FAULT-TOLERANT CONTROL OF A SHIP PROPULSION SYSTEM USING MODEL PREDICTIVE CONTROL

E.C. Kerrigan[*], J.M. Maciejowski[†]

Department of Engineering
University of Cambridge
Trumpington Street
Cambridge CB2 1PZ
United Kingdom
Fax : +44 1223 332662 and e-mail : {[*]eck21,[†]jmm}@cam.ac.uk

Keywords : fault-tolerant control, fault handling, predictive control, ship propulsion benchmark.

Abstract

Recently, it has been shown how model predictive control (MPC) can adapt to faults in certain circumstances. This paper describes how MPC was successfully implemented as a fault-tolerant controller for a single engine/propeller model of a ship propulsion system. It is shown that the MPC controller can be tuned to be robust to internal faults that develop in the ship propulsion system, even in the absence of any fault detection and isolation (FDI) information for the internal faults. For the case of sensor faults, it is assumed that FDI information is available and it is shown how the MPC controller, in combination with a Kalman estimator, can drastically improve the tracking response of the system in the presence of sensor faults. The paper concludes that MPC is a very good candidate for a fault-tolerant controller for the ship propulsion system, requiring re-configuration only at the supervisory level, without the need for additional re-configuration in the lower-level control systems.

1   Introduction

In practice all physical systems have some form of constraint, whether the constraint is due to a physical, economic, safety or performance requirement. These constraints are typically enforced on control inputs, control rates and/or system outputs. The ability of model predictive control (MPC) to handle constraints systematically is one of the primary advantages of MPC over alternate control schemes.

Lately, there have been proposals to apply MPC to areas outside the process industries, where it is the most widely used advanced control technique. In particular, its use for flight control, including the question of fault-tolerance, has been investigated in [1, 2, 3]. It has also been shown how MPC transparently adapts the controller in the presence of actuator faults, even without fault detection and isolation (FDI) information [4, 5, 6, 7].

In [8] a nonlinear model of a low speed marine vehicle is described with the aim of it being used as a benchmark to test a control system's fault-tolerant capabilities. This paper describes the results from designing and implementing an MPC controller for the non-linear single engine/propeller model of the ship.

The benchmark describes requirements for FDI and re-configuration. Only the re-configuration part is studied in this paper and it is assumed that, where necessary, FDI information is available without delay.

Based on the results presented in this paper, some conclusions are drawn and recommendations made as to further work that needs to be undertaken.

2   Controller design

The requirement of the benchmark is that the remedial actions should primarily use re-configuration at the coordination level to accommodate a fault. The performance in the re-configured mode is allowed to be lower than under the no-fault conditions. Bump-less transfer is not required, but large transients should be avoided when re-configuring the controller.

This section starts with a review of the principles behind MPC. A description of the ship propulsion system is given, before the control objectives and constraints are defined. The effect of tuning the different MPC design parameters will also briefly be discussed.

2.1 Review of MPC

The main idea behind MPC is to determine future values of the control inputs by optimising a cost function which expresses the control objectives. The sequence of N_u future control signals $u(k + i)$ is determined by optimising a cost function that typically has the quadratic form

$$J(k) = \sum_{i=N_1}^{N_2} \|M\hat{x}(k + i|k) - r(k + i)\|_{Q(i)}^2$$
$$+ \sum_{i=0}^{N_u-1} \|\Delta u(k + i)\|_{R(i)}^2 \qquad (1)$$

subject to the constraints

$$\Delta u_j(k + i) \in [V_{min_j}, V_{max_j}] \qquad (2)$$
$$u_j(k + i) \in [U_{min_j}, U_{max_j}] \qquad (3)$$
$$(M\hat{x})_j(k + i|k) \in [X_{min_j}, X_{max_j}] \qquad (4)$$

where the control increments are defined as $\Delta u(k) = u(k + 1) - u(k)$ and $Mx(k)$ is the vector of variables to be controlled; $x(k)$ is the state of the plant. $\hat{x}(k + i|k)$ is a prediction of $x(k+i)$ made at time k, and M is some matrix (for example, $M = C$ in the usual state-space model if only outputs are to appear in $J(k)$). $r(k)$ is some reference trajectory for $Mx(k)$.

