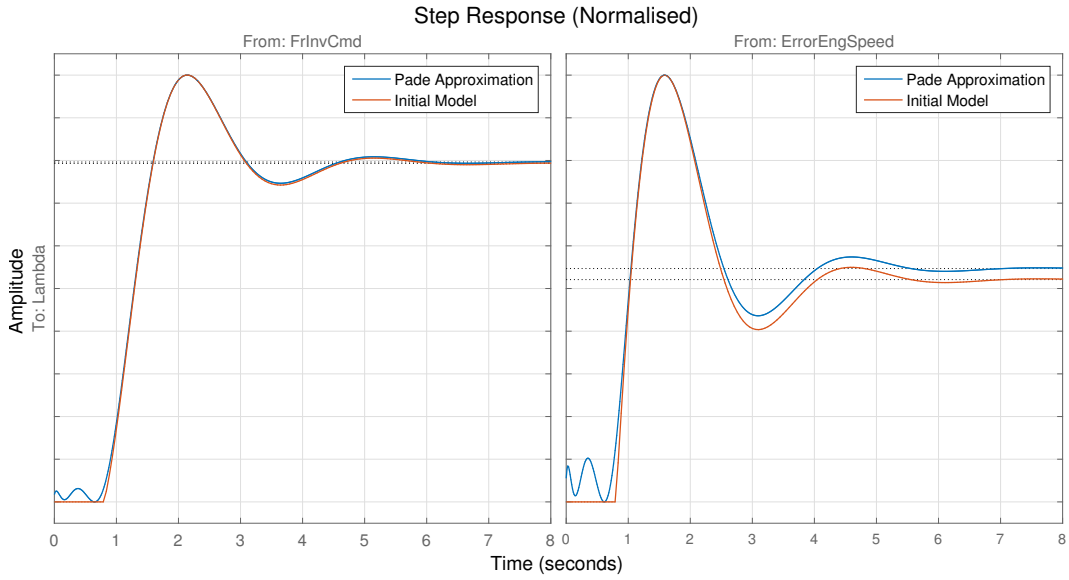


Figure 4.15: Step Input Comparison.

The approximation describes very well the step-response-input behaviour of the initial model, especially with respect to the response derived from the frequency inverter command. The State- Space Model after the Padé approximation is the following:

$$a = \begin{bmatrix} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\ x_1 & -25 & -17.578 & -6.409 & -4.0053 & 240 & 9.328 \\ x_2 & 16 & 0 & 0 & 0 & 0 & 0 \\ x_3 & 0 & 16 & 0 & 0 & 0 & 0 \\ x_4 & 0 & 0 & 4 & 0 & 0 & 0 \\ x_5 & 0 & 0 & 0 & 0 & 0.07556 & -1.739 \\ x_6 & 0 & 0 & 0 & 0 & 3.093 & -1.946 \end{bmatrix}$$

$$b = \begin{bmatrix} & FrInvCmd & ErrorEngSpeed \\ x_1 & 0 & 0 \\ x_2 & 0 & 0 \\ x_3 & 0 & 0 \\ x_4 & 0 & 0 \\ x_5 & 0.3254 & 0.001003 \\ x_6 & -0.7647 & 0.000549 \end{bmatrix} \tag{4.4}$$

$$c = \begin{bmatrix} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\ \lambda & -6.25 & 0 & -1.602 & 0 & 30 & 1.166 \end{bmatrix}$$

$$d = \begin{bmatrix} & FrInvCmd & ErrorEngSpeed \\ \lambda & 0 & 0 \end{bmatrix}$$

In this case, since we have validated the performance of the controller (section 4.1) we can move straight to the simulation of the soft-constrained controller of the full plant, as was implemented in the Simulink environment. The model of the hybrid power plant as well as the waterbrake is considered to be known from previous work at LME [11]. The whole Simulink model is presented in Figure 4.16 and the HIPPO-1 Figure 4.17.

The simulation model consists of a signal builder, providing the plant with the required setpoints, in **total torque demand** and **engine speed (RPM)**. Then there is the plant model, comprised of all the subsystems of the HIPPO-1 testbed, and the respective controller for the water brake. Finally there is the MPC controller in question getting the required data as inputs from the plant as well as the λ setpoint value (constant) and computing the command to the frequency inverter $FrInvCmd$. The tuning parameters in this case are only the weights Q, R since the only constraint is imposed on the control command which

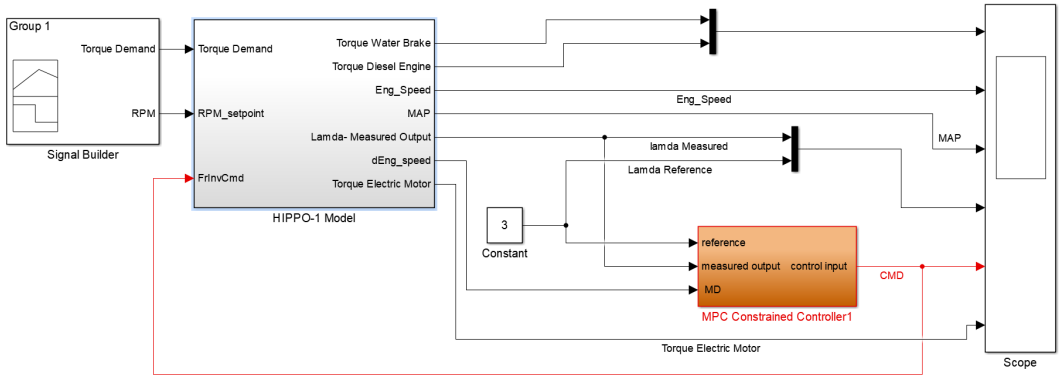


Figure 4.16: The Simulink Model.

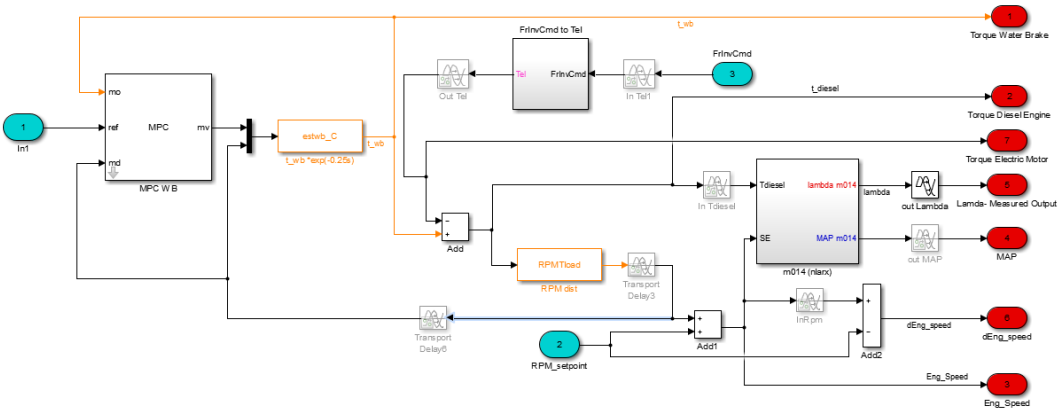


Figure 4.17: The HIPPO-1 Simulink Model.

is a hard constraint, as discussed in section 2.6. We will discuss 2 cases of tuning, one of a 'fast' controller and one of a 'slow' controller, depending on the weights. For the first case the weights then are:

- Case 1 - Fast Controller

$$Q = 2, \quad R = 30, \quad H_u = 1, \quad H_p = 15$$

We can see that since the model has only one controlled output and one measured input the weights degenerate into single numbers. However we will observe how other parameters of the plant behave as well, although they are not directly controlled by the controller. Such parameters are the Manifold Absolute Pressure (MAP), Diesel engine torque, electric motor torque and engine speed. As mentioned above the water brake torque and engine speed are controlled by separate controllers, whose performance will not be investigated here. The results of the simulation are presented in Figure 4.18 and Figure 4.19

