TTK4135 Optimization and Control
Helicopter Lab

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Trondheim
January 2020

Contents

1 Practical Information

In the helicopter lab in the course TTK4115 Linear System Theory, simple PID controllers were implemented to control the helicopter. Below is a short description of how the helicopter was controlled, since this project work to some extent is based on what was done in TTK4115. Yet, you should have no problems doing this project work even if you have not completed the helicopter lab in TTK4115.

Some general information:

- The assignment is compulsory, and should be performed in groups of two to three students.
- The problem formulation and MATLAB files can be downloaded from Blackboard.
- The helicopters are in Elektroblokk B, rooms B117, B121 and B125. The helicopters are numbered 1 to 10. Helicopter number 1 is used as a spare, please do not use this.
- If you can't access the building, you need to have an activated entrance card. Third year students at the Department of Engineering Cybernetics can activate their student card directly at the "Seksjon for bygningstjenester" (Information Desk) at Stripa. All other students following the course must first get a statement from the ITK department office (D144) to confirm that they actually follow the course before they can activate their card.
- If you can't log on to the lab computers, you should get a user id for the computer network at Department of Engineering Cybernetics. This can be obtained by consulting the department office at The Department of Engineering Cybernetics (D144).
- The project work is based on using MATLAB/Simulink and the Optimization Toolbox for optimal control.
- In the lab, MATLAB version R2015a is installed, including Simulink and QUARC.
- Hints on how to solve parts of the exercises might be posted on Blackboard and updated as it becomes clear what most students find difficult.
- The tentative deadline for handing in the project work is March 13, 2020, in D238 by 12:00 (noon). Each group should hand in only one report.

2 Health, Safety, and Environment ("HMS")

The first three items on this list should be obvious, but to be on the safe side:

- Keep fingers off the propellers while they are running.
- Remember to turn off the power-module when you leave the lab.
- Stand by to catch the helicopter when it is flying. The system is rather unstable and quite sensitive, so avoid crash landings. Also, stand by to either turn the power supply off or to stop the program.

Furthermore, you are required to

- Familiarize yourself with the general safety instructions for the department (`https://www.itk.ntnu.no/english/about/hse`), including escape routes, fire extinguishers, and first aid.
- Familiarize yourself with safety instructions for lab.
- Report all equipment errors.
- Think safety: Minimize both the probability and consequences of unwanted events,

$$\min risk = frequency \times consequence \tag{1}$$

3 Objective

The purpose of this exercise can be summarized by the following items:

- The students should get practice in formulating a dynamic optimization problem, as well as discretizing and solving the resulting problem using a computer.
- The exercise should illustrate the implementation and practical use of optimal control with and without feedback.
- A modern environment for real-time system development based on automatic generation of program code from Simulink block diagrams is used.

The students are expected to spend about 24 hours working on this project.

4 Lab report

A lab report describing what has been done in this project work should be handed in after the project work is completed. For each part of the exercise the report should include

- printouts of data from relevant experiments (plots),
- relevant model derivations, calculations and parameter values (remember to include units and scaling),
- discussion and analysis of the results,
- Simulink diagrams and MATLAB code.

All group members are graded based on the report. When the report is evaluated, documented calculations, experimental results, analysis and discussion of the results, skills in MATLAB/Simulink, and the quality of the written documentation of the results will be emphasized. The report counts for 20 % of the final grade in this course.

5 System Description

The helicopter consists of a base with an arm attached, as shown in Figure 1. The arm has the helicopter body at one end and a balance weight at the other end, see Figure 2. The arm can be moved up and down around an elevation axis and rotate around a vertical axis (travel). The helicopter body itself can also rotate around an axis normal to the arm (pitch). These three angles are measured with optical position sensors.

Two manipulated variables are available. These are the two DC motors with propellers attached. The force from the propellers is assumed to be proportional to the voltage applied. The counter balance is adjusted so that the weight to be lifted by the propellers is approximately 50 g. If you apply a voltage of approximately 1.5 V to each motor, the helicopter will take off from the ground.

The physical data (parameter values) can be found in Table 1 (page 11) and in the MATLAB file initF.m which are posted on Blackboard.

5.1 QuaRC, Simulink, and Realtime Workshop

QuaRC is Quanser's new, state-of-the-art rapid prototyping and production system for real-time control. QuaRC integrates seamlessly with Simulink to allow Simulink models to be run in real-time on a variety of targets, such as Windows and QNX.

The control program is made in MATLAB and Simulink, see the example in Figure 3. There are separate Simulink blocks for communication

