The Institute of Space and Astronautical Science
Report No. 665, March 1996

Adaptive Balloon Azimuth Control using a Simple DC Motor Actuator

By

Drago MATKO, Nobuyuki YAJIMA and Motoki HINADA

(February 15, 1996)

ABSTRACT: A control system for the azimuth control of the balloon gondola using a simple actuator (motor and coupling ring) is presented. Three mathematical models of the system are reviewed first. Classical continuous time compensators and robust optimal continuous time and discrete time controllers are designed and compared in frequency and time domain. The rejection of typical disturbances (a ramp type disturbance due to the turning of the entire balloon and a sinusoidal disturbance due to the pendulum motion of the suspended gondola) is investigated by simulations. A feedforward velocity controller applicable in scanning missions and a feedforward position controller for large changes of the azimuth are designed and tested by simulations. A selftuning procedure for the automatic adjustment of the parameters of the controller to the dominant dynamic characteristics of the gondola is given. Finally a comparison between the adaptive notch filter and adaptive Butterworth low pass filter with respect to the elimination of instabilities of pendulum motion around a horizontal axis is presented. The performances of the selftuning and adaptive algorithms are proven by simulations.

1. INTRODUCTION

Typical tasks for a balloon mission [7] are fine pointing to a specified object, (corresponding to the servo problem in terms of control engineering), changing the objects (corresponding to the step response problem in terms of control engineering) and raster scanning of the celestial sphere (corresponding to the tracking problem in terms of control engineering). In Fig. 1 the balloon gondola with typical loading is shown. It should be noted however that even the pointing to a specific object represents a tracking due to the rotation of the earth and this term will be used in the report for the pointing task. Some of the parameters of the system to be controlled are unknown (e.g. spring constant of the suspension rope) or changed between the missions (e.g. moment of the inertia of the gondola) or changed during the mission (e.g. motor electrical parameters due to the environmental temperature). This is the reason for the application of the adaptive and self tuning control respectively.

The authors gratefully acknowledge the support of Prof. Ryojiro AKIBA, former director general of ISAS.

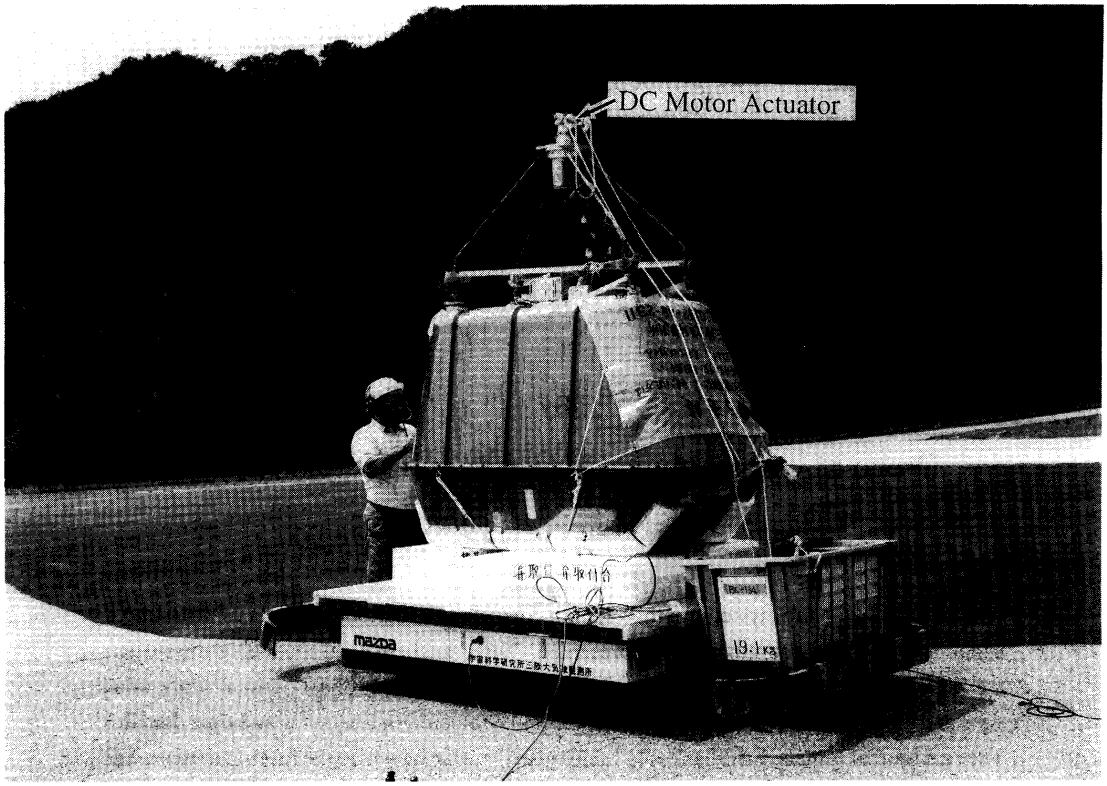


Fig. 1. The balloon gondola with typical loading.

The report is organised as follows: First the mathematical models used for simulation are reviewed, then some preliminary studies of the continuous time compensator are given. The discrete time controller to be used in the on board computer is designed next using optimisation of the third order state space controller and observer while taking into account small uncertainties in the process parameters. The resulting controller can be reduced to the second order controller. Next the self tuning of the controller due to the unknown spring constant of the suspension rope and moment of the inertia of the gondola is designed. Finally an adaptive control strategy for elimination of instabilities in the motion around a horizontal axis which is caused by the torque due to unsymmetric loading of the gondola is proposed. Two strategies are compared. In the first one an adaptive notch filter is used to eliminate the oscillations in the control signal by adjusting its central frequency, i.e. to make the closed loop gain at the unknown frequency of oscillations very small. The undesired oscillations decrease by natural damping. In the second approach the adaptive notch filter is used to detect the frequency and amplitude of undesired oscillations and an adjustable Butterworth filter is applied to change the phase of the control loop in the corresponding frequency range. In this approach the damping of undesired oscillations is controlled. All algorithms were tested by simulations using MATLAB SIMULINK design tool.

2. SIMULATION MODELS

The balloon gondola to be controlled is shown schematically in Fig. 2. The balloon is supposed to be a fixed point due to its large moment of inertia. The suspension rope is modelled as a torsion spring with the spring constant k_s. The coupling ring has a small moment of inertia I_m, so the differential equation of its rotation (its rotation angle is denoted by Θ_m) is

$$\ddot{\Theta}_m = \frac{-T - k_s\Theta_m}{I_m} \tag{1}$$

where T is the motor torque which drives the gondola to turn. It is generated by a DC motor with the torque constant k_T, back generating voltage k_F, coil resistance R and input voltage (control input) U according to

$$T = nk_t\frac{U - nk_F(\dot{\Theta}_g - \dot{\Theta}_m)}{R} \tag{2}$$

where n is the gear ratio and Θ_g the azimuth of the gondola. The motor torque turns the gondola (moment of inertia I_g) according to

$$\ddot{\Theta}_g = \frac{T}{I_g} \tag{3}$$

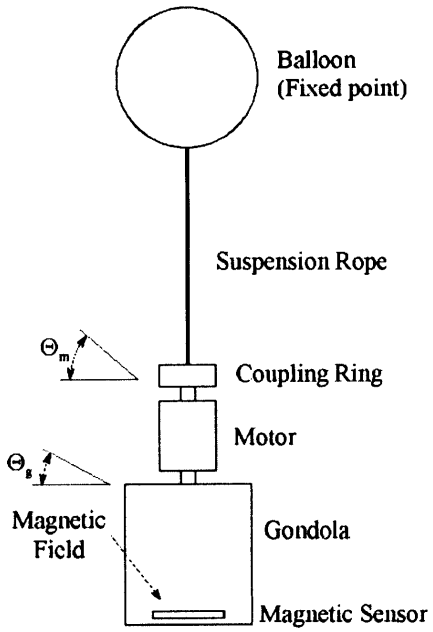


Fig. 2. The scheme of the balloon gondola system.

The simulation scheme is depicted in Fig. 3. The corresponding transfer function is

$$G(s) = \frac{1}{nk_F}\frac{I_m}{I_g}\frac{s^2 + \frac{k_s}{I_m}}{s\left[\frac{RI_m}{n^2k_Tk_F}s^2\left(s + \frac{n^2k_Tk_F}{RI_g}\right) + \left(s^2 + \frac{Rk_s}{n^2k_Tk_F}s + \frac{k_s}{I_g}\right)\right]} \tag{4}$$

The moment of inertia (I_m) of the coupling ring is small, so the term in brackets can be neglected yielding the following transfer function

$$G(s) = \frac{1}{nk_F}\frac{\frac{k_s}{I_g}}{s\left(s^2 + \frac{Rk_s}{n^2k_Tk_F}s + \frac{k_s}{I_g}\right)} \tag{5}$$

and the simulation scheme shown in Fig. 4. If the gear ratio n is large the simulation model depicted in Fig. 5 is obtained having the transfer function

$$G(s) = \frac{1}{nk_F}\frac{\omega_0^2}{s\left(s^2 + \omega_0^2\right)} \qquad \omega_0^2 = \frac{k_s}{I_g} \tag{6}$$

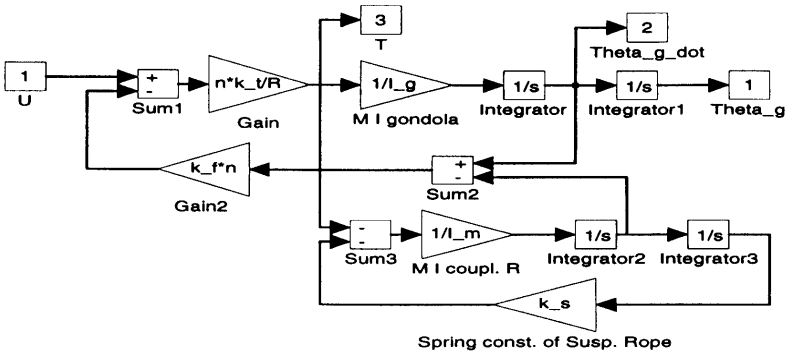


Fig. 3. Simulation scheme for Model 1.

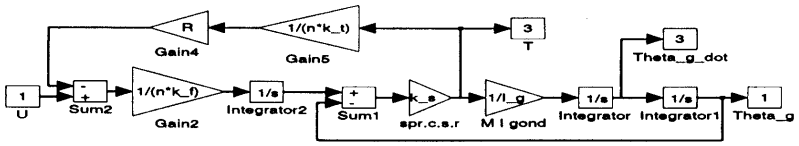


Fig. 4. Simulation scheme for Model 2.

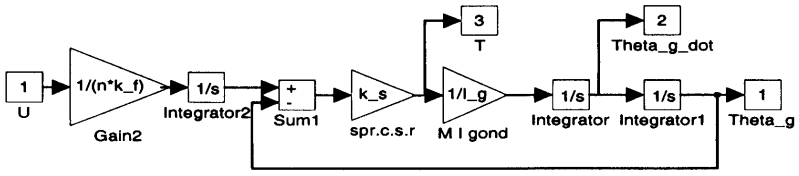


Fig. 5. Simulation scheme for Model 3.

The following nominal values of the physical parameters are used

$$R = 10\,\Omega \quad k_t = 50 \cdot 10^{-3}\text{Nm/A} \quad k_f = 5.5\ V/1000\ \text{rpm} = 0.9167\ \text{mV/degps}$$
$$n = 3000 \quad I_g = 50\ \text{kg m}^2 \quad I_m = 0.1\ \text{kg m}^2$$
$$T_0 = 100\ \text{s} \quad \omega_0 = 2 \cdot \pi / T_0 \quad k_s = \omega_0^2 \cdot I_g$$

However the moments of inertia of the coupling ring and gondola and the spring constant of the suspension rope are supposed to vary in such a way that the period of the dominant dynamics of the controlled system $2 \cdot \pi/\omega_0$ varies between 50 s and 150 s. The nominal values of physical parameters are chosen conservatively, so that the transfer function (4) represents the worst case deviation from the simple model (Eq.6). Robustness studies were made by varying the eigenfrequency ω_0 for $\pm 30\%$.

At some missions an undesired motion of the gondola was observed. This motion occurs around the axis, which is perpendicular to the rotational axis and is denoted here by x. The motion around this axis is influenced by the moment of inertia of the gondola around it (I_{gx}), the mass of the gondola (M_g), gravity ($g = 9.81\ \text{m/s}^2$) and a torque T_x which is due to the unsymmetric loading of the gondola. This torque is very small but its sign is unknown. The gondola is equipped with a simple magnetic sensor which however in some positions detects not only the desired rotation of the gondola but also its movement around the x axis due to the vertical component of the earth magnetic field ($52\ ^0$ in the latitudes around Sanriku). Due to the unknown sign of the torque which is driving the gondola around the x axis, positive feedback can occur, which makes the system unstable. The differential equation of this undesired motion is

$$\ddot{\Theta}_x = \frac{T_x - M_g g l_2}{I_{gx}} \tag{7}$$

where l_2 is the distance between the driving motor and the mass centre of the gondola. The corresponding simulation scheme is shown in Fig. 6 where also a small damping ζ is modelled. Its value is estimated by observations that the settling time for this motion is 5 to 10 minutes. The complete model used in simulation is shown in Fig. 7.

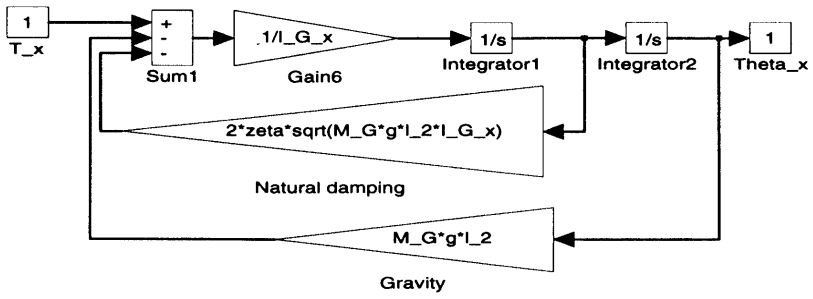


Fig. 6. Simulation scheme for the movement around the x axis.

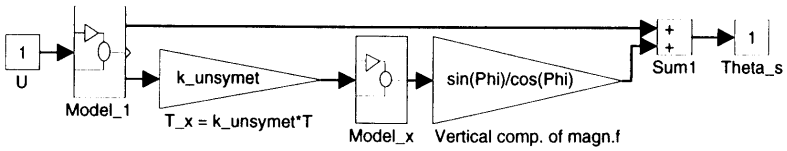


Fig. 7. Simulation scheme for the complete model.

3. PRELIMINARY STUDIES IN CONTINUOUS TIME

First some preliminary studies in continuous time using the Model 1 were performed. Classical compensator design is given in Subsection 3.1, optimal compensator in Subsection 3.2 and optimal state variable controller and observer in Subsection 3.3.

3.1 Classical Compensator Design

Classical controller to feedback stabilise a process with three poles very close to the origin is a double lead lag compensator in the form

$$G_c(s) = K_c \frac{(s+a_1)(s+a_2)}{(s+b_1)(s+b_2)} \tag{8}$$

Fig. 8 shows the simulation scheme of the closed loop. The influence of the b parameters on the closed loop behaviour was studied first by choosing $a_1 = a_2 = 0.1$ and increasing $b_1 = b_2$ from 1 over 2.5 to 5. The corresponding gains for these three cases determined by root locus plots are 280, 2200 and 9900 respectively. Fig. 9 shows the corresponding root locus plots with the most left curve corresponding to the largest b parameter (pole of the lag part). The open loop poles are denoted by $\times$, the open loop zeros by $\circ$ and the closed loop poles with chosen (used in simulation) controller gain by $+$. In Fig. 10 the Nyquist plots are shown for all three cases with the solid line corresponding to the $b_1 = b_2 = 1$, dotted line corresponding to the $b_1 = b_2 = 2.5$ and dashed line corresponding to the $b_1 = b_2 = 5$. It can be observed that for the linear model the increase of the b parameters increases the classical performance criteria such as high and low gain and phase margin.

However the controller gains are very high and increase with the increasing b parameters. Some

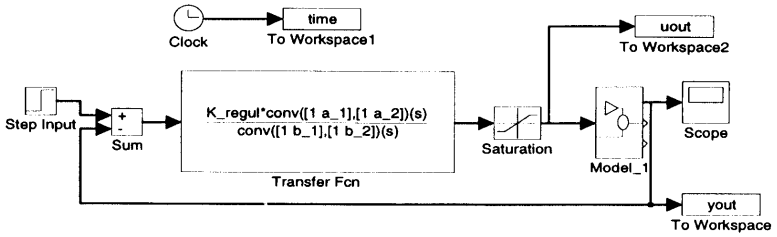


Fig. 8. Simulation scheme of the closed loop.

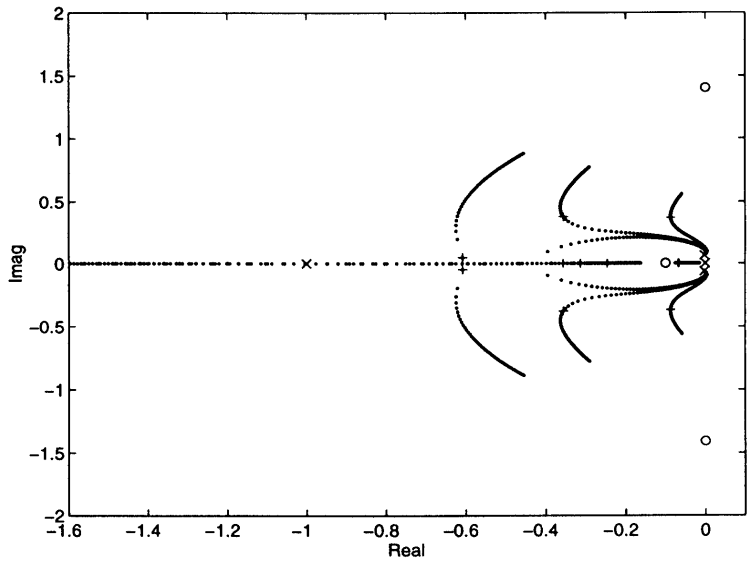


Fig. 9. Root locus plots for $a_1 = a_2 = 0.1$ and $b_1 = b_2 = 1$, 2.5 and 5 (the most left curve corresponds to the largest b parameter).

