Manual for 3-DOF Helicopter Lab

Rita Laezza[*]

Department of Electrical Engineering,
Chalmers University of Technology
November, 2022

1 Introduction

During the Linear Control Systems Design (SSY285) course, you have learned to design optimal control strategies for linear time-invariant (LTI) systems. In this lab, these skills will be put to the test on a real experimental setup of a helicopter with 3 Degrees of Freedom (DOF), designed by Quanser. The 3-DOF Helicopter model that is used in this laboratory is analogous to a tandem rotor helicopter such as the Boeing HC-1B Chinook shown in Figure 1.
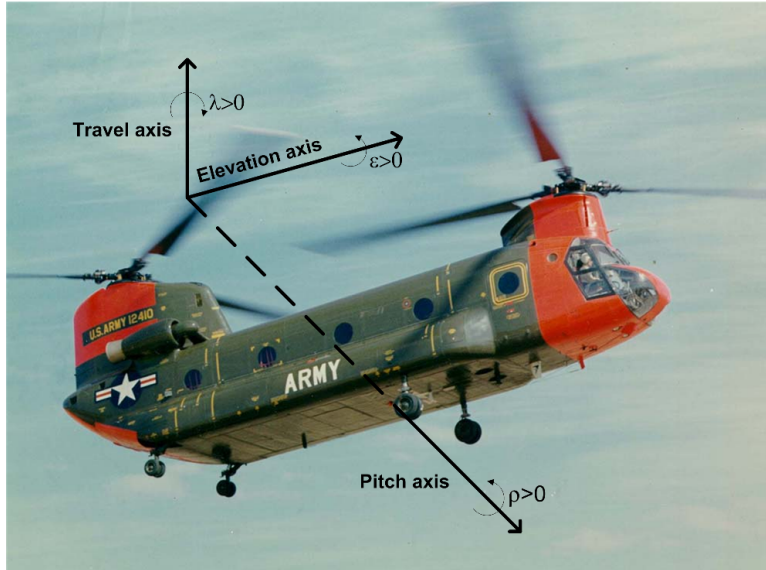

Figure 1: Boeing HC-1B Chinook. Source: [2]

This lab manual provides the necessary background information related to the equipment, **a list of preparation exercises to be completed before the session** (Sections 1-4) and a number of tasks which should be carried out at the lab (Section 5). You are expected to read the entire document and to successfully complete all of the non-optional preparation exercises before your lab session.

The purpose of the lab is for you to familiarize yourself with useful computer software for model analysis and control design. First, you should linearize a given nonlinear model, by hand and/or using tools from Simulink. The resulting linearized model should then be validated, by comparing it to the nonlinear model. Finally, based on the linear model you should perform control design, using Linear Quadratic (LQ) optimization principles to control the process elevation axis, travel axis and travel axis speed.

1.1 Equipment

The 3-DOF Helicopter plant is depicted in Figure 2. Two DC motors are mounted at the two ends of a rectangular frame and drive two propellers. The motors' axes are parallel and the thrust vector is normal to the frame. The helicopter frame is suspended from in instrumented joint mounted at the end of a long arm and is free to pitch about its center. The arm is installed on an additional 2-DOF instrumented joint which allows the helicopter body to move in elevation and yaw directions. The other end of the arm carries a counterweight such that the effective mass of the helicopter is light enough for it to be lifted using the thrust from the motors [2].

Figure 2: 3-DOF Helicopter equipment. Source: [2]