The integers N_1 and N_2 are the minimum and maximum output horizons, respectively. It is assumed that the control signals are constant after the end of the optimisation horizon, namely that $\Delta u(k + i) = 0$ for $i \geq N_u$.

The norm $\|.\|_Q^2$ within the cost function is defined as $\|\alpha\|_Q^2 = \alpha^T Q \alpha$.

In the inequalities $u_j(k)$ denotes the j'th component of the vector $u(k)$, etc, and V_{min_j}, V_{max_j}, U_{min_j}, U_{max_j}, X_{min_j} and X_{max_j} are problem-dependent values.

The first sum in (1) penalises the control error, while the second sum penalises the control effort. The matrices Q and R are used to weight the corresponding control errors and control actions.

For each set of control increments, the predicted values of the state vector are calculated from a (usually) linear internal model. Once an optimal solution has been found, only the first actuation signal is applied to the process. At the next time instant $k+1$, the whole sequence is repeated using the latest plant measurements.

For a detailed discussion of MPC, the reader is referred to the literature [9, 10]

2.2 Description of the ship propulsion system

The focus of this study has been limited to the design of a controller for the single engine system. The propulsion system can be treated as a "black box" with four inputs and four outputs. The known inputs to the system are propeller pitch angle set-point θ_{ref} and shaft speed set-point n_{ref}. Unknown inputs to the system are external forces T_{ext} and friction torque Q_f. The measured outputs are propeller pitch angle θ_m, diesel engine shaft speed n_m, ship speed U_m and fuel index Y_m.

Inside the propulsion system there are two low-level control loops. The first loop controls the propeller pitch angle θ and the second controls the shaft speed n.

This paper assumes that only the lookup table and overload control modules as described in [8] are implemented, without any efficiency optimisation or ship speed control. The lookup table converts a command lever position h into two command signals θ_{com} and n_{com}. The MPC controller was placed in the coordinated control level, with θ_{com} and n_{com} as its inputs and its outputs $\theta_{com,MPC}$ and $n_{com,MPC}$ as inputs to the overload control module as can be seen in Figure 1. The outputs of the overload control module are θ_{ref} and n_{ref}.

The goal of the control system will therefore be for θ_m and n_m to track the command values θ_{com} and n_{com} as closely as possible. The values for θ_{com} and n_{com} and θ_{ref} and n_{ref} might be the same, depending on whether the engine is within its torque limits and no fault has occurred. As will be seen later, the values for θ_{ref} and n_{ref} are adjusted by the MPC and overload controller in order to accommodate a fault or overload condition.

2.3 Obtaining an internal model

For the design of the MPC controller, certain assumptions have to be made and parameters chosen before the controller can be implemented on the non-linear model. The first step in the design of the MPC controller was to obtain a suitable internal linear model of the non-linear plant. The choice of linearisation point is crucial to the stability, performance and robustness of the controller.

In the benchmark, the command lever position h varies from 3 to 10, and a value of $h = 6.5$ was chosen around which to linearise the plant. The linear state-space model was obtained using Matlab's *linmod* command. Using the look-up table for the *economy* mode, the resulting values for pitch angle and shaft speed at which the model was linearised are $\theta_0 = 0.7397$ and $n_0 = 10.866$.