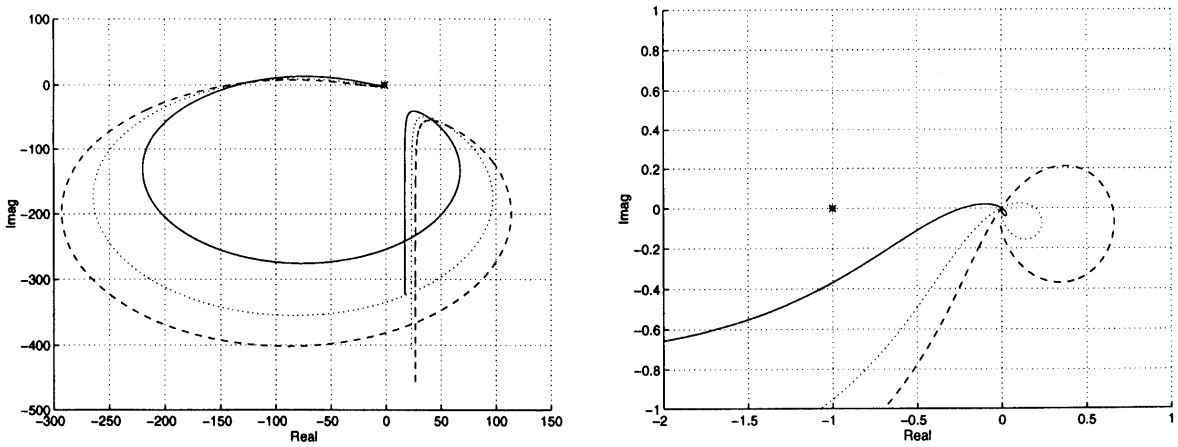


Fig. 10. Nyquist plots for $a_1 = a_2 = 0.1$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line).

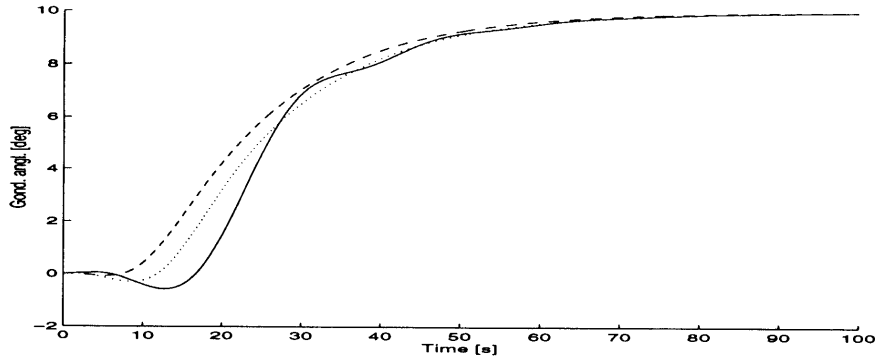


Fig. 11. Rotation angles of the gondola for $a_1 = a_2 = 0.1$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line).

simulations of the closed loop behaviour with the motor voltage limited to 10 V and a position reference changed from $0\ ^0$ to $10\ ^0$ were performed. The response (rotation angle of the gondola) for all three cases is shown in Fig. 11 with the same line type notation as used in the Nyquist diagram (Fig. 10). Fig. 12 depicts the corresponding control signal (motor voltage). It can be observed that

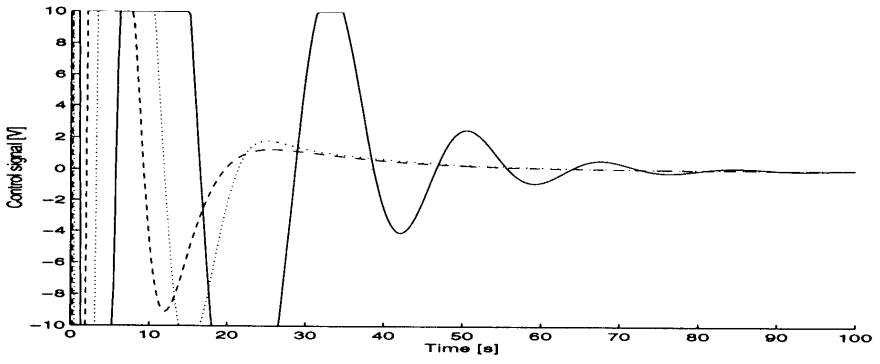


Fig. 12. Control signal for $a_1 = a_2 = 0.1$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line).

the non-linear effect of the saturation results in a response of the closed loop which is similar to the non minimal phase response. The simulations have shown that the increase of the b parameters has positive effect on the closed loop behaviour in the time domain with the non-linear model.

Theoretically there is no limit on the high gain, since the relative degree of the Model 1 is two. For the estimation of the high gain boundary the Model 3 seemes to be suitable since it has relative degree 3 and two purely complex poles. However no significant difference in time domain can be observed while the root locus and Nyquist diagrams, depicted in Figs. 13 and 14 respectively, show little differences at high frequencies.

One drawback of the high gain is small proportional band, i.e. small rotation angle of the gondola which saturates the driving motor. The static gains for the three investigated cases are 2.80, 3.52 and

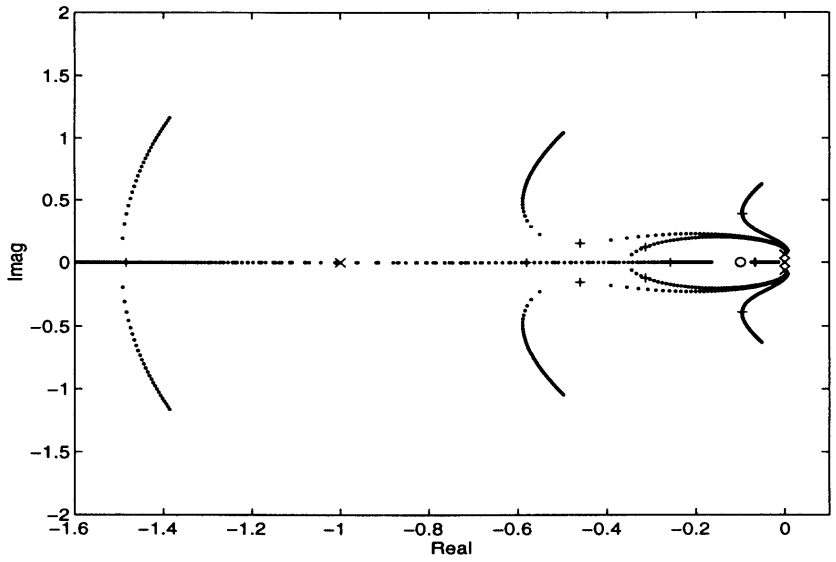


Fig. 13. Root locus plots for the Model 3.

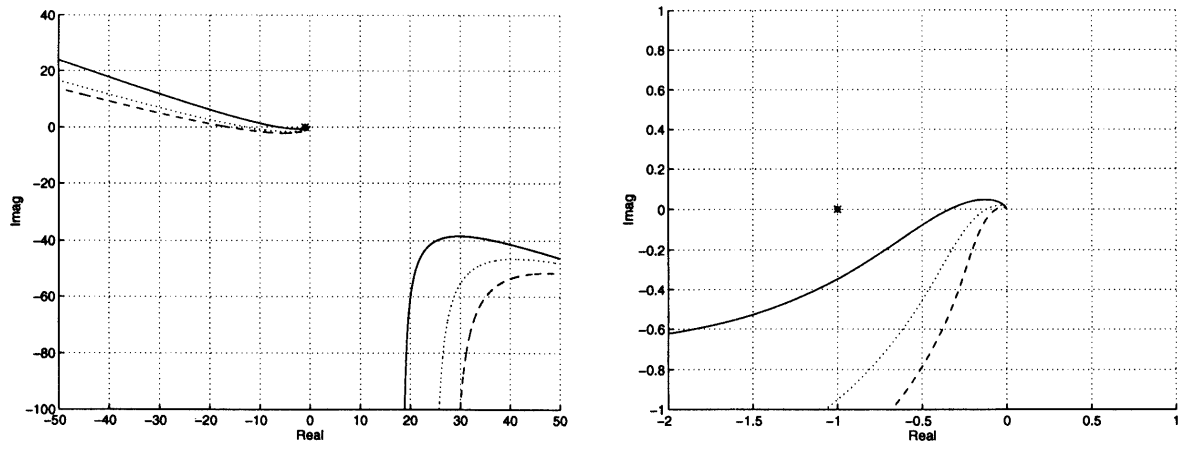


Fig. 14. Nyquist plots for the Model 3.

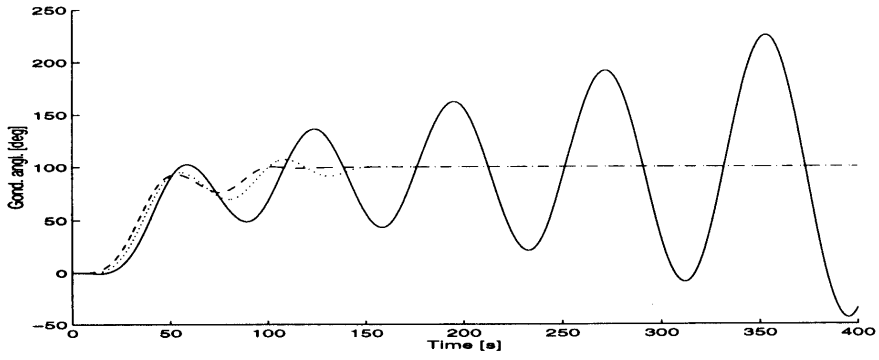


Fig. 15. Rotation angle of the gondola for the 100 ^{0} reference change and $a_1 = a_2 = 0.1$, $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line).

3.96 yielding saturating rotation angles 3.57 ^{0}, 2.84 ^{0} and 2.53 ^{0} respectively.

The saturation, which makes the closed loop system non-linear requires also a study of the global (large rotational angle) stability. This study shows that the closed loop becomes unstable for large reference angles changes. Fig. 15 shows the rotation angle of the gondola response for the 100 ^{0} change of the reference angle. It can be seen that the closed loop system becomes unstable for small b parameters.

Next study which has been made is the study of the influence of the a parameters. Larger a parameters mean smaller dominant time constant and faster response. This is true for the linear case while in the non- linear case the allowable reference position change becomes smaller and smaller. With increasing a parameters the high frequency gain and the static gain, which in the linear case stabilise the closed loop system become very high. The highest still acceptable a parameters are $a_1 = a_2 = 0.15$. Figs. 16, 17, 18 and 19 show the results for these values of the a parameters and $b_1 = b_2 = 5$, 10 and 100 respectively. The corresponding gains were 14000, 58400 and 6.25×10^6 respectively. The simulation results in Figs. 18 and 19 correspond to the 10 ^{0} reference position change. With increasing reference position change the closed loop system becomes oscillatory (as shown in Fig. 20 for the reference position change 15 ^{0}) and for greater changes unstable.

A study of robustness was made for the parameters of the gondola not equal to those used for the controller design. There are only two parameters which significantly influence the closed loop behaviour, namely the eigen frequency ω_0 (influenced by I_g and k_s) and the gain $1/(nk_f)$. The eigen radial frequency is expected to vary significantly, so in the study it was supposed to vary between $2\pi/50$ and $2\pi/150$ with $2\pi/100$ the nominal case.

First the controller with the parameters $a_1 = a_2 = 0.15$, $b_1 = b_2 = 10$ and gain 58400 was examined. In the linear case there seems to be no dangerous influence even for so drastic change in the eigen frequency, as demonstrated in Fig 21, which depicts the closed loop poles for varying ω_0 over the specified range. The poles location for the nominal parameter ($\omega_0 = 2\pi/100$) is denoted by an asterisk (*). However the simulation with the constrained input due to the saturation of the actuator changes the situation dramatically. Fig 22 shows the closed loop response for the nominal ($\omega_0 = 2\pi/100$ depicted by solid line) and two extreme cases ($\omega_0 = 2\pi/50$ - dotted line and

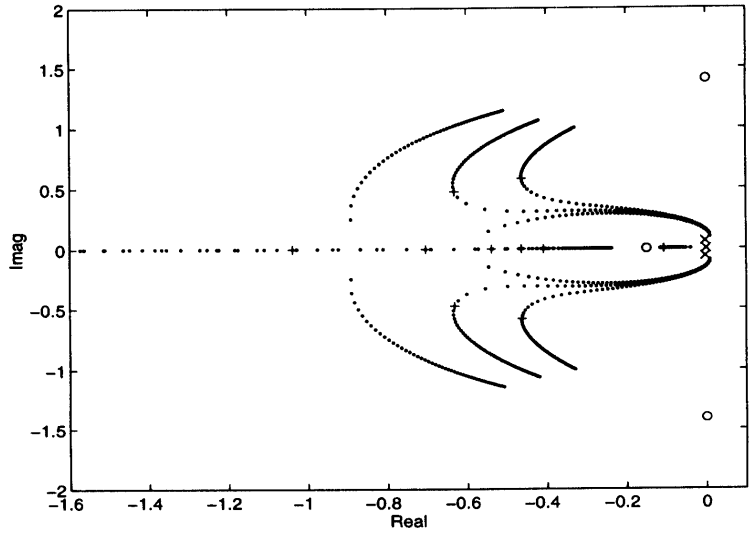


Fig. 16. Root locus plots for $a_1 = a_2 = 0.15$ and $b_1 = b_2 = 5$, $b_1 = b_2 = 10$ and $b_1 = b_2 = 100$ respectively.

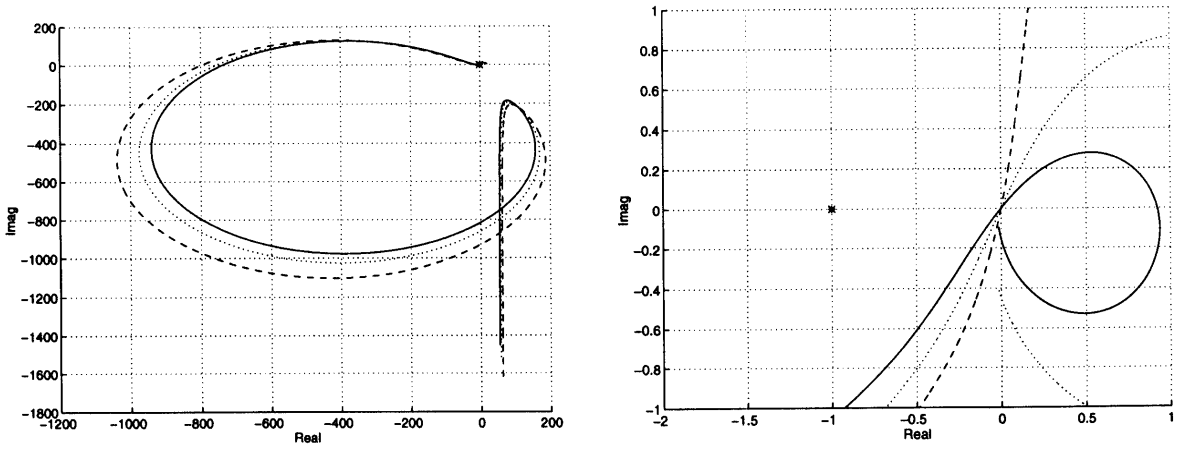


Fig. 17. Nyquist plots for $a_1 = a_2 = 0.15$ and $b_1 = b_2 = 5$ (solid line), $b_1 = b_2 = 10$ (dotted line) and $b_1 = b_2 = 100$ (dashed line).

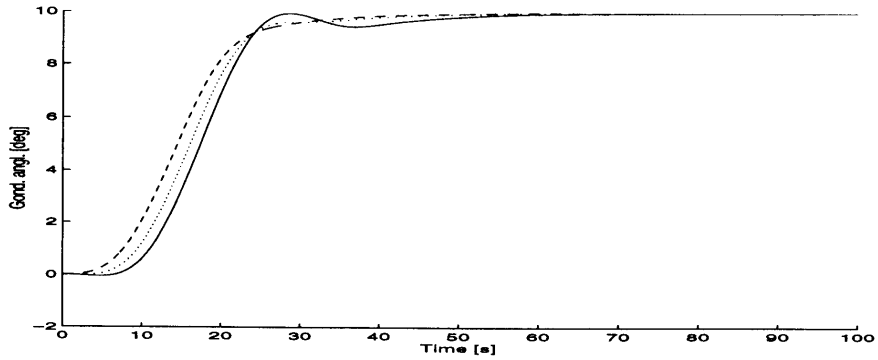


Fig. 18. Rotation angles of the gondola for $a_1 = a_2 = 0.15$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line) - 10^0 step change.

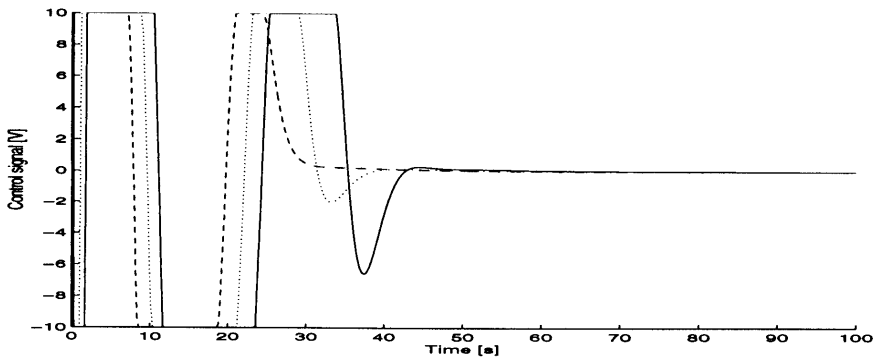


Fig. 19. Control signal for $a_1 = a_2 = 0.15$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (solid line).

