TTK4115 – Linear System Theory

Helicopter lab assignment

Department of Engineering Cybernetics
NTNU

Version 5.1 August 2020, SVR

Previous versions:
5.0, August 2019, SVR
4.5, August 2015, KG
4.4, August 2013, MH
4.3, August 2012, MH
4.2, August 2011, DB
4.1, August 2010, DB
4.0, August 2009, JR
3.8, August 2008, AAE
3.7, August 2007, JM
3.6, August 2006, JB
3.5, August 2005, JH
3.4, August 2005, JS
3.3, August 2003, JS
3.2, September 2002, ESI
3.1, September 2001, ESI
3.0, June 2001, TAJ

1 Introduction

1.1 Evaluation and grading

The completion of this lab assignment is required to pass the course. A report and an oral evaluation will account for 50% of the final grade. Experimentation, also outside of what is asked for in the task, is highly encouraged and will give you a bonus. If you have an idea of how to improve your system, or if you see something interesting to investigate further, we encourage you to test it out! Remember to write about it in the report, and you can bring it up during the evaluation when we are discussing the relevant task :)

1.1.1 Oral evaluation

An oral evaluation will take place after the lab is completed. The evaluation will be an informal conversation where you can show us what you have managed and we will ask questions. There will be questions regarding your implementation, our observations, and questions from theory relevant to the task. Your grade will be based on the performance of the helicopter together with your theoretical understanding.

At the oral evaluation you will show your result from each lab day. At the end of each part of the assignment, you will be asked to tune your system and prepare a file for the evaluation. The performance of the helicopter will be evaluated, so make sure you tune it well. Use the knowledge gained from the experimentation and from theory to guide your tuning process. Good performance can often be achieved by identifying which problems arise when tuning too far in different directions, and then finding a nice trade-off between the problems.

Make sure the file can be quickly started and that a copy of all necessary Matlab files are included, speaking from experience, you will probably change your main Matlab script later which will corrupt all previous lab-days. At the presentation we will probably want to look at some scopes, so make sure you have scopes connected to all interesting signals.

1.1.2 Report

A lab-report should be handed in at the evaluation. It should be printed in color on A4 format. The report should be formatted in the same way as the assignment. Everything relating to part 1 should be written together so that the examinator does not have to jump back and forth. Keep the report brief and to the point, and avoid length appendixes. You do not need to introduce the lab setup. Include the names together with the student number of group participants (not candidate number!). The name is used to verify that I managed to write in the correct student number :)

The report should preferably be written in LaTeX. See the ITK Labreport guide/template on Github for some useful hints. Whenever the template disagrees with the specification in this assignment, disregard the template.[1].

The report should include all experimentation that is done, and answers to the questions that arise throughout the assignment. The derivations and equations from the preparations should

[1] These files are also posted on Blackboard

only be included if they are relevant to the discussion or the results. Do not include your entire Matlab code or Simulink diagram. Instead, only include sections of interest that show how you have solved a problem.

Throughout the lab you will be asked to experiment. The quality of your experimentation, discussion, and your theoretical understanding make up your grade. The experiments should be done systematically and thorough, but not unnecessarily long. You should identify, based on theory, which choice in tuning that will give rise to distinct behavior and compare the observed response with what you expected. You should discuss your findings, and discuss the cause for any observed discrepancies.

Ensure that all plots in the presentation are readable and understandable. The plots need units, axis, and legends. If you want to compare two graphs, make sure they are plotted on top of each other. Avoid having black backgrounds on the plots to save ink and increase readability on paper. You won't get any points for a good conclusion if we cannot understand the discussion or the plots the discussion is based on.

Remember to save plots while experimenting, and note down your experimentation in a systematic manner. Having a lot of data but no memory of what you tested is not worth the ink it will be printed on! Save a copy of your files so that you can easily go back and re-do any experiments if needed. It is highly recommended to complete the report before your next lab-day. When writing the report you will realize that you should have checked something else, this can then easily be done at the start of the next lab-day. Due to the tight schedule, there won't be any extra time at the end of the lab to complete the report, so make sure it is almost complete before the last lab-day!

