1

Approximate Model Predictive Control for Nonlinear Multivariable Systems

Jonas Witt and Herbert Werner
Hamburg University of Technology
Germany

1. Introduction

The control of multi-input multi-output (MIMO) systems is a common problem in practical control scenarios. However in the last two decades, of the advanced control schemes, only linear model predictive control (MPC) was widely used in industrial process control (Maciejowski, 2002). The fundamental common idea behind all MPC techniques is to rely on predictions of a plant model to compute the optimal future control sequence by minimization of an objective function. In the predictive control domain, *Generalized Predictive Control* (GPC) and its derivatives have received special attention. Particularly the ability of GPC to be applied to unstable or time-delayed MIMO systems in a straight forward manner and the low computational demands for static models make it interesting for many different kinds of tasks. However, this method is limited to linear models.

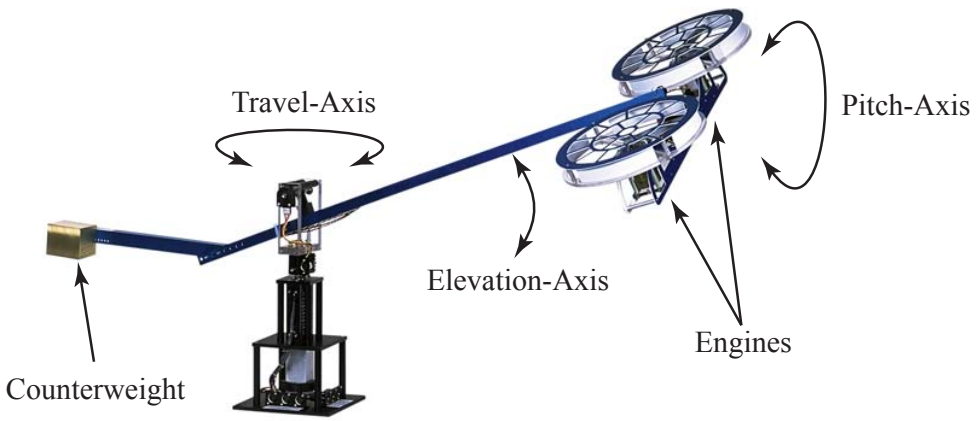


Fig. 1. Quanser 3-DOF Helicopter

If nonlinear dynamics are present in the plant a linear model might not yield sufficient predictions for MPC techniques to function adequately. A related technique that can be applied to nonlinear plants is *Approximate Predictive Control* (APC). It uses an instantaneous linearization of a nonlinear model based on a neural network in each sampling instant. It is similar to GPC in most aspects except that the instantaneous linearization of the neural network yields

an adaptive linear model. Previously this technique has already successfully been applied to a pneumatic servomechanism (Nørgaard et al., 2000) and gas turbine engines (Mu & Rees, 2004), however both only in simulation.
The main challenges in this work were the nonlinear, unstable and comparably fast dynamics of the 3-DOF helicopter by Quanser Inc. (2005) (see figure 1). APC as proposed by Nørgaard et al. (2000) had to be extended to the MIMO case and model parameter filtering was proposed to achieve the desired control and disturbance rejection performance.
This chapter covers the whole design process from nonlinear MIMO system identification based on an artificial neural network (ANN) in section 2 to controller design and presentation of enhancements in section 3. Finally the results with the real 3-DOF helicopter system are presented in section 4. On the way pitfalls are analyzed and practical application hints are given.

2. System Identification

The correct identification of a model is of high importance for any MPC method, so special attention has to be paid to this part of controller design. The success of the identification will determine the performance of the final controlled system directly or even whether the system is stable at all.
Basically there are a few points one has to bear in mind during the experiment design (Ljung, 1999):

- The sampling rate should be chosen appropriately.
- The experimental conditions should be close to the situation for which the model is going to be used. Especially for MIMO systems this plays an important role as this may be nontrivial.
- The identification signal should be sufficiently rich to excite all modes of the system. For nonlinear systems not only the frequency spectrum but also the excitation of different amplitudes should be sufficient.
- Periodic inputs have the advantage that they reduce the influence of noise on the output signal but increase the experiment length.

The following sections guide through the full process of the MIMO identification by means of the practical experiences with the helicopter model.

2.1 Excitation Signal

The type of the excitation signal plays an important role as it should exhibit a few properties which affect the outcome essentially. Generally the input signal should be persistently exciting of at least twice the system order. There are many different types of input signals which are not covered here (see Ljung (1999) for further reading). Despite the desirable optimal Crest factor, for nonlinear system identification binary signals are not an option due to the lack of excitation of different amplitudes. For this work an excitation signal comprised of independent multi-sine signals as described in (Evan et al., 2000) was designed. This is explored in the following section.

2.1.1 Assembling of Multisine Signals

A multisine is basically a sum of sinusoids:

$$u(t) = \sum_{k=1}^{n_s} A_k cos(\omega_k t + \phi_k)$$

where n_s is the number of present frequencies. This parameter should be large enough to guarantee persistent excitation.
A favourable attribute of multisine signals is that the spectrum can be determined directly. By this property it is possible to just include the frequency ranges that excite the system which is done by splitting the spectrum in a low (or main) and a high frequency band. As a rule of thumb one should choose the upper limit of the main frequency band ω_c around the system bandwidth ω_b, since choosing ω_c too low may result in unexcited modes, while $\omega_c \gg \omega_b$ does not yield additional information (Ljung, 1999). In a relay feedback experiment the bandwidth of the helicopters pitch axis was measured to be $f_b \approx 0.67Hz$. As one can see in figure 2 the upper limit of the main frequency band $f_c = \omega_c/2\pi = 1.5Hz$ was chosen about twice as large but the higher frequencies from ω_c up to the Nyquist frequency ω_n are not entirely absent. This serves the purpose of making the mathematical model resistant to high frequency noise as the real system will typically not react to this high frequency band.

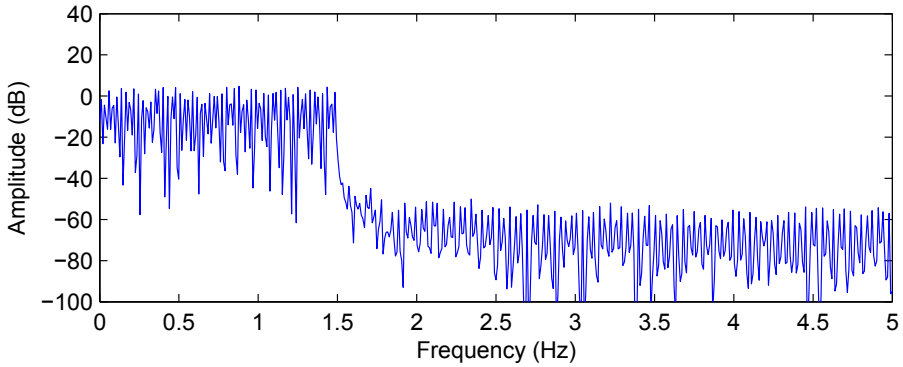


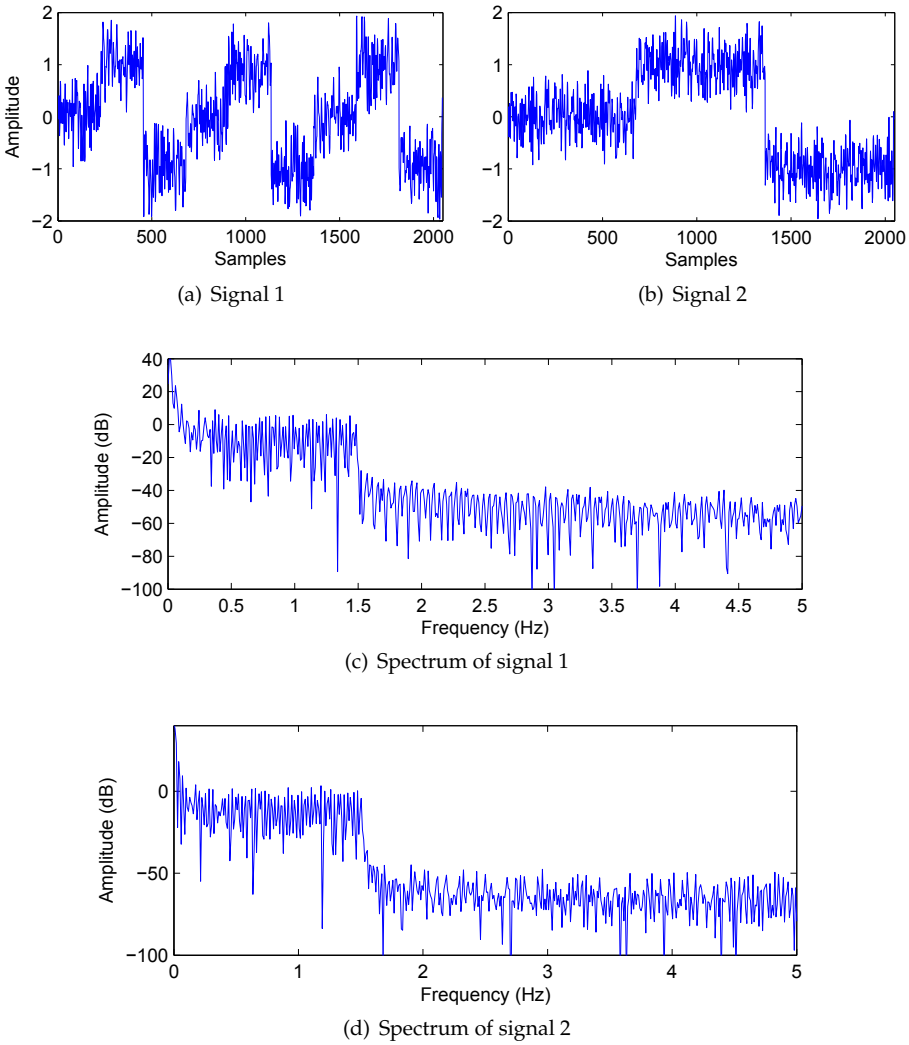
Fig. 2. Spectrum of the multisine excitation signal for the helicopter

2.1.2 Periodic Signals

To reduce the influence of noise present in the output signal of the plant, taking an integer number of periods of the input signal can be considered. If K periods of the input signal are taken, the signal to noise ratio is improved by this factor K. A drawback of periodic inputs is that they generally can not inject as much excitation into the system over a given time span as non-periodic inputs, since a signal of length N can at most excite a system of order N (Ljung, 1999). But as a periodic signal of length $N = KM$ consists of K periods of length M it has the same level of excitation as one period.
In the case of the helicopter three signal periods were chosen, as this proved to give consistent results for the present noise level.