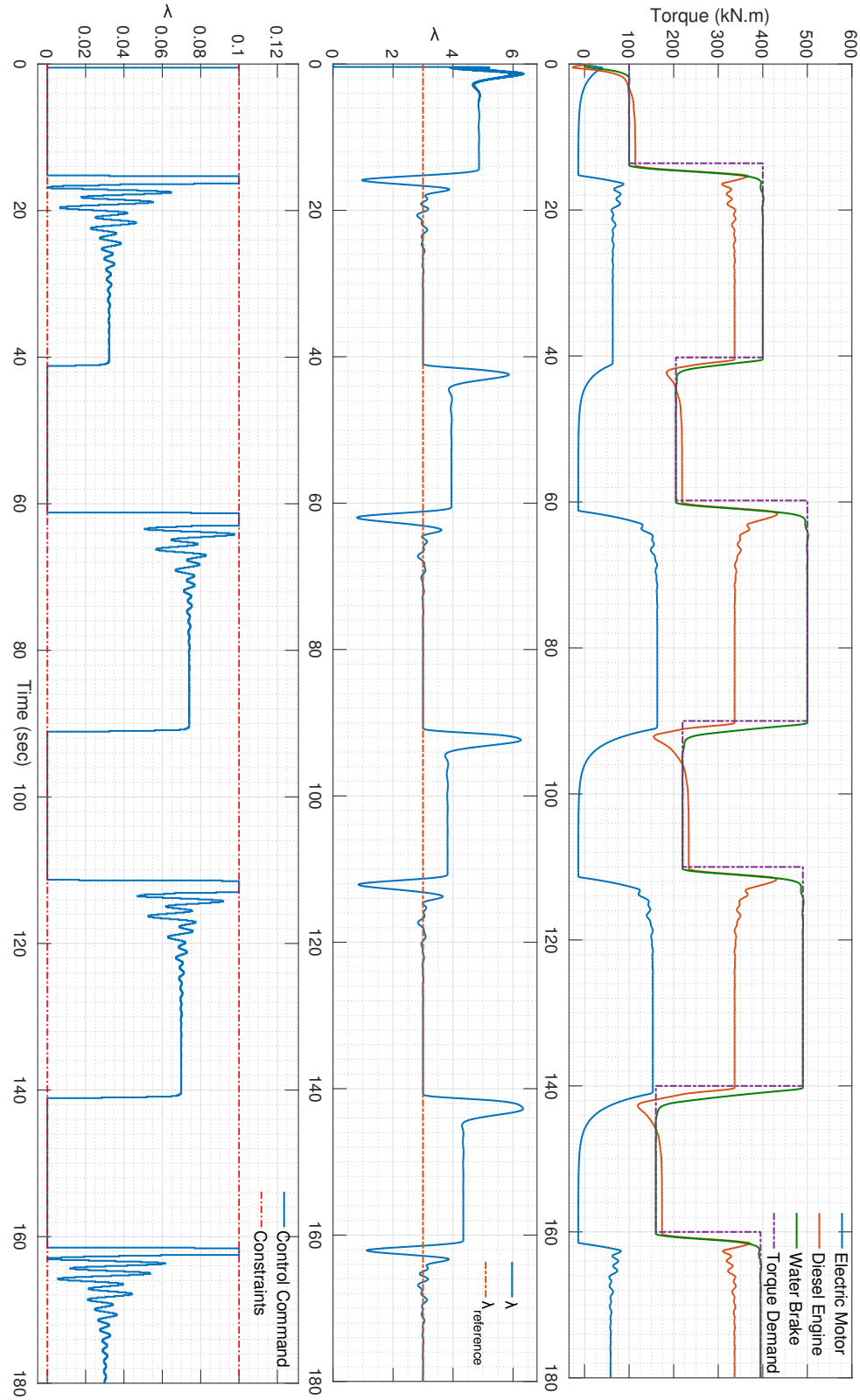


Figure 4.18: Torque, λ and $FrInvCmd$ values

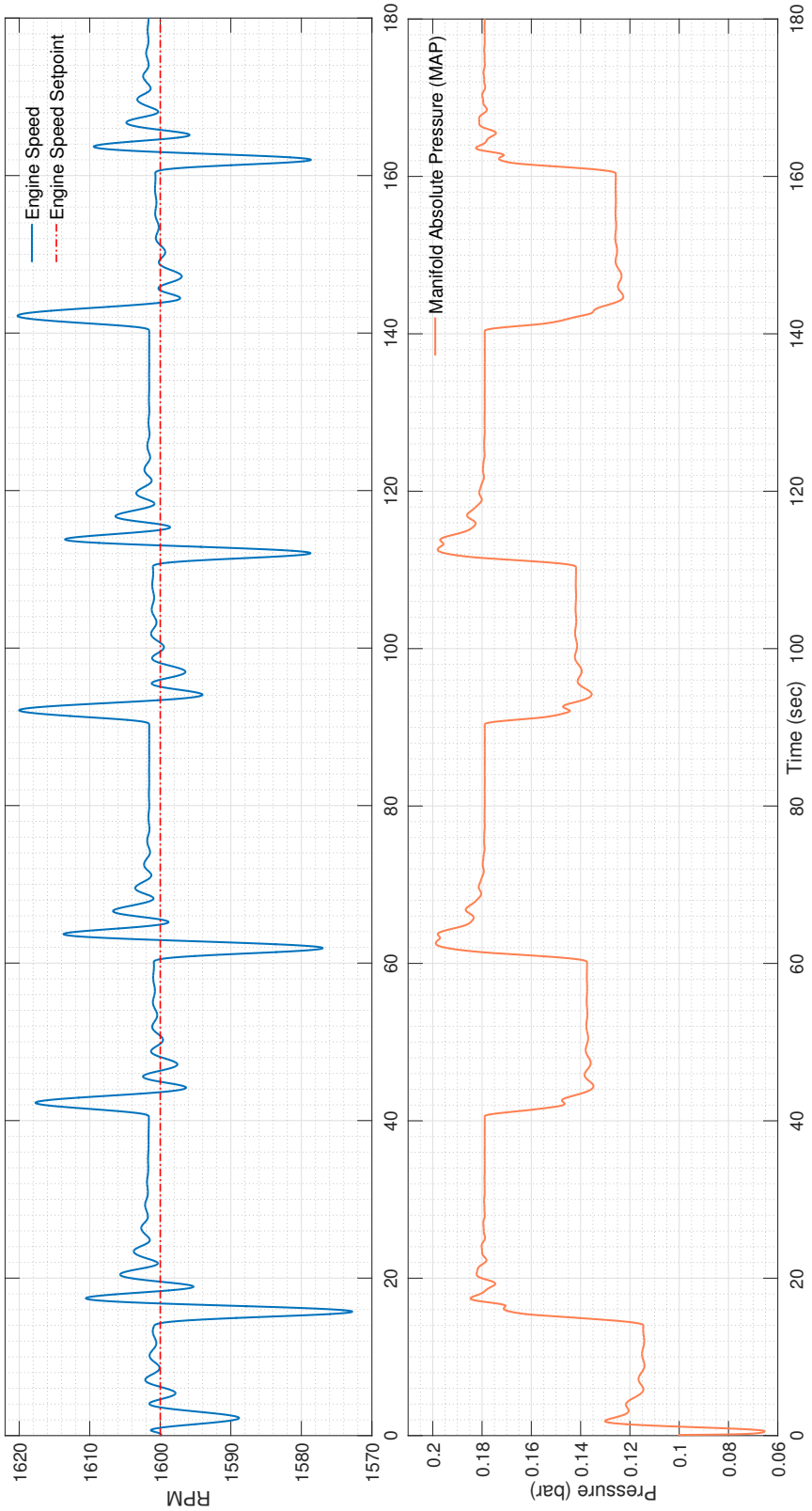


Figure 4.19: MAP and Engine Speed values

By observing Figure 4.18 we can first see that the controller manages to keep the λ values at the set-point value during the transients and the high-load periods. During the low-load periods the ICE operates at higher than intended λ values which means that it produces exhaust gases with low content in fuel which is not an unwanted operating state. We can observe that at these periods of time the controller gives zero command to the frequency inverter. The lower constraint in the control command represents a negative signal, which would turn the electric motor into a generator, absorbing power from the ICE. We can also observe that during the transients the control command reaches its upper boundary, since the controller is quite fast but only for sort periods of time.