1.1.3 Practical information

- Ten helicopters are found in "Elektrobygget", rooms B113 (Helicopters 1 and 2) B117 (Helicopters 3 and 4), B121 (Helicopters 5 – 7) and B125 (Helicopters 8–10). You need your key card to access the room[2]. You are assigned to a specific helicopter, and should use that throughout the lab to avoid conflicts with other groups and to minimize time spent on adapting your code to a new helicopter, as they vary slightly.
- Each group will have 4 whole days reserved in the lab. One day per part of the assignment. A student assistant will be present for 3 hours each day.
- The lab computers are not connected to the internet, which means that there are no e-mail or other internet-based ways to store your data. The easiest way to store your data is to bring a memory stick to the lab. Note that you need the put the Matlab files that are posted on Blackboard on your memory stick before you come to the lab. It is also possible to use the "sambastud" server; see Section 1.4.4 for more details.

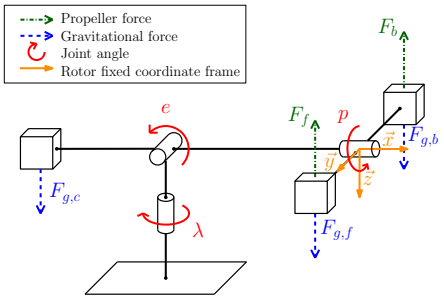
1.2 The helicopter

The helicopter consists of a base on which a long arm is attached; see Fig. 1.1a. The helicopter head, with two motors and propellers, is attached to one side of the arm. On the other side, a counterweight is placed. The helicopter has three rotational joints. The helicopter arm can rotate about the vertical axis. This will be referred to as the travel of the helicopter. The helicopter head can move up and down with respect to the base. This rotation of the helicopter arm will be

[2]If you are denied access, talk to Lill Hege Pedersen who works in the Department's reception (room D144 in "Elektrobygget"), or contact the teaching assistant.

referred to as the elevation of the helicopter. Moreover, the helicopter head can tilt with respect to the arm, which will be referred to as the pitch of the helicopter. All three joint angles are measured using encoders. The helicopter is actuated by the two motors on the helicopter head, which are connected to the propellers.

(a) Helicopter



(b) Model

1.3 Safety

- Before you start, read the safety instructions attached to your work station.
- Keep your fingers (and other limbs) away from the propellers while they are running, and while the power supply is turned on! The DAQ card will output transient signals now and again, which cannot be controlled, e.g. when the computer is switched on or off.
- With respect to fire safety: remember to turn off the power supply when you leave the lab.
- The system is often in an unstable state and behaves badly. It is also expensive and a bit fragile, so try to avoid crash landings! One person in the group should always be prepared to switch off the system when it is running, while another one should be standing prepared to catch the counterweight of the helicopter to avoid crash landing. **Do not grab the end with the spinning blades! The protective grids are very thin to be light enough for flight, it might break when you grab it which can result in serious injuries!**
- It should not be necessary to alter the helicopter setup in any way. If you experience hardware problems, notify the student assistants, teaching assistant, or technical staff. **Do not try to fix it yourself.**

1.4 User guide

1.4.1 Starting up

Before you get started, make sure that the power supply of the helicopter is switched off: the power switch on top of your desk should be on “Stopp”. To turn on the computer, press the power button on the computer below the helicopter platform. Once Windows has started up, click on the “Other User” button. Log in with your student user name and password. Once you are logged in, start Matlab.

A template Simulink model, `heli_q8.slx`, has been made which you can use as a starting point for the assignment. In addition to this file, an initialization file containing the physical parameter values for the helicopter, `init_heli.m`, has been prepared. Note that there are different initialization files for helicopter 1 and 2 than for the rest. The initialization must be run before your Simulink program can be built as it contains calibration data for the encoders and other configuration parameters. These files can be downloaded from "Blackboard"[3].

You can add new blocks into the Simulink diagram by opening the "Library browser" (the symbol left of the gear symbol), or by single-clicking in the diagram and writing the name of the block you want.