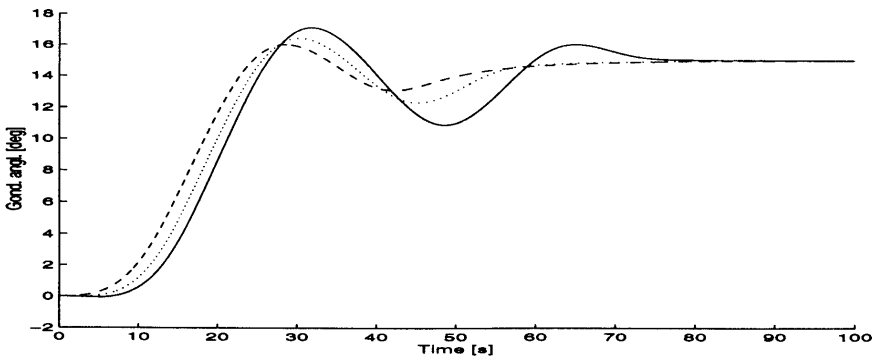


Fig. 20. Rotation angles of the gondola for $a_1 = a_2 = 0.15$ and $b_1 = b_2 = 1$ (solid line), $b_1 = b_2 = 2.5$ (dotted line) and $b_1 = b_2 = 5$ (dashed line) - 15^0 step change.

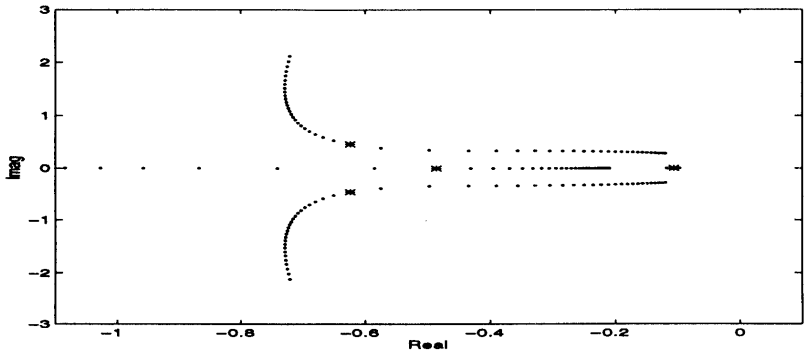


Fig. 21. The closed loop poles for $a_1 = a_2 = 0.15$ and ω_0 varying from $2\pi/150$ to $2\pi/50$.

$\omega_0 = 2\pi/150$ - dashed line respectively). It can be observed that for small ω_0 (large periods) the

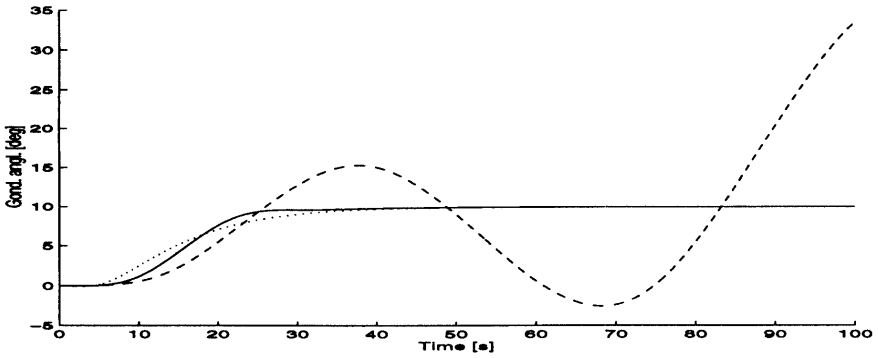


Fig. 22. The closed loop response for the nominal ($\omega_0 = 2\pi/100$ - solid line) and two extreme cases ($\omega_0 = 2\pi/50$ - dotted line and $\omega_0 = 2\pi/150$ - dashed line with the controller parameters $a_1 = a_2 = 0.15$, $b_1 = b_2 = 10$ and gain 58400.

closed loop becomes unstable. The controller with the parameters $a_1 = a_2 = 0.1$, $b_1 = b_2 = 2.5$ and gain 2200 has a better robustness. The corresponding closed loop poles are shown in Fig. 23, the closed loop responses in Fig. 24 and the the corresponding control signals in Figure 25. This controller has better robustness, so the old dilemma between good control and good robustness has to be solved. One solution to avoid the problem of varying eigen frequency is to apply the selftunning procedure for the initial estimate of the eigenfrequency and to adjust the controller to it. This will be done later in Section 8.. However also the tuned controller must posess some robustness. It will be supoosed that the eigenfrequency of the gondola can alter for $\pm$ 30 % and that the moment of inertia of the coupling ring, which is supposed to be small, can lie in the range between 0 and 0.1 kg m^2. Using this superpositions an optimal compensator will be designed in the next Section.

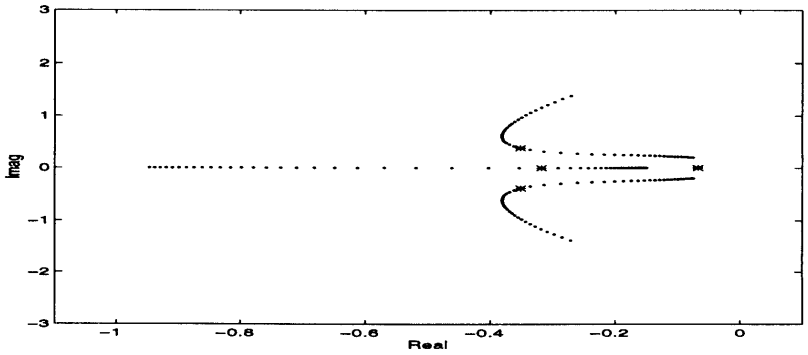


Fig. 23. The closed loop poles for $a_1 = a_2 = 0.1$ and ω_0 varying from $2\pi/150$ to $2\pi/50$

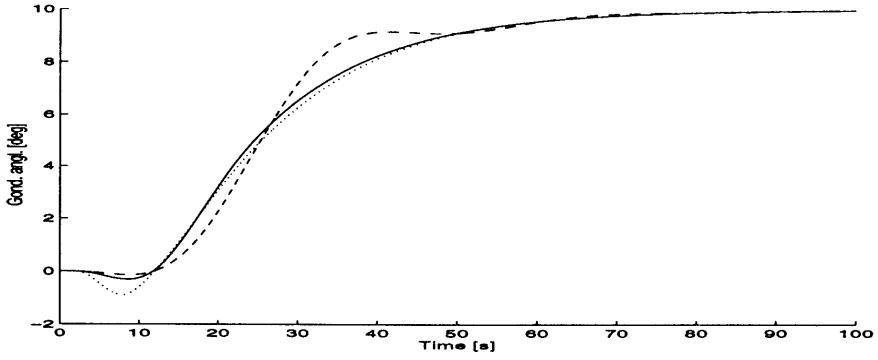


Fig. 24. The closed loop response for the nominal ($\omega_0 = 2\pi/100$ - solid line) and two extreme cases ($\omega_0 = 2\pi/50$ - dotted line and $\omega_0 = 2\pi/150$ - dashed line for controller parameters $a_1 = a_2 = 0.1$, $b_1 = b_2 = 2.5$ and gain 2200.

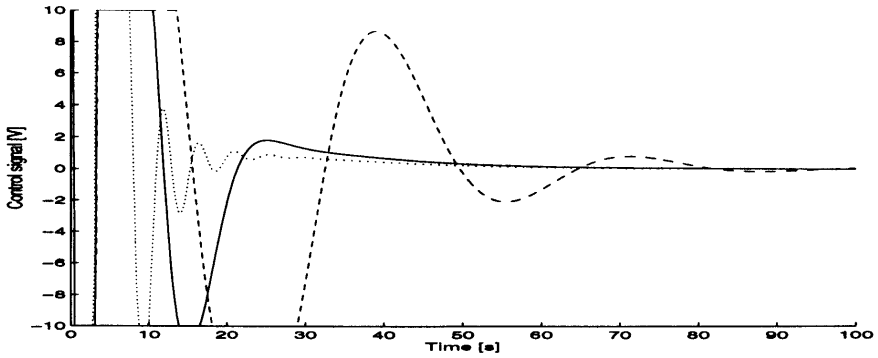


Fig. 25. The control signal for the nominal ($\omega_0 = 2\pi/100$ - solid line) and two extreme cases ($\omega_0 = 2\pi/50$ - dotted line and $\omega_0 = 2\pi/150$ - dashed line for controller parameters $a_1 = a_2 = 0.1$, $b_1 = b_2 = 2.5$ and gain 2200.

3.2 Optimal Compensator

The five parameters of the compensator (K_c, a_1, a_2, b_1 and b_2) were optimised in order to get an optimal and robust controller. The performance criteria was

$$PC = \int_0^T e^2(t)dt \tag{9}$$

where $e(t)$ is the control error and T is the integration time, which is larger than the transition time of the closed loop (200 s were chosen). As input a 10^0 step change in the reference azimuth position was chosen. The performance criteria evaluation scheme is shown in Fig. 26. Actual optimisation

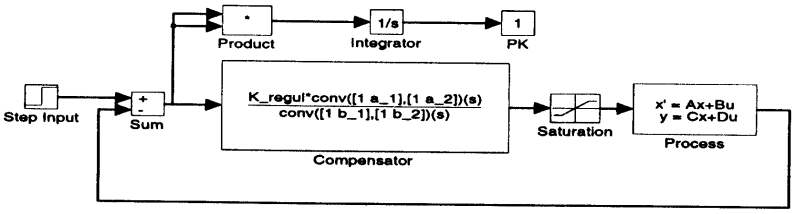


Fig. 26. The performance criteria evaluation scheme.

criteria (cost function) was the sum of the performance criteria for the nominal and ±30% disturbed eigenfrequencies of the gondola. Also the most accurate Model 1 with worst case parameters and the saturation of the actuator at 10 V were used in optimisation. In such way the robustness of the controller was achieved.

The resulting optimal parameters of the controller are:

$$\begin{aligned} K_c &= 249290 \\ a_1 = 0.12077 \quad & a_2 = 0.16195 \\ b_1 = 37.497 \quad & b_2 = 43.578 \end{aligned} \tag{10}$$

The resulting cost function value for the optimal compensator parameters is 3856.4.

3.3 Optimal State Variable Controller and Observer

Next an optimal state controller and observer was designed for simple Model 3. The observer and controller poles were placed to minimise the same cost function as used in the optimal compensator design. Actually the state controller and observer for the Model 3 is a compensator of the third order. The poles were placed as real and also as conjugate complex, however real poles gave better results. The optimal (using the same cost function as in Subsection 3.2) closed loop pole locations for the controller part are (-0.1471 -26.7399 -12.8960) and for the observer part (-0.1530 -14.9725 - 6.3615) respectively. The corresponding state feedback equation is

$$u = -\mathbf{K}\hat{\mathbf{x}} \qquad \mathbf{K} = [109.403 \;\; 35225.2 \;\; 244266.8] \tag{11}$$

and the corresponding observer equation

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{b}u + \mathbf{L}(y - \mathbf{c}^T\hat{\mathbf{x}}) \qquad \mathbf{L}^T = [3691.35 \;\; 21.487 \;\; 98.508] \tag{12}$$

yielding the controller transfer function

$$G_C(s) = 2.52 \times 10^7 \frac{s^2 + 0.2779s + 0.020415}{(s + 31.458)(s^2 + 29.812s + 366.17)} \tag{13}$$

The resulting cost function value for the optimal state controller and observer (third order compensator) is 3796.7. This is a little bit better than the optimal second order compensator; a comparison of control algorithms will be given in Section 5..

4. DISCRETE TIME CONTROLLERS

Discrete time controllers will be reviewed in this Section. First a third order controller, which is an equivalent of the continuous state space controller - observer will be designed. Due to a notch filter or Butterworth filter application for the elimination of undesired oscillations it will be modified to stabilise the closed loop in the presence of the fourth order Butterworth filter for the nominal and for ±30% deviated eigenfrequencies of the gondola. The modified controller has one zero pole pair in the high frequency region of the applied filter, so the controller can be reduced to a second order controller.

4.1 Third Order Controller

Third order discrete time controller, interpreted here as state variable controller and observer was designed by optimisation for sampling frequency $f_s = 1/T_s = 25$ Hz, where T_s is the sampling period. The performance criteria was the discrete time equivalent of Eq. (9)

$$PC = T_s \sum_{k=0}^{f_s T} e^2(k) \tag{14}$$

However a long term instability, which could be detected only by using large values of observation time T, occurs in the discrete case, so a constrained optimisation with the constraints that the closed loop is stable for the nominal and for ±30 % disturbed eigenfrequencies of the gondola was applied. Also the constraint that the resulting controller should be stable was added. The state controller vector $\mathbf{K}$ and observer feedback vector $\mathbf{L}$ were optimised rather than the desired pole locations. This is due to better numerical properties of the algorithm and also due to the simplicity of handling real and conjugate complex poles. The resulting optimal state feedback equation is

$$u(k) = -\mathbf{K}\tilde{\mathbf{x}}(k) \qquad \mathbf{K} = [54.598 \;\; 671.48 \;\; 6966.9] \tag{15}$$

and the resulting optimal observer equation

$$\begin{aligned} \hat{\mathbf{x}}(k+1) &= \mathbf{A}\hat{\mathbf{x}}(k) + \mathbf{b}u(k) + \mathbf{L}[y(k) - \mathbf{c}^T\hat{\mathbf{x}}(k)] \\ \tilde{\mathbf{x}}(k) &= \hat{\mathbf{x}}(k) + \mathbf{L}(y(k) - \mathbf{c}^T\hat{\mathbf{x}}) \qquad \mathbf{L}^T = [21.122 \;\; 0.14408 \;\; 0.31144] \end{aligned} \tag{16}$$

yielding the closed loop poles of the controller part (0.2183, 0.9854, 0.9942) and for the observer part (0.9878, 0.9341±0.0811j) respectively. The resulting controller transfer function is

$$G_C(z) = \frac{3419.7 - 6807.9z^{-1} + 3388.3z^{-2} + 1.2433 \times 10^{-9}z^{-3}}{1 - 2.0413z^{-1} + 1.2365z^{-2} - 0.1830z^{-3}} \tag{17}$$

The resulting cost function value for the optimal state controller and observer (third order compensator) is 4397.8. This is not as good as the continuous time compensators; a comparison of the controllers will be given in Section 5..

4.2 Discrete Controllers Suitable for Butterworth Filter Application

The fourth order low pass Butterworth filter will be used to change the phase of the signals at the eigenfrequency of the undesired motion around the x axis. Fourth order is chosen to change the phase for 180 ^{0} at the cut - off frequency. The phase shift is controlled by moving the cut - off frequency of the filter between 0.3 and 3 Hz. This is the frequency range of undesired oscillations. The controller was optimised with the fourth order Butterworth low pass filter with the worst case cut - off frequency (0.3 Hz). The constraint used in optimisation was the request for the closed loop stability to be assured for all filters with cut - off frequencies between 0.3 and 3 Hz (during the optimisation only filters with cut - off frequencies 0.3, 1 and 3 Hz were tested, however the resulting controller was tested for other frequencies as well). The resulting controller and observer gain vectors were

$$\begin{aligned} \mathbf{K} &= [6.8316 \;\; 708.66 \;\; 3277.8] \\ \mathbf{L}^T &= [2.6895 \;\; 0.39448 \;\; 0.095339] \end{aligned} \tag{18}$$

yielding the closed loop poles of the controller and observer part (0.9902, 0.9533 ±0.0671j) and (0.6154 , 0.9951±0.0045j and the controller transfer function

$$G_C(z) = \frac{610.4 - 1217.0z^{-1} + 606.6z^{-2} - 7.3328 \times 10^{-12}z^{-3}}{1 - 2.4986z^{-1} + 2.0496z^{-2} - 0.5476z^{-3}} \tag{19}$$

The resulting cost function is 4927.1. The cost functions for cut - off frequencies 1, 3 and ∞ = no Butterworth filter are 4837.2, 4963.5 and 4985.5 respectively.

The resulting controller has one zero practically in the origin of the complex z plane, and one pole at high frequencies (z=0.6110 which corresponds to s=–12.3165). Since this is in the frequency range of the Butterworth filter, the controller can be reduced to a second order compensator with dominant poles at $0.9438 \pm 0.0745j$) and zeros at $0.9968 \pm 0.0031j$ which practically (with the neglecting of imaginary parts) corresponds to the lead lag compensator with two zeros at $s = -0.0819$ and poles at $s = 1.45$. The gain of the controller is 610.4 which yields its transfer function

$$G_C(z) = \frac{1569.2 - 3128.4z^{-1} + 1559.2z^{-2}}{1 - -1.8876z^{-1} + 0.8963z^{-2}} \tag{20}$$

The resulting cost function is 4948.8. The cost functions for cut - off frequencies 1, 3 and ∞ = no Butterworth filter are 4847.3, 5007.7 and 5032.1 respectively.

5. COMPARISON OF THE CONTROLLERS

All controllers (continuous second order compensator, continuous third order compensator - state variable controller with observer, discrete third order discrete compensator (discrete state variable controller with observer) and discrete second order compensator suitable for Butterworth filter application were compared in frequency and time domain. Figs. 27, 28 and 29 depict the frequency responses of all controllers. Continuous controllers have higher gain at high frequencies. In Fig. 29

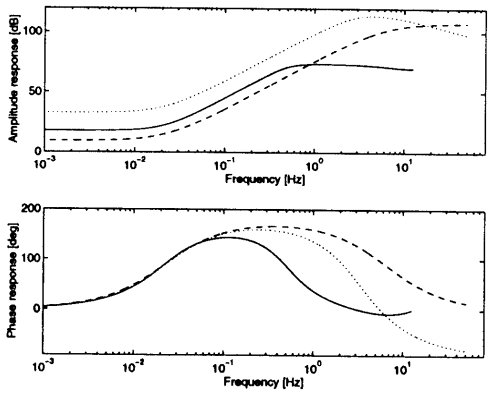


Fig. 27: Frequency response of optimal controllers: second order compensator (dashed line), third order compensator - state variable controller with observer (dotted line) and third order discrete compensator - state variable controller with observer (solid line).