A positive voltage applied to the front motor causes a positive pitch while a positive voltage applied to the back motor causes a negative pitch. A positive voltage to either motor also causes an elevation of the body (i.e., pitch of the arm). If the body pitches, the thrust vectors result in a travel of the body (i.e., yaw of the arm) as well. The vertical base is equipped with an eight-contact slip ring. Electrical signals to and from the arm and helicopter are channeled through the slip ring to eliminate tangled wires, reduce friction, and allow for unlimited and unhindered travel [2].
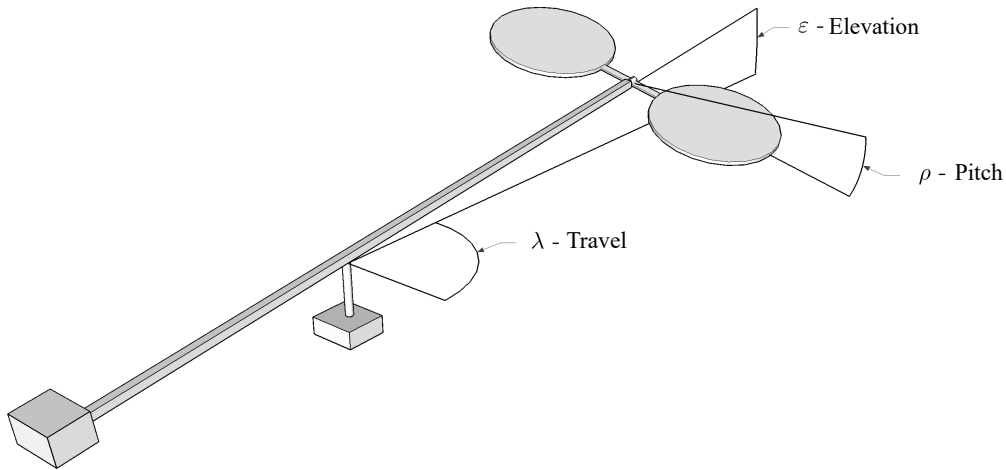
1.2 Model

Figure 3: Schematics of 3-DOF Helicopter, indicating elevation, pitch and travel axes.

Figure 3 depicts the layout of the helicopter device. The three angular position states we will use for our model are the elevation (ε), pitch (ρ) and travel (λ) angles. The input voltage for the front and back motor will be denoted by V_f and V_b, respectively. A positive voltage implies air being blown downwards creating an upwards lift. A simplified nonlinear model of the process is given by the following input-affine system:

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where

$$x = \left[\varepsilon, \rho, \lambda, \dot{\varepsilon}, \dot{\rho}, \dot{\lambda} \right]^T \tag{2}$$

$$u = [V_f, V_b]^T \tag{3}$$

The function $f(x)$ is defined as

$$f(x) = \begin{bmatrix} \dot{\varepsilon} \\ \dot{\rho} \\ \dot{\lambda} \\ p_1 \cos \varepsilon + p_2 \sin \varepsilon + p_3 \dot{\varepsilon} \\ p_5 \cos \rho + p_6 \sin \rho + p_7 \dot{\rho} \\ p_9 \dot{\lambda} \end{bmatrix} \tag{4}$$

and $g(x)$ is defined as

$$g(x) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ p_4 \cos \rho & p_4 \cos \rho \\ p_8 & -p_8 \\ p_{10} \sin \rho & p_{10} \sin \rho \end{bmatrix} \tag{5}$$

Table 1: Modelling parameters

Parameter	Value	Description
M_f	0.71 kg	Mass of front propeller assembly
M_b	0.71 kg	Mass of back propeller assembly
M_c	1.69 kg	Mass of the counterweight
L_d	0.05 m	The length of pendulum for elevation axis
L_c	0.44 m	Distance from pivot point to counterweight
L_a	0.62 m	Distance from pivot point to helicopter body
L_e	0.02 m	The length of pendulum for pitch axis
L_h	0.18 m	The distance from the pitch axis to motor
g	9.81 m/s^2	Gravitational acceleration
K_m	0.12 N/V	Propeller force-thrust constant
J_ε	0.86 $kg \cdot m^2$	Moment of inertia about the elevation axis
J_ρ	0.044 $kg \cdot m^2$	Moment of inertia about the pitch axis
J_λ	0.82 $kg \cdot m^2$	Moment of inertia about the travel axis
η_ε	0.001 $kg \cdot m^2/s$	Coefficient of viscous friction, elevation axis
η_ρ	0.001 $kg \cdot m^2/s$	Coefficient of viscous friction, pitch axis
η_λ	0.005 $kg \cdot m^2/s$	Coefficient of viscous friction, travel axis

