TTK4135 Optimization and Control
Lab Report

750001, 750002, 750003, and 750004
Group 99

May 1, 2016

Department of Engineering Cybernetics
Norwegian University of Science and Technology

Abstract

This document outlines a few important aspects of the lab report. It contains some advice on both content and layout. The Latex source for this document is also published, and you can use it as a template of sorts for your own report.

When you write your own report, this section (the abstract) should contain a *very* short summary of what the lab is about and what you have done.

Contents

Introduction

This report will describe the process we've gone through to model and control the helicopter system as assigned. In particular, the problems deal with dynamic optimization problems, their discretization and solution. The report includes model derivation, calculations and parameter values, plots, discussion about problems and results, Simulink diagrams and MATLAB code.

The problems and their related discussion, plots, etc. are presented as distinct subwsections. However, the MATLAB code and Simulink diagrams are placed at the end of the document, and are referenced where ever necessary.

Problem Description

There are four problems, all of which relate to the table mounted lab helicopter, illustrated in figure (1). It pivots and rotates about a fixed point. The first "problem" is just getting to know the system and verifying that the setup works. The other three problems are concerned with dynamic optimization problems and control related of the helicopter. The individual problems are described in more detail in their respective subsections.

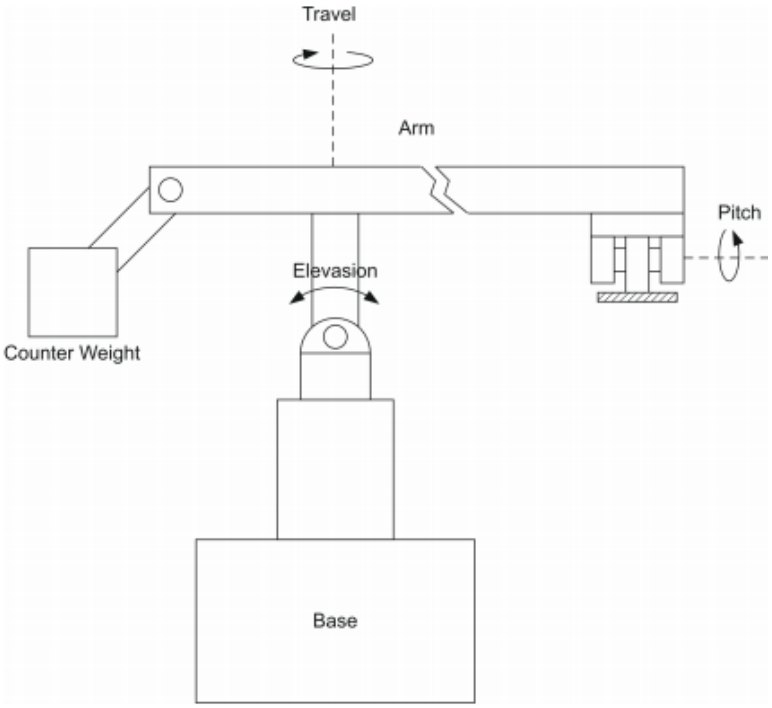

Figure 1: Helicopter model

The system is modeled and controlled using MATLAB and Simulink. All the movement we model and control can be described as elevation, travel and pitch. The system model can be summarized by the equation set (1), with variables from table 1 and table 2.

$$\ddot{e} + K_3 K_{ed} \dot{e} + K_3 K_{ep} e = K_3 K_{ep} e_c \tag{1a}$$

$$\ddot{p} + K_1 K_{pd} \dot{p} + K_1 K_{pp} p = K_1 K_{pp} p_c \tag{1b}$$

$$\dot{\lambda} = r \tag{1c}$$

$$\dot{r} = -K_2 p \tag{1d}$$

Table 1: Parameters and values from (Assignment, 2014, p. 10)

Symbol	Parameter	Value	Unit
l_a	Distance from elevation axis to helicopter body	0.63	m
l_h	Distance from pitch axis to motor	0.18	m
K_f	Force constant motor	0.25	N/V
J_e	Moment of inertia for elevation	0.83	$\mathrm{kg\,m^2}$
J_t	Moment of inertia for travel	0.83	$\mathrm{kg\,m^2}$
J_p	Moment of inertia for pitch	0.034	$\mathrm{kg\,m^2}$
m_h	Mass of helicopter	1.05	kg
m_w	Balance weight	1.87	kg
m_g	Effective mass of the helicopter	0.05	kg
K_p	Force to lift the helicopter from the ground	0.49	N