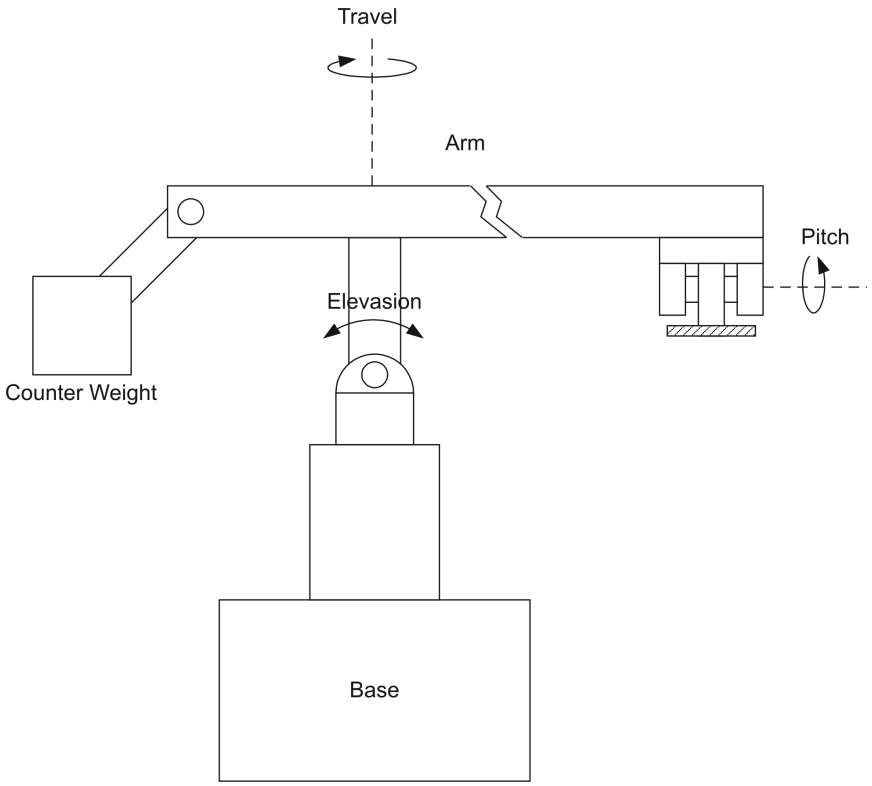

Figure 1: Cross section of the Helicopter system.

through the IO-card. After a block diagram for the controller is made in Simulink, QuaRC and Realtime Workshop are used to run the system. If these programs are installed, the QuaRC menu is available in Simulink. The option QuaRC $\rightarrow$ Build is the only one needed for this exercise, see Figure 4. The following happens when the build command is executed:

- Real-Time Workshop converts the Simulink diagram to C code.
- The code is compiled and linked using Visual C++.
- The executable code is downloaded to QuaRC.

After this, the program is ready to be started. Now the power module can be turned on, and the program is started by pushing "Start" in the QuaRC.

To carry out measurements on the system, the sampled data must be transferred to MATLAB. The "To Workspace"-block in Simulink does not work in real-time, so real-time plots must be used. This is the usual "Scope" plot in Simulink.

Before you start a measurement series, the following parameters must be set in the plot window:

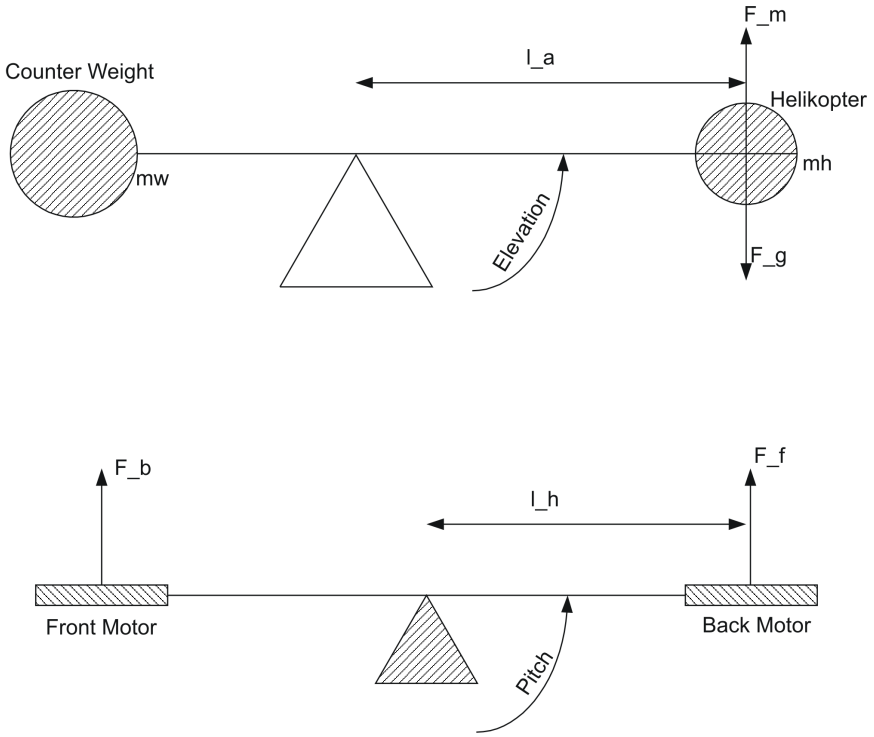

Figure 2: Sketch of the helicopter.

- Buffer size. The length of the measurement series that the plot will "remember" in seconds
- Sampling frequency.
- The time window that will be displayed in the plot. You should set this parameter equal to the buffer size. If the option for Maximize Plotting Frequency is selected, the highest possible sampling frequency will be implemented.

After a measurement series has been recorded, you can export the data to the MATLAB workspace or to a file. Go to File → Save in the plot of interest. The best thing to do is probably to save the data as a mat-file, and keep them for later use.

Notice: If it turns out that the data you have transferred has a much higher sampling frequency than the one you chose, try to record a new measurement series in the same plot-window and transfer the data again.

Note: If the above method of recording data does not work, see Section 11.