Because of problems encountered in the linearisation, the "black-box" model was reduced to a two-input, three-output system; the inputs being θ_{ref} and n_{ref} and the outputs being θ_m, n_m and U_m. As not all seven of the plant states were available for feedback, a Kalman estimator was used to estimate the current state that would be used for prediction in the MPC cost function (1). Future work will involve updating the software to enable the use of the full four-input, four-output system as the MPC controller's internal model.
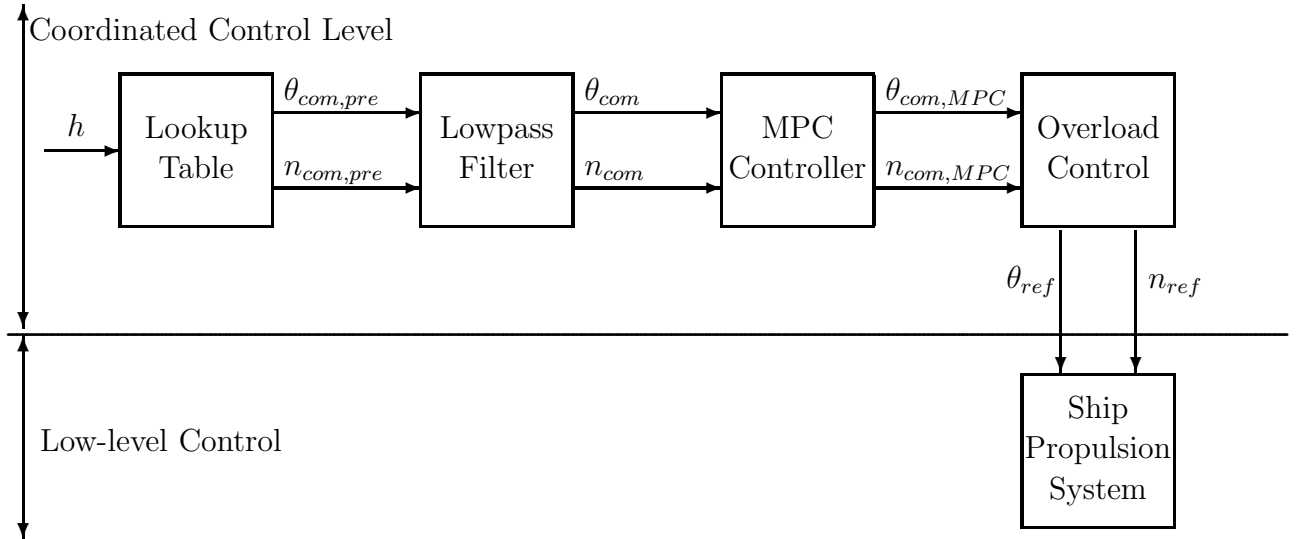
Figure 1: Placement of MPC controller in the coordinated level.

2.4 Defining the constraints

The benchmark gives hard limits on the inputs and outputs. One of the strengths of MPC is its ability to include constraints on the inputs and outputs in the problem formulation. The resulting constraints on the input and output variables are:

$$\theta_{com}, \theta_m \in [-0.7, 1] \tag{5}$$
$$n_{com}, n_m \in [0, 13] \tag{6}$$
$$U_m \in [0, 9.7] \tag{7}$$

Since the engine dynamics are sensitive to abrupt changes in the reference signal, a low-pass filtered version of θ_{com} and n_{com} should be used as described in [8]. Since MPC allows one to limit the control input rate, the input rate limit was chosen to be the maximum rate as implemented by the filter. The control input rate limits used in the controller are:

$$\Delta\theta_{com} \leq 0.1 \tag{8}$$
$$\Delta n_{com} \leq 1.3 \tag{9}$$

2.5 Tuning of MPC parameters

Though theories exist about the choice of prediction horizons and weighting matrices to guarantee certain stability and robustness properties, most of the tuning of these parameters are usually done by iteration and simulation. The parameters that need to be tuned are the sampling time T_s, the control and prediction horizons N_u and N_2 and the output and input weighting matrices Q and R.

2.5.1 Selection of Prediction and Control Horizons

The control and prediction horizons mainly affect the performance of the controlled system, but may also influence the robustness. A long prediction horizon, results in better performance of the control system. In general, a short control horizon makes the system more robust to uncertainties such as parameter variations.