Table 2: Variables (Assignment, 2014, p. 10)

Symbol	Variable
l_a	Distance from elevation axis to helicopter body
l_h	Distance from pitch axis to motor
K_f	Force constant motor
J_e	Moment of inertia for elevation
J_t	Moment of inertia for travel
J_p	Moment of inertia for pitch
m_h	Mass of helicopter
m_w	Balance weight
m_g	Effective mass of the helicopter
K_p	Force to lift the helicopter from the ground

1 Repetition/Introduction to Simulink/QuaRC

2 Optimal Control of Pitch/Travel without Feedback (10.2)

2.1 Continous Linearized Model

We represent equations 1 in continous state space form $\dot{\mathbf{x}} = \mathbf{A_c}\mathbf{x} + \mathbf{B_c}\mathrm{u}$:

$$\begin{bmatrix} \dot{\lambda} \\ \dot{r} \\ \dot{p} \\ \ddot{p} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -K_2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -K_1K_{pp} & -K_{pd} \end{bmatrix} \begin{bmatrix} \lambda \\ r \\ p \\ \dot{p} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ K_1K_{pp} \end{bmatrix} p_c \tag{2}$$

As we can see, the state-space model uses $u = p_c$ which is used as reference for the Pitch's PD-controller as indicated in Figure 2. It should be noted that this continous state-space model is marginally stable, as it has two zero eigenvalues in addition to two negative ones.

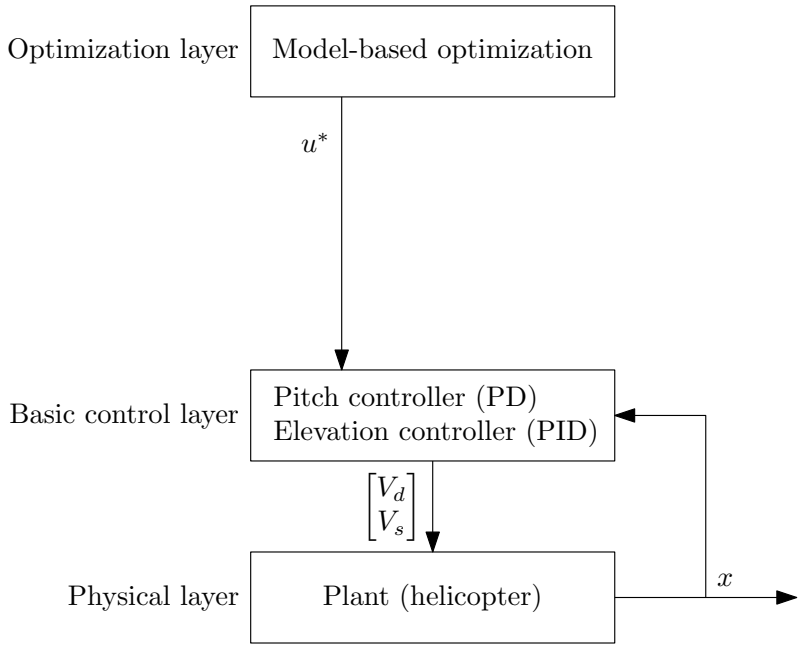

Figure 2: Illustration of the layers in the control hierarchy used in Section 10.2. from (Assignment, 2014, p. 15)

2.2 Discretization of the Continous State Space Model

To apply discrete-time algorithms like finite horizon LQR, the system has to be discretized. We achieve this through the use of the forward Euler

method:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A_c}\mathbf{x} + \mathbf{B_c}u \tag{3a}$$

$$\frac{d\mathbf{x}}{dt} \simeq \frac{\mathbf{x_{k+1}} - \mathbf{x_k}}{h} \tag{3b}$$

$$\mathbf{x_{k+1}} = \mathbf{x_k} + h\frac{d\mathbf{x}}{dt} \tag{3c}$$

Where h is the sampling time. Inserting equation 3a yields us the discrete state space form:

$$\mathbf{x_{k+1}} = \underbrace{(h\mathbf{A_c} + \mathbf{I})}_{\mathbf{A_d}}\mathbf{x_k} + \underbrace{h\mathbf{B_c}}_{\mathbf{B_d}}u_k \tag{4}$$