For a faster building procedure, copy the Matlab/Simulink files to your folder on the hard drive of the computer:

`C:\Users\username`

where `username` is your username. Do not forget to copy the files to your memory stick after you are done! Note that no backups are made of the files on the lab's computers, and the computers are wiped without notice. Thus, it is your own responsibility to back up your files.

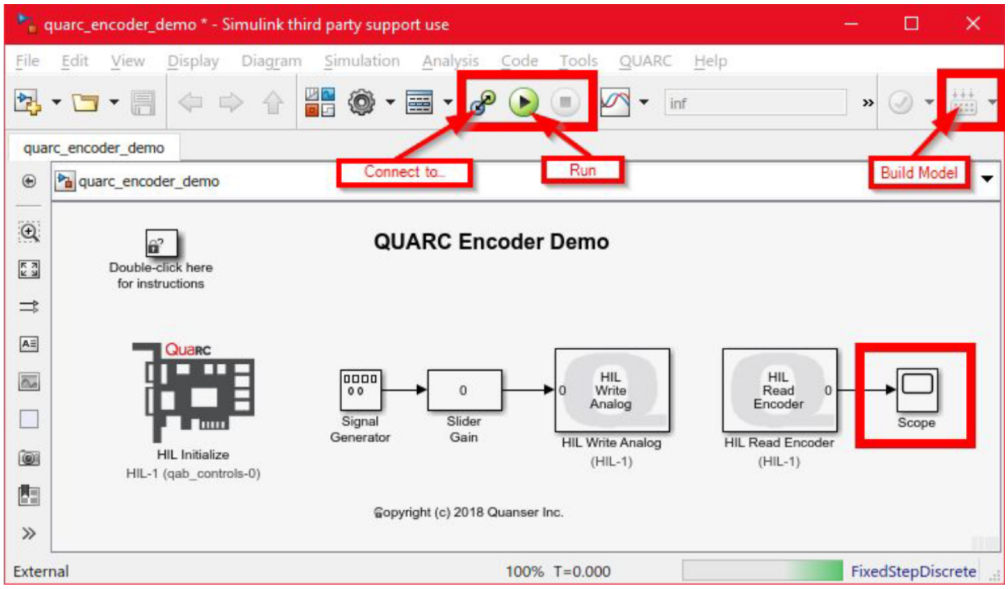


Fig. 1.2: QuaRC buttons in Simulink.

1.4.2 Running

When a block diagram implementing your controller has been made in Simulink, QuaRC is used to automatically generate source code for this model. QuaRC will also automatically compile the source code. The model can be built by pressing the "Build Model" button shown in figure 1.2. When the source code has been built, Simulink is used to run the implementation in real-time. Before running the program you have to power on the helicopter using the power switch on top of the table. It is important that the helicopter is powered on before executing the program. Some drivers might crash if the helicopter does not have power requiring a re-boot of Windows.

[3]The files have been tested on all the helicopters. However, due to some differences between the helicopters, some adjustments of the parameters might improve performance.

Run the program by first pressing "Connect to..." then "Run" as shown in figure 1.2. Press the stop button to stop the execution.

When you make changes to the Simulink diagram, by e.g. adding new blocks, you will have to `Build` again before you can "Connect" and "Play". However changing the values but keeping the dimensions of constants can be done while the helicopter is *running*, and does not require a new `Build` nor a restart of the program.

1.4.3 Recording data

To save measured data from the system, you may use a "To File" or "To Workspace" block. The standard time-series format does not work. Instead use one of the others, such as "Array" to export your data; see Fig. 1.3. You can load the stored data into the Matlab Workspace by using the command `load xxxx`, where `xxxx.mat` is the name of the stored MAT file. Note that the first row of the array in which the data is stored contains the time sequence; the other rows contain the stored signals. You can store multiple signals in one variable by "muxing" the signals with the "Mux" block in the Simulink Library Browser. Due to a limit in the QuaRC driver, the following steps has to be made to store data sequences longer than 10 seconds: `Code` (in the Simulink menu) → `External mode control panel` → `Signals and Triggering`. Under `Trigger options` you should increase the duration to an appropriate value.