As far as the torque is concerned we can see that the electric motor is supplementing the ICE as intended during the high-load periods by "regulating" the λ values to the intended set-point. We can also see that the electric motor is completely shutting down during the periods that its contribution is not needed.

As far as the engine speed behaviour shown in Figure 4.19 is concerned, we can see that the engine speed changes in various ranges. During the transients, the speed varies widely; when the torque of the engine reaches a steady state, then the engine speed settles as well. However, as was also seen in practice, the engine speed has a constant error regarding its set-point, which has to do with the speed controller of the ICE. The deviation is however negligible and is not an issue of further discussion.

The MAP values have the expected behaviour as well. These are however neither controlled nor constrained in any way. We can simply observe that as the load rises, the MAP also rises.

- Case 2 - Slow Controller

$$Q = 2, \quad R = 350000, \quad H_u = 1, \quad H_p = 15$$

In this case we examine a much slower controller, in terms of how fast the Δu is allowed to change. We can see that the corresponding weight R is 4 orders of magnitude larger than the one in the previous case. All other parameters have remained the same since their values were proven to be satisfactory in terms of speed and overall performance Moreover larger prediction horizons H_p do not have a significant impact on the control. On the other hand, larger Control Horizons $H_u > 3$ were proven to make the control input very oscillating since the degrees of freedom increase largely and the controller looses its accuracy.

The results of the simulation are presented in Figure 4.20 and Figure 4.21:

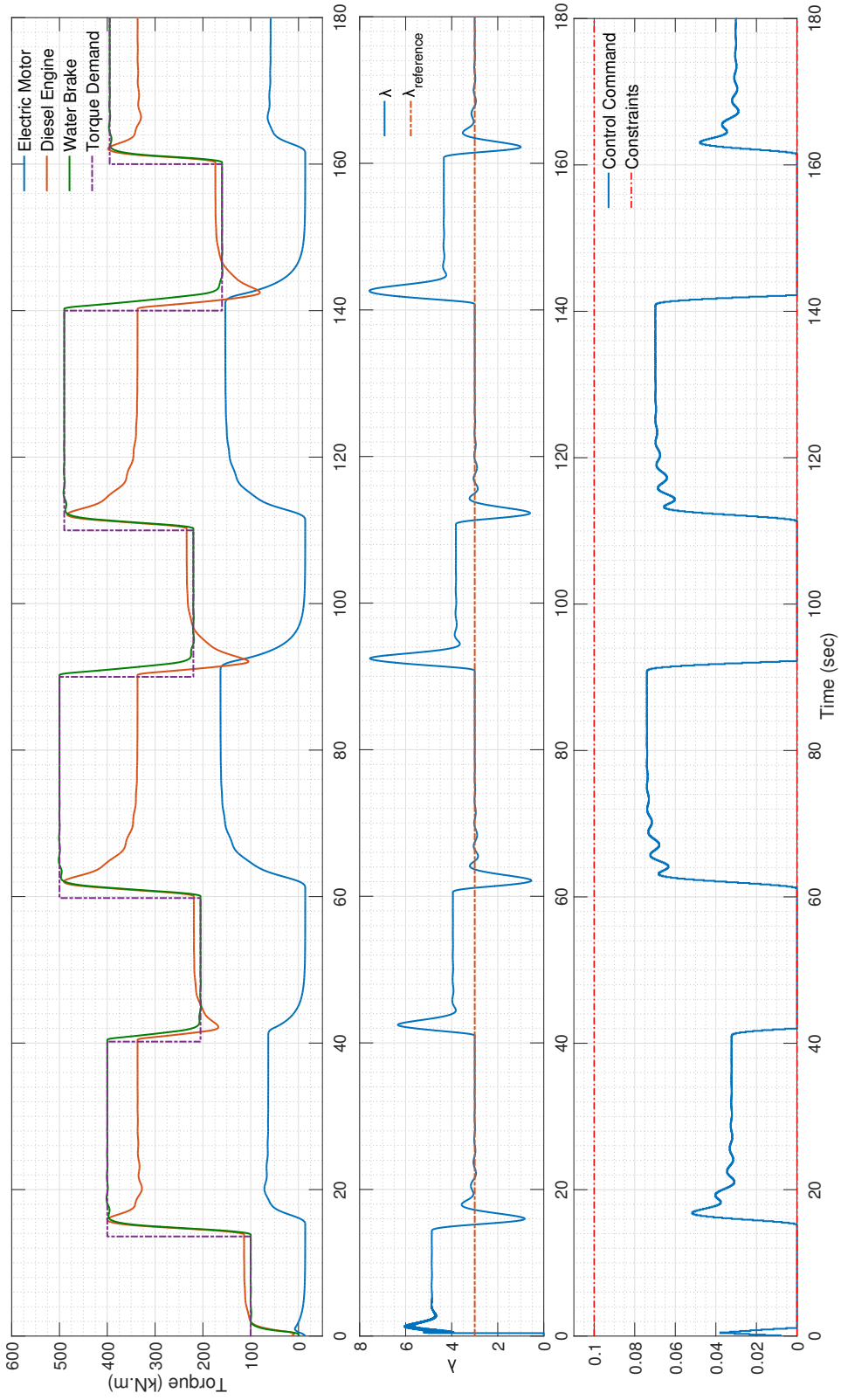


Figure 4.20: Torque, λ and $FrInvCmd$ values

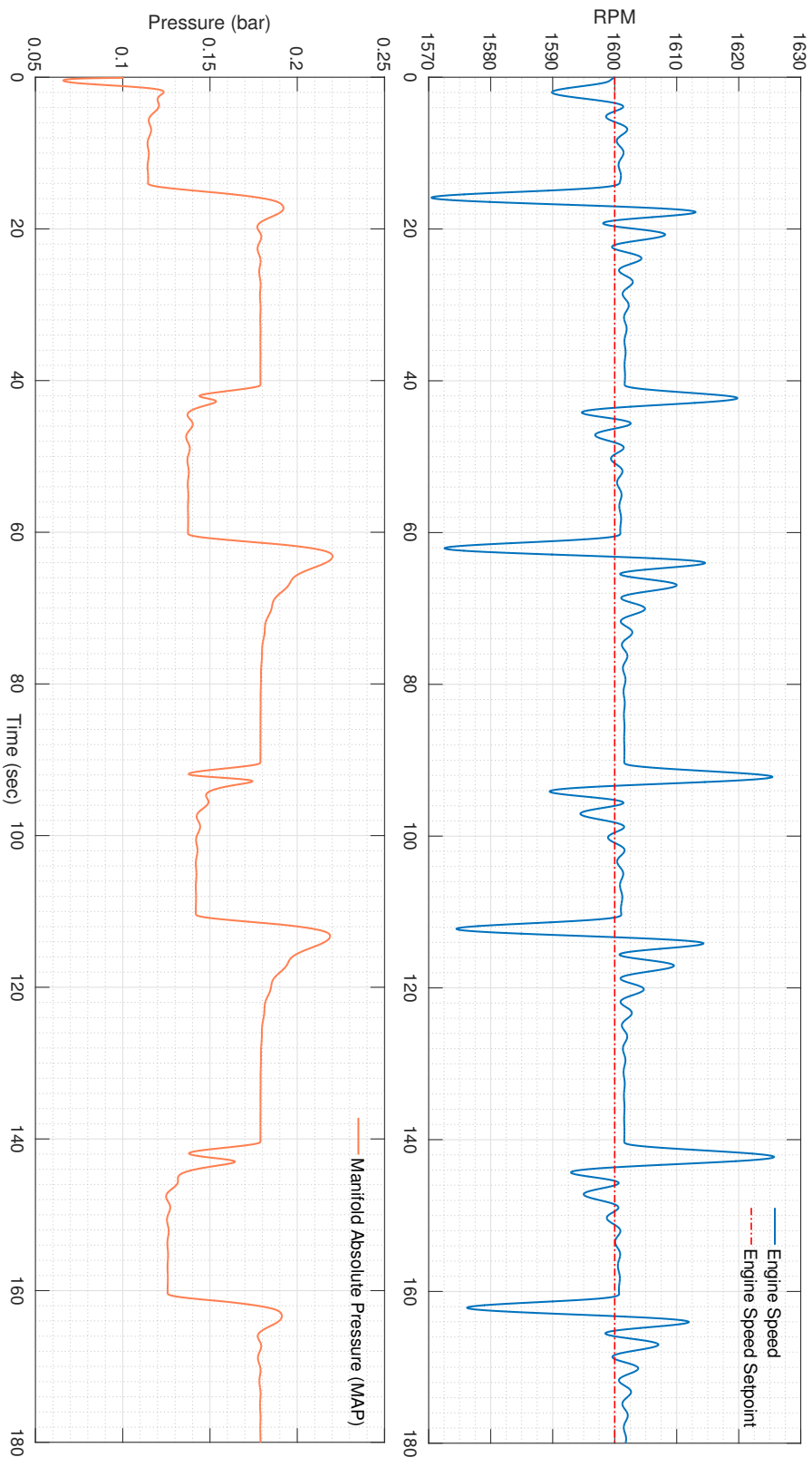


Figure 4.21: MAP and Engine Speed values

We can see that the control input follows indeed a more "conservative/cautious" trajectory. As a result large changes are avoided and the controller never reaches the boundaries imposed. The plant is, however, successfully controlled and the λ values reach their set-point. We could in fact argue that the overall performance is better since the settling time appears to be slightly lower. However we could not expect the controller to behave in a satisfactory way in cases of more frequent changes in power demand and in more complex overall scenarios. We can also sea that the ICE has a larger drop in torque during the transients from high torque to low torque demand due to slower response of the control input with still has non zero values after the transient has occurred This happens because the MPC controller assumes constant reference trajectory. If it took into account the future reference trajectory then this problem could be eliminated. However in practice this is not feasible since the future torque demand could not be known beforehand. Respectively, we can also see a larger increase in ICE torque during the transient from low torque demand to high torque demand since the response of the controller is slower and can not catch up with the large change of the system state.

The slower response of the controller leads to larger overshoots of the λ values as it can be seen from the second diagram of Figure 4.20. This is a direct result of the behaviour of the ICE during the transients as described before. As far as the Engine Speed and MAP behaviour is concerned we can see in Figure 4.21 that larger oscillations occur with the ICE operating farthest from the set-point engine speed

Chapter 5

Experimental Results

In this Chapter we will discuss the results of the experiments conducted at the experimental facility of the Laboratory of Marine Engineering (NTUA) on the test-bed setup HIPPO 1. We will then compare them to the simulation results presented in section 4.2. The controllers tested were tuned so that they resemble the ones used for the simulations. Thus, a fast and a slow controller were used and the behaviour of the plant was assessed in real time. In both cases the results are compared with the MPC controller MPC401 [11], created with the Model Predictive Control Toolbox provided by Simulink [4]. The tuning was roughly the same based on the average values the different parameters take since the way the weights are introduced in the two controllers is different and an exact computation of the equivalent weights is not possible. The MPC controller created by the Simulink Toolbox incorporates scaling factors and therefore the weights introduced by the user are normalized in a way. On the other hand in the MPC controller in question the weights are introduced in their final values so no scaling is needed. The advantage of that is the simpler algorithm since no scaling factors need to be calculated. However the user need to have an idea of the values the different parameters are going to take in order to tune the controller properly and set the rough order of magnitude of the weights accordingly.

The controllers used were the following

- Case 1- Fast Controller

$$Q = 2, \quad R = 35, \quad H_u = 1, \quad H_p = 30$$

Firstly we examine the fast controller which has almost identical weights as the Case 1 examined in section 4.2. The only difference is the twice as long Prediction Horizon H_p.

However we do not expect it to be the root of the differences between the simulation and the model since the unmodelled dynamics of the plant are expected to have a larger impact.

In order to test the performance of the controller a load cycle of alternating load was imposed. The cycle consisted of 2 low load periods of 350 Nm for about 30 seconds and 2 high load periods of 500 Nm of about 30 seconds with 15 second resting periods of 200 Nm between them where the controller is expected to be inactive. The results are presented in Figure 5.1, Figure 5.2 and Figure 5.3.

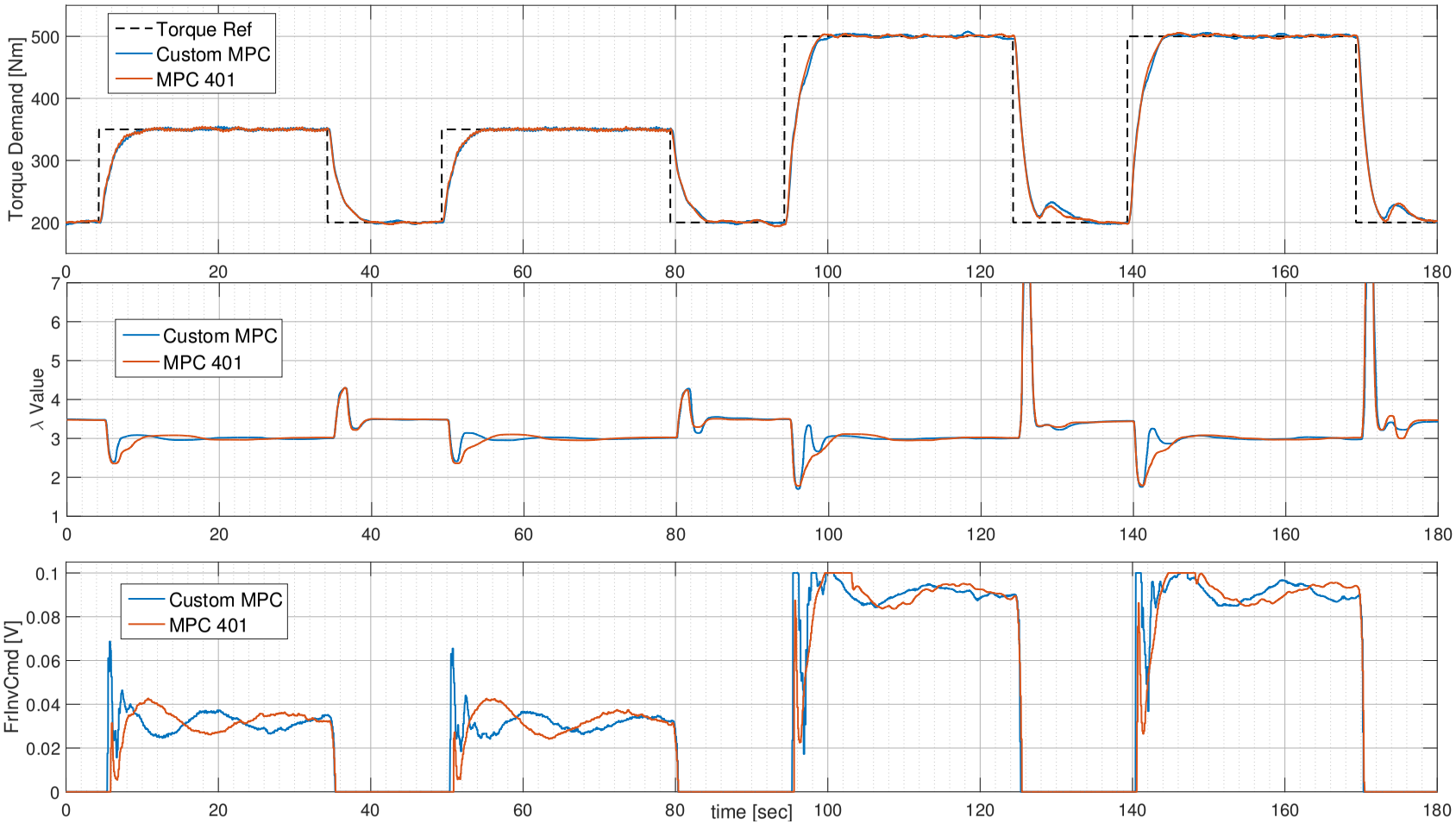


Figure 5.1: Torque, λ and $FrInvCmd$ values.