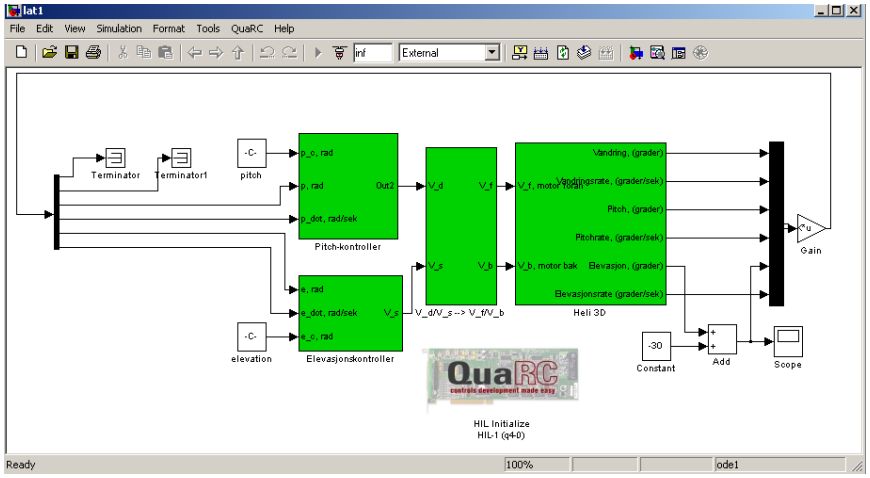

Figure 3: An example of SIMULINK diagram with QuaRC.

5.2 Hardware

5.2.1 Quanser Q4

The Q4 is an innovative HIL ("hardware in the loop") control board with an extensive range of input and output support, see Figure 5. A Quanser Q4 IO card is connected to the PC bus. The Quanser Q4 card measures the physical signals coming from the helicopter and converts them into digital signals that can be read by the computer. The card also converts the signals from the computer that is used as input to the helicopter.

The following blocks are available:

- Analog input.
- Analog output.
- Digital input.
- Digital output.
- Encoder input. (Gives out an integer that indicates the condition of a position sensor.)

5.2.2 Power Module

The power that operating the motors comes from one or two Universal Power Modules.

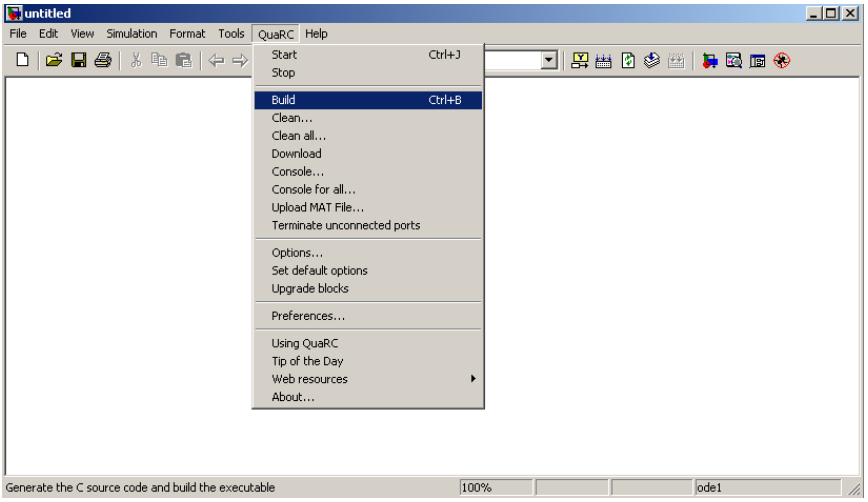

Figure 4: Build QuaRC.

6 Some Advice

- Errors may arise during compilation. It is a good idea to wait to start the helicopter (that is, to push start in the QuaRC), until the compilation is finished. It is obviously also important that the compilation is done by QuaRC → Build.
- If inexplicable error messages arise during compilation, there are a couple of things that may help. First, erase all compiled code and start over again. If this does not help, try restarting the computer.
- If something is not working properly or is broken, notify the student or teaching assistant.

7 Starting Point for the Programming

A ready-made Simulink block diagram should be used as the basis for your program. You will need the following files:

- helikopter: Simulink-diagram of helicopter, half-made.
- init0i.m: contains the physical data for the helicopter number i.

These files can be downloaded from Blackboard. Change the names of the two files to helikopter.mdl and init.m, respectively. No backup is taken of your own files, so remember to do this yourself.

Some files that may be helpful for solving the project work are genA2.m, genQ2.m, and genBegr2.m. Remember to change genQ2.m and genBegr2.m before you use them.

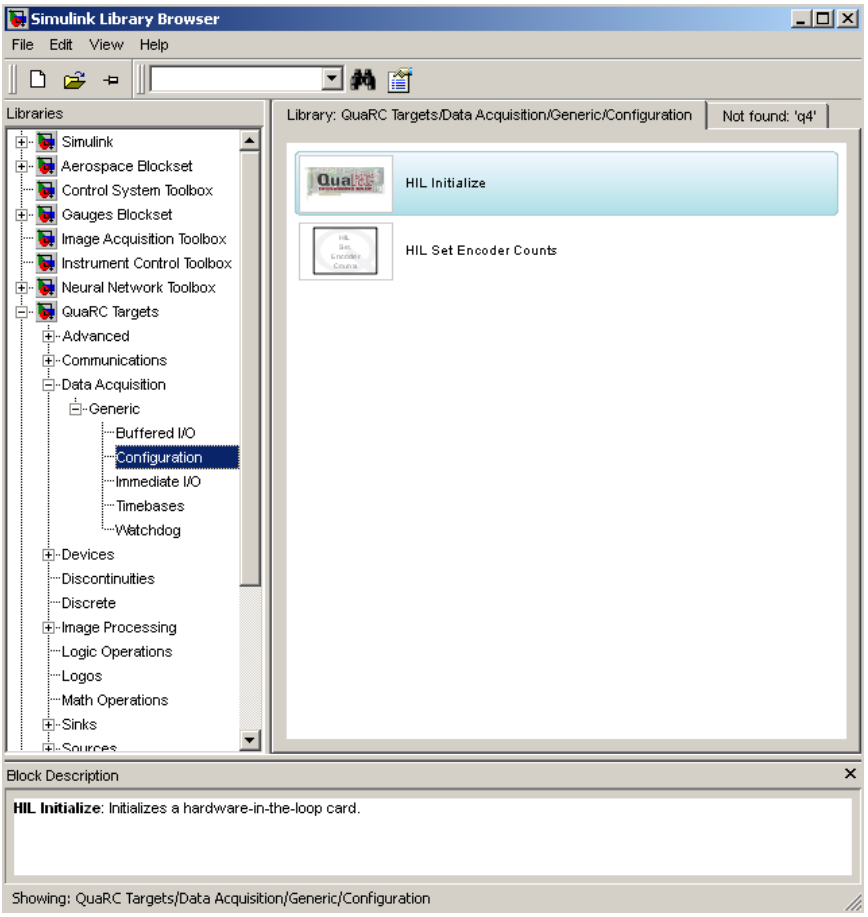

Figure 5: HIL in Simulink Library Browser.