2.3 Optimization of the problem

We now want to find the optimal trajectory for moving the helicopter from:

$$\mathbf{x_0} = \begin{bmatrix} \lambda_0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \Rightarrow \qquad \mathbf{x_N} = \begin{bmatrix} \lambda_f \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

The elevation is assumed constant (and therefore, obviously, the elevation rate as well), while $\lambda_0 = \pi$ and $\lambda_f = 0$. The optimization problem we wish to model is:

$$\phi = \sum_{i=1}^{N}((\lambda_i - \lambda_f)^2 + qp_{ci}^2) \tag{6a}$$

$$s.t. \qquad \mathbf{A_{eq}}\mathbf{z} = \mathbf{B_{eq}} \tag{6b}$$

$$\mathbf{z}_{min} \leq \mathbf{z} \leq \mathbf{z}_{max} \tag{6c}$$

With the z boundary constraints in 6c given by:

$$|p_k| \quad \leq \quad \frac{\pi}{6}, \qquad k\epsilon\{1, \ldots, N\} \tag{7}$$

These constrains causes us to create vectors for maximum and minimum values the z variables can have:

$$\mathbf{x_b} = \begin{bmatrix} \frac{\pi}{6} & 0 & 0 & 0 \end{bmatrix}^T \tag{8}$$

$$z_{\mathbf{min}} = \begin{bmatrix} -\mathbf{x_b} \\ \vdots \\ -\mathbf{x_b} \\ \frac{-\pi}{6} \\ \vdots \\ \frac{-\pi}{6} \end{bmatrix}, \quad z_{\mathbf{max}} = \begin{bmatrix} \mathbf{x_b} \\ \vdots \\ \mathbf{x_b} \\ \frac{\pi}{6} \\ \vdots \\ \frac{\pi}{6} \end{bmatrix} \tag{9}$$

The full horizon state space model serves as the equality constraints, which is given by the matrix equation:

$$\underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \cdots & \mathbf{0} & -\mathbf{B_d} & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ -\mathbf{A_d} & \ddots & \ddots & \ddots & \vdots & \mathbf{0} & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \cdots & \cdots & -\mathbf{A_d} & \mathbf{I} & \mathbf{0} & \cdots & \cdots & \cdots & -\mathbf{B_d} \end{bmatrix}}_{\mathbf{A_{eq}}} \begin{bmatrix} \mathbf{x_1} \\ \vdots \\ \mathbf{x_N} \\ u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{ax_0} \\ \mathbf{0} \\ \vdots \\ \vdots \\ \mathbf{0} \end{bmatrix}}_{\mathbf{B_{eq}}} \tag{10}$$

With four states and one input the full cost matrix becomes a $5N \times 5N$ block diagonal matrix given by:

$$\mathbf{R_i} = \begin{bmatrix} r_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ with } r_1 = 1 \qquad and \qquad q_i, \quad q_i \epsilon \mathbb{R}^{1x1} \tag{11}$$

Which gives the full cost matrix for the entire horizon:

$$H = \begin{bmatrix} \mathbf{R_1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \cdots & \vdots & \vdots & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{R_N} & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ 0 & \cdots & \cdots & 0 & q_0 & 0 & \cdots & 0 \\ \vdots & \ddots & \cdots & \vdots & 0 & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 0 & \cdots & 0 & q_N \end{bmatrix} \tag{12}$$

NB: As the assignment text uses the notation q for the input weight, and as this would colide with the normal notation for the weight matrices for

the states ($\mathbf{Q}$), we chose the notation $\mathbf{R}$ instead.

To solve the optimization problem, as seen in MATLAB code **??**, we use he function `quadprog()` to extract the optimal x, u and λ values. The input arguments were the cost matrix H, $\mathbf{A_{eq}}$ and $\mathbf{B_{eq}}$ as could be found in equation 10, and the constrant vectors $z_{\mathbf{min}}$ and $z_{\mathbf{max}}$. The rest of the arguments are either empty vectors or omitted as they are not used.

2.4 Results & discussions

We simulated the problem with a horizon of N = 100, and timestep of h = 0.25, which meant a 25 second flight.