Finally, the weights p are defined as

$$p_1 = \left[-(M_f + M_b)gL_a + M_cgL_c\right]/J_\varepsilon \qquad p_2 = \left[-(M_f + M_b)g(L_d + L_e) + M_cgL_d\right]/J_\varepsilon$$

$$p_3 = -\eta_\varepsilon/J_\varepsilon \qquad\qquad\qquad\qquad p_4 = K_mL_a/J_\varepsilon$$

$$p_5 = (-M_f + M_b)gL_h/J_\rho \qquad\qquad p_6 = -(M_f + M_b)gL_e/J_\rho$$

$$p_7 = -\eta_\rho/J_\rho \qquad\qquad\qquad\qquad p_8 = K_mL_h/J_\rho$$

$$p_9 = -\eta_\lambda/J_\lambda \qquad\qquad\qquad\qquad p_{10} = -K_mL_a/J_\lambda \qquad\qquad (6)$$

and Table 1 lists the physical quantities used to compute the weights in equation (6), together with a description of the parameters.

Remark 1. *All exercises related to this laboratory are carried out in continuous time.*

1.3 LTI State-Space Model

As this course focuses on linear control methods, the system will have to be linearized in order to perform control design. The standard form for a dynamic model, suitable for simulation and controller design, is the state-space form

$$\begin{aligned}
\dot{x}(t) &= A_\delta x(t) + B_\delta u(t) \\
y(t) &= C_\delta x(t) + D_\delta u(t)
\end{aligned} \qquad (7)$$

where $x(t)$ is the state, $u(t)$ is the control input, A_δ, B_δ, C_δ, and D_δ are state-space matrices. For a 3-DOF Helicopter, the linear states are

$$x(t) = [\delta\varepsilon(t)\ \delta\rho(t)\ \delta\lambda(t)\ \delta\dot{\varepsilon}(t)\ \delta\dot{\rho}(t)\ \delta\dot{\lambda}(t)]^T, \qquad (8)$$

where $\delta\cdot$ refers to small variable changes around the operating point. According to equations (1)-(6) the system state equations are given by

$$\begin{aligned}
\frac{d\varepsilon(t)}{dt} &= \dot{\varepsilon}(t) = h_1(t) \\
\frac{d\rho(t)}{dt} &= \dot{\rho}(t) = h_2(t) \\
\frac{d\lambda(t)}{dt} &= \dot{\lambda}(t) = h_3(t) \\
\frac{d\dot{\varepsilon}(t)}{dt} &= h_4(t) \\
\frac{d\dot{\rho}(t)}{dt} &= h_5(t) \\
\frac{d\dot{\lambda}(t)}{dt} &= h_6(t)
\end{aligned} \qquad (9)$$

The control inputs are

$$u(t) = [\delta V_f(t)\ \delta V_b(t)]^T = [u_1(t)\ u_2(t)]^T. \qquad (10)$$

We would like to linearize the system at the origin, that is to set the initial condition as: $x_0 = [0\ 0\ 0\ 0\ 0\ 0]^T$. For the inputs $u_0 = [V_{f0}, V_{b0}]$, it is reasonable to assume that the front and back voltages are equal, i.e. $V_{f0} = V_{b0}$. To find V_{f0}, simply solve $f(x_0) + g(x_0)u_0 = 0$ for u_0. After determining the operating point, linearization results in matrices A_δ and B_δ as

$$A_\delta = \left.\begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \cdots & \frac{\partial h_n}{\partial x_n} \end{bmatrix}\right|_{(\varepsilon=\rho=\lambda=\dot\varepsilon=\dot\rho=\dot\lambda=0, u_0)}$$

$$B_\delta = \left.\begin{bmatrix} \frac{\partial h_1}{\partial u_1} & \cdots & \frac{\partial h_1}{\partial u_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial u_1} & \cdots & \frac{\partial h_n}{\partial u_n} \end{bmatrix}\right|_{(\varepsilon=\rho=\lambda=\dot\varepsilon=\dot\rho=\dot\lambda=0, u_0)} \tag{11}$$

Only angles can be sensed by encoders, and not angular rates. Hence, the states $x_1(t)$, $x_2(t)$ and $x_3(t)$ are measured and the outputs of the 3-DOF Helicopter are obtained as

$$y(t) = [\delta\varepsilon(t)\ \delta\rho(t)\ \delta\lambda(t)]^T = [x_1(t)\ x_2(t)\ x_3(t)]^T. \tag{12}$$

giving the following matrices, since the control input does not influence the measurement:

$$C_\delta = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad D_\delta = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{13}$$

Exercise 1 (optional) Linearize the nonlinear model defined in (1)-(6), around the operating point. You can use generalized series expansions of the trigonometric functions (for small angles). Find the numerical values for the matrices A_δ and B_δ, using the parameter values from Table 1.