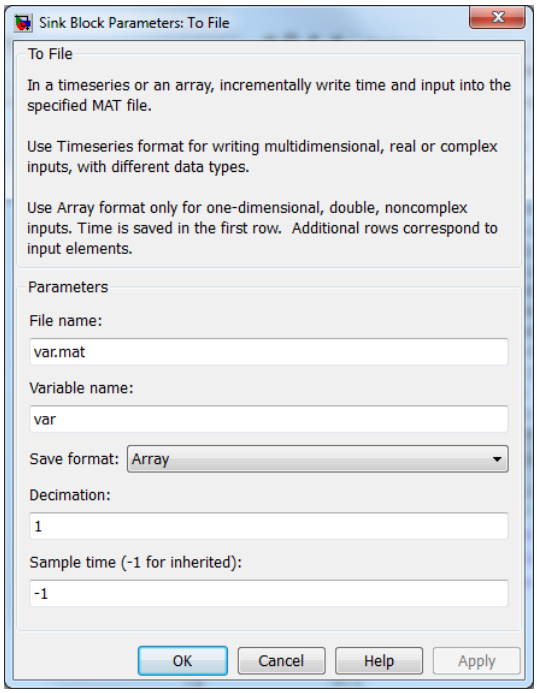


Fig. 1.3: Sending data to a file

1.4.4 Data storage

Because there is **no internet in the lab**, it is not possible to store and transfer your data via e-mail or other internet-based methods. The easiest way to store your data is on a memory stick,

so **dont forget to bring a memory stick to the lab**.

Alternatively, you can store your data on the "sambastud" server, which should be accessible from the lab's computers. Windows should automatically connect to this server when you log on, but in case it does not, you can try the following: right-click on "My Computer", select "Map Network Drive", enter

`\\sambastud.stud.ntnu.no\username`

as the path, and your username in the "Connect As" field. Be sure the "Reconnect at Logon" checkbox is unchecked. Click "OK" and fill in your password. To connect to the "sambastud" server outside the lab, you need to make sure that you are connected to the NTNU network (possibly via a VPN connection).

1.5 Debugging guide

1.5.1 QUARC and Simulink

It can sometimes help to clean and re-build the program. Cleaning can be done by pressing `Clean all...` in the `QUARC` menu in Simulink.

The error "algebraic loops" tells you that you have a loop in your program making a value directly dependent on itself at the current time-step. You are only allowed to loop if there is a time-delay, memory block, integrator, transfer function, or other dynamic block in the loop.

It can be useful to verify the signal dimensions. This can be done by pressing `Display` → `Signals & Ports` → `Signal Dimensions`. Note that it may need to be toggled off and on again to work. The most common mistake causing the wrong dimension is to have "Element-wise(K.*u)" multiplication in gain blocks instead of "Matrix(K*u) (u vector)".

If you receive an error with the following text: "An operating system specific kernel-level driver for the specified card could not be found". Turn off the helicopter for 30sec. Then make sure that the helicopter is turned on and try to clean and re-build the program. This error might be caused by a Quarc driver on the computer that has crashed, this often happens if you attempt to run the program while the helicopter is off. To restart the driver you must unfortunately restart the computer.

1.5.2 IMU

If you receive an error with the following text: "It was not possible to connect to the specified URI". This is either caused by having specified the wrong PORT for the IMU application, or by the Arduino not being detected by Windows. Open device manager and check that you have the correct port. If the Arduino does not show up then plug it out and in again. If neither of these things works, contact the teaching assistant.

If you do not receive an error but there is no new data from the IMU. The Arduino might have crashed and requires a re-start. Press the restart button (the only button) on the Arduino. If you continue to not receive data contact the teaching assistant.

If the values that come from the IMU-block are unreasonable, either constant or super high, restart the Arduino. If this does not work contact the teaching assistant. Make sure to look at the values that come out of the IMU block to ensure that the problem is not caused by your code.

If you receive new data, and the data looks good. Then the problem is in your code.

2 Tasks

Make sure you read the introduction before continuing, especially the chapter on safety. The introduction contains vital information for avoiding severed fingers and saving time-series longer than 10s. Follow the debugging guide before asking for help due to technical difficulties.