A long prediction horizon is necessary for systems with slow dynamics. A long horizon will result in an unnecessarily long computation time for each control input and should therefore be kept as short as possible, without sacrificing performance too much. The prediction horizon must contain at least the non-minimum phase behaviour of the system.

2.5.2 Selection of Weighting Matrices

The output performance weighting matrix Q and the control increment weighting matrix R are two very important design parameters. They play an important role in determining the stability and performance of the system.

The R matrix penalises the control increments and helps to keep the control inputs within bounds, making sure that smooth control actions result. The Q matrix, in addition to helping ensure that no output constraints are violated, penalises the tracking errors and therefore improves the servo performance of the control system.

2.5.3 Tuning Parameters used in the MPC Controller

The tuning of the control parameters was done using the linear and non-linear model. The results from adjusting the various parameters are given in [11].

The minimum output horizon was set to $N_1 = 1$. The sampling time has been fixed by the benchmark to $T_s = 1.0$s. This is the sampling time that was used for the MPC controller. After several iterations, the final values for the implemented controller were chosen to be $N_2 = 10$, $N_u = 5$, $Q = \texttt{diag}(10, 1, 0)$ and $R = \texttt{diag}(20, 2)$.

Event	Severity	Start time	End time
$\Delta\theta_{high}$	High	180s	210s
$\Delta\dot{\theta}_{inc}$	Medium	800s	1700s
$\Delta\theta_{low}$	Very high	1890s	1920s
Δn_{high}	High	680s	710s
Δn_{low}	Very High	2640s	2670s
Δk_y	Medium	3000s	3500s

Table 1: *Test sequence for different fault events.*

3 Fault simulation

Once satisfied that the controller was stable and that it performed well over the operating range, faults were introduced one at a time.

In the benchmark six faults and their effect on the system and degree of severity are discussed. The faults are related to propeller pitch, shaft speed measurement and the diesel engine. The faults can be divided into two categories:

Internal faults There are two internal faults defined in the benchmark. The first one, represented as $\Delta\dot{\theta}_{inc}$, is due to a leak in the hydraulic system and is an additive error inside the pitch control module. The second fault is interpreted as a gain change in the diesel engine, represented as Δk_y.

Sensor faults Sensor faults occur on two of the measured outputs, namely n_m and θ_m. The sensors saturate either at their maximum or minimum output levels. These faults are represented as Δn_{high}, Δn_{low}, $\Delta\theta_{high}$ and $\Delta\theta_{low}$ for the shaft speed and propeller pitch sensors.

The test sequence for the system and time intervals for different fault events are shown in Table 1. The total simulation time, as defined by the benchmark, is 3500s.

If the effects of the internal faults are sufficiently small, then they are accommodated by the inherent robustness of the feedback system. More serious faults can be dealt with by updating the internal model used by the controller, if FDI information is available [3, 5, 6].

Sensor faults are potentially the most difficult to deal with from the point of view of MPC. It might be necessary to adjust the MPC formulation by abandoning the control of the corresponding variable, estimating it from other measurements, replacing the cost function or re-tuning the parameters of the MPC controller.

3.1 Internal faults

In the case of the internal faults, e.g. the hydraulic leak $\Delta\dot{\theta}_{inc}$ and engine gain change Δk_y, no FDI information was assumed. The same MPC controller as in Section 2 was used without modifications.

The simulation results for all the internal and sensor faults are shown in Figure 2. The fault events can clearly be seen as those cases where the original output, without MPC, deviates quite significantly from the set-point.

Compared to the case without MPC, it can be seen that the MPC controller handles the hydraulic leak by appropriately increasing the propeller pitch command $\theta_{com,MPC}$ from $t = 800$s to $t = 1700$s. Without MPC the propeller pitch θ_m drifts away from the set-point, resulting in a slowing down of the ship as can be seen in Figure 3(a), but with MPC it can be seen that this has been corrected. This is due to the inherent integral action of the MPC controller; the MPC controller treats the hydraulic leak as an output disturbance on θ_m that will continue at the same rate into the future.