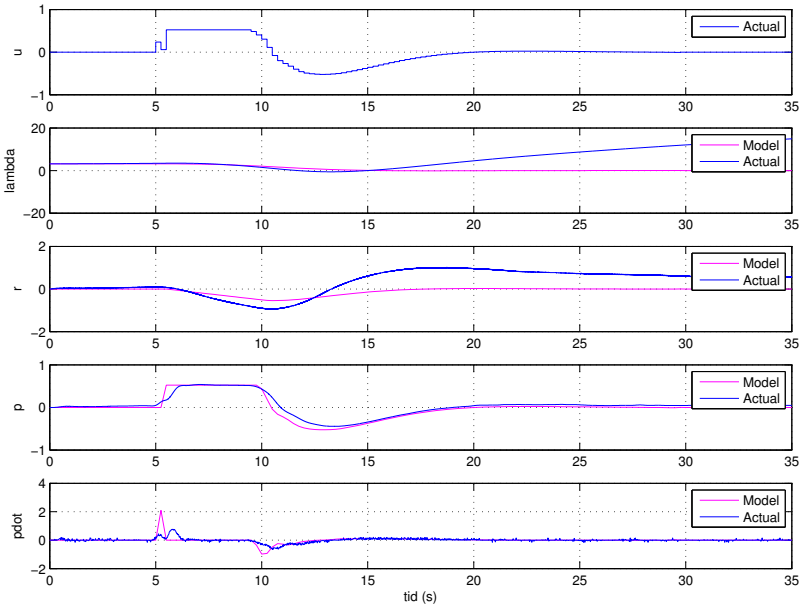

Figure 3:

The deveations are as seen relative small in the "active" part, but the ending $\dot{\lambda}$ causes the linear increase in λ error from the model. This problem is further discussed at the next paragraph.
The deviations in f.ex r is best explained by us using a simplified model. The linearization causes our model to be more faulty as we move further away from the linearization point, which is why the error first appears when the model starts to move, after the initial 5 seconds. This is not true however for the p and $\dot{p}$, as they are linearly

The results made it clear that even though the helicopter found the optimal trajectory, it made no requirements for the other states, which caused the helicopter to not be at rest after 25 seconds. This is mainly because the state $\dot{\lambda}$ was non-zero at the end of the input-serie given to the helicopter. When given extra simulation time, after 25 sec, with no other control input then the elevation regulator, the helicopter continous to spin with the same travel rate it had at t=25. This can however be solved by adding zeroes as constraints for $\mathbf{x_N}$, which will force the helicopter to be at rest at the last timestep.

NB: It should be noted that we nowhere in our simulink diagram (figure 5) do anything with the issue that λ in the model ($\lambda_0 = \pi$) and according to the helicopter ($\lambda_0 = 0$). This is because we use no form of feedback, and therefore only imported inputs from our data model. We only needed to manipulate the plot data by adding π to every actual λ output from the helicopter.
This will however be an issue in later tasks when feedback is added, so it will be further discussed then.

3 Optimal Control of Pitch/Travel with Feedback (LQ)

Next an infinite horizon quadratic controller, LQR, will be implemented according to the feedback function:

$$u_k = u_k^* - K^T(x_k - x_k^*) \tag{13}$$

Where x_k^* and u_k^* is the estimated state and input respectively. This ensures that deviations from the estimated states leads to a change in input. The LQR is implemented as shown in simulink diagrams 6 and 7. The elevation and elevation rate states are not included in the LQR as they are assumed to be constantly zero, and are therefore removed by a simple matrix multiplication. To find the gain matrix K the MatLab function `dlqr()` is used. The weighting matrices **Q** and **R** are given in the MatLab code for problem three as:

$$\mathbf{Q} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 2.5 \end{bmatrix}, \mathrm{R} = 1 \tag{14}$$

For states and input respectively.