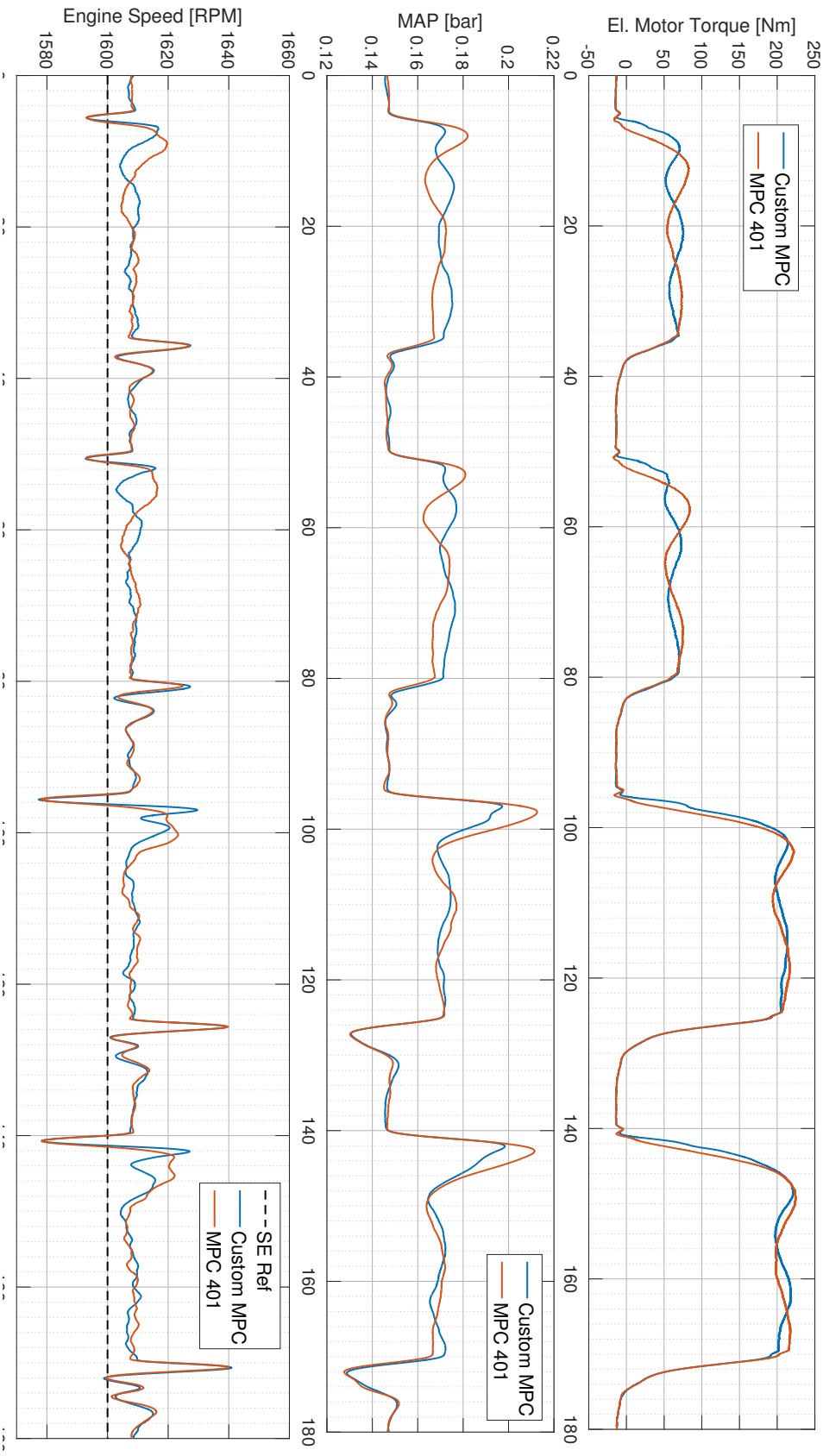


Figure 5.2: Electric Motor Torque, MAP and Engine Speed values.

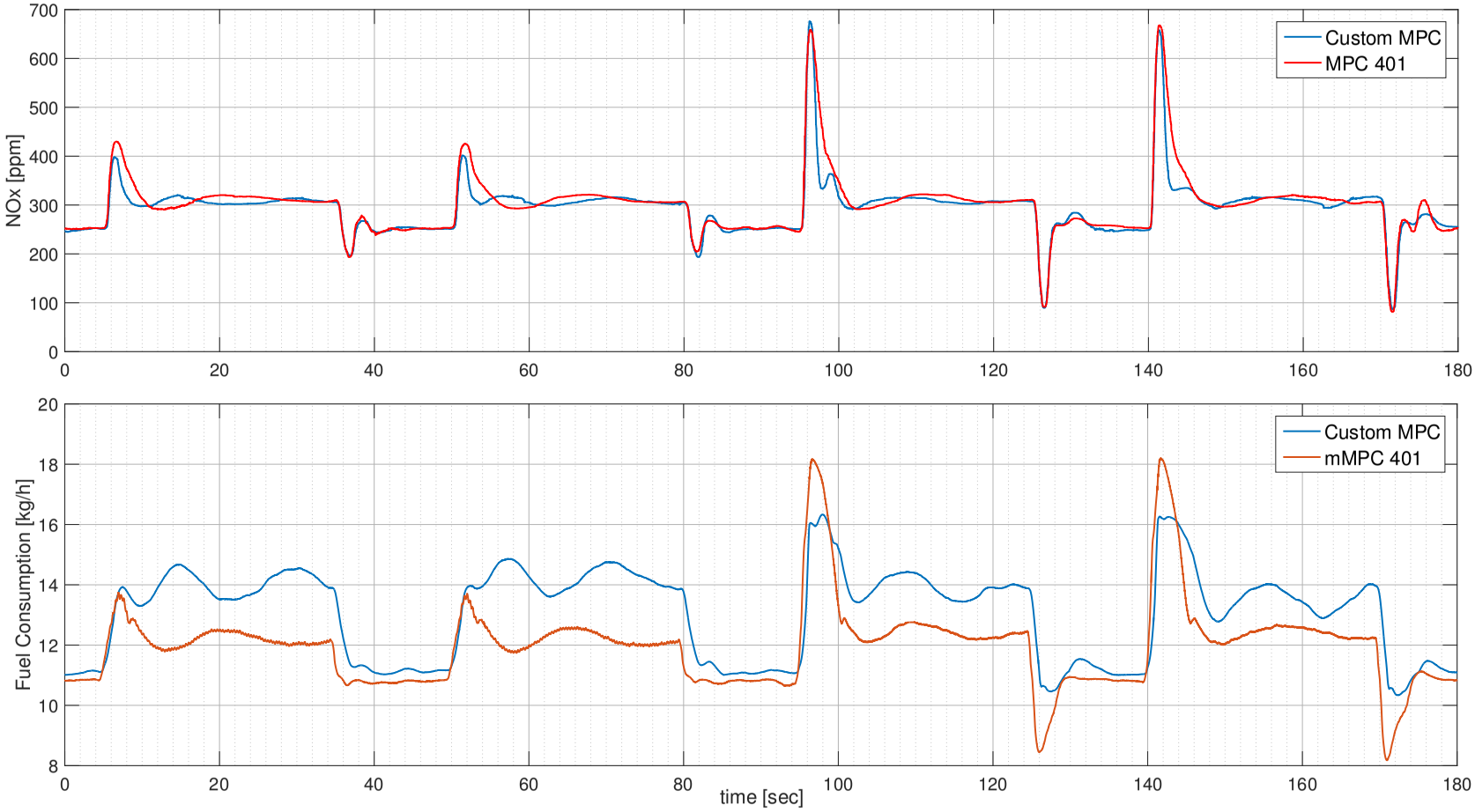


Figure 5.3: NOx values and Fuel Consumption.