Note that the duration of the fault in the benchmark is, fortunately, short enough that the control input does not saturate. If saturation occurs the controller will not be able to correct for the fault. However, this might be accounted for by reformulating the control objectives at a higher level. For example, at a higher level it might be more important to control the ship speed than track the propeller pitch correctly. The controller would then try to find a combination of propeller pitch and shaft speed that would track the ship speed, while minimising some efficiency criteria.

The fault related to the diesel engine has the effect of overloading the engine, resulting in engine wear and slowdown. In Figure 2 it can be seen that the fault had very little effect on the tracking of θ_m and n_m, the MPC controller making very little difference. The only effect that the fault had on n_m, was a slight undershoot then overshoot as the sudden change in engine dynamics was "detected" and corrected. However, Figure 3(b) shows how the diesel fault causes a jump in the fuel index at time $t = 3000$s.

This has the effect of decreasing the fuel or propeller efficiency. Including the control of the fuel index explicitly would be necessary in order to accommodate this fault. Based on the results in this section, it is likely that MPC could be a good candidate for such a scheme.

3.2 Sensor Faults

As mentioned above, sensor faults are potentially the most difficult to deal with from the point of view of MPC. Initially no FDI information was assumed and the original controller was applied to the ship model, but with little success. This was due to the sensor faults directly affecting the lower-level control loops. This has the result of moving the actual system further away from the MPC controller's internal model. Therefore, FDI information was assumed in order to deal with this type of fault. It was assumed that the FDI information became available before the next sampling instant and that the nature and magnitude of the sensor fault was available.

The approach implemented, used a Kalman estimator with the two reference signals θ_{ref} and n_{ref} as inputs and the three correct output measurements to estimate
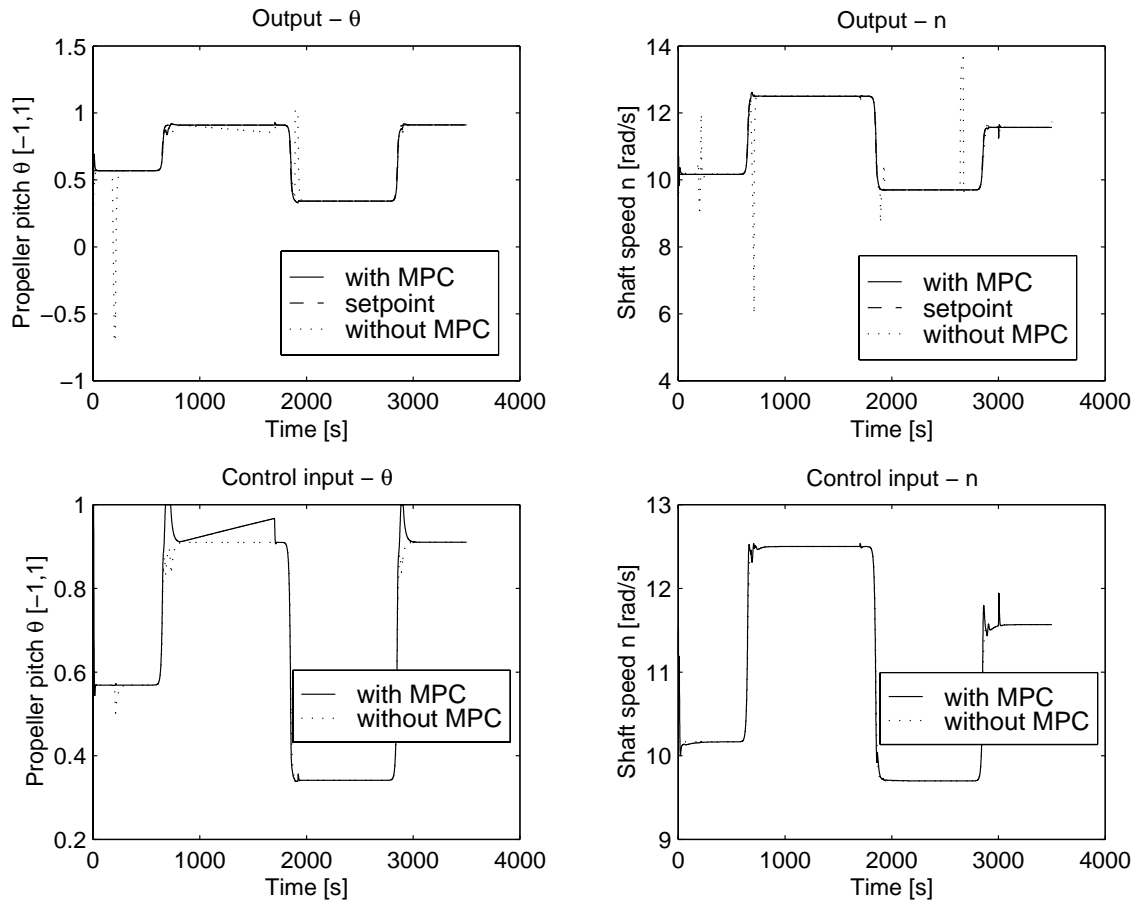
Figure 2: *Performance of controller, simulating the internal and sensor faults.*