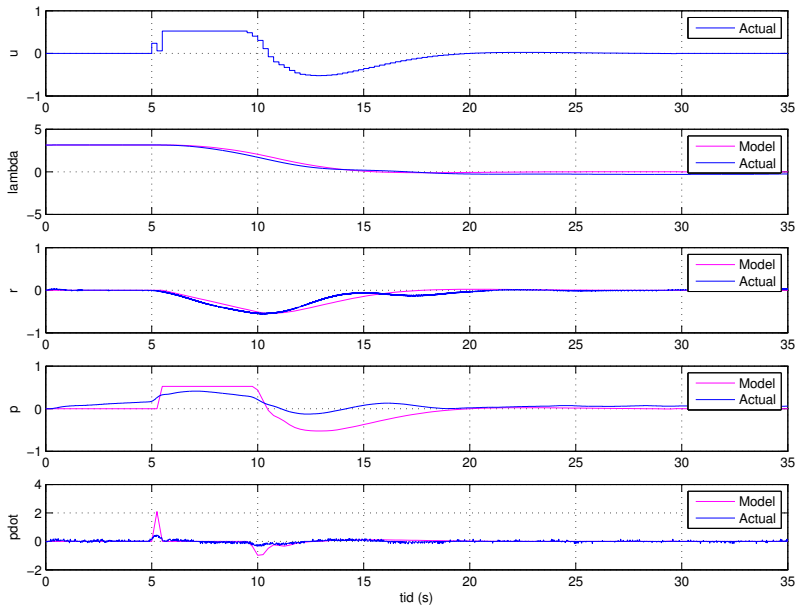

Figure 4:

3.1 Results and Discussion

The data gathered from the model with LQR implemented is presented in figure 4, as before compared to the predicted optimal states. With LQR we see considerable less deviations in the travel and travel rate states, although at a price of accuracy in pitch. It is easy to connect these results to the mathematics of the system by looking at the state weighting matrix $\mathbf{Q}$. Deviations in the travel state are penalized more than any of the other states and should therefore follow the predictions nicely. This will also lead to some penalties in travel rate as these two states are closely connected. However this will in turn lead to deviations in pitch, a less penalized state, as it is used to regulate the travel and travel rate states especially these states move away further away from their linearization point.

4 Optimal Control of Pitch/Travel and Elevation with and without Feedback

4.1 Results and Discussion

5 Discussion

A section like this does not have to be long, but write a few short paragraphs that show you understand what you have been doing and how the different results relate to each other.

A MATLAB Code

A.1 Problem 2

A.1.1 System description

```
%%%  Parameters  %%%
x0 = [pi 0 0 0]';
h  = 0.25;
N  = 100;
M = N;
nx = 4;
nu = 1;
n  = N * (nx + nu);

%%%  Continous State Space %%%
Ac = [0 1     0          0 ;
      0 0   -K_2         0 ;
      0 0     0          1 ;
      0 0 -K_1*K_pp -K_1*K_pd];

Bc = [0 ; 0 ; 0 ; K_1*K_pp];


%%%  Discrete State Space %%%
Ad = eye(4) + h * Ac;
Bd = h * Bc;
```

A.1.2 Constraints

```
%%%  Constraints  %%%
Q1 = diag([1,0,0,0]);
R1  = 1;
p_max = 30*pi/180;
x_max = [inf; inf; p_max; inf];
x_min = [-inf; -inf; -p_max; -inf];
u_max = p_max;
u_min = -p_max;

Q = genq2(2*Q1, 2*R1, N, M, nu);

Aeq = gena2(Ad,Bd,N,nx,nu);
Beq = zeros(N*nx,1);
```

```
Beq(1:nx) = Ad*x0;

[z_min,z_max] = genbegr2(N,M,x_min,x_max,u_min,u_max);
z_min((N-1)*nx+1:N*nx) = [0.00;0.00;0.00;0.00]; % Goal state
z_max((N-1)*nx+1:N*nx) = [0.00;0.00;0.00;0.00]; % Goal state

z_min(N*nx+(N-1)*nu+1:n) = 0; % Goal input
z_max(N*nx+(N-1)*nu+1:n) = 0; % Goal input

[z,lambda] = quadprog(Q,[],[],[],Aeq,Beq,z_min,z_max);
```

A.1.3 Zeropadding

```
Antall = 5/h;
Nuller = zeros(Antall,1);
Enere  = ones(Antall,1);

u   = [Nuller; u; Nuller];
x1  = [pi*Enere; x1; Nuller];
x2  = [Nuller; x2; Nuller];
x3  = [Nuller; x3; Nuller];
x4  = [Nuller; x4; Nuller];


x = [x1'; x2'; x3'; x4'];
```

Adds a five second pause at the beginning and ending of flight too let the helicopter reach a desired initial value and to hold it's end value, respectively.