As we see, initially in Figure 5.1 the custom MPC manages to successfully control the plant. We can also see that it is a little faster than MPC401 in the way the control input behaves. Our controller reaches almost immediately the boundary during the high-load cycle. The controller MPC401 reaches the boundary as well, later in the cycle but for a longer period of time. The end result is that the plant controlled by Custom MPC reaches faster the λ set-point value of "3". As we would expect from a faster controller the λ values are more oscillating when the Custom MPC is used. We can also observe that since the controller has no way for predicting the change in torque, it can not pro-actively compensate for the transient behaviour, hence the same drop/ overshoot of λ in both cases.

As far as Figure 5.2 is concerned we can see that the experiments agree with the simulations in the behaviour ot the Engine Speed. We observe that the ICE operates constantly at slightly higher rpm than the set-point value. As far as the MAP values are concerned, we can see here as well as in the simulations the "spikes" that occur during the transients. However the behaviour is much more oscillating since there is a number of unmodelled, non-linear dynamics that were not accounted for in the simulation model. The same goes for the Electric Motor torque as well. The dynamics of the electric motor appear to be slower than these of the model and the behaviour is more oscillating. Interestingly, due to the different tuning of the controllers there appears to be a phase difference in the behaviour of the plant between the two controllers. Finally we observe that there is a smaller overshoot in the Electric Motor torque, when the Custom MPC is used, which leads in the smoother operation of the whole plant.

In Figure 5.3 we can see how the NOx in the exhaust gases behave during the different loading cycles. First of all we observe that they follow the behaviour of λ with "spikes" of very high values during the transient from low to higher torque. In this phase of the operation we could expect higher particulate matter and lower opacity as well (black smoke). However these were not measured in the current study. During the un-loading of the system the NOx values decrease dramatically as the λ increases, meaning that the mixture of exhaust gases is more rich in atmospheric air. In general we see that the NOx amount in the exhaust gases is kept under a constant limit, the same in both cases, although not directly controlled by either of the controllers. Only during the transients from lower to higher torque demand the Custom MPC brings the NOx values slightly faster down in comparison to the MPC 401. As far as the fuel consumption is concerned, we can see an unexpected behaviour between the two controllers. Although the ICE and the electric motor operate roughly at the same points under the same conditions, the values that the Fuel Consumption seems to settle during the various load cycles appears to be a lot different between the two cases. However the way the fuel consumption is measured is not exact and the measurement results should be treated with caution.

In order to evaluate the performance of the two controllers more accurately we could calculate the error as follows

$$E = \int_{t=0}^{t=\tau} (\lambda - setpoint)^2 \, dt, \quad setpoint = 3 \tag{5.1}$$

The error is displayed in Figure 5.4

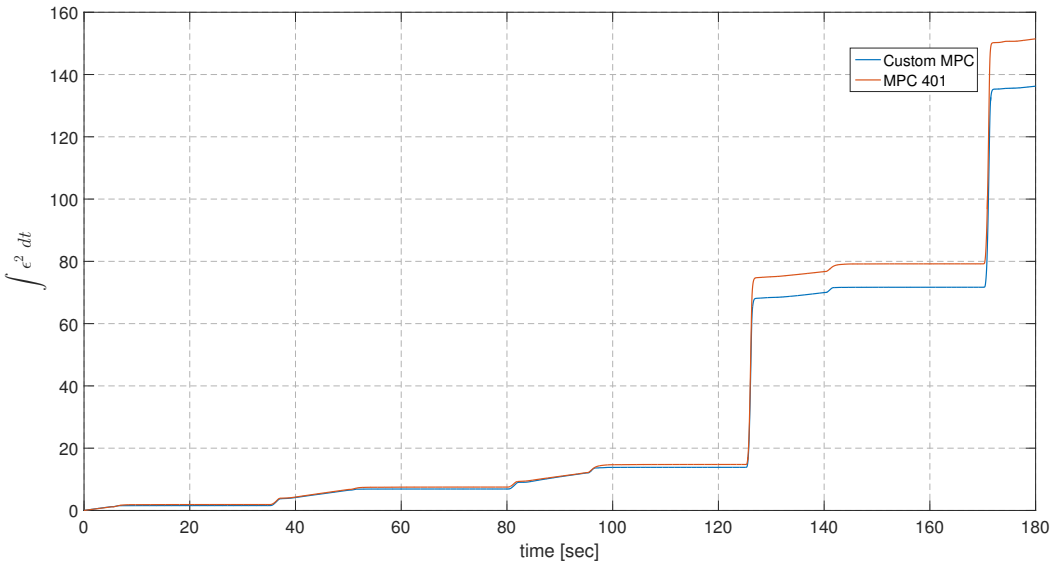


Figure 5.4: Error of the two controllers

We can see that during the low-load cycles the difference is very small. However when we move to the high-load cycles then the difference becomes significant. We can therefore come to the conclusion that the Custom MPC has a better performance in this particular case.

- Case 2- Slow Controller

$$Q = 2, \quad R = 27\ 10^4, \quad H_u = 1, \quad H_p = 30$$

In this case we will examine a slower controller, in terms of response speed since the weight on Δu is much larger than in the previous case. We will compare it with the faster Custom MPC used in the case presented above and discuss the differences. The loading cycles are the same as in the previous case. The results are presented Figure 5.5, Figure 5.6 and Figure 5.7.

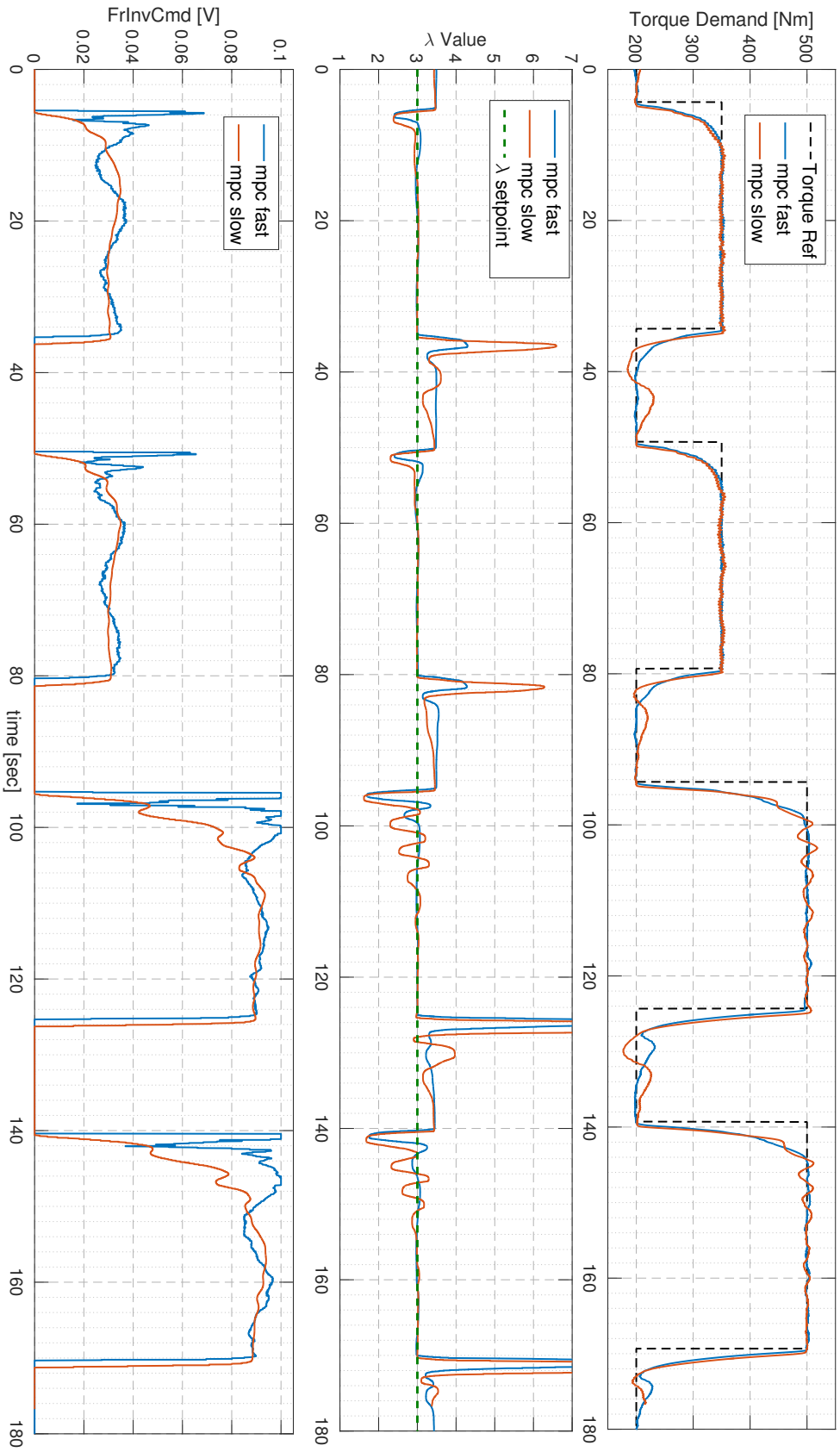


Figure 5.5: Torque, λ and $FrInvCmd$ values.