Answer

Exercise 2 A Simulink model describing the nonlinear system is available on the course home page, in `nlmodel_3D.mdl`. Instead of computing the linearized system by hand, it is possible to use the **Model Linearizer** app which is part of the **Simulink Control Design** add-on. Use this Simulink tool to obtain a linearized model. In the LINEAR ANALYSIS tab, the point around which the system is linearized can be set to "Model Initial Condition" [3]. Make sure to compute V_{f0} and V_{b0} in the `nl_parameters_3D.m` script and run it before running the simulation. Load the resulting $(A_\delta, B_\delta, C_\delta, D_\delta)$ matrices into the same script.

1.4 Elevation and Travel Tracking Errors

In this laboratory, you will control the elevation and travel of the helicopter. Note that the pitch will indirectly be controlled in order to move the helicopter around the travel axis. To this end, one can define the appropriate reference tracking errors. Let us augment the LTI model from equation (7) with two integral states:

$$x_e(t) \equiv \begin{bmatrix} x(t) \\ z_\varepsilon(t) \\ z_\lambda(t) \end{bmatrix} \tag{14}$$

with,

$$z_\varepsilon(t) = \int (\delta\varepsilon(t) - \delta\varepsilon_{ref}(t))\mathrm{d}t \tag{15}$$

$$z_\lambda(t) = \int (\delta\lambda(t) - \delta\lambda_{ref}(t))\mathrm{d}t \tag{16}$$

where $\delta\varepsilon_{ref}(t)$ and $\delta\lambda_{ref}(t)$ are the reference signals. Note that these integral states correspond to the cumulative error of the elevation and travel angles with respect to their desired references.

Exercise 3 (optional) Expand the linear model with the above elevation tracking error and find the numerical values for the augmented system as (A, B, C, D). Include this augmented system in `nl_parameters_3D.m`. Note that when augmenting the linearized system with two new states then $C \in \mathbb{R}^{2\times8}$.

Answer

2 Open-Loop System Analysis

Exercise 4 It is important to investigate how well the linear model represents the nonlinear one. Simulate both models in open loop using an all zero initial state and constant input u_0. Implement the linearized model in `nlmodel_3D.mdl`, without changing the nonlinear system. Analyze the output, what happens to the various states? Compare the results, how do the linear and nonlinear models' reactions differ?

Answer

2.1 Stability

Stability of a state space model can be analyzed based on the eigenvalues of the matrix A.

Exercise 5 (optional) Are the linearized $(A_\delta, B_\delta, C_\delta, D_\delta)$ and augmented (A, B, C, D) systems asymptotically stable? Does the inclusion of extra integral states change the stability property? Are they input-output stable? Find the transmission zeros of the systems. What can you conclude from the zeros?

Answer

2.2 Observability

If by only measuring the output $y(t)$ and input $u(t)$, the states of the system can be deduced at any time in final time, the system is called observable. Observability can be checked by different alternatives, such as by checking the rank of the observability matrix or by means of the PBH (Popov-Belevich-Hautus) test.

Exercise 6 (optional) Determine the observability matrix and check if the system is observable using Kalman's rank test. Check both systems, i.e. $(A_\delta, B_\delta, C_\delta, D_\delta)$ and (A, B, C, D). Does the inclusion of additional integral states change the observability property?

Answer

（空白框）

Exercise 7 (optional) Check observability by means of the PBH test.
Answer

（空白框）

2.3 Controllability

If the control input, $u(t)$, of a system can take the state vector, $x(t)$ from any non-zero initial state to the origin in finite time, the LTI system is said to be (full state) controllable. When checking controllability there are different possible approaches, namely by checking the rank of the controllability matrix or by means of the PBH test.