2.1 Part I – Monovariable control

2.1.1 Task 1 - Manual control

The first attempt to control the helicopter will be done using feedforward. Let the signal from the x-axis of the joystick connect directly to the voltage difference V_d, and the signal from the y-axis of the joystick connect directly to the voltage sum V_s. The motors do not need a large voltage difference to increase the pitch angle. It might therefore be a good idea to add a small gain from the output of the joystick to the voltage difference, V_d. It is rather difficult to control the helicopter this way. Be careful not to crash the helicopter! One person on the group should stand ready to catch the counterweight.

2.1.2 Task 2 - Parameter identification

Before you continue, identify that the positive direction given by the encoders matches the positive direction given in the task. Handle any deviations. The identification can be done by manually moving the helicopter with the rotors off.

The encoder values are set to zero every time Simulink is connected to the helicopter. Assuming that the helicopter head rests on the table when Simulink is connected, the encoder outputs are not the same as the elevation angle, e. Add constants to the encoder outputs such that the output of the encoder for the elevation is zero when the arm between the elevation axis and the helicopter head is horizontal.

Before we can implement the controller developed in the preparations, we need to determine the motor force constant K_f. By measurement, obtain the value for the voltage V_s which makes the helicopter maintain the equilibrium value $e = 0$ (the arm between the elevation axis and the helicopter head is horizontal). Note that this values is equal to $V_{s,0}$. Use this value to calculate the motor force constant K_f. Use the calculated value of K_f for the remainder of the lab.

2.1.3 Task 3 - PD control

In this task, the elevation controller already given in the Simulink diagram should be used. The output from the elevation controller is $\tilde{V}_s$. Add $V_{s,0}$ to the output of the controller to get V_s.

Implement the PD pitch controller developed in the preparations. Use the x-axis of the joystick as the reference signal, p_c.

Experiment with different pole placements, and see how the system behaves. Discuss your findings, and discuss the cause for any observed discrepancies.

Tune in a good performance that you can demonstrate in the oral evaluation. Make a presentation friendly copy that can be quickly started and contains a copy of all necessary Matlab scripts.

2.2 Part II – Multivariable control

2.2.1 Task 1 - Implementation

Implement the controller (without integral effect) developed in the preparations. Let the x-axis of the joystick provide the reference p_c for the pitch angle, and the y-axis the reference $\tilde{e}_c$ for the elevation rate.

2.2.2 Task 2 - Pole placement

Use pole placement to design the $\boldsymbol{K}$ matrix. You may use the Matlab function $K = place(A, B, p)$ where p is a vector of poles. Experiment with different pole placements and see how the system behaves.

2.2.3 Task 3 - LQR

Instead of using pole-placement to design $\boldsymbol{K}$, we will instead use the linear quadratic regulator (LQR) algorithm. An LQR uses the same regulator as before $\boldsymbol{u} = \boldsymbol{F}\boldsymbol{r} - \boldsymbol{K}\boldsymbol{x}$, but optimizes the $\boldsymbol{K}$ matrix such that the following cost function is minimized

$$J = \int_0^\infty (\boldsymbol{x}^\intercal(t)\boldsymbol{Q}_{\mathrm{LQR}}\boldsymbol{x}(t) + \boldsymbol{u}^\intercal(t)\boldsymbol{R}_{\mathrm{LQR}}\boldsymbol{u}(t))\, dt \tag{2.1}$$

For simplicity, let the weighting matrices $\boldsymbol{Q}_{\mathrm{LQR}}$ and $\boldsymbol{R}_{\mathrm{LQR}}$ be diagonal. The gain matrix $\boldsymbol{K}$ can be computed by using the Matlab command `K=lqr(A,B,Q,R)`.

Experiment with different values for $\boldsymbol{Q}_{\mathrm{LQR}}$ and $\boldsymbol{R}_{\mathrm{LQR}}$. Discuss the results. Explain how the values in $\boldsymbol{Q}_{\mathrm{LQR}}$ and $\boldsymbol{R}_{\mathrm{LQR}}$ affect the experienced behavior of the helicopter.