2.1.3 MIMO Considerations

For MIMO systems the design of the input signal is a bit more complex, as there may be cross couplings between the different inputs which drive the outputs out of desired limits or the signal does not excite all modes sufficiently. The design process of the identification signal involves the consideration of system specifics and can not be generalized.



(a) Signal 1

(b) Signal 2

(c) Spectrum of signal 1

(d) Spectrum of signal 2

Fig. 3. MIMO signals with appropriate setpoints

In the case of the helicopter three axes need to be excited in all modes. A first attempt was to directly apply multisine signals to both inputs. For this attempt both inputs were limited

to low amplitudes though, as coincidental add-up effects quickly drove the system out of the operating bounds. Naturally this yielded bad models that did not resemble the actual plant very well.
A way to drive a MIMO system to different operating states is the use of setpoints that are added to the multisine signal. This enables the selective identification of certain modes of the system. At the same time this can be used as a means to keep the outputs inside of valid operating bounds, since the amplitude of the multisine signal can be chosen to be much lower than without setpoints. This enables much safer operation during the experiment, as the energy of the random signal can be reduced. Of course one has to keep in mind that the actual excitation signals amplitude has to be as large as possible to assure maximal excitation around each setpoint.
The spectrum of a multisine signal with additive setpoints does not differ much from the original multisine (figure 2) as can be seen in figure 3. The only difference is a peak in the low frequency band and a general small lifting in the upper band. Both signals are composed of multisines of same spectrum with unit variance and additive setpoints in the range of $[-1, 1]$. This assures overlapping amplitude ranges, which is desirable for a consistent model.

2.2 Closed Loop Identification

In recent years the interest in closed loop identification has generally risen, due to its importance for practical system identification. In many industrial processes existing control loops cannot be switched off during system identification for safety reasons or process restrictions. Likewise when dealing with unstable systems every experiment setup must involve stabilizing control loops to keep the output in a valid operating range. Another advantage of models computed from closed loop data is their better approximation of the behavior of a process under feedback which is important for successful controller design (Pico & Martinez, 2002). There are a few different approaches to closed loop identification of which the two most general are covered here:

- Direct Approach: ignore the presence of the feedback and directly identify the plant by plant input and output data. This has the advantage that no knowledge about the type of control feedback or even linearity of the controller is required.
- Indirect Approach: identify the closed loop and obtain the open loop model by deconvolution if possible. Obtaining the open loop model is only possible if the controller is known and both the closed loop plant model and the controller are linear.

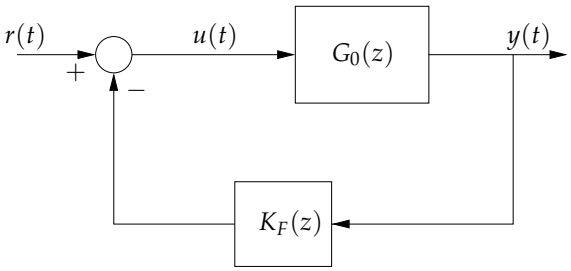


Fig. 4. Closed loop setup for identification

This section is limited to the techniques and practical experiences of the identification of the helicopter model and does not cover the whole theory of closed loop identification. For further information on this matter see Pico & Martinez (2002) or Ljung (1999).

2.2.1 Direct Identification

This is the natural approach to closed loop identification as it is similar to open loop identification if one keeps some fallacies in mind. In general this method applies a straightforward identification by taking the raw data from plant input $u(t)$ and plant output $y(t)$ and thus computes the model as if in open loop. Figure 4 shows the basic setup of the experiment.
A few general statements can be made about the direct identification in closed loop (Pico & Martinez, 2002):

- The experiment is informative if $r(t)$ is persistently exciting.
- Even if $r(t)$ is not a rich signal, the experiment can be informative if the system is driven by the output-noise and the feedback mechanism is of adequate structure avoiding a linear dependency between $y(t)$ and $u(t)$.
- Problems can arise if the amplitude of $r(t)$ is small in comparison to $u(t)$ and the feedback mechanism is approximately linear, i.e. $u(t) \approx -K_F(z)y(t)$.
- The direct approach can be problematic if the open loop plant is unstable, since the spectrum of $u(t)$ may be altered in a suboptimal way.

So if it is possible to use a rich signal for $r(t)$, the experiment is informative. But as this may not always be the case in a practical scenario, informative experiments can still be achieved by choosing an appropriate controller. Generally if $r(t)$ has a very low amplitude in comparison to $u(t)$ or if $r(t)$ is not a rich signal, problems arise if a dependency like $u(t) \approx -K_F(z)y(t)$ exists. But this can be avoided by choosing high order, time varying or nonlinear feedback mechanisms. For further information about appropriate controllers see (Pico & Martinez, 2002).

Issues with Unstable Systems

In the case of the helicopter the multisine from section 2.1 was used which exhibits a high level of excitation, thus a simple proportional derivative controller (PD controller) could be used for stabilization since the main source of exitation is the reference signal $r(t)$ in this case. Figure 5 shows the spectrum of the input signal $u(t)$ that was recorded during a SISO experiment to identify the helicopters pitch axis. The elevation axis was controlled separately by a PID controller. As one can see, the spectrum has changed significantly if compared to the original spectrum in figure 2. The low frequencies are heavily damped which comes naturally for an unstable plant like the helicopters pitch axis, as a constant input would drive the system to infinity. For identification though this is unfavourable, since the low frequencies are not sufficiently excited although these frequencies lie in the normal operating band.
More problems arise during validation, since with the direct approach unstable plants yield unstable models directly and validating these is tedious. The problem with validating unstable models is that the validation is usually open loop, as the recorded input sequence is applied to the model and the output is compared to the recorded actual output. Errors are not compensated and thus add up as there are no control loops during validation so the model response may ascend to infinity - even with a decent model. A better way to verify the quality of an unstable model is to look at a k-step-ahead prediction, because errors do not have as much time to add up.

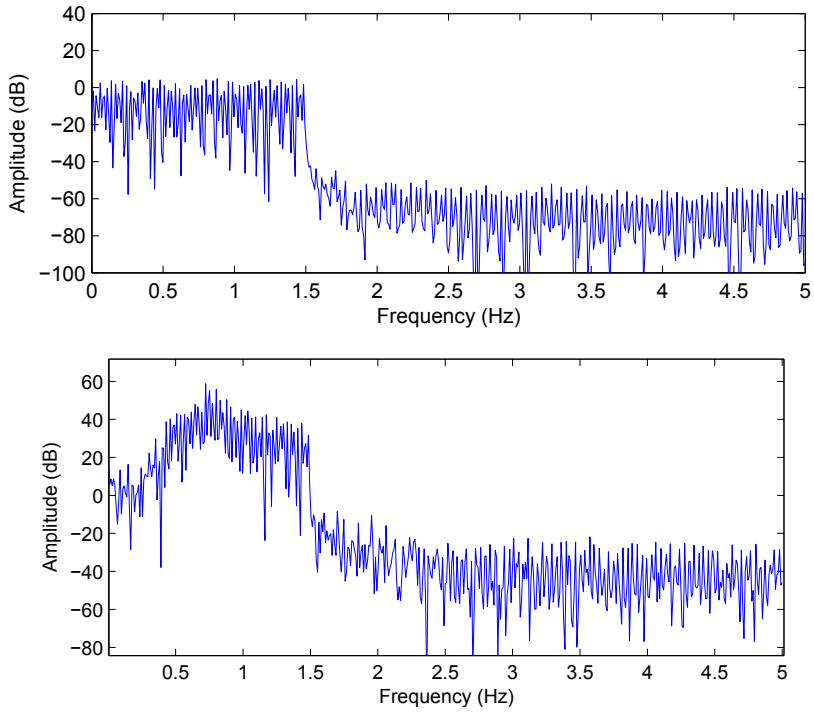


Fig. 5. Spectrum of the plant input $u(t)$ in closed loop excited by multisine signal (lower picture). Note that this signal is scaled in comparison to the original spectrum of the reference $r(t)$ (upper picture).

A comparison of the direct and the indirect approach by example of the helicopter's unstable pitch axis is presented in section 2.2.3.

2.2.2 Indirect Identification

The basic concept of indirect identification is to identify the closed loop as a whole and validate this model of the closed loop in the first place. Consecutive steps may include a deconvolution of the plant and controller to obtain the open loop model. This requires the regulator and the plant model to be both linear, and of course the regulator transfer function must be known. So in the case of nonlinear system identification these steps are not possible, even if the controller is linear and known, because in this case the equation can usually not be solved for $G_0(z)$ analytically.