Exercise 8 (optional) Find the controllability matrix and check if the system is controllable using Kalman's rank test. Check both systems, i.e. $(A_\delta, B_\delta, C_\delta, D_\delta)$ and (A, B, C, D). Does the inclusion of additional integral states change the controllability?
Answer

（空白框）

Exercise 9 (optional) Check controllability by means of the PBH test.
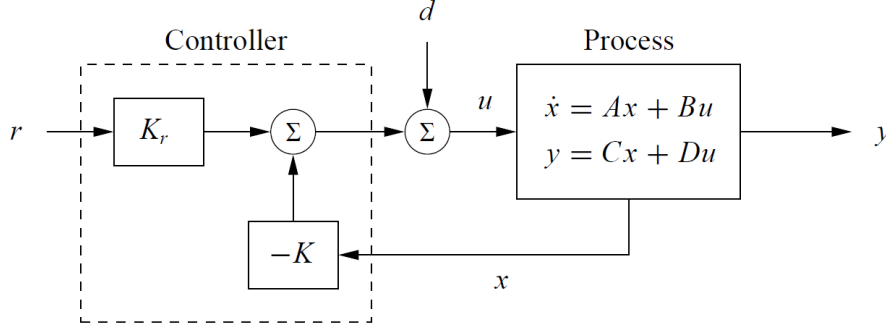**Answer**

（空白框）

3 Feedback Control

Figure 4: A feedback control system with state feedback. The controller uses the system state x and the reference input r to command the process through its input u. We model disturbances via the additive input d. Source: [4]

Consider the system defined in (7). A feedback control loop is to be designed to drive the helicopter to follow a given time varying reference signal $r(t)$. For a linear feedback law (without disturbances d), the input can be written as:

$$u(t) = -Kx(t) + K_r r(t), \tag{17}$$

For this control input, illustrated in Figure 4, the closed-loop state-space equation becomes:

$$\dot{x}(t) = A_\delta x(t) + B_\delta(-Kx(t) + K_r r(t)) = (A_\delta - B_\delta K)x(t) + B_\delta K_r r(t) \tag{18}$$

The state feedback gain K can be obtained in different ways, for example by pole placement design or by linear quadratic optimization (see Section 3.1).

Note that K_r does not affect the stability of the system (determined by eigenvalues of $A_\delta - B_\delta K$) but it does affect the steady-state solution. Since the equilibrium point ($\dot{x}(t) = 0$) and steady state output $y_e(t)$ for the closed loop system (for $D_\delta = 0$) are given by:

$$x_e = -(A_\delta - B_\delta K)^{-1} B_\delta K_r r \tag{19}$$
$$y_e = C_p x_e$$

where C_p is the matrix selecting which states are to be controlled, and the value of K_r should be chosen such that the output equals the desired reference $y_e = r$. In other words, such that the stationary gain from r to x is the identity matrix I. Therefore, K_r can be computed as :

$$K_r = -(C_p(A_\delta - B_\delta K)^{-1} B_\delta)^{-1} \tag{20}$$

3.1 Linear Quadratic Regulator (LQR)

A stationary LQR can be used to find the feedback control gain K, which minimizes the following quadratic cost, with respect to a control input $u(t)$:

$$J = \frac{1}{2} \int_0^\infty \left(x(t)^T Q_x x(t) + u(t)^T Q_u u(t) \right) dt. \tag{21}$$

where Q_x and Q_u are penalty weightings, $Q_x \in \mathbb{R}^{n \times n}$, $Q_u \in \mathbb{R}^{2 \times 2}$, where n is the number of states. By solving the Control Algebraic Riccati Equation (CARE) offline we may find the solution of $\min_{u(t)} J$. Based on the solution matrix of the Riccati equation, K can be obtained.

3.2 LQR with Integral Action

An alternative way to achieve reference tracking, is to augment the system with integral states z, as in Section 1.4, which allow the computation of the cumulative tracking error[1], $\int(y-r)\mathrm{d}t$. If for example we would like an integral action at the plant outputs, then the augmented system (for $D_\delta = 0$) is given by:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{z}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} A_\delta & 0 \\ C_I & 0 \end{bmatrix}}_{A} \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} + \underbrace{\begin{bmatrix} B_\delta \\ 0 \end{bmatrix}}_{B} u(t) + \begin{bmatrix} K_r \\ -I \end{bmatrix} r(t) \tag{22}$$

$$y(t) = \underbrace{\begin{bmatrix} C_I & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} \tag{23}$$

where C_I selects the system states needed to compute the new integral states. With this formulation, the result is a dynamic controller which gets rid of the steady state error by means of tracking error integration (*consequently K_r is not needed and can be set to zero*). Therefore the control policy becomes:

$$u(t) = -K \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} = -[K_P \ K_I] \begin{bmatrix} x(t) \\ z(t) \end{bmatrix} \tag{24}$$

where K is the feedback gain, which can be decomposed into proportional K_P and integral K_I gains. The resulting equilibirum point for the system (for $D = 0$) is then:

$$x_e = -(A - BK)^{-1}B(K_r r - K z_e) \tag{25}$$
$$y_e = C x_e$$

Note that z_e is not specified. Since the equilibrium point is given by $\dot{x} = 0$, this automatically implies that $\dot{z} = y - r = 0$, at equilibrium, i.e. the output equals the reference.

Exercise 10 Design a feedback controller for the linearized system $(A_\delta, B_\delta, C_p)$ in order to achieve elevation and travel angle tracking, based on LQ principles. In the `nl_parameters_3D.m` script, compute K based on LQ optimization and then obtain K_r. In `nlmodel_3D.slx`, implement also the closed loop system from Figure 4, while still keeping the previous open-loop implementations of the linearized and nonlinear models.

Exercise 11 (optional) Design an LQR controller with integral action for the system augmented system (A, B, C), in the `nl_parameters_3D.m` script. Implement the LQR closed loop in `nlmodel_3D.mdl`, while keeping previous implementations.

---

[1]Depending on how the tracking error is defined, i.e. $\pm(y - r)$, the signs in equation (22) change.

4 Closed-Loop Simulation

In this section you will use the Simulink model to simulate the closed-loop behaviour of designed controllers.

Exercise 12 Simulate your controllers from **Exercise 10** and **11 (optional)** on both the linear and nonlinear system. Make sure to add u_0 to the control input of the nonlinear model such it rests at steady state. Use reference steps on ε and λ separately and pay specific attention to their outputs. Remember that your state space model uses radians. Comment on your results.
Answer

5 Laboratory Session

If you have successfully completed the preparatory exercises above, you should be ready for controlling the physical system. Bring your version of the `nl_parameters_3D.m` script with a loaded version of the linearized model. Be prepared to present your results and explain your conclusions.

5.1 Introduction

In order to start the lab, log into the PC at your station, with your CID and password. Download from the Canvas page the Helicopter lab files. Unzip the *3DOFHelilab* file into a folder on your Desktop. In this part of the laboratory you will examine how regulation of the helicopter's elevation works based on the linearized model defined in `linnlsys.m`. For all tasks, you are required to specify four matrices: $K \in \mathbb{R}^{2 \times n}$, where n is the number of states, $K_I \in \mathbb{R}^{2 \times 2}$, $K_r \in \mathbb{R}^{2 \times 2}$ and $V_K \in \mathbb{R}^2_+$, which represent the proportional gain matrix, integral gain matrix and voltage bias respectively. The latter is simply a constant voltage added to the motors, which may also be a scalar since at equilibrium the voltage for both motors should be equal. This matrices are defined in the `lab_init.m` script, which must be modified. **Save these matrices in each task, so that you can easily go back to previous controllers.**

In Sections 5.3 and 5.4 you will be working on elevation and travel angle control, respectively, while in Section 5.5 you will move to travel speed control. In order to execute the files for both these experiments the procedure is the same, for the *initial* run:

1. Open the correct Simulink file.

2. Click on "Build Model" ⊞.

3. Click on "Connect to Target" ⊙, within the **Simulation** menu in Simulink.

4. Click on "Run" ▶, after you have called the lab assistant to verify your solution.

Every time a change is made in the `lab_init.m` script, it must be executed before you run the simulation again, i.e. before you press "Run".

Remark 2. *You only need to build the code once. If the build is unsuccessful, make sure that the correct compiler is selected in MATLAB. In the Command Window execute the following command:* `mex -setup`*. The result should state:* "MEX configured to use '**Microsoft Visual C++ 2013 Professional (C)**' for C language compilation". *If something else is printed, execute the appropriate suggested command to change to the correct compiler.*

5.2 Warnings

It is very **important that you at all times are ready to physically intervene** in case the process and the controller turns out to be unstable. There is also an emergency stop button which will cut power to the motors when pressed, please use this option or the stop button in Simulink at any sign of trouble. **Before you run the first controller on the real system, call the laboratory assistant**.