the actual value of the faulty measurement. In the MPC controller, the estimated value was then substituted for the faulty output.

The benchmark desired that only re-configuration at the coordinated level take place, without changing any of the lower-level control blocks. Since the faulty measurement is subtracted from the reference signal in the lower-level control loop, the reference signal needs to be adjusted in order to accommodate for this. As the FDI information provides the control system with the nature, and hence the magnitude of the fault, this was exploited and implemented in the coordinated control level. When a sensor fault is detected, the faulty measurement is added to the reference signal. The estimated value of the output is subtracted from the new reference signal and this signal is then fed to the lower-level control loop, resulting in a better control action.

In Figures 2 and 3, it can be seen that the controller drastically improved the performance of the system during the sensor faults. The ship speed and fuel index profiles are also much smoother under MPC control.

One of the key factors involved in implementing MPC on a physical system, is the time required to compute the desired control action, as this could be longer than

the sampling time of the system. The actual MPC controller and Simulink model simulation time ranged from 3000–4000s on a Pentium II processor, compared to the "real-world" time of 3500s in the benchmark. This suggests that MPC could be implemented real-time on an actual ship propulsion system, using currently available computing power.

4 Conclusions

The successful implementation of an MPC controller for the single engine/propeller model of the ship propulsion benchmark has been presented. This paper showed that the MPC controller could be tuned to be robust to the internal faults as described by the benchmark, thereby giving improved performance over the original low-level controllers.

In order to accommodate the sensor faults, FDI information was assumed and a Kalman estimator was used to estimate the faulty outputs. Only re-configuration at the coordinated level was necessary, by appropriately adding the faulty measurement and subtracting the estimated output from the reference signal.

Despite using a single, simplified internal model for the

MPC controller, the controller proved to be robust over the operating range as implemented in the benchmark. Measurement noise, friction torque and external forces were added and the controller still performed well, accommodating all the internal and sensor faults. Due to space restrictions, these results have not been presented here, but can be found in [11].

Future work should investigate the ability of MPC to handle the higher level control objectives during failures, such as speed control and efficiency optimisation.

References

[1] S.A. Heise. *Multivariable Constrained Model Predictive Control*. PhD thesis, University of Cambridge, December 1994.

[2] G. Papageorgiou, M. Huzmezan, K. Glover, and J. Maciejowski. Combined MBPC/$\mathcal{H}_\infty$ autopilot for a civil aircraft. In *Proceedings of American Control Conference*, New Mexico, USA, 1997.

[3] M. Huzmezan. *Theory and Aerospace Applications of Constrained Model Based Predictive Control*. PhD thesis, University of Cambridge, March 1998.