In contrast to the direct approach the indirect approach avoids any alteration of the input signal's spectrum since the input to the closed loop directly is the reference signal $r(t)$ which has no dependence on other signals. This has the advantage, that even if $r(t)$ is small compared to $u(t)$ and the feedback mechanism is of a simple linear kind, an informative experiment is still achieved as long as $r(t)$ is persistently exciting (which the multisine signal from section 2.1 ensures). Another advantage is that unstable systems can be handled intuitively as the resulting model is stable. This eliminates the problems discussed in 2.2.1 in the validation phase

since the closed loop model does not have unstable poles as the open loop model would. A drawback of this method is, that the feedback mechanism increases the model order since it is identified along with the actual open loop model.
For the case of a linear known controller and a linear closed loop model, the open loop model can be obtained by deconvolution as was already mentioned. The closed loop transfer function corresponding to figure 4 is:

$$G_{cl}(z) = \frac{G_0(z)}{1 + G_0(z)K_F(z)}. \quad (1)$$

Solving for $G_0(z)$ yields:

$$G_0(z) = \frac{G_{cl}(z)}{1 - G_{cl}(z)K_F(z)}, \quad (2)$$

which is the final formula for obtaining the open loop model $G_0(z)$. So if either $G_{cl}(z)$ or $K_F(z)$ are nonlinear both formulas cannot be applied and a deconvolution is not possible. However for the linear case it will be shown that a controller design for the closed loop model $G_{cl}(z)$ can yield exactly the same overall system dynamics as a controller design for the open loop model $G_0(z)$. For control strategies that utilize linearizations of a nonlinear model like APC, this similarly implies that the direct use of a (in this case nonlinear) closed loop model has no adverse effects on the final performance.

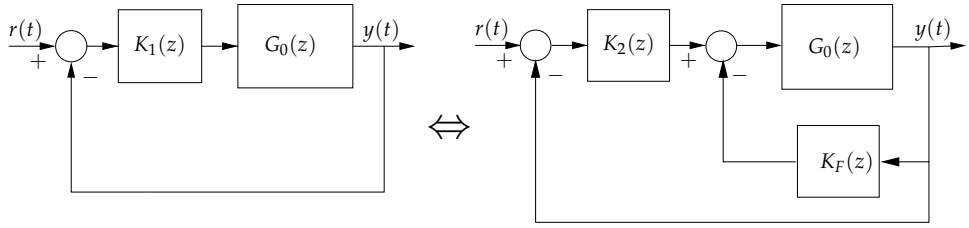


Fig. 6. Controller Setup for open loop and closed loop models

Theorem. *Given the closed loop system $G_{cl}(z)$ consisting of the open loop plant $G_0(z)$ and the controller $K_F(z)$, a controller $K_2(z)$ can be found that transforms the system to an equivalent system consisting of an arbitrary controller $K_1(z)$ that is applied to the plant $G_0(z)$ directly.*

Proof. The two system setups are depicted in figure 6. The transfer function of the left system is:

$$G_1(z) = \frac{K_1(z)G_0(z)}{1 + K_1(z)G_0(z)}$$

while the right system has the transfer-function:

$$\begin{aligned} G_2(z) &= \frac{K_2(z)G_{cl}(z)}{1 + K_2(z)G_{cl}(z)} \\ &= \frac{K_2(z)\frac{G_0(z)}{1+G_0(z)K_F(z)}}{1 + K_2(z)\frac{G_0(z)}{1+G_0(z)K_F(z)}} \\ &= \frac{K_2(z)G_0(z)}{1 + (K_F(z) + K_2(z))G_0(z)}. \end{aligned}$$

Now it has to be proved that there exists a controller $K_2(z)$ that transforms $G_2(z)$ to $G_1(z)$ for any given $K_1(z)$. It is clear that the system $G_2(z)$ with the $K_F(z)$ feedback controller can achieve exactly the same performance as the $G_1(z)$ system if this is the case.

$$\begin{aligned} \frac{K_1(z)G_0(z)}{1+K_1(z)G_0(z)} &= \frac{K_2(z)G_0(z)}{1+(K_F(z)+K_2(z))G_0(z)} \\ K_1(z)G_0(z)+K_1(z)K_F(z)G_0(z)G_0(z) &= K_2(z)G_0(z) \\ K_2(z) &= K_1(z)+K_1(z)K_F(z)G_0(z) \\ &= K_1(z)(1+K_F(z)G_0(z)) \end{aligned}$$

2.2.3 Indirect vs. Direct Approach

The quality of the models heavily depends on the experiment setup and the identification approach chosen. This is valid even more for the identification of unstable systems. This will be shown in the following with the example of the SISO identification of the helicopter's pitch axis. Consider the experiment setup from figure 4. As discussed in section 2.1 three periods of a multisine signal with spectrum as in figure 2 are applied to the reference input $r(t)$. The feedback controller $K_F(z)$ is a hand tuned PD controller. The input data for the identification process are $r(t)$ for the indirect approach and $u(t)$ for the direct approach respectively, the output data is $y(t)$ in both cases. The spectrum of $u(t)$ is shown in figure 5 (the spectrum of $r(t)$ is just the one of the multisine in figure 2). To obtain a fair comparison of the approaches the open loop model is computed for both (for the indirect approach (2) is used to compute $G_0(z)$). Since direct validation of unstable models is not very meaningful in most cases, a stabilizing controller is added for the simulation. Here, the natural choice is the same PD feedback controller as used with the real helicopter. The comparison of the model outputs to the original helicopter output is shown in figure 7.

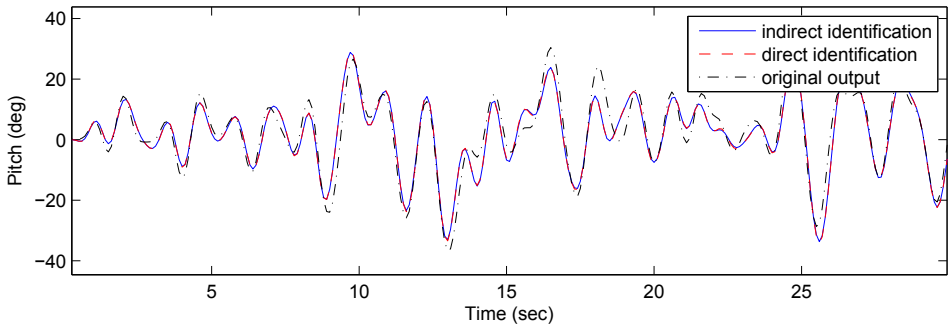


Fig. 7. Simulated closed loop response of models from direct and indirect approach compared to experimental measurement

Judging from the predicted outputs of both models they seem almost identical, as it is even difficult to distinguish between both model outputs. Both are not perfectly tracking the real output but it seems that decent models have been acquired. In figure 8 the bode plots of both

open loop models are shown and this illustrates that the models are not as similar as it had seemed in the closed loop validation, since the static gain differs in a few orders of magnitude. The high frequency part of the plot is comparable, though.

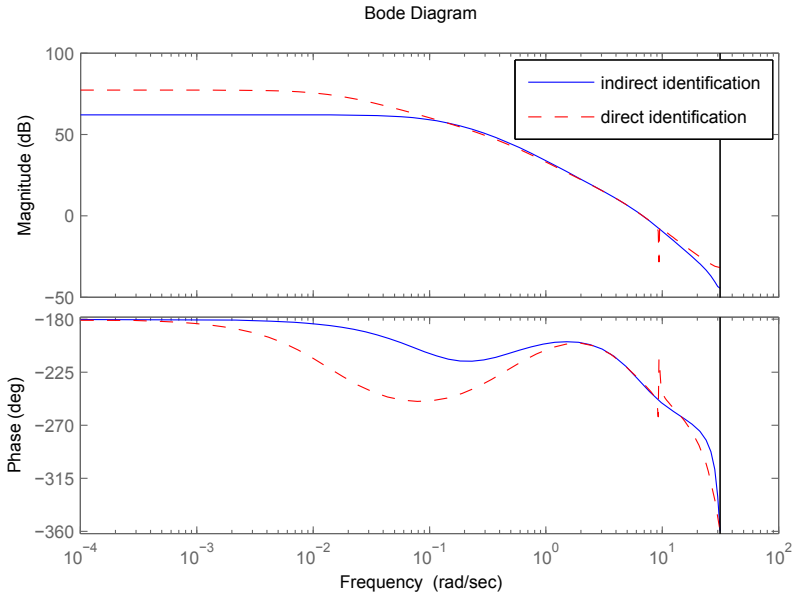


Fig. 8. Bode plot of models from direct and indirect approach

This correlates with the spectra of $r(t)$ and $u(t)$ since they are similar for higher frequencies, too. Taking a close look at the step responses in figure 9 of both stabilized open loop models, although looking similar over all, the model of the direct approach shows a small oscillation for a long time period which seems negligible at first.
To see the consequences of the differences in the models they have to be used in a controller design process and tested on the actual plant. Figure 10 shows the responses of the helicopters pitch axis to a rectangular reference stabilized by two LQG controllers. Both controllers were designed with the same parameters differing only in the employed plant models.
The controller designed with the model of the indirect approach performs well and is also very robust to manual disturbances. In contrast the LQG controller designed with the model of the direct approach even establishes a static oscillation indicating that the model is not a good representation of the real plant. During all identification approaches the indirect method performed superiorly, which led to the conclusion that the direct approach is not ideal for our setup.