A.2 Problem 3 LQR

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% LQ Feedback System
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Weighting matrices %%%
Qlq = diag([10,0.2,0.2,5]);
Rlq  = 1;

%Calculate gain
[K, S, e] = dlqr(Ad,Bd,Qlq,Rlq);

% Extract control inputs and states
u  = [z(N*nx+1:N*nx+M*nu);z(N*nx+M*nu)]; % Control input from solution

x1 = [x0(1);z(1:nx:N*nx)];              % State x1 from solution
x2 = [x0(2);z(2:nx:N*nx)];              % State x2 from solution
x3 = [x0(3);z(3:nx:N*nx)];              % State x3 from solution
x4 = [x0(4);z(4:nx:N*nx)];              % State x4 from solution
```

A.3 Problem 4

A.3.1 SQP

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% SQP - Solver
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Q1 = diag([1 0 0 0 0 0]);
R1  = diag([1 1]);

Aeq = gena2(Ad,Bd,N,nx,nu);
Beq = zeros(N*nx,1);
Beq(1:nx) = Ad*x0;

Q = genq2(Q1, R1, N, M, nu);

objective = @(z) z'*Q*z;


alpha = 0.2;
beta = 20;
lambda_t = 2*pi/3;

p_max = 30*pi/180;
x_max = [inf; inf; p_max; inf; inf; inf];
x_min = [-inf; -inf; -p_max; -inf; -inf; -inf];
u_max = [p_max; inf];
u_min = [-p_max; -inf];

e_max = alpha*exp(-beta*(-lambda_t).^2);
```

A.3.2 mycon alt pls rename

```
function [c, ceq] = mycon_alt(x)
% system parameters
alpha = 0.2;
beta = 20;
lambda_t = 2*pi/3;
N = 40;

lambda = x(1 : 6 : 6*N);
e = x(5 : 6 : 6*N);

c = zeros(N, 1);

for n = 1:N
    c(n) = alpha * exp(-beta .* (lambda(n) - lambda_t).^2) - e(n);
end

ceq = [];
%mycon = @(x) [c(x), []];

end
```