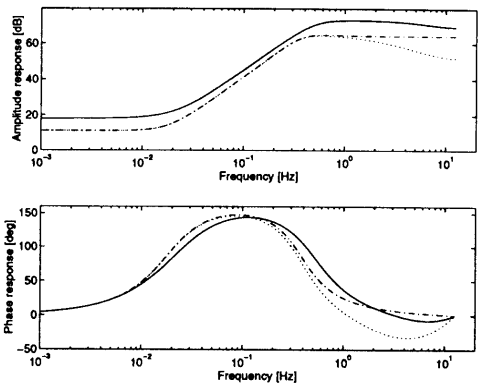


Fig. 28: Frequency response of discrete controllers: optimal third order (solid line), optimal third order suitable for the Butterworth filter application (dotted line) and second order (dashed line).

also the mechanism to change the phase of the controller and Butterworth filter in the frequency range of undesired oscillations around a horizontal axis can be seen.

Fig. 30 shows the Bode diagram for the most precise Model 1 and continuous second order compensator, continuous third order compensator (state variable controller with observer) and third order discrete compensator (state variable controller with observer) respectively. The peak at 0.01 Hz is due to the nominal eigenfrequency of the gondola (its moment of inertia and suspension rope as a spring); the low and change of the phase at 0.3 Hz is due to the zero of the coupling ring (its moment of inertia and suspension rope as a spring - see Eq. 4). From the Nyquist diagram shown in Fig. 31 the gain, phase and stability margins can be evaluated. It can be seen that the continuous state variable controller with observer has the lowest stability margin.

The characteristics of the controllers in the time domain can be seen in Figs. 32, 33 and 34. A 10^{0} change in the reference position was applied and the closed loop was simulated with the most precise Model 1 taking into account also the saturation of the actuator. It can be seen again that the continuous controllers are superior to the discrete ones. This is due to the high frequencies which occur if the actuator is saturated and can destabilise the closed loop especially if the actual parameters (e.g. eigenfrequency of the gondola) are apart from the nominal ones (that ones for which the controller was designed). So the digital controllers with relative low sampling rate (25 Hz) have lower gain at high frequencies to make them more robust.

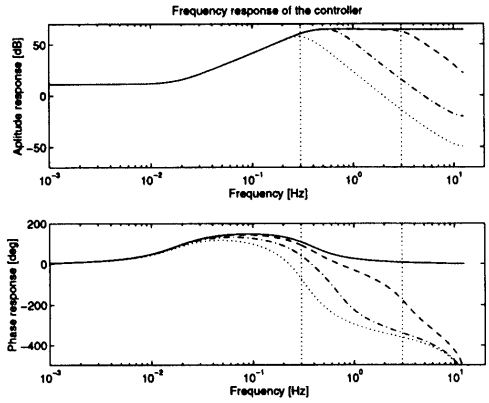


Fig. 29: Frequency response of the second order discrete controller (solid line) and of the discrete controller and fourth order Butterworth filter with cut - off frequencies from 0.3 Hz (dotted line) to 3 Hz (dashed line).

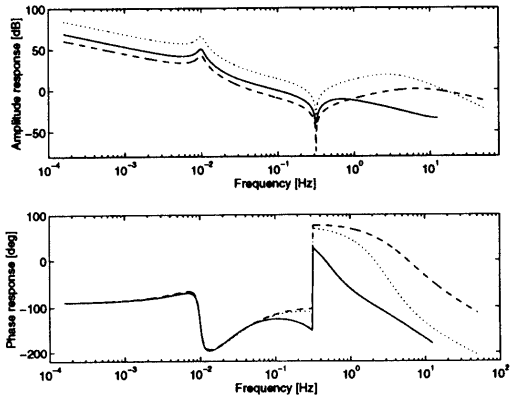


Fig. 30: Bode diagram using: second order compensator (dashed line), third order compensator - state variable controller with observer (dotted line) and third order discrete compensator - state variable controller with observer (solid line).

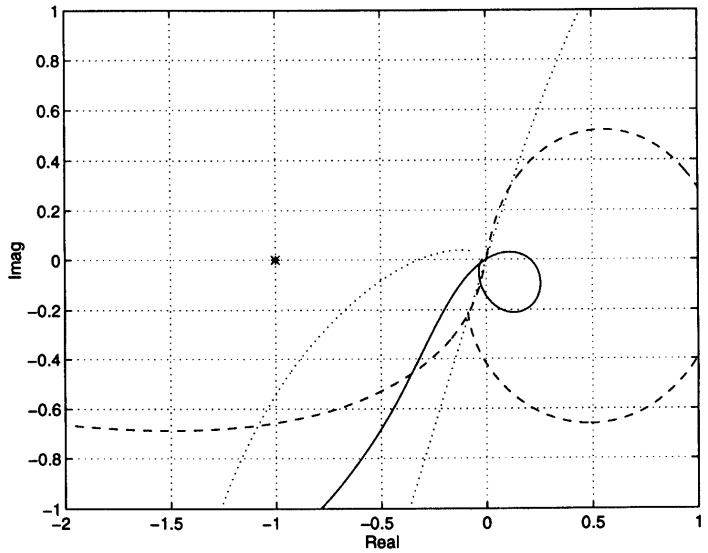


Fig. 31. Nyquist diagram using: second order compensator (dashed line), third order compensator - state variable controller with observer (dotted line) and third order discrete compensator - state variable controller with observer (solid line).

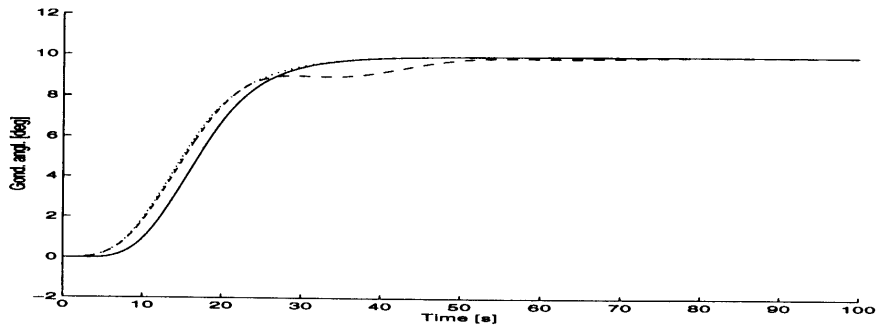


Fig. 32. Step responses using: second order compensator (dashed line), third order compensator - state variable controller with observer (dotted line) and third order discrete compensator - state variable controller with observer (solid line).

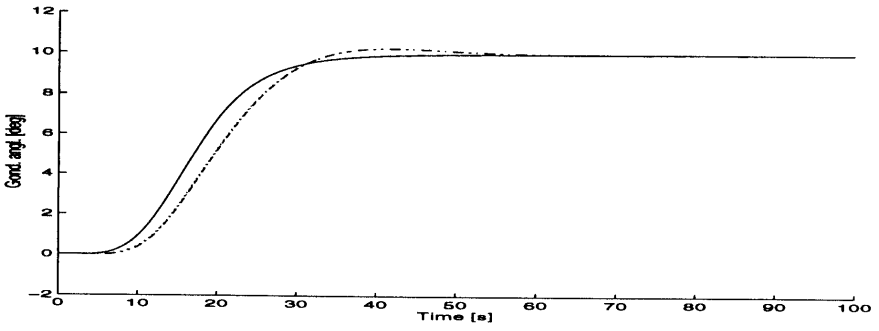


Fig. 33. Step response of the closed loop with the discrete controllers: optimal third order (solid line), optimal third and second order suitable for the Butterworth filter application (dotted and dashed line overlapped).

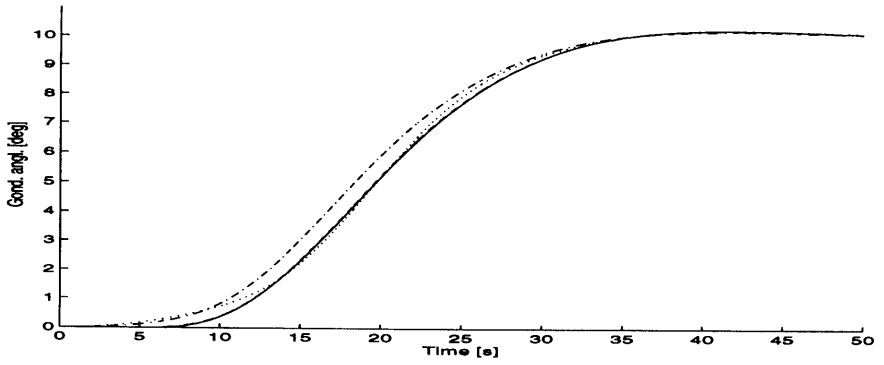


Fig. 34. Step response of the closed loop with the discrete controller only (solid line) and with the discrete controller and fourth order Butterworth filter with cut - off frequencies from 0.3 Hz (dotted line) to 3 Hz (dashed line).

6. DISTURBANCE REJECTION

There are two kinds of disturbances which should be rejected by the controller:

- A ramp type disturbance due to the turning of the balloon
- A sinusoidal disturbance due to the pendulum motion of the suspended gondola.

The rejection of both type disturbances (worst case amplitudes were appiled) by the digital controller was tested by simulations.

To simulate the impact of the turning balloon, a ramp signal with the slope of 360^0 in 30 minutes was added to the position of the gondolla. The results of the simulation are shown in Fig. 35 which depicts the position error (in arc ') and in Fig. (36) showing the correspondig control signal. It can be seen that the controller does not reject the applied ramp disturbance entirely, a steady state position error of 10 arc ' remains. This is due to the proportional character of the controller. An integral type controller would reject this type of disturbances entirely, but with the integral type process in the control loop it would produce huge overshoots for step responses and other stability problems. Other measure to reduce the error is to increase the controller gain.

The results for the sinusoidal disturbance due to the pendulum motion of the suspended gondola are shown in Fig. 37 (the position error) and in Fig. 38 (the corresponding control signal). It can be observed again that the applied controller does not reject the disturbance, a oscillation with the amplitude of 40 arc ' remains. This is again due to the unsufficient gain of the controller. The disturbance rejections could be improved by the costs of the controller's robustness.

7. FEEDFORWARD CONTROLLERS

For tracking a ramp position reference signal and for big changes in the position (where closed loop goes unstable as shown in the section 3.), the velocity and the position feedforward controllers

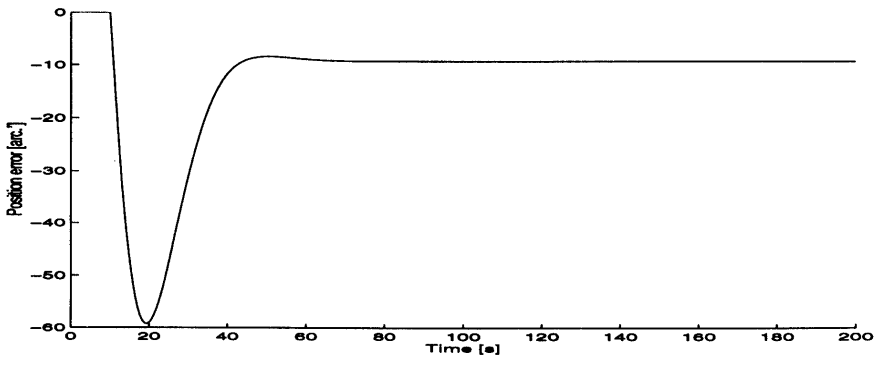


Fig. 35. The position error for the ramp disturbance.

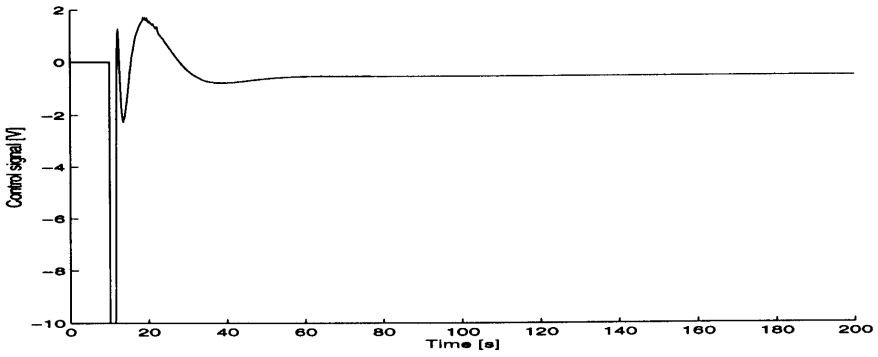


Fig. 36. The control signal for the ramp disturbance.

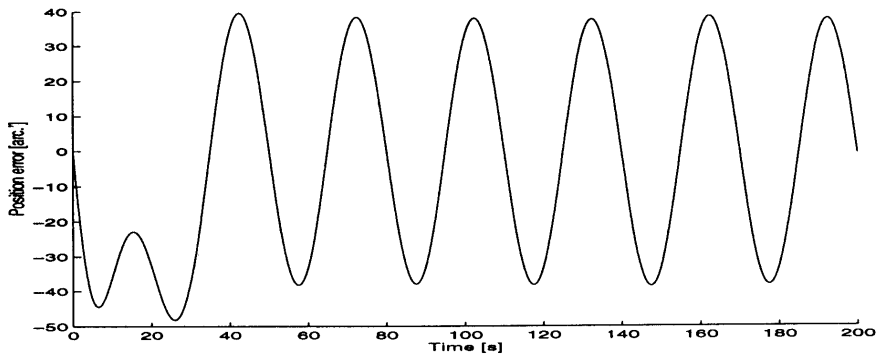


Fig. 37. The position error for the sinusoidal disturbance.

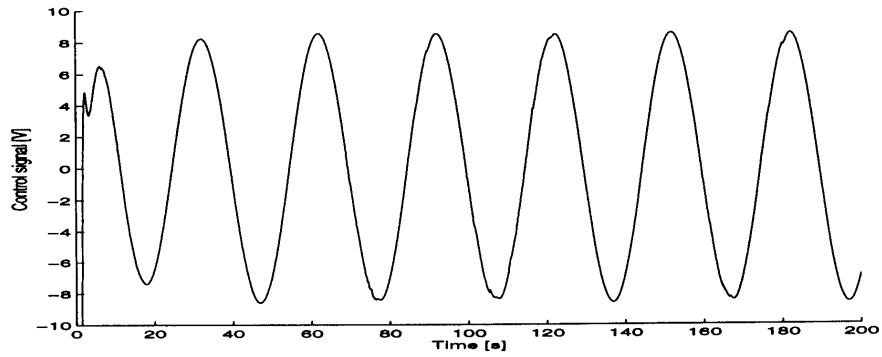


Fig. 38. The control signal for the sinusoidal disturbance.

were designed.

7.1 FEEDFORWARD VELOCITY CONTROLLER

The time optimal velocity controller is based on the simple Model 3. Its principle is shown in Fig. 39. Time optimal control of the second order velocity model (an oscillator) is obtained by a

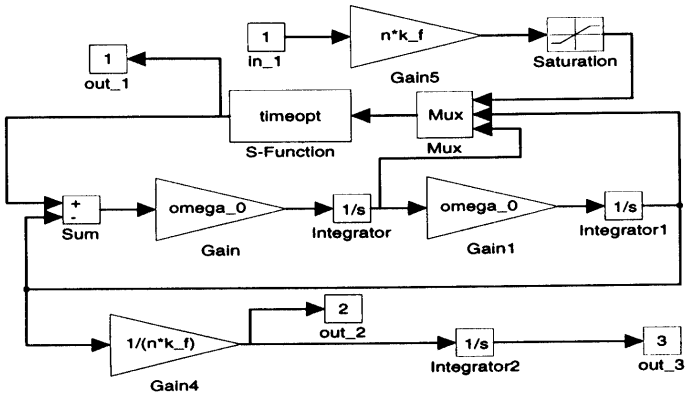


Fig. 39. The time optimal velocity controller.

bang - bang control law. The switching curve is shown in Fig. 40 and is in Fig. 39indicated as an MATLAB S-function. Actual realisation however was done as an S function of the entire controller as a discrete block and is given in Appendix 1. The form of the S-function is suitable to be translated into a C code for actual realisation in an on board computer.

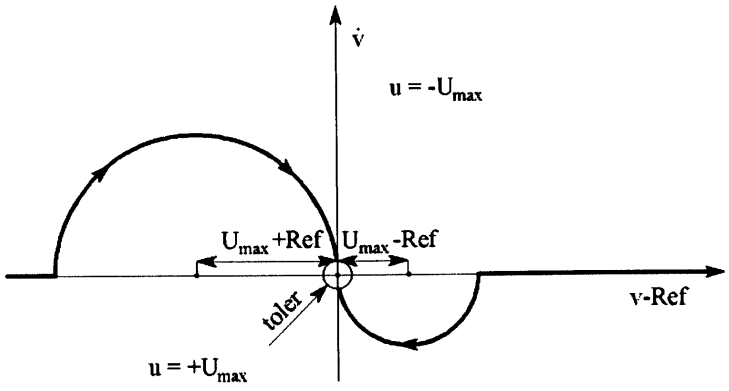


Fig. 40. The switching curve

For smooth behaviour in the vicinity of the desired velocity the bang - bang switching is terminated and a linear control with the damping 0.707 is applied. The input to the time optimal velocity controller is the desired velocity, the outputs are the feedforward control signal which is added to the feedback control signal, the velocity of the model, which is not used for control purposes and the

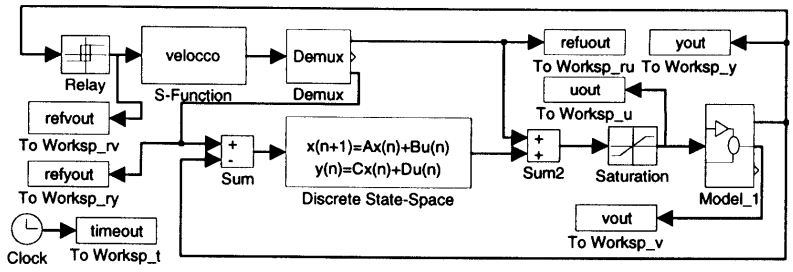


Fig. 41. The simulation scheme for the scanning application of the feedforward velocity controller.

position of the model, which is applied as the reference signal for the classical feedback loop. The complete control configuration for the application of 5 ^{0} scanning is shown in Fig. 41 The control scheme consists of the feedforward velocity controller and the feedback controller. Second order discrete compensator suitable for the application of the Butterworth filter was used as the feedback controller. The feedforward control signal is the signal, which would drive the simplified model of the gondola in shortest time to the desired (reference) velocity. The actual control signal however differs from the feedforward one because the actual system to be controlled differs from the ideal one. The difference is due to the simple Model 3 used for the feedforward controller design and due to non exact parameters of the gondola (mainly its eigenfrequency) used for the feedforward controller design. For this reason the magnitude of the bang - bang control signal should be smaller than the maximum applicable control signal (determined by the saturation of the actuator); only 50% of it (5 V) were used in the simulation.