2.3 Linear Identification Results

With the bad results for the direct identification in the SISO case the MIMO identification was attempted with the indirect approach only.
The output of a MIMO model computed from a data set with usage of setpoints as described in 2.1.3 is shown in figure 11. The model used for the output in figure 11 is a state space model of order 16 computed with the prediction error/maximum likelihood (PEM) method of the

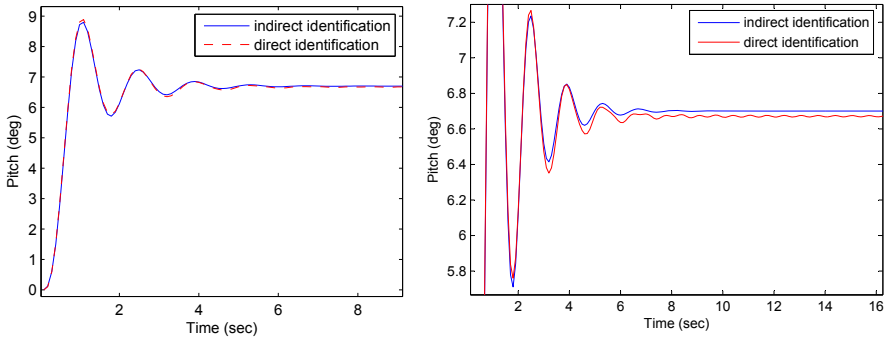


Fig. 9. Simulated step response of models from direct and indirect approach (plotted at different scales)

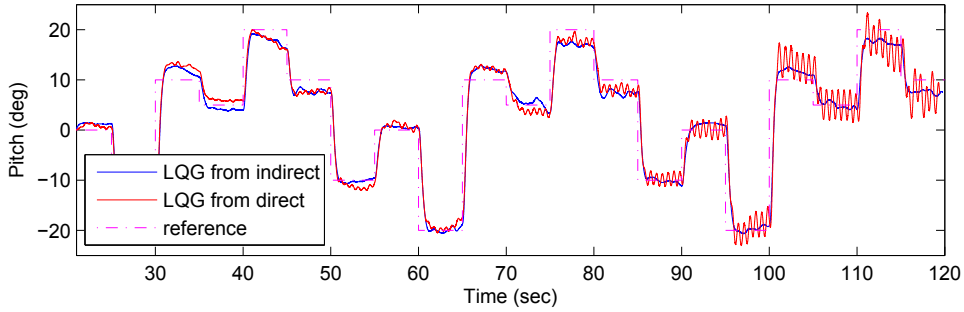


Fig. 10. Experimental results of controllers tracking a rectangular reference on the pitch axis. The LQG controller design was done with models from the direct and indirect approach respectively.

identification toolbox in Matlab. This method uses an iterative search starting at the result of the subspace-method. Other methods like MIMO ARX or directly the sub-space method also yielded good results.
From the model output it can be seen that the characteristics of the model seem to resemble the real ones correctly. During more dynamic maneuvers a discrepancy between the measurement and the prediction becomes visible, though.

2.4 Neural Networks for System Identification

Opposed to the common linear plant models with widely spread structures like ARX (AutoRegressive with eXogenous input), ARMAX (AutoRegressive Moving Average with eXogenous input) or state space models the use of neural networks for system identification is a relatively new approach. Traditionally the identification of models with neural networks falls in the category of black box modelling as typically very few information about the system can be incorporated in the process of identification. Neural networks as they are used most often are very general approximators which can be trained to resemble any given function (given that the network complexity is sufficient). Many different approaches and network structures

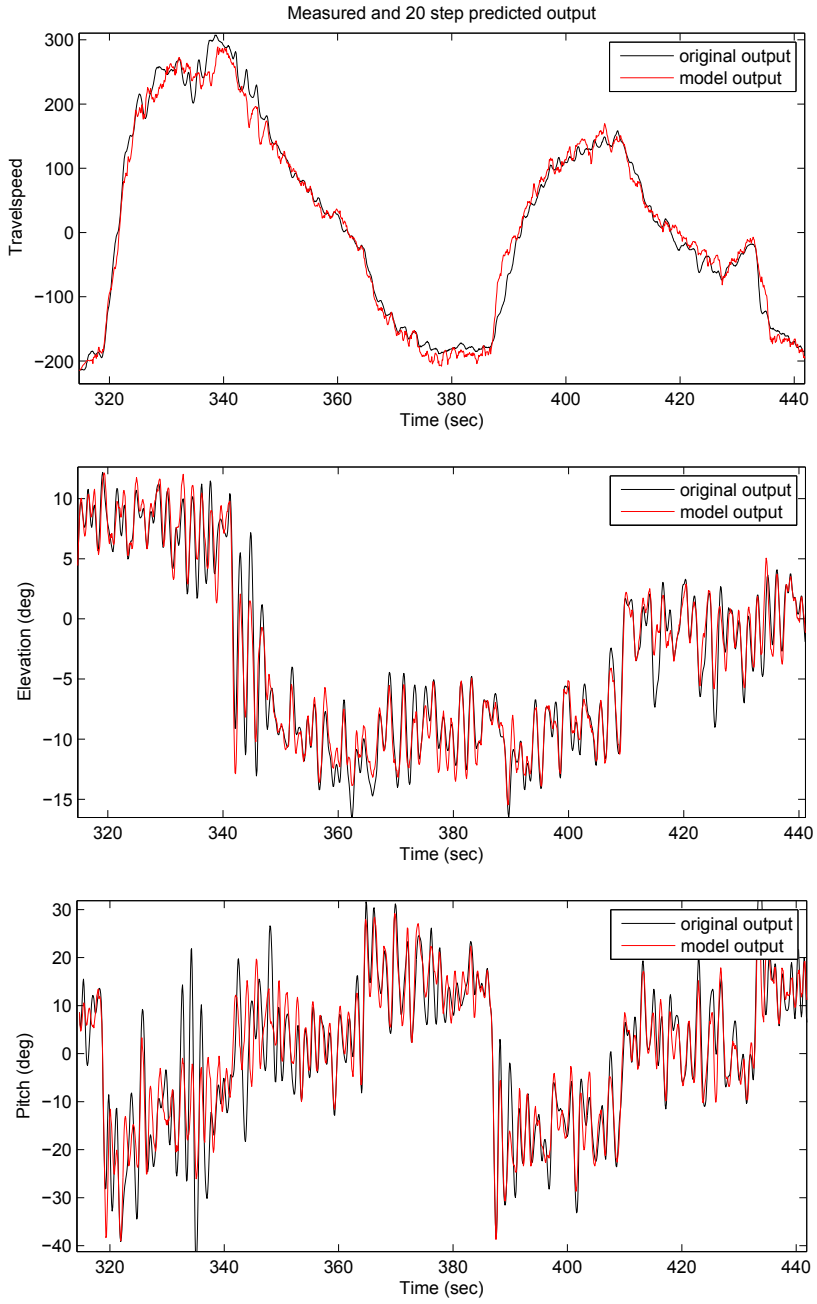


Fig. 11. 20-step ahead prediction output of the linear model for a validation data set

exist. For an introduction to the field of neural networks the reader is referred to Engelbrecht (2002). The common structures and specifics of neural networks for system identification are examined in Nørgaard et al. (2000).

2.4.1 Network Structure

The network that was chosen as nonlinear identification structure in this work is of NNARX format (Neural Network ARX, corresponding to the linear ARX structure), as depicted by figure 12. It is comprised of a multilayer perceptron network with one hidden layer of sigmoid units (or tanh units which are similar) and linear output units. In particular this network structure has been proven to have a universal approximation capability (Hornik et al., 1989). In practice this is not very relevant knowledge though, since no statement about the required number of hidden layer units is made. Concerning the total number of neurons it may still be advantageous to introduce more network layers or to introduce higher order neurons like product units than having one big hidden layer.

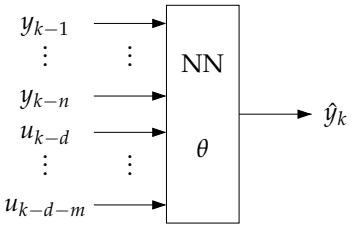


Fig. 12. SISO NNARX model structure

The prediction function of a general two-layer network with tanh hidden layer and linear output units at time k of output l is

$$\hat{y}_l(k) = \sum_{j=1}^{s^1} w_{lj}^2 \tanh\left(\sum_{i=1}^{r} w_{ji}^1 \varphi_i(k) + w_{j0}^1\right) + w_{l0}^2 \tag{3}$$

where w_{ji}^1 and w_{j0}^1 are the weights and biases of the hidden layer, w_{lj}^2 and w_{l0}^2 are the weights and biases of the output layer respectively, $\varphi_i(k)$ is the ith entry of the network input vector (regression vector) at time k which contains past inputs and outputs in the case of the NNARX structure. The choice of an appropriate hidden layer structure and input vector are of great importance for satisfactory prediction performance. Usually this decision is not obvious and has to be determined empirically. For this work a brute-force approach was chosen, to systematically explore different lag space and hidden layer setups, as illustrated in figure 13.
From the linear system identification can be concluded that significant parts of the dynamics can be described by linear equations approximately. This knowledge can pay off during the identification using neural networks. If only sigmoid units are used in the hidden layer the network is not able to learn linear dynamics directly. It can merely approximate the linear behavior which would be wasteful. Consequently in this case it is beneficial to introduce linear neurons to the hidden layer. The benefits are twofold as training speed is greatly improved when using linear units (faster convergence) and the linear behavior can be learned "natively". Since one linear neuron in the hidden layer can represent a whole difference equation for an output the number of linear neurons should not exceed the number of system outputs.

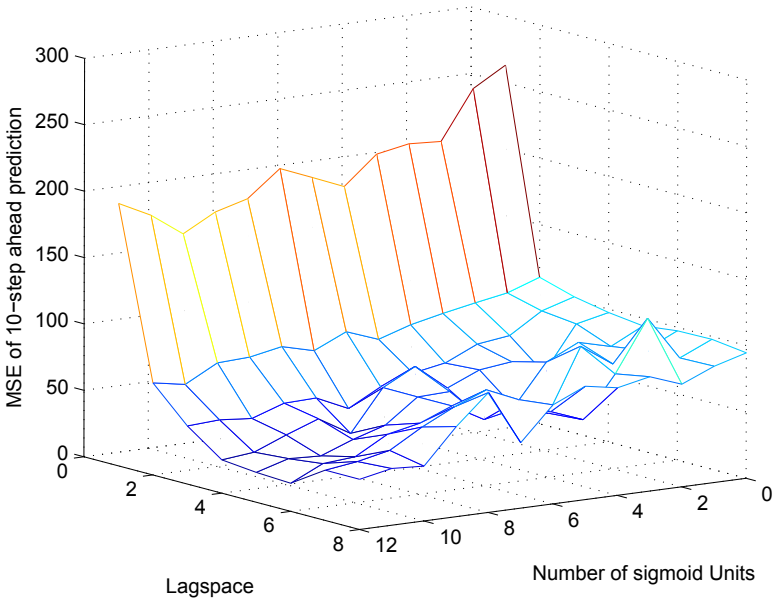


Fig. 13. Comparison of network structures according to their MSE of the 10-step ahead prediction using a validation data set (all networks include three linear units in the hidden layer). Each data point reresents the best candidate network of 10 independent trainings.