B Simulink Diagrams

B.1 Problem 2

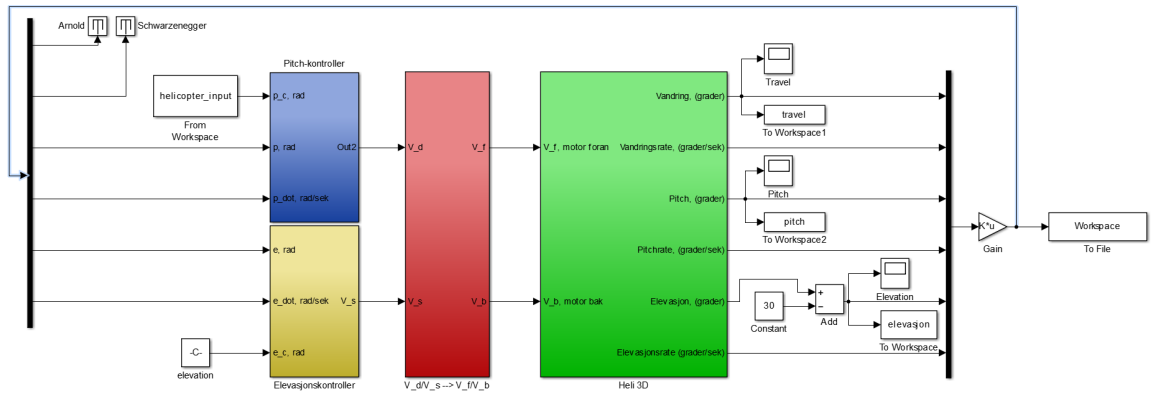

Figure 5: problem 2 Simulink model

B.2 Problem 3

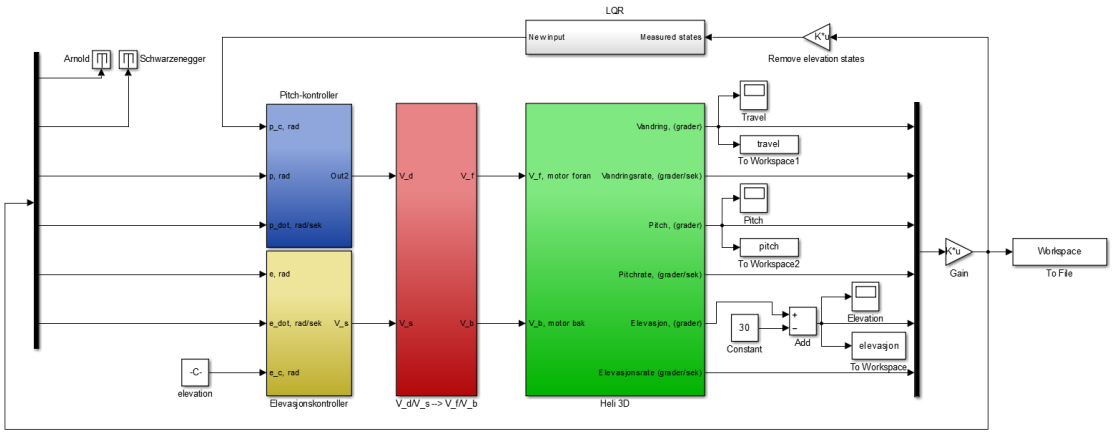

Figure 6: problem 3 Simulink model: Full system diagram with LQR

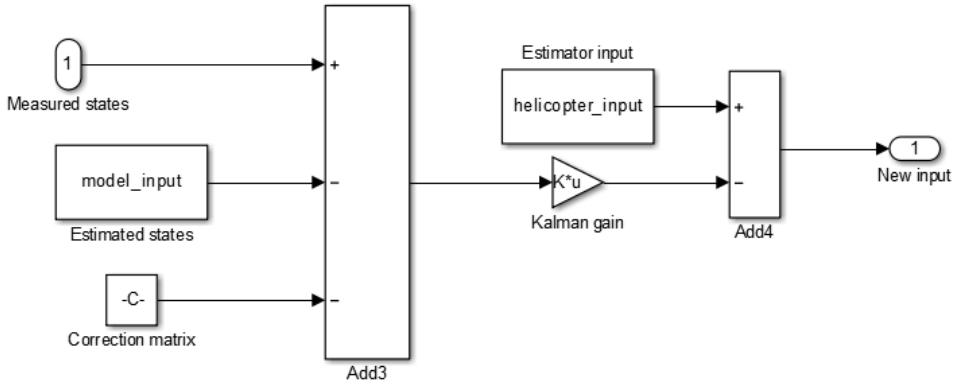

Figure 7: problem 3 Simulink model: LQR

B.3 Problem 4

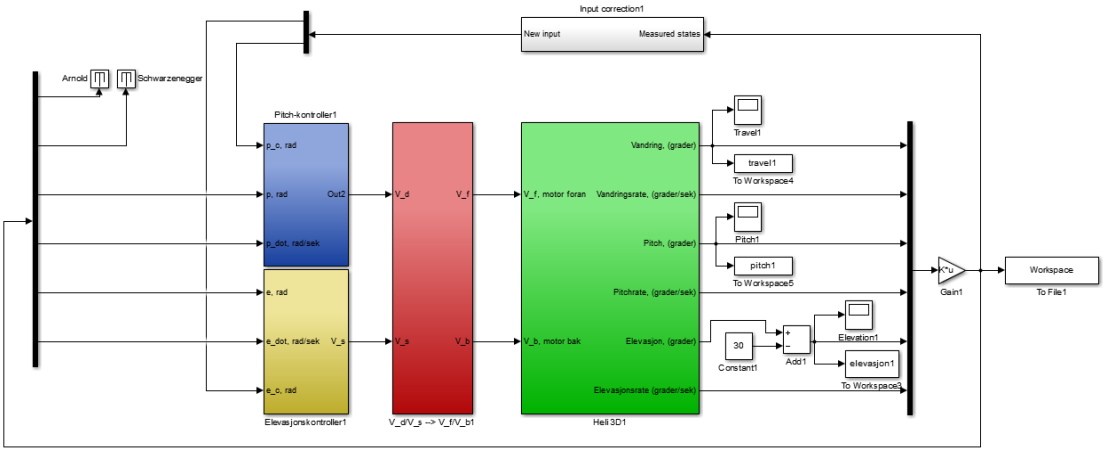

Figure 8: problem 4 Simulink model

Bibliography

Assignment, T. L. (2014). *Assignment text*. NTNU.