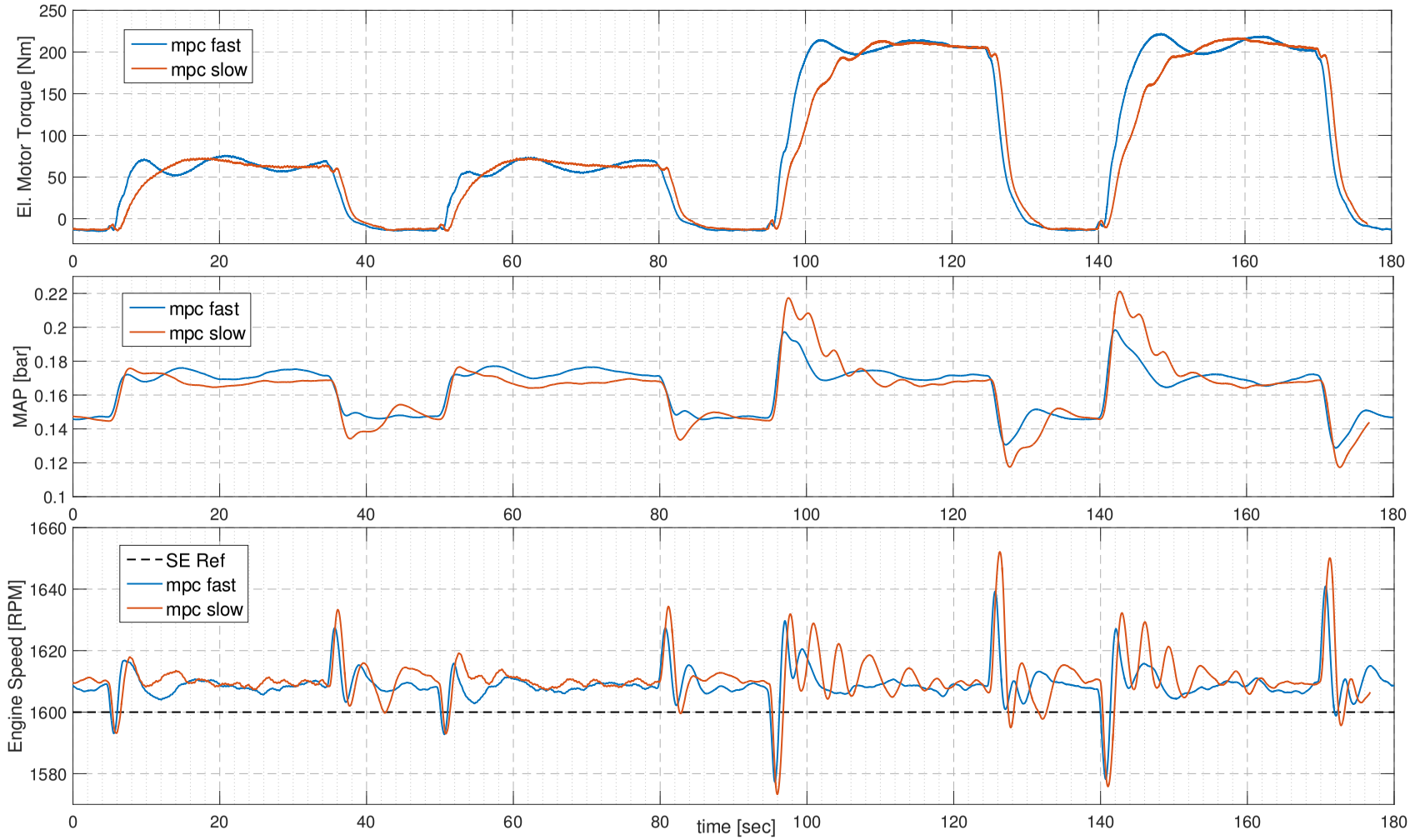


Figure 5.6: Electric Motor Torque, MAP and Engine Speed values.

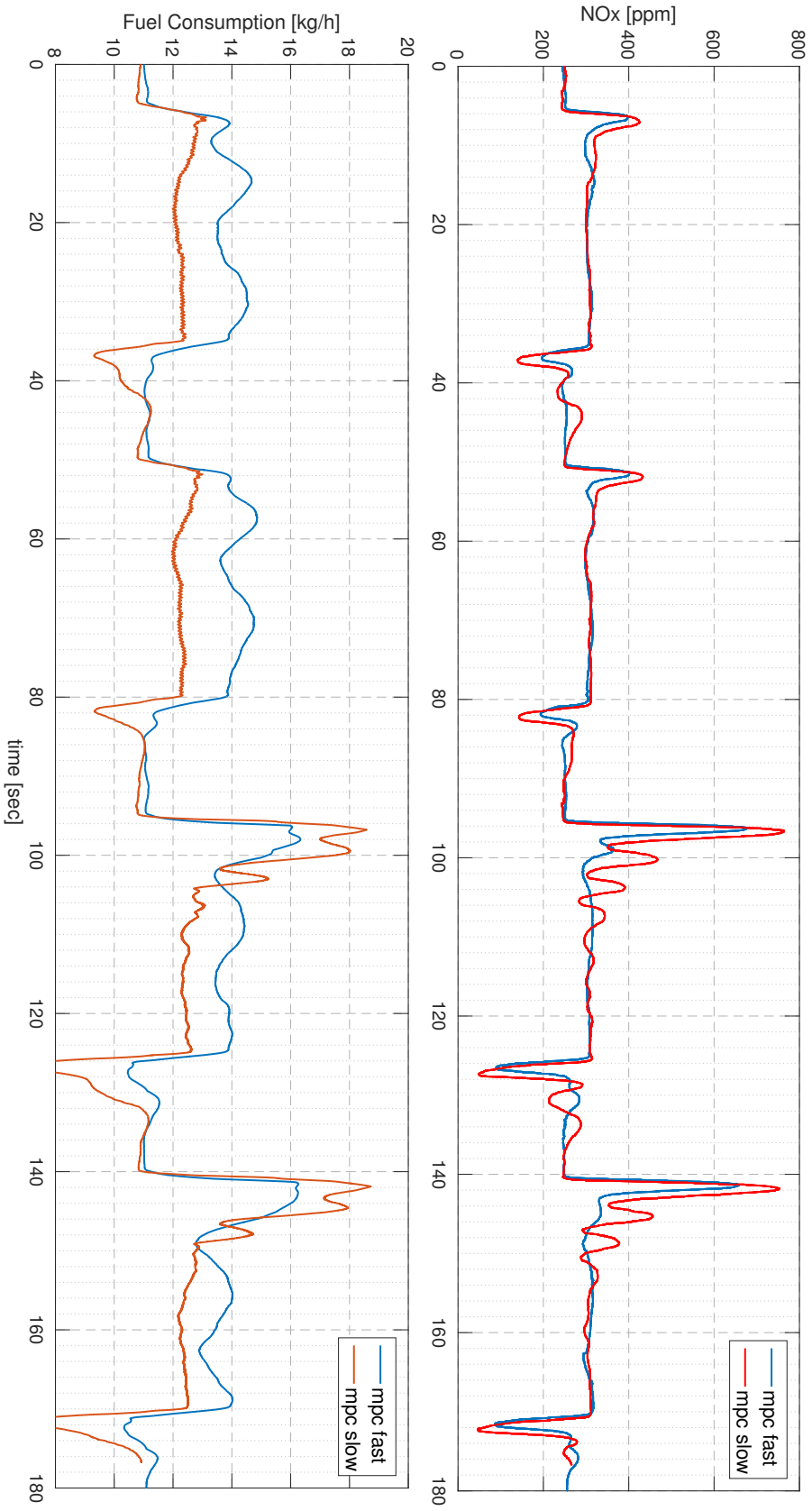


Figure 5.7: NOx values and Fuel Consumption.