The final structure that was chosen according to the results depicted by figure 13 includes three linear and twelve sigmoid units in the hidden layer with a lag space of six for both inputs and the three outputs. For this network accordingly $((2+3)\cdot 6+1)\cdot(12+3)+(12+3+1)\cdot 3 = 513$ weights had to be optimized.

2.4.2 Instantaneous Linearization

To implement APC, linearized MIMO-ARX models have to be extracted from the nonlinear NNARX model in each sampling instant. The coefficients of a linearized model can be obtained by the partial derivative of each output with respect to each input (Nørgaard et al., 2000). Applying the chain rule to (3) yields

$$\frac{\partial \hat{y}_l(k)}{\partial \varphi_i(k)} = \sum_{j=1}^{s^1} w_{lj}^2 w_{ji}^1 \left[1 - \tanh^2\left(\sum_{i=1}^{r} w_{ji}^1 \varphi_i(k) + w_{j0}^1\right)\right] \tag{4}$$

for tanh units in the hidden layer. For linear hidden layer units in both the input and the output layer one yields

$$\frac{\partial \hat{y}_l(k)}{\partial \varphi_i(k)} = \sum_{j=1}^{s^1} w_{lj}^2 w_{ji}^1 \,. \tag{5}$$

2.4.3 Network Training

All networks were trained with *Levenberg Marquardt Backpropagation* (Hagan & Menhaj, 1994). Due to the monotonic properties of linear and sigmoid units, networks using only these unit types have the inherent tendency to have only few local minima, which is beneficial for local optimization algorithms like backpropagation. The size of the final network (513 weights) that was used in this work even makes global optimization techniques like *Particle Swarm Optimization* or *Genetic Algorithms* infeasible. Consequently for a network of the presented size, higher order units such as product units cannot be incorporated due to the increased amount of local minima, requiring global optimization techniques (Ismail & Engelbrecht, 2000).
But also with only sigmoid units, based on the possibility of backpropagation getting stuck in local minima, always a set of at least 10 networks with random initial parameters were trained. To minimize overfitting a *weight decay* of $D = 0.07$ was used. The concept of regularization to avoid overfitting using a weight decay term in the cost function is thoroughly explored by Nørgaard et al. (2000).

2.5 Nonlinear Identification Results

For the nonlinear identification the same excitation signal and indirect measurement setup was used as for the linear identification. Thus a stabilized closed-loop model was aquired. The controller that was inevitably identified along with the unstable plant model cannot be removed from the model analytically. In section 2.2.2 we showed that the stabilizing controller will not hinder the final control performance in the case of APC, though.
The prediction of the finally chosen network with a validation data set is depicted in figure 14. If one compares the neural network prediction with the prediction of the linear model in figure 11 it is obvious, that the introduction of nonlinear neurons benefited the prediction accuracy. This is underlined by figure 13 also visualizing a declining prediction error for increasing sigmoid unit numbers. Whether the improvements in the model can be transferred to an improved controller yet remains to be seen, though.

2.6 Conclusion

This section demonstrated successful experiment design for an unstable nonlinear MIMO system and showed some pitfalls that may impede effective identification. The main approaches to closed loop identification have been presented and compared by means of the helicopters unstable pitch axis. It was shown that the identification of unstable systems can be just as successful as for stable systems if the presented issues are kept in mind. Both linear and nonlinear identifications can be regarded as successful, although the nonlinear predictions outperform the linear ones.

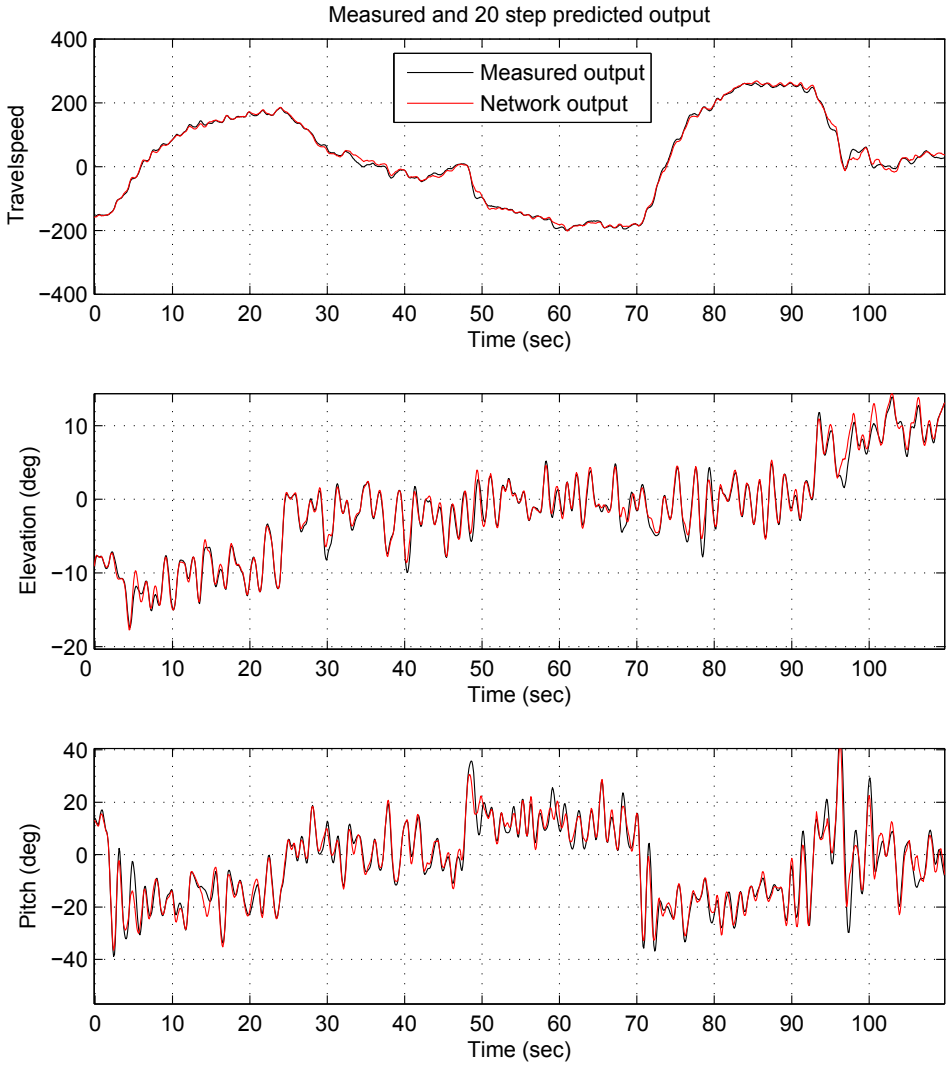


Fig. 14. 20-step ahead prediction output of the best network for a validation data set

3. Approximate Model Predictive Control

The predictive controller that is discussed in this chapter is a nonlinear adaptation of the popular *Generalized Predictive Control* (GPC), proposed in (Clarke et al., 1987a;b). *Approximate Predictive Control* (APC) as proposed by Nørgaard et al. (2000) uses the GPC principle on instantaneous linearizations of a neural network model. Although presented as a single-input single-output (SISO) algorithm, its extension to the multi-input multi-output (MIMO) case with MIMO-GPC (Camacho & Borbons, 1999) is straightforward. The scheme is visualized in figure 15.

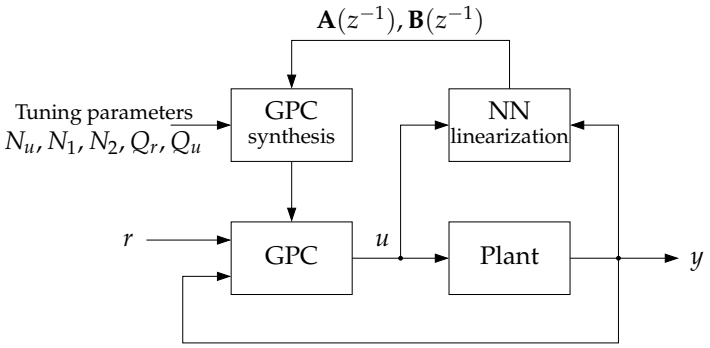


Fig. 15. Approximate predictive control scheme

The linearized model that is extracted from the neural network at each time step (as described in section 2.4.2) is used for the computation of the optimal future control sequence according to the objective function:

$$\begin{aligned} J(k) &= \sum_{i=N_1}^{N_2} \Big(r(k+i) - \hat{y}(k+i)\Big)^T Q_r \Big(r(k+i) - \hat{y}(k+i)\Big) \\ &+ \sum_{i=1}^{N_u} \Delta u^T(k+i-1)\; Q_u\; \Delta u(k+i-1) \end{aligned} \tag{6}$$

where N_1 and N_2 are the two prediction horizons which determine how many future samples the objective function considers for minimization and N_u denotes the length of the control sequence that is computed. As common in most MPC methods a *receding horizon strategy* is used and thus only the first control signal that is computed is actually applied to the plant to achieve loop closure.

A nice property of quadratic cost functions is that a closed-form solution exists, enabling its application to fast processes under hard realtime constraints (since the execution time remains constant). If constraints are added an iterative optimization method has to be used in either way, though. The derivation of MIMO-GPC is given in the following section for the sake of completeness.