8 Model Derivation

To derive a model for the helicopter, one can formulate the moment balances; this is done below. For a list of parameters and variables, see Table 1 and 2.

We start with the derivation of the model for elevation:

$$J_e\ddot{e} = l_a K_f V_s - T_g \tag{2a}$$

so that

$$\ddot{e} = K_3 V_s - \frac{T_g}{J_e}, \quad K_3 = \frac{l_a K_f}{J_e} \tag{2b}$$

The following PD controller is implemented to control the height:

$$V_s = K_{ep}(e_c - e) - K_{ed}\dot{e}, \quad K_{ep},\, K_{ed} > 0 \tag{3}$$

Table 1: Parameters and values.

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.

Symbol	Variable
p	Pitch
p_c	Setpoint for pitch
λ	Travel
r	Speed of travel
r_c	Setpoint for speed of travel
e	Elevation
e_c	Setpoint for elevation
V_f	Voltage, motor in front
V_b	Voltage, motor in back
V_d	Voltage difference, $V_f - V_b$
V_s	Voltage sum, $V_f + V_b$
$K_{pp}, K_{pd}, K_{ep}, K_{ei}, K_{ed}$	Controller gains
T_g	Moment needed to keep the helicopter flying

This leads to the system

$$\ddot{e} = -K_3 K_{ed}\dot{e} - K_3 K_{ep} e - \frac{T_g}{J_e} + K_3 K_{ep} e_c \tag{4}$$

In addition, an integral term is added, giving a PID controller. We assume that the integral term counteracts the effect from the constant term $-\frac{T_g}{J_e}$, so that these two terms can be ignored. The resulting system for elevation is then

$$\ddot{e} + K_3 K_{ed}\dot{e} + K_3 K_{ep} e = K_3 K_{ep} e_c \tag{5}$$

The same can be done for the pitch angle:

$$J_p \ddot{p} = K_f l_h V_d \tag{6a}$$

so that

$$\ddot{p} = K_1 V_d, \quad K_1 = \frac{K_f l_h}{J_p} \tag{6b}$$

The PD controller

$$V_d = K_{pp}(p_c - p) - K_{pd}\dot{p}, \quad K_{pp},\, K_{pd} > 0 \tag{7}$$

is used to control the pitch angle. This leads to the following closed-loop system

$$\ddot{p} = K_1(K_{pp}(p_c - p) - K_{pd}\dot{p}) \tag{8a}$$

or

$$\ddot{p} + K_1 K_{pd}\dot{p} + K_1 K_{pp} p = K_1 K_{pp} p_c \tag{8b}$$

Finally, we need a model for the travel ("vandring"). We start from the moment balance. For small pitch angles, the force needed to keep the helicopter flying is approximately K_p. The horizontal component of the force gives the acceleration around the travel axis:

$$J_t \dot{r} = -K_p l_a \sin p \tag{9}$$

We assume that the angle p is small, so that $\sin p \approx p$. Then,

$$\dot{r} = -K_2 p, \quad K_2 = \frac{K_p l_a}{J_t} \tag{10}$$

Note that a positive pitch angle gives a negative travel acceleration. Also remember that $\dot{\lambda} = r$.

The model we will use can then be summarize by

$$\ddot{e} + K_3 K_{ed} \dot{e} + K_3 K_{ep} e = K_3 K_{ep} e_c \tag{11a}$$

$$\ddot{p} + K_1 K_{pd} \dot{p} + K_1 K_{pp} p = K_1 K_{pp} p_c \tag{11b}$$

$$\dot{\lambda} = r \tag{11c}$$

$$\dot{r} = -K_2 p \tag{11d}$$

All angles in the equations above are given in radians. Remember that the measured angles from the helicopter model are recorded in degrees. The controllers for pitch angle and elevation are tuned, and further tuning of these controllers should not be necessary.

Based on the model, you may get the impression that there are no interactions between elevation and pitch angle/travel. This is a result of the assumptions made in the model derivation. Interactions between these variables are of course present.

9 Before you Start

The arrangement of the helicopters is as shown in Figure 6.

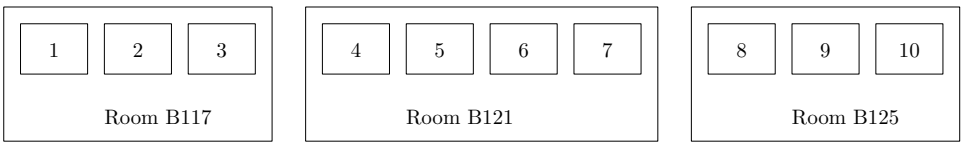

Figure 6: Arrangement of helicopters.

Since the helicopters are different, using the same helicopter for the entire lab project is recommended.