Fig. 42 depicts the velocity of the gondola response for step change in the velocity reference from 0 ^{0}/s to 1 ^{0}/s. In Fig. 43 the corresponding feedforward and actual control signals are shown. It can

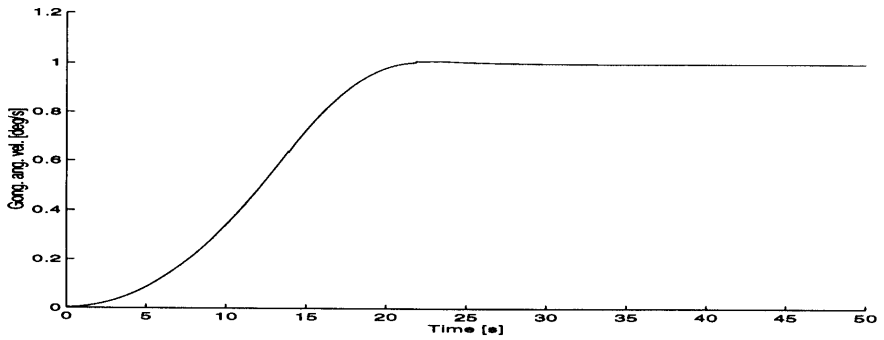


Fig. 42. The angular velocity response of the gondola to the step velocity reference signal of 1 ^{0}/s.

be seen that the change of the velocity is smooth and that the control signal does not saturate which would be the case if only the position feedback controller would be used. The examples shown in Figs. 42 and 43 illustrate the principle of the feedforward velocity controller, actual application of it is the scanning mission.

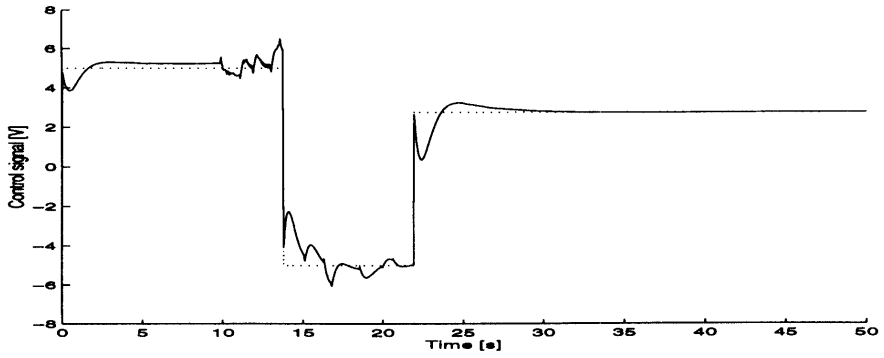


Fig. 43. The feedforward (dotted line) and actual (solid line) control signal of the velocity feedforward control scheme due to the step velocity reference change of 1 ^{0}/s.

The simulation scheme for a typical scanning mission (between 0 ^{0} and 5 ^{0}; period 2 minutes) is shown in Fig. 41. The scanning movement of the gondola is achieved by a relay with hysteresis which toggles the velocity reference between $\pm 5/60 = 0.0833$ ^{0}/s. The time course of the azimuth of the gondola with 30 % increased parameters and the corresponding control signals are shown in Figs. 44 and 45 respectively. It can be seen that the actual scanning range is bigger than 5 ^{0} and that

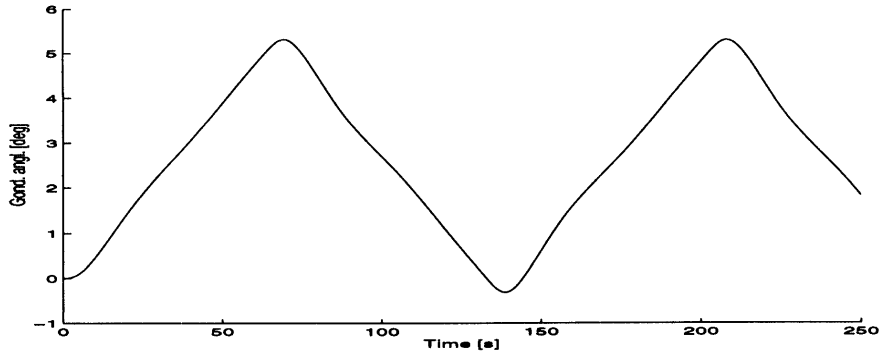


Fig. 44. The azimuth of the gondola using feedforward velocity controller in a typical scanning mission; the actual parameters of the gondola are +30 % increased to the nominal values.

the period is bigger than 2 minutes. This is due to the time lag of the controlled system. An other possibility to realise the scanning reference signal is to apply a square wave ($\pm 5/60 = 0.0833$ ^{0}/s) velocity reference with the period 2 minutes. In this case the period would be exact but the azimuth range would be biased.

Fig. 46 depicts the azimuth errors (in arc ') to the extrapolated position using feedforward control scheme (Fig. 41) for the nominal and ± 30 % deviated parameters (eigenfrequency) of the gondola.

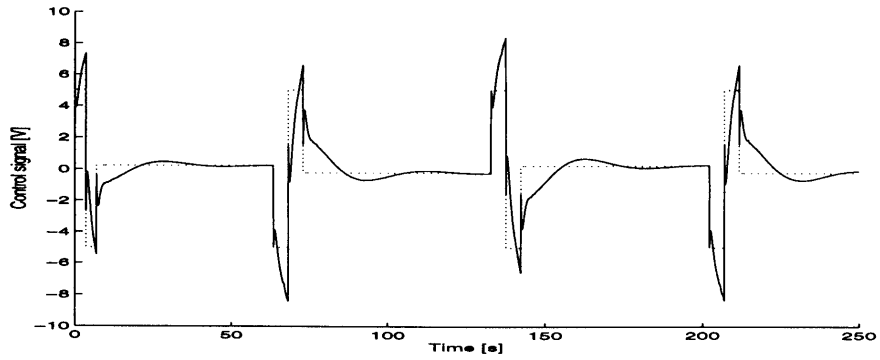


Fig. 45. The feedforward (solid line) and actual control signal of the velocity feedforward control scheme in a typical scanning mission; the actual parameters of the gondola are +30 % increased to the nominal values.

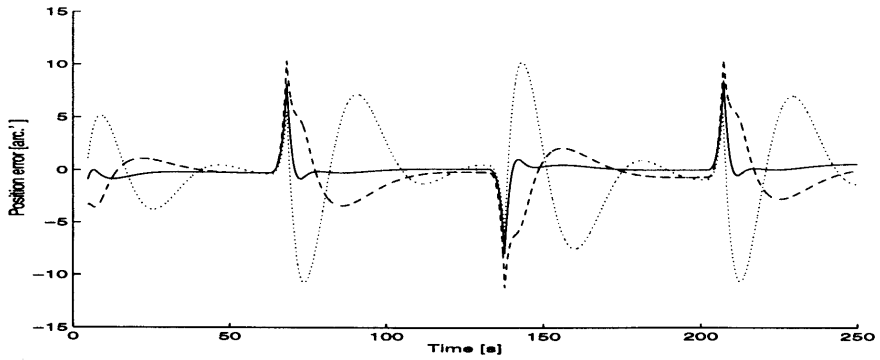


Fig. 46. The azimuth errors (in arc ') to the extrapolated position using feedforward control scheme in a typical scanning mission; the nominal parameters (solid line), 30 % increased eigenfrequency (dotted line) and 30 % decreased eigenfrequency (dashed line).

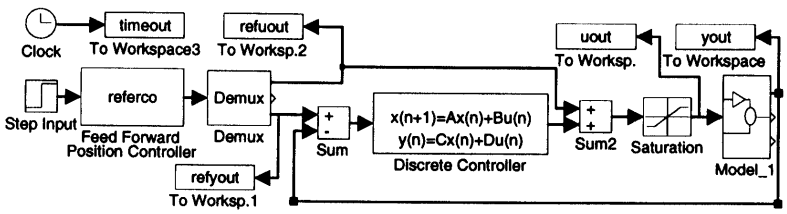


Fig. 47. Simulation scheme of the feedforward position controller.

7.2 Feedforward Position Controller

Due to the saturation of the actuator the designed controllers become unstable for large changes of the reference position signal. The controllers designed in Sections 3. and 4. were optimised for maximal ± 30 % deviations in gondola eigenfrequency and a maximum 10^0 reference change. So a feedforward position controller was designed to be used if larger changes in the reference position are performed. The controller produces a sequence of positive - negative - positive or negative - positive - negative feedforward control signals, which would transfer the simple model of the gondola with nominal parameters from one to another position and a position reference signal, which corresponds to the current position of the simple model. This is not an optimal closed loop position control, it is an open loop point to point transfer. As shown in Fig. 47 the feedback controller corrects the actual control signal due to the deviations of the actual gondola azimuth from the position reference signal of the feedforward controller. The mechanism of the feedforward controller is illustrated in Figs. 48 and 49, which show the position reference signals and the corresponding feedforward control signals respectively, for maximum feedforward signals 10 V, 5 V and 2 V. To allow the feedback controller to be able to compensate the deviations due to non exact model and parameters, the magnitude of the feedforward signal should be smaller than the maximum allowable control signal. The consequence of this is slower amplitude change response but higher robustness against changes in the model structure and parameters.

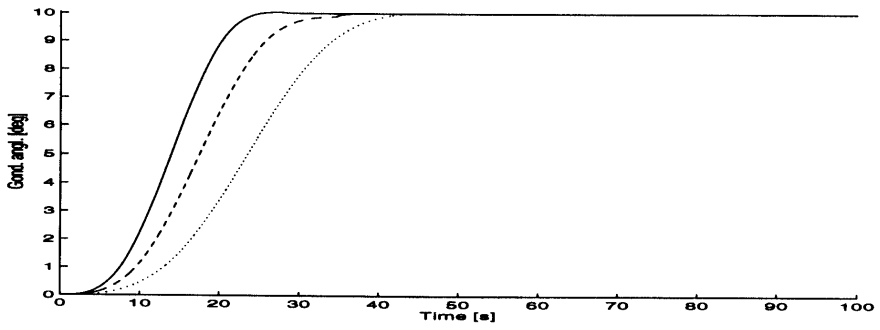


Fig. 48. The position reference signal (ideal response) of the feedforward controller for maximum feedforward control signals 10 V (solid line), 5 V (dashed line) and 2 V (dotted line) respectively.

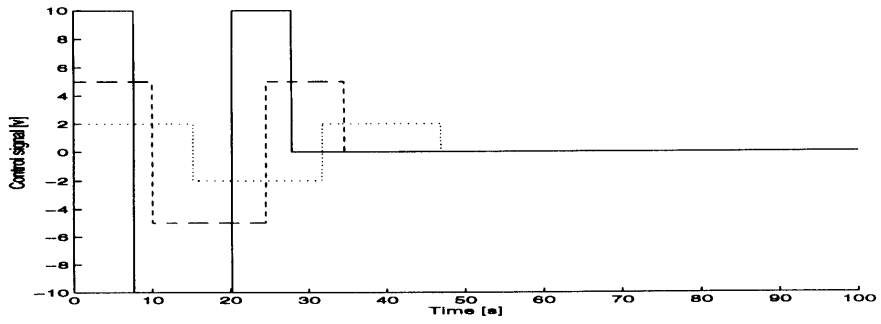


Fig. 49. The feedforward control signal for maximum feedforward control signals 10 V (solid line), 5 V (dashed line) and 2 V (dotted line) respectively.

Figs. 50 and 51 depict the azimuth of the gondola and the corresponding control signals respectively for the nominal and ±30 % deviated eigenfrequency of the gondola and for a 10^0 change in the reference position.

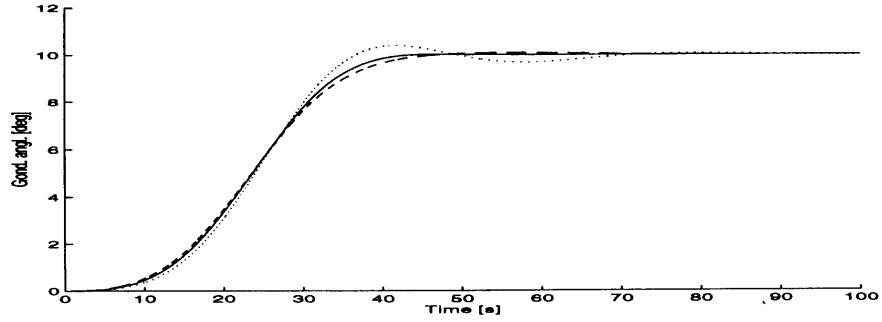


Fig. 50. The closed loop responses to 10^0 reference change with the feedforward controller for the nominal ($\omega_0 = 2\pi/100$ - solid line), 30 % decreased ($\omega_0 = 2\pi/130$ - dotted line) and 30 % increased ($\omega_0 = 2\pi/70$ - dashed line) eigenfrequency respectively.

In Figs. 52 and 53 the same signals for a 100^0 change in the reference position are shown. It can be seen that the signals for a 10^0 change in the reference position are smoother than the corresponding signals without feedforward controllers, while in the case of the 100^0 change in the reference position the closed loop remains stable. There is again a trade-off between the duration of the transition (which may be decreased by increasing the feedforward control signal's magnitude) and the robustness of the control scheme.

8. SELFTUNING

The parameters of the gondola as system to be controlled are not exactly known and change between missions. The dominant dynamics of the gondola is influenced mainly by the moment of iner-

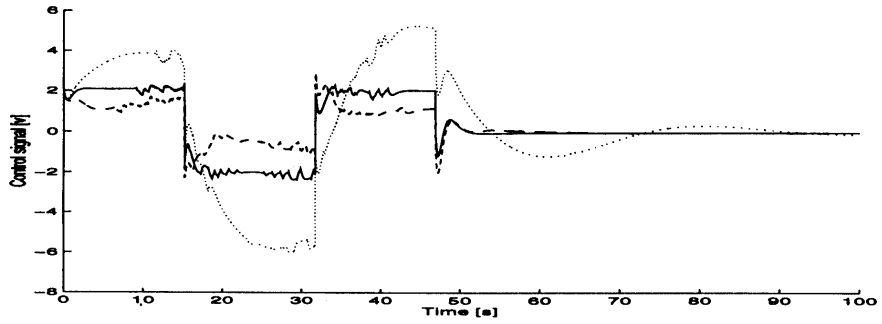


Fig. 51. The control signals for 10^0 reference change (feedforward + feedback controller's signals for the nominal ($\omega_0 = 2\pi/100$ - solid line), 30 % decreased ($\omega_0 = 2\pi/130$ - dotted line) and 30 % increased ($\omega_0 = 2\pi/70$ - dashed line) eigenfrequency respectively.

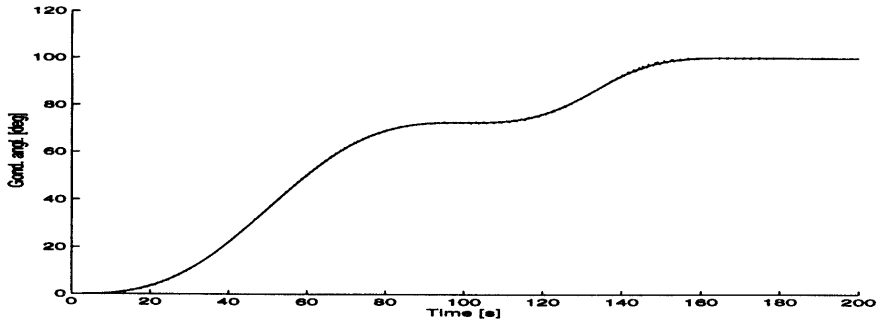


Fig. 52. The closed loop responses to 100^0 reference change with the feedforward controller for the nominal and ±30 % disturbed eigenfrequency respectively (all signals overlapped).

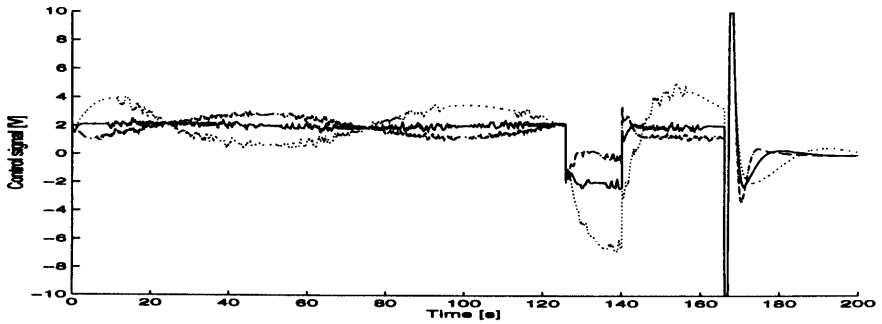


Fig. 53. The control signals for 100^0 reference change (feedforward + feedback controller's signals for the nominal ($\omega_0 = 2\pi/100$ - solid line), 30 % decreased ($\omega_0 = 2\pi/130$ - dotted line) and 30 % increased ($\omega_0 = 2\pi/70$ - dashed line) eigenfrequency respectively.

tia of the gondola and by the spring constant of the rope, which may change significantly. The gain of the controlled system depends mainly on the gear ratio, which is fixed, and on the back generating voltage constant, which may change with the temperature, but not significantly. So the designed controller should be (self)adjustable to the dominant dynamic parameter i.e. eigenfrequency of the gondola around vertical axis, ω_0, see Eq. (6) and robust against unknown dynamics in higher frequency range (due to unknown moment of inertia of the coupling ring, coil resistance, torque constant), see Eq. (4).