3.1 Generalized Predictive Control for MIMO Systems

In GPC, usually a modified ARX (AutoRegressive with eXogenous input) or ARMAX (AutoRegressive Moving Average with eXogenous input) structure is used. In this work a structure like

$$\mathbf{A}(z^{-1})y(k) = \mathbf{B}(z^{-1})u(k) + \frac{1}{\Delta}e(k) \tag{7}$$

is used for simplicity, with $\Delta = 1 - z^{-1}$ where $y(k)$ and $u(k)$ are the output and control sequence of the plant and $e(k)$ is zero mean white noise. This structure is called ARIX and basically extends the ARX structure by integrated noise. It has a high relevance for practical applications as the coloring polynomials for an integrated ARMAX structure are very difficult to estimate with sufficient accuracy, especially for MIMO systems (Camacho & Borbons, 1999). The integrated noise term is introduced to eliminate the effects of step disturbances.
For an n-output, m-input MIMO system $\mathbf{A}(z^{-1})$ is an $n \times n$ monic polynomial matrix and $\mathbf{B}(z^{-1})$ is an $n \times m$ polynomial matrix defined as:

$$\begin{aligned} \mathbf{A}(z^{-1}) &= I_{n\times n} + A_1 z^{-1} + A_2 z^{-2} + ... + A_{n_a} z^{-n_a} \\ \mathbf{B}(z^{-1}) &= B_0 + B_1 z^{-1} + B_2 z^{-2} + ... + B_{n_b} z^{-n_b} \end{aligned}$$

The output $y(k)$ and noise $e(k)$ are $n \times 1$-vectors and the input $u(k)$ is an $m \times 1$-vector for the MIMO case. Looking at the cost function from (6) one can see that it is already in a MIMO compatible form if the weighting matrices Q_r and Q_u are of dimensions $n \times n$ and $m \times m$ respectively. The SISO case can easily be deduced from the MIMO equations by inserting $n = m = 1$ where $\mathbf{A}(z^{-1})$ and $\mathbf{B}(z^{-1})$ degenerate to polynomials and $y(k)$, $u(k)$ and $e(k)$ become scalars.

To predict future outputs the following Diophantine equation needs to be solved:

$$I_{n\times n} = \mathbf{E}_j(z^{-1})(\mathbf{A}(z^{-1})\Delta) + z^{-j}\mathbf{F}_j(z^{-1}) \tag{8}$$

where $\mathbf{E}_j(z^{-1})$ and $\mathbf{F}_j(z^{-1})$ are both unique polynomial matrices of order $j-1$ and n_a respectively. This special Diophantine equation with $I_{n\times n}$ on the left hand side is called Bizout identity, which is usually solved by recursion (see Camacho & Borbons (1999) for the recursive solution). The solution to the Bizout identity needs to be found for every future sampling point that is to be evaluated by the cost function. Thus $N_2 - N_1 + 1$ polynomial matrices $\mathbf{E}_j(z^{-1})$ and $\mathbf{F}_j(z^{-1})$ have to be computed. To yield the j step ahead predictor, (7) is multiplied by $\mathbf{E}_j(z^{-1})\Delta z^j$:

$$\mathbf{E}_j(z^{-1})\Delta\mathbf{A}(z^{-1})y(k+j) = \mathbf{E}_j(z^{-1})\mathbf{B}(z^{-1})\Delta u(k+j-1) + \mathbf{E}_j(z^{-1})e(k+j) \tag{9}$$

which by using equation 8 can be transformed into:

$$y(k+j) = \underbrace{\mathbf{E}_j(z^{-1})\mathbf{B}(z^{-1})\Delta u(k+j-1)}_{past\ and\ future\ inputs} + \underbrace{\mathbf{F}_j(z^{-1})y(k)}_{free\ response} + \underbrace{\mathbf{E}_j(z^{-1})e(k+j)}_{future\ noise} \tag{10}$$

Since the future noise term is unknown the best prediction is yielded by the expectation value of the noise which is zero for zero mean white noise. Thus the expected value for $y(k+j)$ is:

$$\hat{y}(k+j|k) = \mathbf{E}_j(z^{-1})\mathbf{B}(z^{-1})\Delta u(k+j-1) + \mathbf{F}_j(z^{-1})y(k) \tag{11}$$

The term $\mathbf{E}_j(z^{-1})\mathbf{B}(z^{-1})$ can be merged into the new polynomial matrix $\mathbf{G}_j(z^{-1})$:

$$\mathbf{G}_j(z^{-1}) = G_0 + G_1 z^{-1} + \ldots + G_{j-1} z^{-(j-1)} + (G_j)_j z^{-j} + \ldots + (G_{j-1+n_b})_j z^{-(j-1+n_b)}$$

where $(G_{j+1})_j$ is the $(j+1)$th coefficient of $\mathbf{G}_j(z^{-1})$ and n_b is the order of $\mathbf{B}(z^{-1})$. So the coefficients up to $(j-1)$ are the same for all $\mathbf{G}_j(z^{-1})$ which stems from the recursive properties of $\mathbf{E}_j(z^{-1})$ (see Camacho & Borbons (1999)). With this new matrix it is possible to separate the first term of (10) into past and future inputs:

$$\begin{aligned}\mathbf{G}_j(z^{-1})\Delta u(k+j-1) \quad = \quad & \underbrace{G_0\Delta u(k+j-1) + G_1\Delta u(k+j-2) + \ldots + G_{j-1}\Delta u(k)}_{future\ inputs} \\ & + \underbrace{(G_j)_j\Delta u(k-1) + (G_{j+1})_j\Delta u(k-2) + \ldots + (G_{j-1+n_b})_j\Delta u(k-n_b)}_{past\ inputs}\end{aligned}$$

Now it is possible to separate all past inputs and outputs from the future ones and write this in matrix form:

$$\underbrace{\begin{bmatrix}\hat{y}(k+1|k)\\ \hat{y}(k+2|k)\\ \vdots\\ \hat{y}(k+N_u|k)\\ \vdots\\ \hat{y}(k+N_2|k)\end{bmatrix}}_{\hat{\mathbf{y}}} = \underbrace{\begin{bmatrix}G_0 & 0 & \cdots & 0\\ G_1 & G_0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ G_{N_u-1} & G_{N_u-2} & \cdots & G_0\\ \vdots & \vdots & \ldots & \vdots\\ G_{N_2-1} & G_{N_2-2} & \cdots & G_{N_2-N_u}\end{bmatrix}}_{\mathbf{G}} \underbrace{\begin{bmatrix}\Delta u(k)\\ \Delta u(k+1)\\ \vdots\\ \Delta u(k+N_u-1)\end{bmatrix}}_{\tilde{\mathbf{u}}} + \underbrace{\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\\ \vdots\\ \mathbf{f}_{N_u}\\ \vdots\\ \mathbf{f}_{N_2}\end{bmatrix}}_{\mathbf{f}} \tag{12}$$

which can be condensed to :

$$\hat{\mathbf{y}} \;=\; \mathbf{G}\tilde{\mathbf{u}} \;+\; \mathbf{f} \tag{13}$$

where $\mathbf{f}$ represents the influence of all past inputs and outputs and the columns of $\mathbf{G}$ are the step responses to future $\tilde{\mathbf{u}}$ (for further reading, see (Camacho & Borbons, 1999)). Since each G_i is an $n \times m$ matrix $\mathbf{G}$ has block matrix structure.

Now that we obtained a j-step ahead predictor form of a linear model this can be used to compute the optimal control sequence with respect to a given cost function (like (6)). If (6) is written in vector form and with (13) one yields:

$$\begin{aligned}J(k) \quad &= \quad (\mathbf{r}-\hat{\mathbf{y}})^T Q_r(\mathbf{r}-\hat{\mathbf{y}}) + \tilde{\mathbf{u}}^T Q_u \tilde{\mathbf{u}} \\ &= \quad (\mathbf{r}-\mathbf{G}\tilde{\mathbf{u}}-\mathbf{f})^T Q_r(\mathbf{r}-\mathbf{G}\tilde{\mathbf{u}}-\mathbf{f}) + \tilde{\mathbf{u}}^T Q_u \tilde{\mathbf{u}}\end{aligned}$$

where

$$\mathbf{r} = [r(k+1),\ r(k+2),\ \ldots,\ r(k+N_2)]^T$$

In order to minimize the cost function $J(k)$ for the future control sequence $\tilde{\mathbf{u}}$ the derivative $dJ(k)/d\tilde{\mathbf{u}}$ is computed and set to zero:

$$\begin{aligned}
\frac{dJ(k)}{d\tilde{\mathbf{u}}} &= 0 \\
&= 2\mathbf{G}^T Q_r \mathbf{G}\tilde{\mathbf{u}} - 2\mathbf{G}^T Q_r(\mathbf{r}-\mathbf{f}) + 2Q_u\tilde{\mathbf{u}} \\
(\mathbf{G}^T Q_r \mathbf{G} + Q_u)\tilde{\mathbf{u}} &= \mathbf{G}^T Q_r(\mathbf{r}-\mathbf{f}) \qquad (14) \\
\tilde{\mathbf{u}} &= \underbrace{(\mathbf{G}^T Q_r \mathbf{G} + Q_u)^{-1}\mathbf{G}^T Q_r}_{K}(\mathbf{r}-\mathbf{f}) \qquad (15)
\end{aligned}$$

Thus the optimization problem can be solved analytically without any iterations which is true for all quadratic cost functions in absence of constraints. This is a great advantage of GPC since the computation effort can be very low for time-invariant plant models as the main computation of the matrix K can be carried out off-line. Actually just the first m rows of K must be saved, because of the receding horizon strategy using only the first input of the whole sequence $\tilde{\mathbf{u}}$. Therefore the resulting control law is linear, each element of K weighting the predicted error between the reference and the free response of the plant.
Finally for a practical implementation of APC one has to bear in mind that the matrix $(\mathbf{G}^T Q_r \mathbf{G} + Q_u)$ can be singular in some instances. In the case of GPC this is not a problem since the solution is not computed online. For APC in this work a special Gauss solver was used which assumes zero control input where no unambiguous solution can be found.