10 Exercises

The assignment is divided in four parts:

1. Repetition/introduction to use Simulink/QuaRC to control the helicopter.

2. Optimal control of pitch/travel with no feedback. An optimal input sequence that moves the helicopter 180 degrees should be calculated and implemented on the helicopter.

3. Optimal control of pitch/travel with feedback (LQ control). Feedback is introduced in the controller from part 2.

4. Optimal control of pitch/travel and elevation with and without feedback; that is, we have an optimal trajectory in two dimensions.

In all the exercises above you can assume that the pitch and the elevation controllers are "turned on" and that they are well tuned. We consider these controllers as inner control loops, and we will in the rest of the exercise use the setpoints to these two controllers (p_c and e_c) as manipulated variables ($u(t)$).

10.1 Repetition/introduction to Simulink/QuaRC

This first exercise is meant to be a repetition for those who have completed the helicopter lab in TTK4115 Linear System Theory and a quick introduction to Simulink/QuaRC to those who have not.

Set up a work folder locally. Copy the files helicopter.slx and initXX.m from Blackboard. Choose the init file that corresponds to your helicopter.

Start MATLAB and go to your work folder. Run init.m and open the helicopter model in Simulink. Go through the different blocks in the Simulink diagram and figure out what they do. To run the helicopter, go to QuaRC $\rightarrow$ Build. This command converts the Simulink block diagram to C-code, compiles, links and downloads the program to QuaRC. When this is done, the QuaRC will start. Start the real-time program, and verify that the controllers work. Try changing the setpoint for pitch and elevation.

Try opening some of the real-time plots (remember that this must be done from the QuaRC). See for example the plots for pitch angle and elevation. Notice that all the states are not measured. The three angles are measured, but the angular velocities are estimated.

10.2 Optimal Control of Pitch/Travel without Feedback

In this part of the exercise we will disregard elevation, that is, we assume $e = 0$. We will then calculate an optimal trajectory x^* and a corresponding optimal input sequence u^*. This input sequence will be implemented as setpoints for the inner controllers, but we will not feed back the measured state to correct for deviations from the optimal trajectory. This control hierarchy is illustrated in Figure 7.

1. Write the model on continuous time state space form

$$\dot{x} = A_c x + B_c u \tag{12}$$

   with $x = \begin{bmatrix} \lambda & r & p & \dot{p} \end{bmatrix}^\top$ and $u = p_c$. What are we modeling here? Is it just the helicopter? Discuss what the model includes, and how it relates to Figure 7.

2. Discretize the model using the forward Euler method and write the resulting model on discrete time state space form

$$x_{k+1} = Ax_k + Bu_k \tag{13}$$

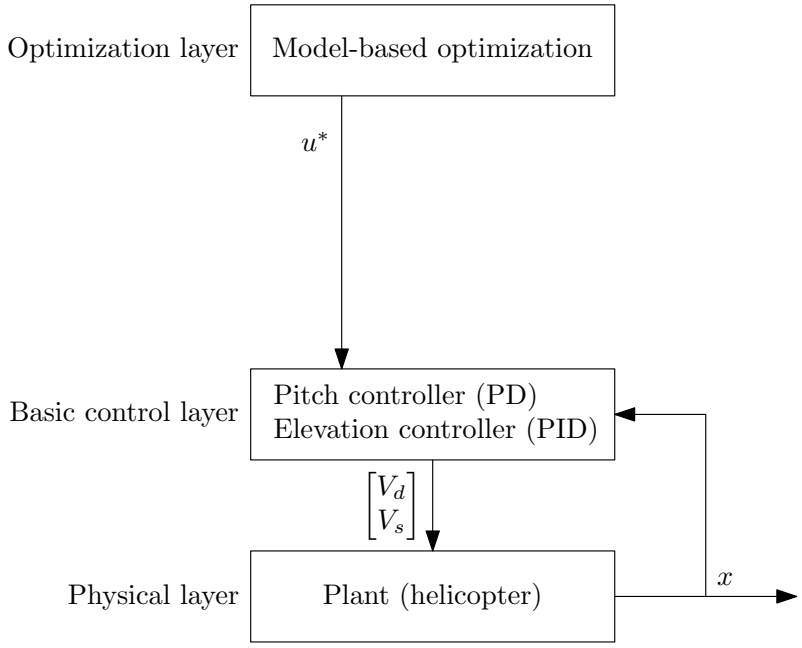

Figure 7: Illustration of the layers in the control hierarchy used in Section 10.2.

3. Calculate an optimal trajectory for moving the helicopter from $x_0 = \begin{bmatrix} \lambda_0 & 0 & 0 & 0 \end{bmatrix}^\top$ to $x_f = \begin{bmatrix} \lambda_f & 0 & 0 & 0 \end{bmatrix}^\top$ when the elevation angle is assumed to be constant. Use $\lambda_0 = \pi$ and $\lambda_f = 0$. Also implement the constraint

$$|p_k| \leq \frac{30\pi}{180}, \quad k \in \{1, \ldots, N\} \tag{14}$$

Since the manipulated variable p_c in this case is the setpoint for the p controller, the constraint should also be implemented for the manipulated variable. We want to minimize the cost function

$$\phi = \sum_{i=1}^{N} (\lambda_i - \lambda_f)^2 + q p_{ci}^2, \quad q \geq 0 \tag{15}$$

Solve the optimization problem using the MATLAB function quadprog. Try using the values 0.1, 1, and 10 as weights q. Plot the manipulated variable and the output. Comment the results with respect to the different weights chosen. Remember that some useful files are posted on Blackboard. Use a sampling time of 0.25 s and $N = 100$. Furthermore, discuss the objective function (15), in particular the term $(\lambda_i - \lambda_f)^2$. For instance, could any unwanted effects arise from steering the helicopter to $\lambda = \lambda_f$ with this objective function?