First of all, from Figure 5.5 we can see that the slower controller manages to control successfully the plant. As expected the controller in question reacts much slower than the faster one in terms of Δu. This does not seem to be a problem for small changes in the torque demand. Instead in these cases the slower MPC yields a better result as far as the speed that the λ values approach their set-point value. However, during the high-load cycles as well as during the de-loading of the hybrid plant the behaviour seems to be a lot worse. During the de-loading there is small time period of about 2 seconds in which the slower controller continues to operate the electric motor, thus overshooting the λ values before it reacts and "cuts" the control input to the electric motor. During the high-load cycle it is obvious that the slower MPC does not have enough time to react which therefore leads to a very oscillating behaviour that cannot be accepted since the controller reaches a steady state much later than the faster MPC, almost 12 seconds. On the other hand the slower controller since it is moving much more "cautiously" than the faster one, never reaches the boundaries, which could be a desired behaviour for some applications.

Moving on to Figure 5.6 we can see, as expected that when controller by the slower MPC the plant needs a lot more time to reach its steady state as fas as the Electric Motor Torque is concerned. As a result the plant operates farther from its RPM set-point, being more oscillating as well, especially during the high-load cycles. This is definitely an undesirable behaviour which can lead to damaging components of the plant.

As far as Figure 5.7 is concerned we can see that the plant controlled with the faster MPC has lower NOx emission during all phases of the experiment. An interesting point in this figure is the values the Fuel Consumption takes. We can see that the consumption measured during the experiment with the slower MPC is comparable with the consumption measured in the experiments with the MPC 401 examined in the previous case. This strengthens our initial assumption that the measurements of the experiment in which the faster MPC was used are inaccurate. However it is obvious that with the way the fuel consumption is measured, the large oscillations in the operation point of the hybrid plant during its operation with the slower MPC cannot be seen.

In general we can see that the faster MPC controls the plant in all aspects of the operation much better than the slower one. It is therefore not necessary to compare the error between the two controllers to evaluate their performance as was done in the previous case we examined.

Chapter 6

Conclusions

In this thesis the feasibility of designing and implementing a MPC controller in order to control a hybrid Diesel- Electric marine powerplant was investigated. First the controller was designed according to the available literature and then was tested in simulations as well as full-scale experiment successfully. The controller was able to handle a variety of input and output constraints, account for modelled an unknown disturbances and generally handle multi-variable problems. During the development the need for future work, additions and modifications arose.

- The need for on-line tuning became apparent during the experiments. The controller could be modified in order to support on-line weight definition and prediction/ control horizons. This could make the controller much more flexible and allow the testing in a much broader field of conditions.

- During the development of the controller and especially during the experiments the importance of a robust, reliable and fast QP optimizing tool became obvious. The QP optimizer should also built in a way so that it would be flexible and compatible with a variety of computational platforms. It is strongly recommended that a specialized QP optimizer should be developed and incorporated into the current controller. The development of such an optimizer would also make the controller much more flexible and more easily customizable for each application.

- On a more general note, this study does note account for the overall efficiency rate of the plant in terms of energy delivered / energy used. A study on this matter would greatly complete the picture we have about the hybrid diesel-electric plant and investigate how the different tuning parameters and controller set-up act not only on emissions but also on the overall performance. In order to do that the fuel consumption on the diesel engine as well as the electric current on the electric motor should be

precisely measured. With the current setup this was not possible as was mentioned in chapter 5.

- Finally a plant also powered by batteries would be an evolutionary step of the experimental facility, since the charge-discharge cycles of batteries would complicated the task of the controller even further but also bring the plant closer to the marine applications of hybrid plants found today on-board ships.

Bibliography

[1] J.M. Maciejowski. 2000. Predictive Control with constraints. *PRENTICE HALL PTR*

[2] G.F. Franklin, J.D. Powell, A. Emami- Naeini. 1994. Feedback Control Of Dynamic Systems *PRENTICE HALL PTR*

[3] L. Wang Model Predictive Control System Design and Implementation using MATLAB®. 2009. *Springer*

[4] A. Bemporad, M. Morari, N.L Ricker . 2016. Model Predictive Control Toolbox, User's Guide. *MathWorks*

[5] Z. Li, J. Sun. 2012. Disturbance Compensating Model Predictive Control With Application to Ship Heading Control. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 20, NO. 1 January 2012*

[6] E. C. Kerrigan, J. M. Maciejowski, . Soft Constraints and Exact Penalty Functions in Model Predictive Control. *UKACC International Conference 2000*

[7] S. Adachi, M. Iwadare, M. Ueno. 2009. Multi-Variable Air-Path Management for a Clean Diesel Engine Using Model Predictive Control. *SAE International.*

[8] 2017. Optimization Toolbox, User's Guide. *MathWorks*

[9] J.B. Rawlings, D.Q. Mayne. 2015. Model Predictive Control: Theory and Design. *Nob Hill Publishing LLC*

[10] G. Ripaccioli, A. Bemporad, F. Assadian, C. Dextreit, S. Di Cairano I. Kolmanovsky. 2009.Hybrid modeling, identification, and predictive control: an application to hybrid electric vehicle energy management: hybrid systems: computation and control. *Lecture notes in computer science, vol. 5469. Springer, p. 321-335*

[11] N. Planakis. 2016. Predictive Control of a Hybrid Diesel - Electric Marine Propulsion Plant, Diploma Thesis

[12] A. Taghavipour, N. L. Azad, J. McPhee. 2015. Real-time predictive control strategy for a plug-in hybrid electric powertrain. *Mechatronics vol. 29 p. 13-27*

[13] P. Ortner, L. del Re. 2007. Predictive control of a Diesel Engine Air Path. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 15, NO. 3*

[14] S. Trimboli, S. Di Cairano, A. Bemporad, I. Kolmanovsky. 2009. Model predictive control for automotive time-delay processes: an application to air-to-fuel ratio. *Proceedings of 8th IFAC workshop time-delay systems, p. 1-6.*

[15] P. Majecki, G.M. van de Molen, M. Grimble, I. Haskara, Y. Hu, C.F.Chang. 2015. Real-Time Predictive Control for SI Engines Using Linear Parameter-Varying Models. *IFAC-PapersOnLine 48-23 (2015) p. 094-101*

[16] P. Falcone, F. Borrelli, J. Asgari, H. Tseng, D. Hrovat. 2007. Predictive active steering control for autonomous vehicle systems *IEEE Trans Control Syst Technol, 15 (3) (2007), pp. 566-580*

[17] L. Guzzella, C. Onder. 2004. Introduction to Modeling and Control of Internal Combustion Engine Systems. *Springer, 2nd edition*

[18] R.E Kalman. On the General Theory of Control Systems

[19] K.C. Daly A.P. Colebourn (1979) Pad approximation for state space models *International Journal of Control, 30:1, p. 37-47*

[20] Hybrid/Battery Systems.2017. *Wärtsilä Electrical and Automation, www.wartsila.com*

[21] H.E. Lindstad, G.S. Eskeland, A. Rialland.2017. Batteries in offshore support vessels-Pollution, climate impact and economics *Transportation Research, pp. 409-417*