3.2 Reducing Overshoot with Reference Filters

With the classic quadratic cost function it is not possible to control the overshoot of the resulting controller in a satisfying manner. If the overshoot needs to be influenced one can choose three possible ways. The obvious and most elaborate way is to introduce constraints, however the solution to the optimization problems becomes computationally more expensive. Another possible solution is to change the cost function, introducing more tuning polynomials, as mentioned by Nørgaard et al. (2000) referring to *Unified Predictive Control*.
A simple but yet effective way to reduce the overshoot for any algorithm that minimizes the standard quadratic cost function (like LQG, GPC or APC) is to introduce a reference prefilter which smoothes the steep areas like steps in the reference. For the helicopter, the introduction of prefilters made it possible to eliminate overshoot completely, retaining comparably fast rise times. The utilized reference prefilters are of first order low-pass kind

$$G_{RF} = \frac{1-l}{1-lz^{-1}}$$

which have a steady-state gain of one and can be tuned by the parameter l to control the smoothing.

3.3 Improving APC Performance by Parameter Filtering

A problem with APC is that a network that has a good prediction capability does not necessarily translate into a good controller, as for APC the network dynamics need to be smooth for consistent linear models which is not a criterion the standard Levenberg-Marquardt backpropagation algorithm trains the network for. A good way to test whether the network dynamics are sufficiently smooth is to start a simulation with the same neural network as the plant and

as the predictive controllers system model. If one sees unnecessary oscillation this is good evidence that the network dynamics are not as smooth as APC desires for optimal performance. The first solution to this is simply training more networks and test whether they provide a better performance in the simulation.

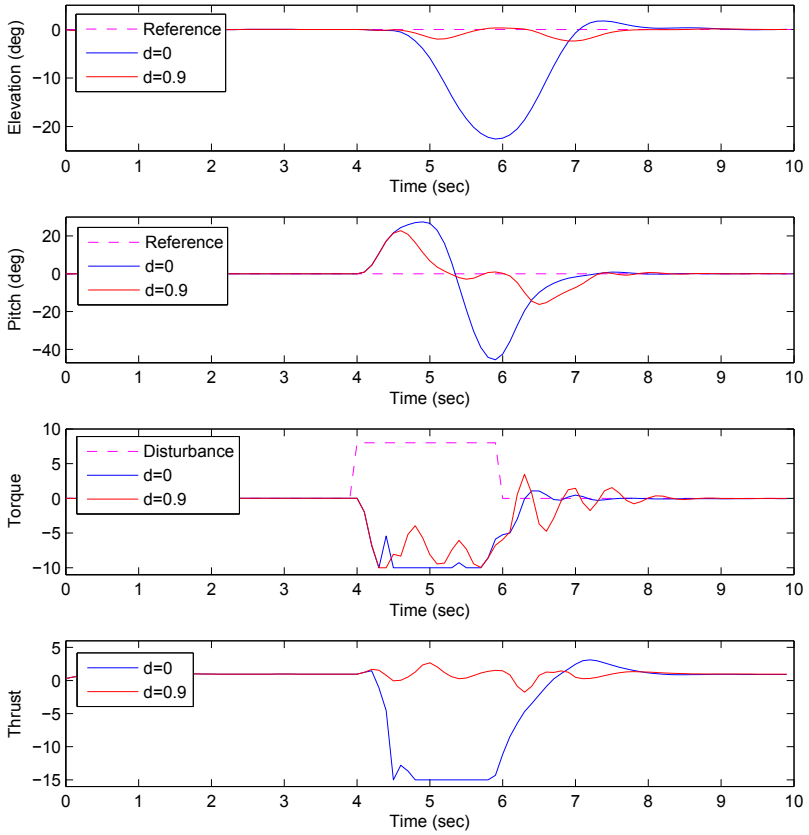


Fig. 16. Simulation results of disturbance rejection with parameter filtering. Top two plots: Control outputs. Bottom two plots : Control inputs

In the case of the helicopter a neural network with no unnecessary oscillation in the simulation could not be found, though. If one assumes sufficiently smooth nonlinearities in the real system, one can try to manually smooth linearizations of the neural network from sample to sample, as proposed in (Witt et al., 2007). Since APC is not able to control systems with nonlinearities that are not reasonably smooth within the prediction horizon anyway the idea of smoothing the linearizations of the network does not interfere with the basic idea of APC being able to control nonlinear systems. It is merely a means to flatten out local network areas where the linearized coefficients start to jitter within the prediction horizon.

This idea has been realized by a first order low-pass filter:

$$G_{PF} = \frac{1-d}{1-dz^{-1}}$$

with tuning parameter d. When applied to the polynomial matrix $\mathbf{A}(z^{-1})$, (3.3) results in the following formula:

$$\hat{\mathbf{A}}_k(z^{-1}) = (1-d)\mathbf{A}_k(z^{-1}) + d\hat{\mathbf{A}}_{k-1}(z^{-1})$$

where $\hat{\mathbf{A}}_k(z^{-1})$ contains the filtered polynomial coefficients $\mathbf{A}_k(z^{-1})$. For prediction horizons around $N_2 = 10...20$ a good starting value for the tuning parameter d was found to be 0.9, however this parameter depends on the sampling rate.
If the filtering parameter d is increased, the adaptivity of the model decreases and shifts towards a linear model (in the case of $d = 1$). The importance of parameter filtering in the case of the helicopter is displayed in figure 16 where an input disturbance acts on the torque input of a standard APC controller and the parameter filtered version.

4. Experimental Results

During the practical experiments the setup shown in figure 17 was used. It necessarily incorporates the stabilizing proportional derivative controller that is included in our nonlinear model from section 2. The sampling time is 0.1 seconds and the experiments were run on a 1 GHz Intel Celeron CPU. All APC related algorithms were implemented in C++ to achieve the computational performance that was necessary to be able to compute the equations in realtime on this system at the given sampling rate.

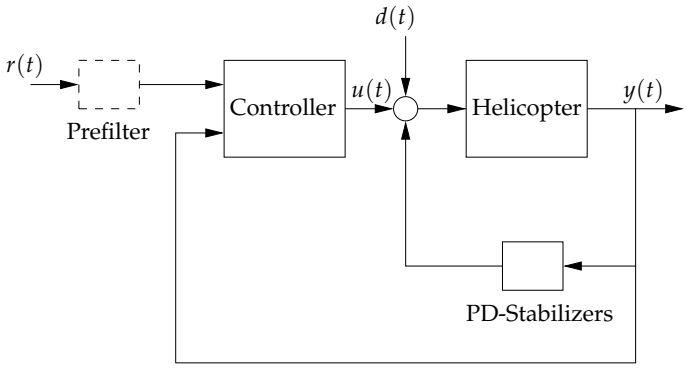


Fig. 17. Control setup for helicopter with inner stabilizing control loop and reference prefilter.

For our experiments only the control of the pitch and elevation axis was considered as the travelspeed axis has significantly longer rise times (about factor 15) than the other two axes, making predictive control with the same sampling rate and prediction horizons impractical. To control the travelspeed axis in this setup one could design an outer cascaded control loop with a slower sampling rate, but this is beyond the scope of this work.
APC as well as GPC were tuned with the same 5 parameters, being the horizons N_1, N_2, N_u and the weighting matrices Q_r and Q_u. The tuning was done as suggested in (Clarke et al., 1987a;b) and resulted in $N_1 = 1$, $N_2 = 10$, $N_u = 10$ and the weighting matrices $Q_r =$

$diag(0,1,1)$ and $Q_u = diag(20,10)$. The choice of Q_r disables weighting for the first output which is the uncontrolled travelspeed-axis.
The computational limits of the test platform were found at horizons of $N_2 = N_u = 20$ which does not leave too much headroom.

4.1 Tracking Performance

APC has been benchmarked with both tracking and disturbance rejection experiments. We also designed a linear GPC and an integrator augmented LQG controller for comparison. The benchmark reference signals are designed to cover all operating ranges for all outputs. All controllers were benchmarked with identically parameterized reference prefilters to eliminate overshoot.
In figure 18 it can be seen that LQG achieves a suitable performance only for the pitch axis while performance on the elevation axis is much poorer than both APC and GPC. For both outputs, APC yields slightly better performance than linear GPC which is most visible for the large reference steps on the more nonlinear elevation axis. However looking at the plant input signals one can see that the APC signals have less high frequency oscillation than for GPC which is also an important issue because of actuator stress in practical use. Parameter filtering does not change the response to the benchmark sequence up to about $d = 0.9$ but significantly improves the performance for disturbance rejection as will be shown in the next section.

4.2 Disturbance Rejection

The performance of the benchmarked controllers becomes more diverse when disturbance rejection is considered. In figure 19 one can see the response to disturbances applied to the two inputs. Again LQG can be tuned to satisfactory performance only for the pitch axis, but also the standard APC and GPC do not give satisfying results. Considering input disturbance rejection the standard APC even shows a lower stability margin than GPC. The introduction of parameter filtering however changes this aspect significantly. With parameter filtering of $d = 0.9$ the stability margin of APC becomes much larger than the one of GPC and it can be seen in the plot that it shows the best disturbance response of all tested controllers. Especially note the low input signal amplitude, while superiorly managing the disturbance.

4.3 Conclusion

With this work it has been shown that MIMO APC for a fast process is indeed feasible with mid-range embedded hardware. It was found that standard APC can be problematic if the network dynamics are unsmooth. For this purpose, parameter filtering was presented as an improvement to the standard APC implementation with which it was possible to enhance the stability margin and overall performance of APC in the face of disturbances significantly. Still the aquisition of a decent model should be the first step before one should tune the performance with parameter filtering, since it remains the most important constituent to good control performance.
Finally although the helicopter is not a highly nonlinear system, APC with parameter filtering was able to outperform the linear GPC while being the more generally applicable control scheme.