Hints: To use the quadprog function in MATLAB, you need to formulate the optimization problem as a QP problem in standard form. Start by defining a large vector z containing all the variables that should be optimized:

$$z = \left(x_1^\top, \ldots, x_N^\top, u_0^\top, \ldots, u_{N-1}^\top\right),$$

and use this definition when constructing the objective function and constraints. For more hints read Chapter 3.5 in "Merging Optimization and Control" by Foss and Heirung, which should be on Blackboard. A template and some helper functions to this assignments has also been posted on blackboard, using this template is optional but try to use a similar structure if you implement everything from scratch. Having a similar structure will make it easier for the student assistants to help you and evaluate your results.

4. The specific implementation of the QP algorithm used here can only weight deviations from origin ($\lambda_f = 0$). The position sensors on the helicopter are relative, which means that the position is reset to zero each time the helicopter is started. To get the helicopter to start in x_0 you have to add a suitable value to the measurement. For example, you can subtract 30° from the elevation measurement to get the helicopter to fly at a reasonable height when $e_c = 0$ (this is done in the given file). Implement the input sequence generated in c) with $q = 1$.

   The optimization in c) gives the manipulated variable u. It may be a good idea to add some zeros at the beginning of the input vector, so that the helicopter has time to stabilize in x_0 before the calculated sequence is implemented. Also, adding zeros at the end of the vector is recommended to keep the helicopter stable after the input sequence is over. Add enough zeros to keep the helicopter stable for at least 5 s before and after the optimal input sequence is implemented. The easiest way to transfer the input sequence to Simulink is to use a "From Workspace" block (which can be found under "Sources" in the Library Browser). This block imports two vectors, a vector u which is connected to the output and a time vector t which determines when the different elements of u should be transfered to the output.

   Does the helicopter end in the desired point x_f? What causes the observed deviation?

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

1. We now introduce feedback in the optimal controller. This is done by using an LQ controller. LQ stands for linear quadratic, which means that this controller minimizes a quadratic criteria for a linear

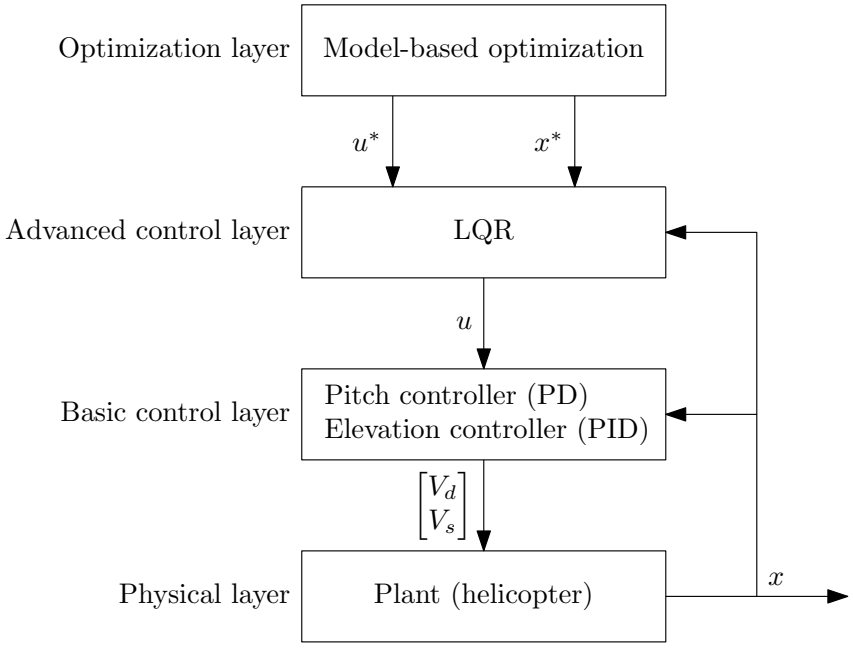

Figure 8: Illustration of the layers in the control hierarchy used in Sections 10.3 and 10.4.

model. From the optimal trajectory calculated in exercise 2, we have an optimal input sequence u^* and an optimal trajectory x^*. Using the following manipulated variable will introduce feedback:

$$u_k = u_k^* - K^\top (x_k - x_k^*) \tag{16}$$

As long as the system is following the calculated optimal trajectory x^* the calculated input sequence u^* is implemented. If a deviation from x^* is observed, the manipulated variable will be modified by the feedback term. This new control hierarchy is illustrated in Figure 8.

A good choice for the gain matrix K must be found. The gain could be calculated in several ways (for example using pole placement), but here we will calculate it as an LQ controller. An LQ-controller (in discrete time) minimizes the quadratic objective function

$$J = \sum_{i=0}^{\infty} \Delta x_{i+1}^\top Q \Delta x_{i+1} + \Delta u_i^\top R \Delta u_i, \quad Q \geq 0,\ R > 0 \tag{17}$$

for a linear model

$$\Delta x_{i+1} = A\Delta x_i + B\Delta u_i \tag{18}$$

without including inequality constraints. Here, Δx and Δu are devia-

tions from the optimal trajectory,

$$\Delta x = x - x^* \tag{19a}$$

$$\Delta u = u - u^* \tag{19b}$$

In MATLAB you can use the function dlqr to calculate the optimal K matrix. Q and R indicates how much you want to penalize deviation in the states, and how much you want to penalize use of the manipulated variable. Use diagonal matrices for Q and R. A diagonal matrix is easily made in MATLAB using the function diag.