A selftuning feature was added to the controller, and was realised with the MATLAB - SIMULINK design tools as two S functions; the identification part of the selftuning controller which is given in the Appendix 3 and the adjustable controller, which is given in the Appendix 4.

The identification procedure (Appendix 3) starts after applying a 0 to 1 step change at the input 1 of the controller and excites the system by a maximum allowable voltage pulse of a duration, which can be set by procedure's parameters (2 seconds were used in simulation studies). The gondola starts to oscillate due to this pulse and an estimation of the turning rate (provided by the Butterworth filter, which yields in the interesting low frequency band quite good estimation of the turning rate; only the angle of the gondola is measured!) is fed to the second input of the procedure. Maximum and minimum of the gondola angle are detected by zero crossings of the turning rate. For safety reasons two successive extremes are evaluated and the oscillation period is given as result if the estimated half periods do not differ for more than a threshold, specified as a parameter of the procedure. After successful estimation the procedure deexcitates the system (not completely, but to a certain degree) by applying a pulse in inverse direction at the appropriate moment. For the time of estimation the procedure serves a "busy" signal which opens the control loop and switches the actuator input to the estimation procedure's control signal output. Other outputs of the procedure are the estimated period and the "data valid" signal. The estimated period is also available in the global area of variables.

The parameter determination of the adjustable controller (Appendix 4) starts after applying a 0 to 1 step change at its input No. 1. Input No. 2. is the normal input to the controller (filtered control error signal). The adjustable controller is a time scaled version of the discrete controller designed in Section 4.. This is possible due to the simple Model 3, which is actually an oscillator plus an integrator. The eigenfrequency of the oscillator part of the model changes with the changing moment of inertia of the gondola and with the spring constant of the rope. The integration part of the simple Model 3 (its gain) does not change with the changing eigefrequency of the gondola. If this would be the case (i.e. if the gain of the integration part would be proportional to the eigenfrequency), the simple model would change with the eigenfrequency of the gondola only in time scaling and the only modification of the controller to be performed would be to scale it in time accordingly. So the controller must be scaled in time and in gain with respect to the changing eigenfrequency of the gondola system. The controller designed in Section 4. was optimised with the nominal period of the gondola oscillations 100 s and with the sampling rate 25 Hz. The time scaling was realised by transforming the designed discrete controller to the continuous time (continuous time controller's parameters are stored in the on board computer) and then transform it back to discrete time with respect to the sampling time and the scaling (ratio of the estimated period and 100 s). Taylor series expansion was applied for the discretisation of the system and all terms with degree higher than 2 were neglected. This design was tested by simulations on the most accurate Model 1 and was proven to be adequate.

Fig. 54 depicts the simulation scheme for a typical mission. It consists of a selftuning controller,

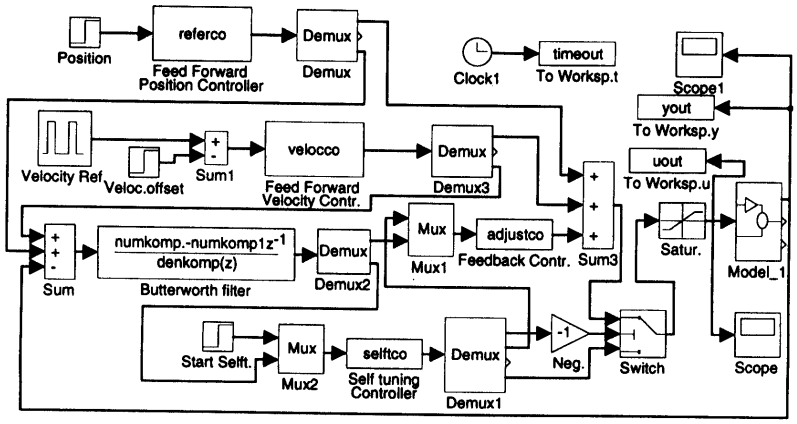


Fig. 54. Simulation scheme for a typical mission.

an adjustable feedback controller, a feedforward position controller and a feedforward velocity controller. The sequence of events is as follows: first the selftuning procedure is started (at $t = 1$ s after beginning of the simulation). At $t = 500$ s the gondola is turned for $50\ ^0$ and at $t = 700$ s the scanning ($2.5\ ^0$ amplitude, 1 minute period is started. The results of the simulation are shown in Fig. 55 (azimuth of the gondola) and Fig. 56 (corresponding control signal) for the actual parameter (oscillating period) being 200 s and in Fig. 57 (azimuth of the gondola) and Fig. 58 (corresponding control signal) for the actual parameter (oscillating period) being 50 s.

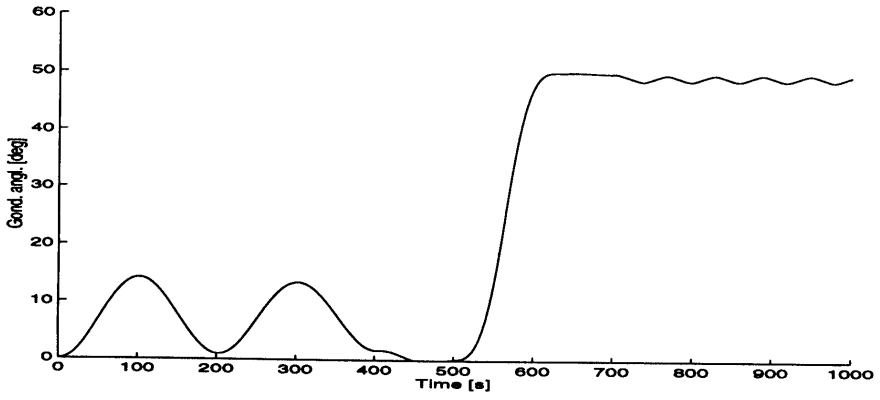


Fig. 55. The time course of the gondola azimuth for a typical mission and the actual period of the gondola oscillations 200 s.

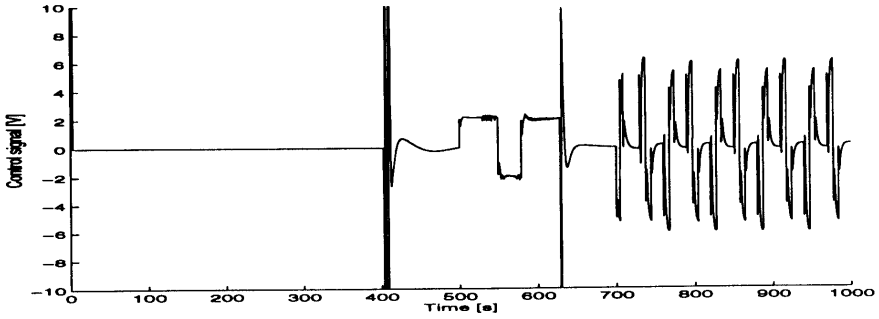


Fig. 56. The time course of the control signal for a typical mission and the actual period of the gondola oscillations 200 s.

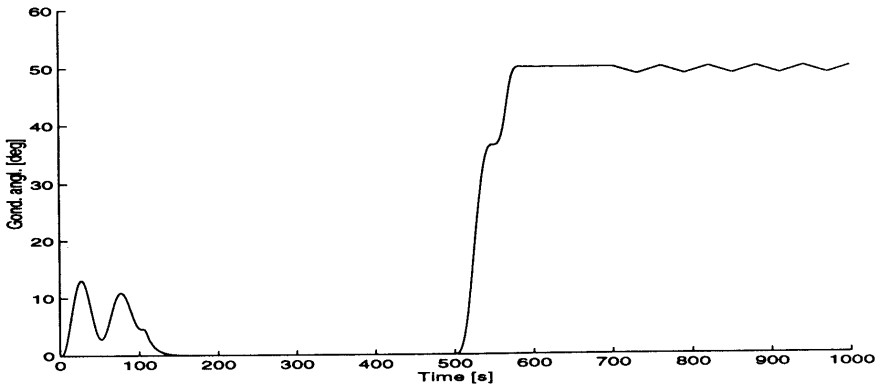


Fig. 57. The time course of the gondola azimuth for a typical mission and the actual period of the gondola oscillations 50 s.

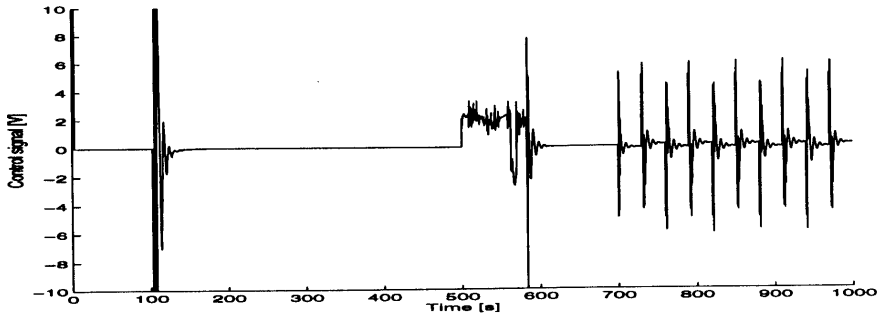
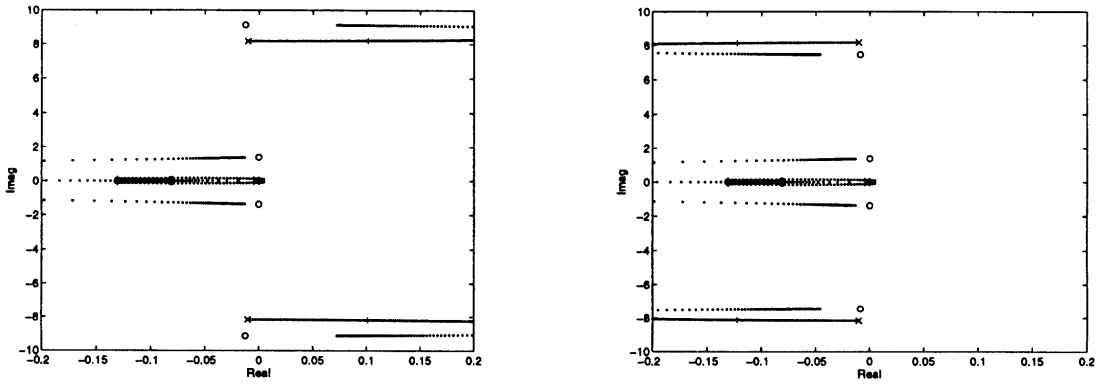


Fig. 58. The time course of the control signal for a typical mission and the actual period of the gondola oscillations 50 s.

9. ADAPTIVE CONTROL OF OSCILLATIONS

As described in Section 2. an undesired motion around the x axis can occur due to the unsymmetric loading of the gondola. The frequency of these oscillations is unknown but it is supposed to be between 0.3 and 3 Hz. More critical however is that the sign of the driving torque, which depends on the relative position of the loading unsymmetry and the current position of the gondola, is not known. In the root locus diagram the submodel for the oscillations around the x axis corresponds to a pair of conjugate complex poles (very close to the imaginary axis) and to a pair of conjugate complex zeros, which in the case of symmetric loading cancels out the poles making the horizontal oscillation part uncontrollable from the input (motor turning the balloon around the vertical axis). In the case of unsymmetric loading however, the root locus may pass to the right half of the complex plane as illustrated in Fig. 59, making the closed loop unstable.



a b

Fig. 59. Root locus plots for a negative (a) and a positive (b) unsymmetric loading.

Two methods for adaptive elimination of this fast parasite part unstability were realised and compared. The first one is an adaptive notch filter [4, 3] which automatically detects the unknown frequency and correspondingly sets its centre frequency. Second one is a fourth order adaptive Butterworth filter which sets its cut off frequency to the undesired motion frequency. This changes the phase of the input signal with this frequency for 180^0. In this application the notch filter is used as a frequency identifier only.

9.1 ADAPTIVE OBSERVER - NOTCH FILTER

Continuous time state space description of the observer - notch filter is as follows:

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} 0 & -\omega_x \\ \omega_x & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 2\zeta\omega_x \\ 0 \end{bmatrix} y \\ y &= [-1 \quad 0]\mathbf{x} + u \end{aligned} \tag{21}$$

where ω_x is the centre notch frequency and ζ is the desired damping of the observer error; $2\zeta\omega_x$ corresponds to the bandwidth of the notch filter. If the centre notch filter frequency ω_x is unknown,

the adaptive notch filter must be applied which automatically adjust its centre frequency according to

$$\dot{\omega}_x = \frac{k_{adapt}}{k_{norm} + \|\mathbf{x}\|^2} \times y \times \mathbf{x}_2 \tag{22}$$

where k_{adapt} is the adaptive gain, k_{norm} the normalising gain and $\|\mathbf{x}\|$ the Euclid norm of the notch filter state. The normalised version of adaptation law is used in order to make the adaptation independent of the amplitude of the input signal.

The discrete version of the adaptive observer - notch filter can be obtained by discretisation of the continuous time form (Eqns. 21 and 22). For realisation in a small on board computer a Taylor series expansion is used and only two terms are taken into account yielding

$$\begin{aligned}
\mathbf{x}(k+1) &= \begin{bmatrix} 1 - \frac{t_s^2}{2}\omega_x^2(k) & -t_s\omega_x(k) \\ t_s\omega_x(k) & 1 - \frac{t_s^2}{2}\omega_x^2(k) \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} 2t_s\zeta\omega_x(k) \\ t_s^2\zeta\omega_x^2(k) \end{bmatrix} y(k) \\
y(k) &= [-1 \quad 0]\,\mathbf{x}(k) + u(k) \\
\omega_x(k+1) &= \omega_x(k) + \frac{t_s k_{adapt}}{k_{norm} + \|\mathbf{x}(k)\|^2} \times y(k) \times \mathbf{x}_2(k)
\end{aligned} \tag{23}$$

Also an "Update Enable " signal is added which disables the frequency estimation update during the application of feedforward signals. This is to prevent the estimated frequency drift due to the non periodic signals, which change the direction of the gondola rotation. The corresponding M file is given in Appendix 5.

Fig. 60 depicts the simulation scheme for a typical scanning mission with the adaptive observer - notch filter in the loop. The selftuning and position feedforward controllers are not shown on the

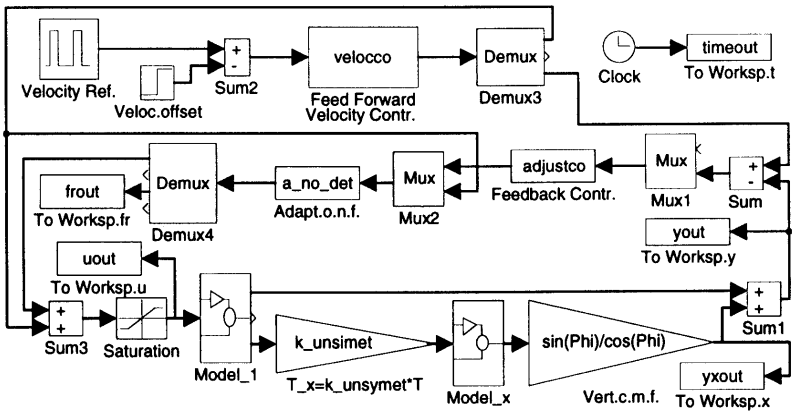


Fig. 60. Simulation scheme for a typical scanning mission with the adaptive observer - notch filter in the loop.

scheme. The parameters for the pendulum motion around the x axis are chosen to be $M_g = 200$ kg, $l_2 = 1$ m , $I_{gx} = 30$ kg m^2 and $T_x = 0.1T$ yielding the eigenfrequency $f_x = \omega_x/2\pi = 1.3$ Hz. The time course of the identified notch filter frequency is shown in Fig. 61. The drift of the estimated frequency after the feedforward signal comes to its steady state is not completely eliminated due

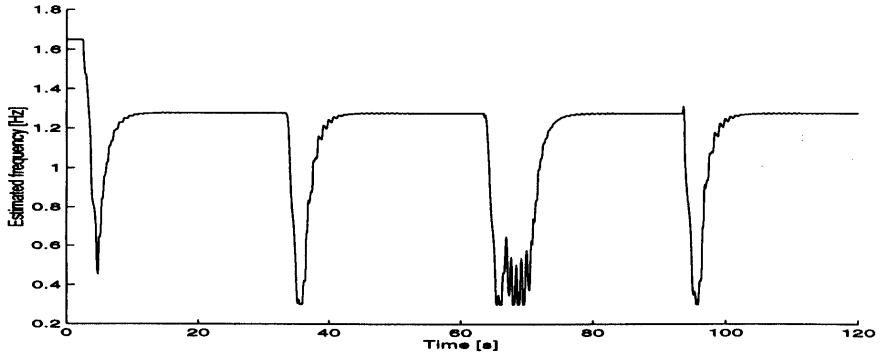


Fig. 61. The time course of the identified notch filter frequency.

to control signal needed to compensate the mismatch of the ideal model used for the velocity feed-forward controller an the actual process model. Fig. 62 depicts the angle Θ_x, i.e. the angle of the gondola with respect to the x axis while in Fig. 63 the corresponding control signal can be seen.