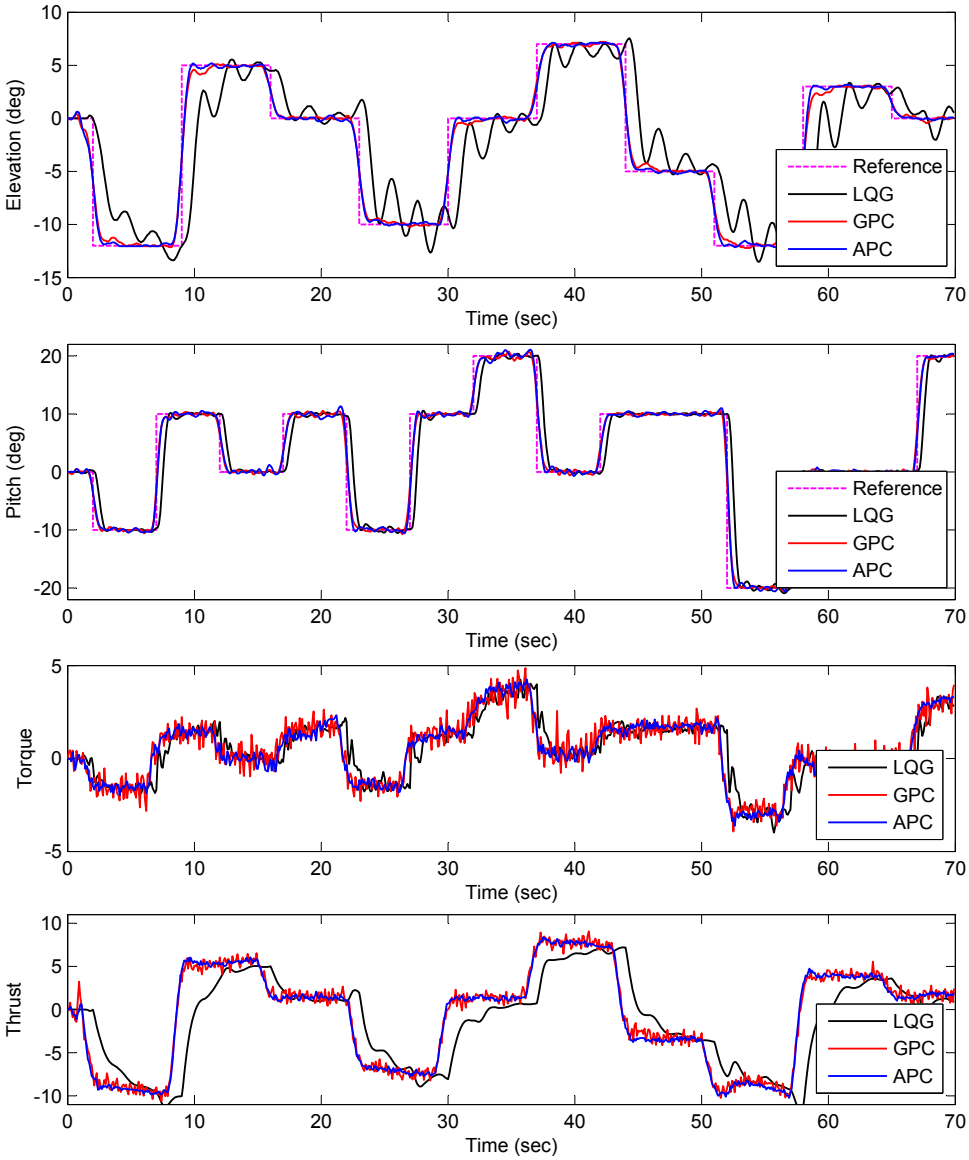


Fig. 18. Experimental results for tracking performance of APC compared to GPC and LQG. Top two plots: Control outputs. Bottom two plots: Control inputs.

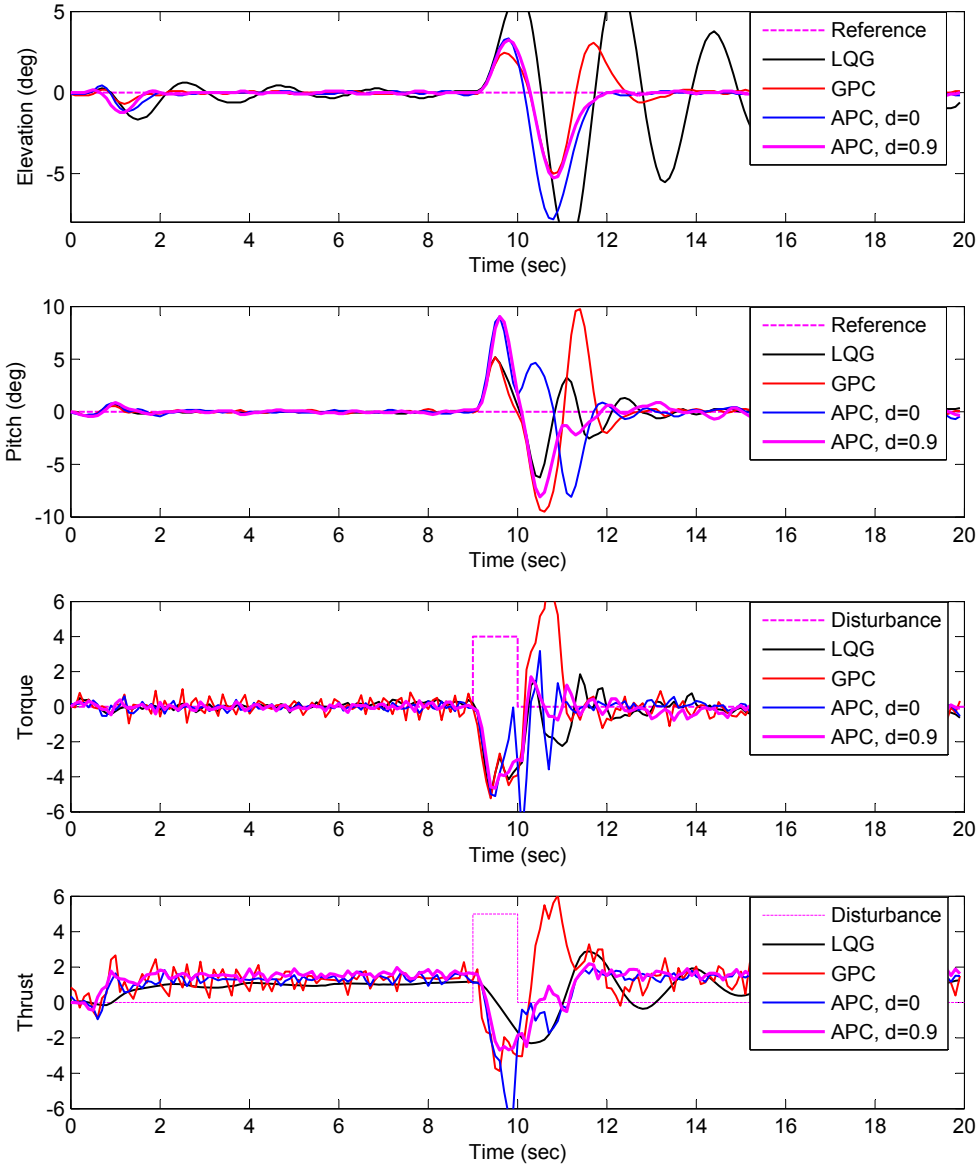


Fig. 19. Experimental results for disturbance rejection performance. Top two plots: Control outputs. Bottom two plots: Control inputs.

5. References

Camacho, E. & Borbons, C. (1999). *Model Predictive Control*, Springer-Verlag, London.

Clarke, D., Mohtadi, C. & Tuffs, P. (1987a). Generalized Predictive Control – Part I. The basic algorithm, *Automatica* **23**(2): 137–148.

Clarke, D., Mohtadi, C. & Tuffs, P. (1987b). Generalized Predictive Control – Part II. Extension and Interpretations, *Automatica* **23**(2): 149–160.

Engelbrecht, A. (2002). *Computational Intelligence: An Introduction*, Halsted Press New York, NY, USA.

Evan, C., Rees, D. & Borrell, A. (2000). Identification of aircraft gas turbine dynamics using frequency-domain techniques, *Control Engineering Practice* **8**: 457–467.

Hagan, M. & Menhaj, M. (1994). Training feedforward networks with the Marquardt algorithm, *IEEE transactions on Neural Networks* **5**(6): 989–993.

Hornik, K., Stinchcombe, M. & White, H. (1989). Multilayer Feedforward Networks are Universal Approximators, *Neural networks* **2**(5): 359–366.

Ismail, A. & Engelbrecht, A. (2000). Global Optimization Algorithms for Training Product Unit Neural Networks, *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, Vol. 1, Como, Italy, pp. 132–137.

Ljung, L. (1999). *System Identification Theory for the User*, 2nd edn, Prentice Hall PTR, Upper Saddle River, NJ.

Maciejowski, J. (2002). *Predictive Control with Constraints*, Prentice Hall.

Mu, J. & Rees, D. (2004). Approximate model predictive control for gas turbine engines, *Proceedings of the 2004 American Control Conference*, Boston, Massachusetts, USA, pp. 5704–2709.

Nørgaard, M., Ravn, O., Poulsen, N. K. & Hansen, L. K. (2000). *Neural Networks for Modelling and Control of Dynamic Systems*, Springer-Verlag, London, UK.

Pico, J. & Martinez, M. (2002). *Iterative Identification and Control*, Springer-Verlag London, chapter System Identification. Performance and Closed-loop Issues.

Quanser Inc. (2005). *3 DOF Helicopter System*, www.quanser.com.

Witt, J., Boonto, S. & Werner, H. (2007). Approximate model predictive control of a 3-dof helicopter, *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, USA, pp. 4501–4506.