2. Implement the feedback on the helicopter and run the helicopter. This is done in Simulink using the following blocks Mux, Demux, Matrix Multiplication, Sum, and "From Workspace". Notice that the "From Workspace" block can import several values (a vector), meaning only one block is required for x.

3. An alternative strategy would be to use an MPC controller. How would you realize this controller? Discuss advantages and disadvantages with an MPC controller compared to the controller you have implemented. Also, think about how the structure in Figure 8 would look if you used MPC.

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

Now, the task is to calculate an optimal trajectory in two dimensions. The helicopter is moved from x_0 to x_f past a restriction causing the elevation angle to change during the flight. We must therefore add the elevation to the state vector; the setpoint e_c is a new manipulated variable in the system.

1. Write the system on continuous state space form with the two extra states e and $\dot{e}$. Use $x = \begin{bmatrix} \lambda & r & p & \dot{p} & e & \dot{e} \end{bmatrix}^\top$ and $u = \begin{bmatrix} p_c & e_c \end{bmatrix}^\top$.

2. Discretize the model using the forward Euler method and write the resulting model on discrete state space form.

3. Inequality constraints on the elevation should now be implemented. The constraint is

$$e_k \geq \alpha \exp\left(-\beta(\lambda_k - \lambda_t)^2\right) \ \forall k \in \{1, \ldots, N\} \tag{20}$$

This is a nonlinear constraint. A QP solver can only solve problems with linear constraints, so we must use another function to solve the optimization problem. Instead we will use fmincon as this an SQP-type

algorithm. See the documentation for a description of the function and how to use it.

The criteria to be minimized is now

$$\phi = \sum_{i=1}^{N} (\lambda_i - \lambda_f)^2 + q_1 p_{ci}^2 + q_2 e_{ci}^2 \tag{21}$$

Start with $q_1 = q_2 = 1$ and see if there are other values that give a better result. Remember to include the equality constraints given from the model equations in the optimization problem. Use $\alpha = 0.2$, $\beta = 20$ and $\lambda_t = \frac{2\pi}{3}$. For every point in time a constraint on the form

$$c(x_k) = \alpha \exp\left(-\beta(\lambda_k - \lambda_t)^2\right) - e_k \leq 0 \tag{22}$$

must be implemented and passed to fmincon.

Start with a relatively short time horizon (12–15 time steps). Use $\Delta t = 0.25$ s. Try optimizing over a horizon of approximately 10 s, that is, $N = 40$. We use a shorter horizon in this exercise than in the previous exercises because performing the optimization over the same horizon will take some time.

4. Implement the optimal input sequence on the helicopter. Introduce feedback in the same way as in exercise 3. Compare the performance with and without feedback.

5. Notice that the first 4 states in the model are completely decoupled from the last 2. How does this fit with reality? What effect does this have on the calculated optimal trajectory? Suggest (but do not implement) a solution to improve this.

6. Optional exercise: Try adding more constraints on the states. These may be constraints on maximal allowed speed $\dot{e}$ and $\dot{\lambda}$.

11 Additional Information

11.1 Recording Data

If the method for recording data described in Section 5.1 does not work, try the following method instead.

1. Use the "To file" block, see Figure 9.
2. Connect the block to a signal you want to measure. Input some suitable parameters, and note that the format has to be Array — see Figure 10.
3. After running the system, you can access and plot the data using something like the following commands:

   ```
   >> load('e.mat')

   >> plot(e_measured(1,:0), e_measured(2,:0))
   ```

   where the first row of `e_measured` is time and the second row is the data. En example result is shown in Figure 11.

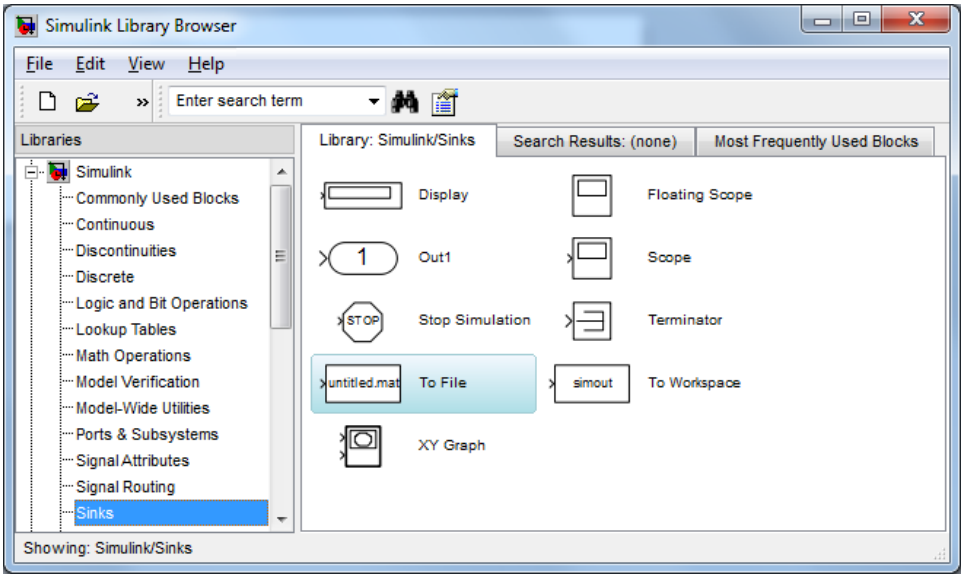

Figure 9: Location of the "To file" block in the Simulink library browser.