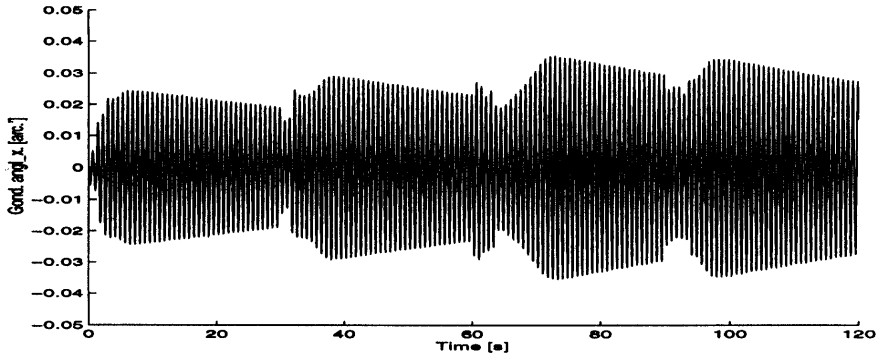


Fig. 62. The time course of Θ_x, (the angle of the gondola with respect to the x axis); adaptive observer - notch filter in the loop.

The notch filter eliminates the undesired oscillations from the input signal by introducing a pair of zeros which cancels out the pair of conjugate complex poles. The oscillatory submodel becomes in this way uncontrollable and oscillations decrease by natural (very low) damping. However due to periodic turn around of the gondola azimuth velocity the undesired oscillations are periodically excited. Since the frequency tracking is lost during the turn around of the speed of the gondola, the oscillations increase until the correct notch filter frequency is recovered.

9.2 Adaptive Butterworth Filter

An other approach to cope with the undesired oscillations is the fourth order adaptive Butterworth filter. The idea of this approach is to control the oscillations rather than to cancel them out of the control signal. The fourth order filter is chosen because it turns the phase of the cut of frequency for 180^0. In this approach the adaptive observer - notch filter is used to estimate the unknown undesired frequency and to detect whether or not the oscillations are increasing. So an oscillation's detection

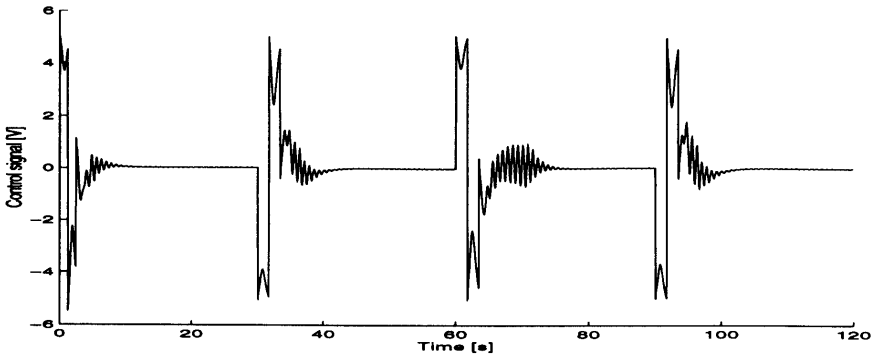


Fig. 63. The control signal; adaptive observer - notch filter in the loop.

algorithm was added to the adaptive notch filter listed in Appendix 5. The algorithm detects the periods of oscillations by detecting the crossing of the zero value of the filtered signal velocity. A period is denoted as an oscillatory period if:

- the integrals of positive and negative half periods do not mismatch for more than a threshold (10 % is used in simulations) and
- the amplitude of oscillations is bigger than a threshold (1V is used in simulations) and
- the estimated frequency is more than a threshold (0.1 rad/s used in simulation) apart from the frequency range (0.3 Hz to 3 Hz) margins.

If the number of successive oscillatory periods is beyond a threshold (3 is used in simulations) the output signal "Oscillations Detected" is issued, the amplitude threshold is set to the current amplitude of oscillations and the threshold for required successive oscillatory periods is doubled. This procedure corresponds to the Nussbaum gain [5, 6] in the adaptive control of systems with an unknown high frequency gain.

If no zero crossing is detected for more than the longest expected period, or if the amplitude of the oscillations decreases below 25 % of initial amplitude threshold, all thresholds are reset to the initial ones.

Initially the adaptive Butterworth filter is turned off. If the oscillations are detected by the adaptive observer - notch filter, then the adjustable Butterworth filter, listed in Appendix 6, is turned on with the cut - off frequency set to the estimated frequency of oscillations. If the Butterworth filter is on and oscillations are detected, it is turned off. For simple realisation in an on board computer the discrete time version of the filter is transformed from the continuous time version by the Taylor expansion (two terms only).

Fig 64 depicts the simulation scheme for a typical scanning mission with the adaptive Butterworth filter in the loop. Fig. 65 depicts the angle Θ_x, i.e. the angle of the gondola with respect to the x axis while in Fig. 66 the corresponding control signal can be seen.

The undesired oscillations increase until the adaptive observer - notch filter detects them and the Butterworth filter is switched on (at approximately 11 seconds). The phase of the control signal

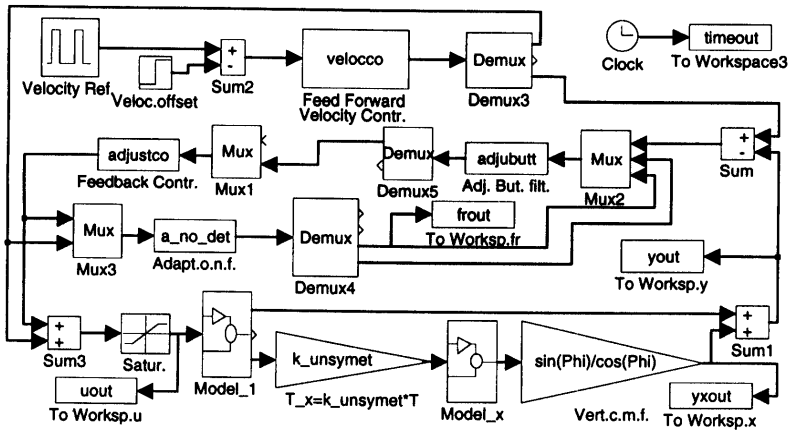


Fig. 64. Simulation scheme for a typical scanning mission with the adaptive Butterworth filter in the loop.

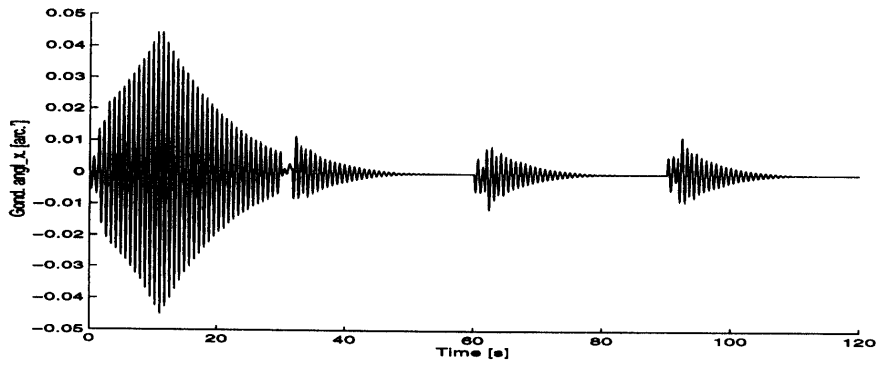


Fig. 65. The time course of Θ_x, (the angle of the gondola with respect to the x axis); adaptive Butterworth filter in the loop.

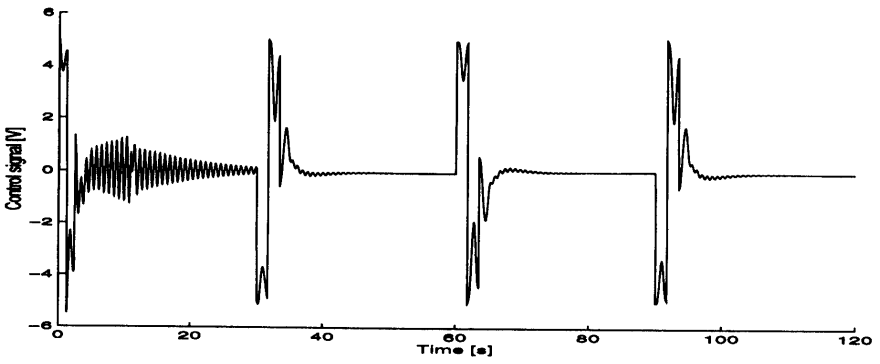


Fig. 66. The control signal; adaptive Butterworth filter in the loop.

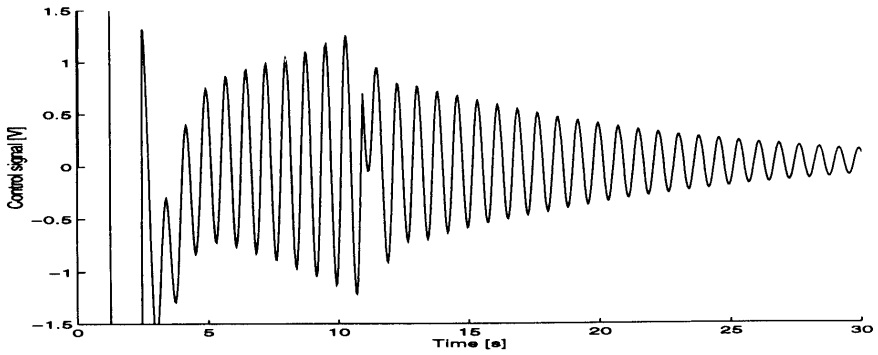


Fig. 67. The control signal - adaptive Butterworth filter in the loop; a detail.

oscillations is changed for 180 ^{0} as shown in Fig. 67, which depicts a detail from the Fig. 66. This stabilises the system and the oscillations decrease with a higher damping than the natural one.

10. CONCLUSION

A control system for the azimuth control of the balloon gondola using a simple actuator was developed. After preliminary studies of the continuous time controllers an optimal discrete time controller suitable for the realisation in a small on - board computer was designed. The controllers were compared in the frequency and in the time domain.

The disturbance rejection of typical disturbances (a ramp signal with the slope 360 ^{0} in half an hour or so; a sinusoidal disturbance with the period 30 seconds or so do not meet the specifications for missions where precise tracking is required. This drawback could be improved by the cost of the controller's robustness. The controller was designed to be stable for 10 ^{0} step reference change and ± 30 % mismatch of the dominant dynamic parameters. Increasing these tolerances improves the controllers characteristics with respect to the disturbance rejections, however at a risk the closed loop to become unstable if the actual tolerances are greater or if the Model 1 is not precise enough and there are some uncertainties in the modelling.

Feedforward position and velocity controllers were designed for application in typical missions such as scanning, changing targeting objects etc. A selftuning procedure for selfadjusting of the digital controller to the dominant dynamic characteristics of the controlled balloon gondola is presented. Finally two adaptive approaches for the elimination of undesired oscillation around a horizontal axis are given and compared. In the first one an adaptive observer - notch filter is used to eliminate the undesired oscillations from the control signal and the oscillations of the gondola vanish by natural damping. The second approach uses the adaptive observer - notch filter as a frequency identifier only. An algorithm for the detection of the oscillation is added to the filter and an adaptive fourth order Butterworth filter is used to invert the phase of the undesired high frequency oscillations which decrease with a higher than the natural damping.

References

[1] Franklin G.F., J. D. Powell and M. L. Workman, *Digital Control of Dynamic Systems*, Addison Wesley, Reading, 1990.

[2] Isermann, R., K. H. Lachmann and D. Matko, *Adaptive Control Systems*, Prentice Hall, New York, 1992.

[3] Kawaguchi J., S. Sawai, H. Yamakawa and S. Matsuda, S - 520 # 18 canard Control Using Adaptive Notch Filter, 4 th Workshop on Astrodynamics and Flight Mechanics, ISAS, Sagamihara, 1994, pp. 175 - 9.

[4] Matko, D., B. Zupančič, F. Bremšak, P. Šega and R. Karba, Phasor Diagram Analysis of the LMS Adaptive Notch Filter, IEEE Tr. A. C. val. AC-25, June 1980, pp. 582 - 5.

[5] Morse, A. S., An Adaptive Control for Globally Stabilizing Linear Systems with Unknown High Frequency Gains, in Analysis and Optimization of Systems, Proceedings of the Sixth International Conference on Analysis and Optimization of Systems, Nice 1984, Springer Verlag, Berlin, pp. 58 - 68.

[6] Willems, J. C. and C. I. Byrnes, Global Adaptive Stabilization in the Absence of information on the Sign of the High Frequency Gain, in Analysis and Optimization of Systems, Proceedings of the Sixth International Conference on Analysis and Optimization of Systems, Nice 1984, Springer Verlag, Berlin, pp. 49 - 57.

[7] Yajima, N., S. Kokaji, S. Hashino and J. Nishimura, A pointing Control System for a Balloon - borne Telescope, Proceedings of the Thirteenth International Symposium on Space Technology and Science, Tokyo 1982, pp. 1189 - 94.

APPENDIX 1

FEEDFORWARD VELOCITY CONTROLLER

```
function[sys, x0]=velocco(t,x,u,flag,nxk_f,U_max_m2,toler,Tsamp)
% Feedforward velocity controller;
% u is the new velocity reference
% x(1) and x(2) are states of the oscillatoty part of the model
% x(3) is integrator of the model
% x(4) is the storage for the calculated control signal
% nxk_f is the product of n and k_f
% U_max_m2 is the maximum magnitude of the feedforw.control signal
% toler is the tollerance to switch off the bang - bang charact.
% Tsamp is the sampling time
%
% omega_0 is calculated from 2*pi/T_estimated;
```

```
% T_estimated is transferred by common block
%
if abs(flag) == 2,
 % the discrete states:
 sys=x;
 if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps,
 % temp is the control signal;
 % temp1 is the velocity in the transient or ref. veloc.elsewhere
 % the switching curve
   Ref=nxk_f*u;
   if abs(Ref)>U_max_m2
      Ref=U_max_m2*sign(Ref);
   end
   y=x(2)-Ref;
   yd=x(1);
   temp1=x(2);%  valid only out of toler
   temp2=0;  % this is the damping inside toler, outside=0!
   dvaummr=2*(U_max_m2-Ref);
   dvaumpr=2*(U_max_m2+Ref);
   if y^2+yd^2<toler^2
      temp=min(abs(Ref),U_max_m2)*sign(Ref);
      temp2=-1.414*yd;
      temp1=Ref;
   elseif y>0
      if y>dvaummr
        temp=-U_max_m2*sign(yd);
      else
        temp=-U_max_m2*sign(yd+sqrt(dvaummr*y-y^2));
      end
   else
      if y<-dvaumpr
         temp=-U_max_m2*sign(yd);
      else
        temp=-U_max_m2*sign(yd-sqrt(-dvaumpr*y-y^2));
      end
   end
   % model for the reference evaluation
   global T_estimated Acd Bcd Ccc Dcc
   omega_0_m=2*pi/T_estimated;
   sys(1)=x(1)-(omega_0_m*Tsamp)*x(2)+Tsamp*omega_0_m*(temp+temp2);
   sys(2)=(omega_0_m*Tsamp)*x(1)+x(2);
   sys(3)=x(3)+(Tsamp/(nxk_f))*temp1;
   sys(4)=temp;
  end
```

```
elseif flag == 3,
  % the output ot the M function:
  sys(1) = x(4);
  sys(2) = x(2)/(nxk_f);
  sys(3) = x(3);
elseif flag == 4,
 % next event (sampling time)
 numSamp = floor(t/Tsamp + 1e+8*eps);
 sys = (numSamp + 1)*Tsamp;
elseif flag  == 0,
 % initialization
 % No. of continuous states, discrete states, system outputs,
 % length of input u, unused feature, direct feedthrough flag
 sys = [0 4 3 1 0 1];
 % initial conditions
 x0 = [0 0 0 0];
else
  % Flags not considered here are treated as unimportant.
    sys = [];
end
```

APPENDIX 2

FEEDFORWARD POSITION CONTROLLER

```
Look up table:
omxx = [0, 0.02, 0.04, 0.06, 0.08, 0.10, 0.20,
        0.40, 0.60, 0.80, 1.00, 1.50,  4.00]
omyy = [0, 0.2799, 0.3590, 0.4172, 0.4653, 0.5074, 0.6730
        0.9184, 1.1220, 1.3068, 1.4807, 1.8895, 3.7880]
function  [sys, x0] = referco(t,x,u,flag,nxk_f,U_max_m2,...
                              omxx,omyy,Tsamp)
% Feedforward position controller;
% u is the new  position reference
% x(1), x(2) & x(3) are times to change output
% x(4) is the reference to be reached
% x(5) is the sign of the change;
% x(6) and x(7) are states of the oscillatoty part of the model
% x(8) is integrator of the model
% nxk_f is the product of n and k_f
% U_max_m2 is the maximum magnitude of the feedforw.contr.signal
% omxx,omyy is the look up table
% Tsamp is the sampling time
```

```
%
% omega_0 is calculated from 2*pi/T_estimated;
% T_estimated is transferred by common block
%
if abs(flag) == 2,
 % the sates:
 sys=x;
 if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps,
  if (x(1)==0)&(x(2)==0)&(x(3)==0) %Contr. is not in transition!
     if abs(u-x(4))>0.1 % reference has changed!
       sys(4)=u;
       % calculate the bang - bang points!
       oma=u-x(4);
       sys(5)=sign(oma);
       temp=abs(oma)*nxk_f/(2*U_max_m2);
       global T_estimated Acd Bcd Ccc Dcc
       omega_0_m=2*pi/T_estimated
       temp1=temp*omega_0_m;
       temp2=fix(temp1/pi);
       temp3=temp1-temp2*pi;
       x1=interp1(omxx,omyy,temp3)/omega_0_m;
       sys(1)=x1+temp2*2*pi/omega_0_m;
       sys(2)=2*(x1-temp3/omega_0_m);
       sys(3)=x1;
     end
     sys(6)=0; % transition is finished; reset the model!
     sys(7)=0;
     sys(8)=x(4)+(x(8)-x(4))*0.96; %bumpless transfer
  else % Transition in progress
     temp=x(1)-Tsamp;
     sys=x;
     sys(1)=temp;
     if temp<=0
       sys(1)=x(2);
       sys(2)=x(3);
       sys(3)=0;
       sys(5)=-x(5);
       if x(2)==0
         sys(5)=0;
       end
     end
   % model for the reference evaluation
     global T_estimated Acd Bcd Ccc Dcc
     omega_0_m=2*pi/T_estimated;
```

```
      sys(6)=x(6)-(omega_0_m*Tsamp)*x(7)+...
             x(5)*U_max_m2*(omega_0_m*Tsamp);
      sys(7)=(omega_0_m*Tsamp)*x(6)+x(7);
      sys(8)=x(8)+(Tsamp/(nxk_f))*x(7);
  end
 end
elseif flag == 3,
  % output:
    sys(1) = x(5)*U_max_m2;  % delj z 2 zaradi varnosti
    sys(2) = x(7)/(nxk_f);
    sys(3)=  x(8);
elseif flag == 4,
  % next event (sampling time)
    numSamp = floor(t/Tsamp + 1e+8*eps);
    sys = (numSamp + 1)*Tsamp;
elseif flag  == 0,
 % initialization
 % No. of continuous states, discrete states, system outputs,
 % length of input u, unused feature, direct feedthrough flag
   sizes(6) = 1;
   sys=[0 8 3 1 0 1];
 % initial conditions
   x0 = [0 0 0 0 0 0 0 0];
else
  % Flags not considered here are treated as unimportant.
    sys = [];
end
```