2.2.4 Task 3 - LQR with integral action

Implement the integral controller developed in the preparations. Use the LQR algorithm to design the $\boldsymbol{K}$ matrix. Experiment with different values for $\boldsymbol{Q}_{\mathrm{LQR}}$ and $\boldsymbol{R}_{\mathrm{LQR}}$. Compare the response with and without integral effect. Discuss how introducing integral effect affects how $\boldsymbol{F}$ can be chosen. Test out your hypothesis.

2.2.5 Task 4 - Tuning

At the presentation present the last task of part II you managed to complete. Tune in a good response guided by your observations in the experimentation.

2.3 Part III – Luenberger observer

The IMU communicates with an Arduino Micro that sends the resulting data to the computer over a serial communication port. To communicate with the Arduino you need to download the "IMU.slx" file from Blackboard and copy the content into your project. The serial communication sets a random port number for the Arduino. This information must be passed on to Simulink. Follow the setup guide written in the Simulink diagram to set the correct port. Note that the port might change from one lab day to the next! Measurements will not arrive in every Simulink time-step. The new-signal output will be equal to 1 when there are new measurements, and equal to 0 when there are no new measurements. The gyroscope and accelerometer outputs are vectors with 3 elements given in the order x, y, z as defined by Fig. 1.1b.

2.3.1 Task 1 - IMU characteristics

Disconnect the voltage-signals into the helicopter block. Plot the gyroscope values against the encoder-rates and plot the accelerometer values by themselves. In this task, we will look at the data from the IMU when we manually move the helicopter (and the rotors are not running!). Write down what you observe, with special emphasis on the following points:

- Hold the helicopter at the linearization point and rotate the helicopter about one axis at the time. How does the gyroscope output compare to the encoder-rates? You might need be rearrange the gyro-outputs so that the first element corresponds to pitch, the second to elevation, and the third to travel. This error is caused by the IMU not having the same coordinate system as our lab-setup. If alterations are needed, apply them to the accelerometer as well.
- Try to rotate the helicopter around one axis, and then rotate it around a second axis while maintaining the rotation about the first axis. Explain what you see.
- Look at the accelerometer output. What do you see?

2.3.2 Task 2 -Transfomrations

You probably noticed that we could not use the gyro-measurements directly as they did not give correct results when the helicopter was rotated. To counteract this we will need to apply a rotation that uses the pitch and elevation angles to calculate the correct pitch-, elevation-, and travel-rate. To save you some cumbersome trigonometry, a Simulink block that performs this rotation is supplied in the "IMU.slx" file. Use the encoder angles as input to this block.

We wish to use the accelerometer measurements to indirectly measure the orientation of the helicopter. Implement the equations for pitch and elevation derived in the preparations in a subsystem. During the start-up of the program measurements from the IMU will be zero, leading to a division by zero. Implement some logic in the sub-system such that the output will be zero when a division by zero happens.

Compare the transformed gyro output with the rates from the encoder, and the transformed accelerometer measurements with the elevation and pitch angle from the encoder. Verify that they match, and discuss any discrepancies.

2.3.3 Theoretical discussion

Compare and discuss your results regarding observability in problem 2 in the preparations. What is needed for a state to be observable? Why must some states be measured while others don't

need to?

Discuss the assumptions underlying the angle measurements based on the accelerometer readings.

2.3.4 Problem 4 - State estimator

In this task we will use all the measurements we have available. As there is a lot of noise in the measurements, we will use a state estimator to smooth the signal. A linear observer for the system of the following form is warranted.

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{L}(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}}) \tag{2.2}$$

Implement the observer in Simulink and use the latest well-working controller you developed in the previous days. It is preferred that you use the LQR with integral action if it worked well. Continue using the encoders for feed-back for now, and compare the state estimates with the encoder values and the measurement. Note that the lower inputs on the scope block are drawn on-top of higher inputs. Use this to your advantage so that noisy signals end in the back of the plot.

We wish to tune the observer using pole-placement. This can be done using the Matlab function $L = place(A', C', p)'$ where p is a vector of poles. Note that A, C, and the result are transposed in the Matlab function when placing poles for observers. Experiment with different pole placements. Discuss your findings and compare them with the theory.