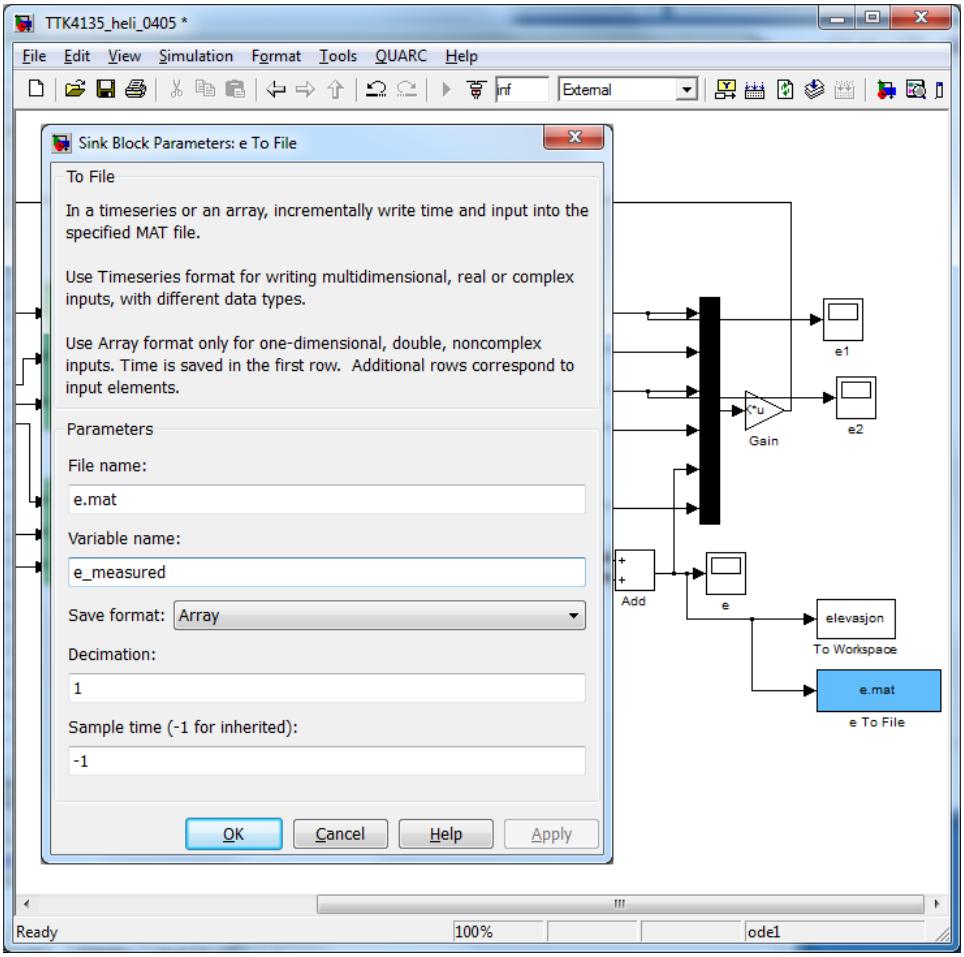

Figure 10: The block (in light blue) connected to the elevation measurement, and the parameters dialogue. Note that format has to be "Array".

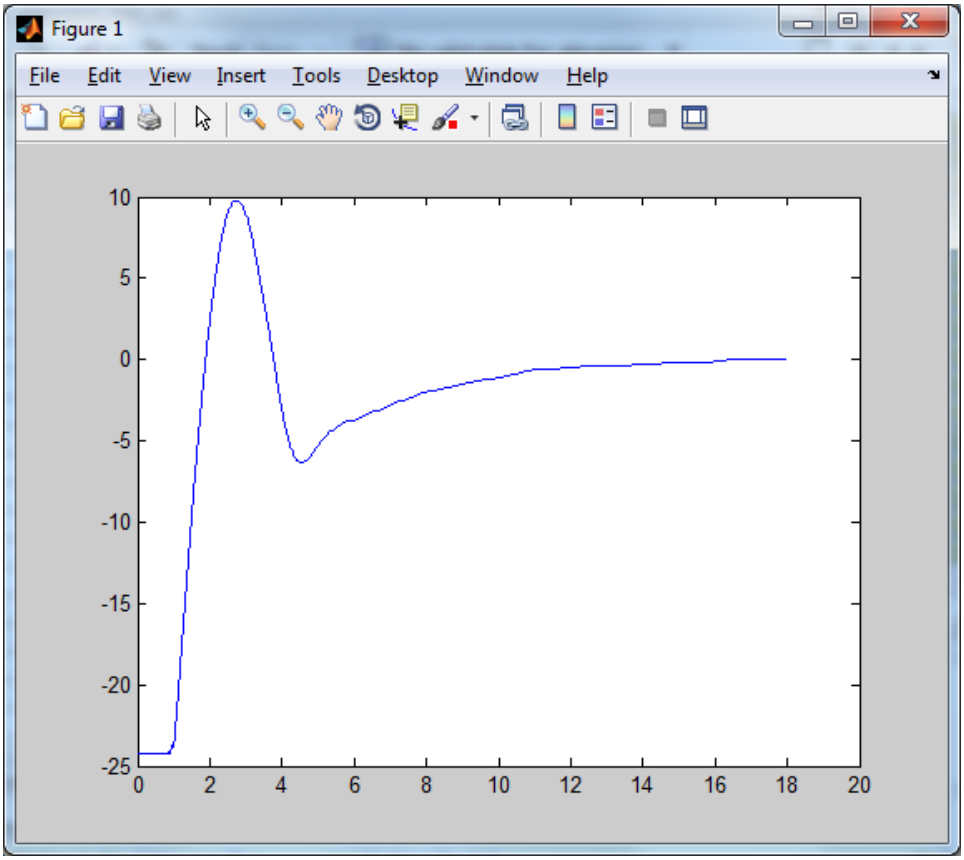

Figure 11: Resulting plot.