APPENDIX 3

SELFTUNING CONTROLLER

```
function  [sys, x0] = selftco(t,x,u,flag,imp_dur,wait1_t,...
                      wait2_t,n_of_trials,toler,U_max,Tsamp)
% selftuning controller;
% u(1) is signal to start selftuning 0 to 1 transition
% u(2) is the filtered derivative of the azimuth
% x(1) is the current state of the selftunning procedure
%     -2 - waiting for start on beginning
%     -1 - waiting for start after unsucessful trial
%      0 - waiting for start after sucessful trial
%      1 - applying positive signal to actuator for imp_dur sec.
%      2 - waiting for first maximum
```

```
%       3 - waiting for first minimum
%       4 - waiting for second maximum
%       ... and so on until x(1) = n_of_trials
%    -998 - waiting for a minimum to deexcitate the system
%    -999 - deexitating the system
% x(2) is internal timer for calculating the period
% x(3) is the memory for old input u
% x(4) is the memory: half period if x(1)>0 or period if x(1)=0
% imp_dur is the duration of exitation impulse
% wait1_t is the waiting time to start detection of the next extr.
% wait2_t is the maximum waiting time for an extrem
% n_of_trials is the maximum number of extrema to be evaluated
% toler in % is the allowable tolerance of the time difference
%       between two succesive extrema
% Tsamp is the sampling time
%
% output(1) = busy flag
%              0 - not busy
%              1 - selftunning in progress
% output(2) = flag of succesful estimation
%              0 - estimation in progress; data not valid
%              1 - estimation finished; data valid
%             -1 - estimation uncussesful; data not valid
% output(3) = estimated period
% output(4) = control signal
%
if abs(flag) == 2,  %discrete states
 sys=x;
 if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps
    if x(1)<=0 & u(1)>0.5 & x(3)<0.5
       sys(1) = 1;%  Start the selftuning
       sys(2) = 0;
       disp('Starting selftunning')
    elseif x(1)==1 % exitation phase
          if x(2) >= imp_dur;
             sys(1) = 2;  %  excitation phase over, start phase 2
             sys(2) = 0;  % reset timer
          else
             sys(2)=x(2)+Tsamp;
          end
    elseif x(1) == 2 % looking for first maximum
          if x(2) > wait1_t & u(2) <0
             disp('First maximum detected')
             sys(1) = 3;  % detected, go to the next phase
```

```
        sys(2) = 0;  % reset timer
    else
        sys(2)=x(2)+Tsamp;
    end
elseif x(1) == 3 % looking for the first minimum
    if x(2) > wait1_t & u(2) >0
        disp('First minimum detected')
        sys(1) = 4;  % detected, go to the next phase
        sys(2) = 0;  % reset timer
        sys(4) = x(2);
    else
        sys(2)=x(2)+Tsamp;
    end
elseif x(1)>=4 & fix(x(1))-x(1)<0.5 %looking for next maximum
    if x(2) > wait1_t & u(2) <0
        disp('Next maximum detected')
        if (x(4)-x(2))/x(2)<=toler % OK success!
            sys(1)=-998;
            sys(2)=0;  % reset the timer
            sys(4)=x(2)+x(4); % put the estimated period
            disp('OK')        % to internal memory
        else
            sys(1) = x(1)+1;  % go to the next phase
            sys(2) = 0;  % reset timer
            sys(4) = x(2);
            disp('Failed')
        end
    else
        sys(2)=x(2)+Tsamp;
    end
elseif x(1)>=5 & fix(x(1))-x(1)>0.5 % looking for minimum
    if x(2) > wait1_t & u(2) >0
        if (x(4)-x(2))/x(2)<=toler % OK success!
            sys(1)=-999;
            sys(2)=0;  % reset the timer
            sys(4)=x(2)+x(4)  % put the estimated period
            disp('OK')        % to internal memory
        else
            sys(1) = x(1)+1;  % go to the next phase
            sys(2) = 0;
            sys(4) = x(2);
            disp('Failed')
        end
    else
```

```
            sys(2)=x(2)+Tsamp;
          end
    elseif x(1)==-998 %waiting for the deexcitation of the system
          if x(2) > wait1_t & u(2) >0
              sys(2) = 0;    % reset the timer
              sys(1) = -999
              disp('Starting deexcitate')
          else
            sys(2)=x(2)+Tsamp;
          end
    elseif x(1)==-999  % deexcitate the system
          if x(2) >= imp_dur;
              sys(1)=0   % OK finished!!!!
              global T_estimated Acd Bcd Ccc Dcc
              T_estimated=x(4); % put the value to global
          else
              sys(2)=x(2)+Tsamp;
          end
    end
    if x(1)>=2 & (x(2)>=wait2_t|x(1)>n_of_trials);
       sys(1)=-1;  %test for time over or No. of trials over
    end
    sys(3)=u(1);
 end
elseif flag == 3,
  %outputs
  if x(1)==0 | x(1)== -1 | x(1)== -2
      sys(1)=0;
  else
      sys(1)=1;
  end
    if x(1)==0
      sys(2) = 1;
  elseif x(1) == -1;
      sys(2) = -1;
  else
      sys(2) =0;
  end
  if x(1) == 0
      sys(3) = x(4);
  else
      sys(3)=0;
  end
  if x(1)==1
```

```
      sys(4)=U_max;
  elseif x(1)==-999
      sys(4)=-U_max;
  else
      sys(4)=0;
  end
elseif flag == 4,
  % next event (sampling time)
  numSamp = floor(t/Tsamp + 1e+8*eps);
  sys = (numSamp + 1)*Tsamp;
elseif flag  == 0,
  % initialization
  % No. of continuous states, discrete states, system outputs,
  % length of input u, unused feature, direct feedthrough flag
  sys = [0 4 4 2 0 0]';
  % initial conditions on the states
  x0 = [-2 0 0 0 ];
else
  % Flags not considered here are treated as unimportant.
  sys = [];
end
```

APPENDIX 4

ADJUSTABLE DISCRETE CONTROLLER

```
function  [sys, x0] = adjustc1(t,x,u,flag,Tsamp)
% adjustable controller;
% u(1) is signal to set parameters if u(1) goes from 0 to 1
% u(2) is the input of the controller
% x(1,2) is the state of the controller
% x(3) is the memory for old input u
% x(4) is the memory for the scaling value
% x(5) is the memory for the output value (sample and hold)
% Tsamp is the sampling time
%
% output(1) = control signal
if abs(flag) == 2,   % discrrete states
 sys=x;
 if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps
   scalling=x(4);
   if u(1)==1 & x(3)==0  %set period value
      global T_estimated Acd Bcd Ccc Dcc
```

```
      sys(4)=100/T_estimated;
      scaling=100/T_estimated;
    % Calculation of controller parameters
    Acc=[ 23.579385   25.0259002;
         -25.0259002 -26.3170467];
    Bcc=[-4215.6125;
          4073.0102];
    Acd=eye(2)+Acc*(Tsamp*scaling)+Acc^2/2*(Tsamp*scaling)^2;
    Bcd=(Tsamp*scaling*Bcc+(Tsamp*scaling)^2/2*Acc*Bcc)*scaling;
    Ccc=[1  0];
    Dcc=1569.2313*scaling;
 end
   % Calculation of the new controller states
   global T_estimated Acd Bcd Ccc Dcc
   sys(1:2)=Acd*x(1:2)+Bcd*u(2);
   % Calculation of output
   sys(5)=Ccc*x(1:2)+Dcc*u(2);
   sys(3)=u(1);
 end
elseif flag == 3,
  %  output
  sys(1)=x(5);
elseif flag == 4,
  % next event (sampling time)
  numSamp = floor(t/Tsamp + 1e+8*eps);
  sys = (numSamp + 1)*Tsamp;
elseif flag  == 0,
  % initialization
  % No. of continuous states, discrete states, system outputs,
  % length of input u, unused feature, direct feedthrough flag
  sys =  [0 5 1 2 0 1]';
  % initial conditions
  x0 = [0 0 0 1 0];
  % Initialization : a backup robust controller
  Acc=[ 23.579385   25.0259002;
       -25.0259002 -26.3170467];
  Bcc=[-4215.6125;
        4073.0102];
  global T_estimated Acd Bcd Ccc Dcc
  T_estimated=100;
  scaling=100/T_estimated;
  Acd=eye(2)+Acc*(Tsamp*scaling)+Acc^2/2*(Tsamp*scaling)^2;
  Bcd=(Tsamp*scaling*Bcc+(Tsamp*scaling)^2/2*Acc*Bcc)*scaling;
  Ccc=[1  0];
```

```
  Dcc=1569.2313*scaling;
else
  % Flags not considered here are treated as unimportant.
  sys = [];
end
```

APPENDIX 5

ADAPTIVE OBSERVER - NOTCH FILTER

```
function [sys,x0]=a_no_det(t,x,u,flag,k,lb,ub,Tsamp,zeta,toler)
%  Adaptive Notch filter + oscillation detector
% x(1:2) - states of the filter
% x(3) - estimated frequency
% x(4) - timer for detecting of nonoscillatory behaviour
% x(5) - integral of positive halfperiods
% x(6) - integral of negative halfperiods
% x(7) - counter of detected oscillation periods
% x(8) - flag for detected oscillations
% x(9) - treshold of numbers of detected osc. periods.
% x(10)- treshold for amplitude of oscillations
%
% u(1) - noth filter input
% u(2) - adaptation disable signal (u(2)=0 for adaptation)
% k  - constant for adaptation of the notch filter(-0.2 recomm.)
% lb - lower bound of oscillations (2*pi*0.3)
% ub - upper bound on oscillations (2*pi*3)
% Tsamp - sampling time (1/25)
% zeta - width of the notch filter, ...
%        also damping of transition (0.7 recomm.)
% toler(1) - treshold on the amplitude of oscill. (0.1 recomm)
% toler(2) - initial treshold of nubers of detected oscillatori
%            periods for rising the output (3 recomm.)
% toler(3) - tollerance on frequency bounds (0.05 recomm.)
%
% output(1) - filtered output of the notch filter
% output(2) - estimated oscillations
% output(3) - estimated radial frequency
% output(4) - "oscillations detected" signal
toler_amp=toler(1);
toler_succ_osc=toler(2);
toler_band=toler(3);
if abs(flag)==2
```

```
  % the discrete states:
  sys=x;
if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps,
  % Notch filter parameters
  a=[1-1/2*Tsamp^2*x(3)^2,-Tsamp*x(3);...
      Tsamp*x(3), 1-1/2*Tsamp^2*x(3)^2];
  k_notch=[2*Tsamp*zeta*x(3);...
             Tsamp^2*zeta*x(3)^2];
  sys(1:2)=a*x(1:2)+k_notch*(u(1)-x(1));
  % frequency update
    if abs(u(2))<1
      omg_new=x(3)+k*x(2)*(u(1)-x(1))/(0.01+x(1)^2+x(2)^2);
      if omg_new <= lb
           sys(3)=lb;
      elseif omg_new >= ub
           sys(3)=ub;
      else
           sys(3)=omg_new;
    end
  end
  % Oscillation detector
  if x(1)>0     %  integrate positive and negative halfperiods
     sys(5)=x(5)+x(1);
  else
     sys(6)=x(6)-x(1);
  end
  if sys(1)>=0 & x(1)<0  % detect positive crossing of 0
     if abs((x(5)-x(6)+x(1))/max(x(6),toler_amp/4))<0.1 & ...
            (x(5)+x(6)-x(1))*x(3)*Tsamp/4>x(10) & ...
            x(3)>(lb*(1+toler_band)) & x(3)<(ub*(1-toler_band))
        sys(7)=x(7)+1;  %increase counter of succesf. periods
     else  % reset
           sys(7)=0;
           sys(8)=0;
     end
     sys(4)=0;   % reset watchdog for oscillations
     sys(5)=0;   % Reset integrals of halfperiods
     sys(6)=0;
     if sys(7)>=x(9); % oscillation detected
        sys(7)=0;     % reset the counter
        sys(8)=1;     % set flag for detected oscillations
        sys(9)=2*x(9); % double the treshold
        sys(10)=(x(5)+x(6)-x(1))*x(3)*Tsamp/4; % set ampl. tr.
     end
```

```
        if x(4)>2*pi/lb|(x(5)+x(6)-x(1))*x(3)*Tsamp/4<toler_amp/4
           sys(4)=0;     % no oscillations; reset everything
           sys(5)=0;
           sys(6)=0;
           sys(7)=0;
           sys(8)=0;
           sys(9)=toler_succ_osc;
           sys(10)=toler_amp
      end
     end
   end
elseif flag==3
   % the output ot the M function:
   sys=[u(1)-x(1); x(1); x(3); x(8)];
elseif flag==4
 % next event (sampling time)
    numSamp = floor(t/Tsamp + 1e+8*eps);
    sys = (numSamp + 1)*Tsamp;
elseif flag==0
 % initialization
 % No. of continuous states, discrete states, system outputs,
 % length of input u, unused feature, direct feedthrough flag
   sys=[0 10 4 2 0 0];
 % initial conditions
   x0=[0 0 (lb+ub)/2 0 0 0 0 0 toler_succ_osc toler_amp];
else
  % Flags not considered here are treated as unimportant.
    sys = [];
end
```

APPENDIX 6

ADJUSTABLE BUTTERWORTH FILTER

```
function  [sys, x0] = adjubutt(t,x,u,flag,Tsamp)
% adjustable Butterworth filter;
% u(1) is the input of the filter
% u(2) is signal to set parameters if u(2) goes from 0 to 1
% u(3) is the cuttoff frequency
% x(1:4) is the state of the controller
% x(5) id the memory for old u(2)
% x(6) is the state for "on - off"
%
```

```
% Tsamp is the sampling time

% output(1) = control signal
% output(2) = derivative of the filtered output
if abs(flag) == 2,   % discrrete states
 sys=x;
 if abs(round(t/Tsamp)-(t/Tsamp)) < 1e+8*eps
   if u(2)==1 & x(5)==0  % oscillation detected
      global T_estimated Acd Bcd Ccc Dcc Akomp Bkomp Ckomp Dkomp
      if x(6)==0  % Filter is off, turn it on
        disp('turn filter on')
        Apom=[[-2.6131 -3.4142 -2.6131 -1.0000];
              eye(3),[0;0;0]];
         Bpom=[1;0;0;0];
         %controller parameters
         % use global for memory
         uuu=u(3);
         Akomp=eye(4)+Apom*(Tsamp*uuu)+Apom^2/2*(Tsamp*uuu)^2;
         Bkomp=Tsamp*uuu*Bpom+(Tsamp*uuu)^2/2*Apom*Bpom;
         Ckomp=[0     0     0     1];
         Dkomp=0;
      else  % turn it off
          disp('turn filter off')
          Ckomp=[0     0     0     0];
          Dkomp=1;
      end
   end
   % Calculation of the new controller states
   global T_estimated Acd Bcd Ccc Dcc Akomp Bkomp Ckomp Dkomp
   sys(1:4)=Akomp*x(1:4)+Bkomp*u(1);
   sys(5)=u(2);   % old u(2) value, to detect front
 end
elseif flag == 3,
   % output
   global T_estimated Acd Bcd Ccc Dcc Akomp Bkomp Ckomp Dkomp
   sys(1)=Ckomp*x(1:4)+Dkomp*u(1);
   sys(2)=[0 0 1 0]*x(1:4);    % derivative
elseif flag == 4,
   % next event (sampling time)
   numSamp = floor(t/Tsamp + 1e+8*eps);
   sys = (numSamp + 1)*Tsamp;

elseif flag  == 0,
  % initialization
```

```
  % No. of continuous states, discrete states, system outputs,
  % length of input u, unused feature, direct feedthrough flag
  sys =  [0 6 2 3 0 1]';
  % initial conditions
  x0 = [0 0 0 0 0 0];
  % Initialization : a backup filter cutt off = 2*pi*1Hz
    Apom=[[-2.6131 -3.4142 -2.6131 -1.0000];
          eye(3),[0;0;0]];
     Bpom=[1;0;0;0];
     %controller parameters
     % use global for memory
     global T_estimated Acd Bcd Ccc Dcc Akomp Bkomp Ckomp Dkomp
     uuu=2*pi*1;
     Akomp=eye(4)+Apom*(Tsamp*uuu)+Apom^2/2*(Tsamp*uuu)^2;
     Bkomp=Tsamp*uuu*Bpom+(Tsamp*uuu)^2/2*Apom*Bpom;
     Ckomp=[0     0     0     0];
     Dkomp=1;    % controller is off
else
  % Flags not considered here are treated as unimportant.
  sys = [];
end
```