Find a somewhat good tuning. Now disconnect the encoder and use the estimated states from the encoder instead. The measurements to the encoder should be the transformed values from the IMU. Tune the filter based on experience from the experimentation. Make a separate file that is to be presented.

2.4 Part IV – Kalman filter

2.4.1 Task 1 - Noise estimate

To begin with, we need an estimate of the measurement noise, $\boldsymbol{R}_\mathrm{d}$. Produce an experimental time-series when the helicopter is laying still on the ground as well as a time-series when the helicopter is kept stationary at the linearization point while flying. You can use one of the controllers from the subsequent tasks to stabilize the system at the linearization point. Discuss the appearance of the noise in both cases. Would you classify it as white noise? Compare the signal covariances for the two cases. Explain the difference. Use the experimental covariance while flying as your $\boldsymbol{R}_\mathrm{d}$ matrix. The covariance can be evaluated using the $cov(A)$ function in Matlab.

2.4.2 Task 2 - Discretization

For all of the subsequent tasks we will use the full state-space model with $\boldsymbol{x} = [p \quad \dot{p} \quad e \quad \dot{e} \quad \lambda \quad \dot{\lambda}]$. Discretize the continuous-time deterministic system with the same time-step as Simulink to obtain $\boldsymbol{A}_\mathrm{d}$, $\boldsymbol{B}_\mathrm{d}$ and $\boldsymbol{C}_\mathrm{d}$. The provided Simulink diagram operates in fixed-step mode with a time-step of 0.002s. You may employ the `c2d` function in Matlab.

2.4.3 Task 3 - implementation

Implement the prediction step as a Matlab function in Simulink. Implement the correction step as a separate Matlab function that also includes the new-data signal.

Data from the IMU arrives at a slower rate than the update frequency of the Simulink program. This can be identified by looking at the "New data" output from the IMU. This results in the output from the IMU being constant over multiple time-steps. In the Luenberger observer we ignored this fact, in this task we will take this into account by only correcting our predictions when new data has arrived. When there is no new measurement available, make the corrected state estimate and error covariance be equal to the predicted state estimate and error covariance.

$$\hat{\boldsymbol{x}} = \bar{\boldsymbol{x}} \tag{2.3}$$

$$\hat{\boldsymbol{P}} = \bar{\boldsymbol{P}} \tag{2.4}$$

Nb: Take extra care with regards to the time-steps of the predictions and corrections. The unit-delay block in Simulink can be used to delay the signal one time-step.

2.4.4 Task 4 - experimentation

Use the encoders for feedback to your regulator. Use the same regulator as with the Luenberger observer.

The matrix $\boldsymbol{Q}_\mathrm{d}$ describes the uncertainty in our model. Experiment with different diagonal $\boldsymbol{Q}_\mathrm{d}$ matrices. Compare the corrected state estimates, $\hat{\boldsymbol{x}}$, with the encoders and the measurements. Hypothesize based on theory what would happen if the elements of Q were 0 or infinity? Test out your hypothesis and discuss the results. You might have to use a large number instead of infinity.

Add a manual switch to the new-data signal, such that you can artificially disconnect it. Experiment with disconnecting and connecting the signal while the program is running. Look at the corrected state estimate, $\hat{\boldsymbol{x}}$, and the covariance of the estimate, $\hat{\boldsymbol{P}}$. Discuss your findings with basis in theory.

Discuss similarities and differences between the Luenberger observer and the Kalman filter. Compare their performance (after tuning the Kalman filter as well).

2.4.5 Task 5 - Tuning

Find a somewhat good tuning for the Kalman filter. Use the Kalman filter estimate for feedback instead of the encoder values. Continue tuning until you are satisfied with the performance. Make a copy of your files that is to be presented.

2.4.6 Task 6 - Oral evaluation

Ensure that you have all the files that you need ready for the oral evaluation. The schedule at the evaluation is tight, so you need to be able to get the helicopter up and running as quick as possible.

Bring a stabled color printed, **A4**, copy of the report to the oral presentation. The printers are guaranteed to fail when you are in a hurry, so don't attempt to print the report 15 min before the evaluation.