[4] J.M. Maciejowski. Reconfigurable control using constrained optimization. In *Proceedings of European Control Conference*, Brussels, July 1997.

[5] J.M. Maciejowski. Modelling and predictive control: Enabling technologies for reconfiguration. In *Proceedings of IFAC Symposium on System Structure and Control*, Bucharest, October 1997.

[6] J.M. Maciejowski. The implicit daisy-chaining property of constrained predictive control. *Applied Mathematics and Computer Science*, 8(4):101–117, 1998.

[7] C. Rowe. The fault tolerant capabilities of constrained model predictive control. Master's thesis, University of Cambridge, August 1997.

[8] R. Izadi-Zamanabadi and M. Blanke. A ship propulsion system model for fault-tolerant control. Technical report, Department of Control Engineering, Aalborg University, 1998.

[9] D.M. Prett and C.E. Garcia. *Fundamental Process Control*. Butterworth Publishers, 1988.

[10] D.W. Clarke, editor. *Advances in Model-Based Predictive Control*. Oxford University Press, Oxford, 1994.

[11] E.C. Kerrigan. Fault-tolerant control of the COSY ship propulsion benchmark using model predictive control. Technical Report CUED/F-INFENG/TR.336, University of Cambridge, November 1998.
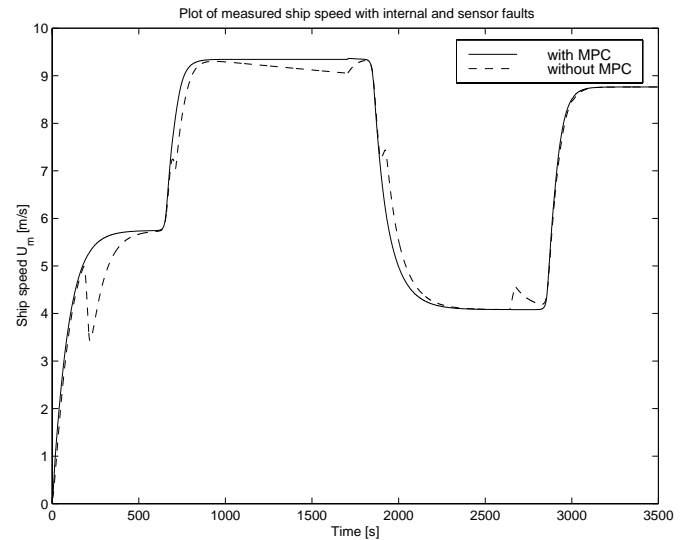
(a) *Ship speed*
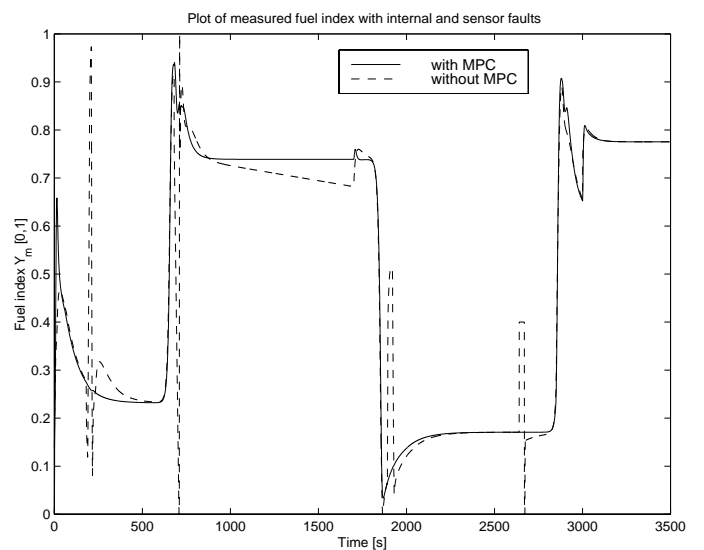
(b) *Fuel index*

Figure 3: *Plots of ship speed and fuel index, simulating the internal and sensor faults.*