When running your controllers, always start with the helicopter at its lowest elevation. Furthermore make sure to start at zero travel and pitch angles. Note that for the latter, one person should gently hold up one of the propellers, otherwise the helicopter will start sideways.

When stopping your controllers, make sure to soften the fall by reaching out and grabbing the helicopter with your hand.

5.3 Elevation Control (approx. 2 hours)

In these first experiments, you will use `lab_elev_travel.mdl` to control the process. Make sure to follow the procedure explained in Section 5.1 and respect the warnings in Section 5.2.

Task 1 (Feedback control) Fill in all the blank matrices and run the script. You should set K_I and V_K to zero for this task and keep Q_u fixed at `diag([0.1 0.1])` for all tasks. In this section, the system has two outputs, ε and λ. Also compute K_r using equation (20). First increase the weight on ε just enough to get the helicopter to lift off. Then find the weights on ρ and λ needed to keep the travel movement to a minimum.

Task 2 (LQ parameter tuning) Start with a conservative weight for ε, progressively choose a higher weight and record your results, keep the rest of the weights fixed. For each attempt record the weight, steady state elevation and steady state control signal.

In the previous task, the steady state error is never completely eliminated. We will try to add a base voltage V_K to the control signal. This means that without any regulator control action, the propellers should deliver enough thrust for the helicopter to be close to zero elevation, and then the controller will just need to steer the helicopter in this region.

Task 3 (Estimating the required base voltage) Using the data from the previous task, make a linear or second order approximation of the control signal, i.e. voltage to the motors, to the steady state elevation. Extrapolate your results and determine the control signal needed to reach zero elevation. Run the process with the added base voltage V_K and record its behaviour.

Task 4 (Feedback control of elevation, with added base voltage) Now try exchanging the constant elevation reference for a square wave which alternates between $-10°$ and $10°$, inside "Desired Angle from Program" block. Is the steady state error eliminated, and why?

As you should have noticed, the helicopter steers very close to zero elevation, but it does not work as well for the square wave reference.

We will keep the base voltage and use integration to remove the small steady state error. In the `lab_init.m` file, just after the linearized system has been loaded, extend the A and B (not C) matrices to include an integral state for the elevation tracking error, as in equation (15). Do not forget to also extend Q_x. The resulting K matrix should be split into proportional integrative parts as follows:

```
K = lqr(A,B,Q_x,Q_u);
K_I = [K(:,7) zeros(2,1)];
K = K(:,1:6);
```

Task 5 (Selecting an appropriate feedforward term) What is a reasonable choice for K_r now that we are using LQR with integral action? Discuss among yourselves and present your hypothesis to the lab assistant.

Task 6 (LQR control of elevation, with added base voltage) Start out with the same weights for the states as in the previous task, run the system and iterate the values of the integral state weight. Use the same square wave as before. What values for Q_x give good results? Keep a plot of the best result.

Even though you get good results in the previous task, those results are only good around the region close to zero elevation.

Task 7 (Robustness for larger set point changes) With your preferred settings from the previous task, try a square wave which alternates between $-15°$ and $15°$ instead. How do your results look now? Keep a plot of the result.

We will now remove the base voltage, i.e. Set V_K to zero, and try to control the system using only the regulator.

Task 8 (LQR control of elevation, without base voltage) Go back to the smaller square wave and try different settings for the weighting matrices. Once you've found good settings, compare your best result with the plot from **Task 6**. With the same settings, test your controller for the larger square wave and compare your result with the plot from Task **7**.

Task 9 (Robustness) You will now try adding a disturbance to the system and supply the smaller square wave as reference. Apply the physical weight supplied by the laboratory assistant to the system. Use the controller from **Task 8** and see how it reacts to the disturbance. Save a plot of the result. Next, use the feedback controller with added base voltage from **Task 4** and compare the results.

5.4 Travel Angle Control (approx. 0.5 hours)

You will now attempt to control the position of the helicopter around the travel axis, still using the `lab_elev_travel.mdl` file to test your controllers. Make sure to follow the procedure explained in Section 5.1 and respect the warnings from Section 5.2.

For travel control, there is really no use for a base voltage as the steady state input is independent of the travel angle. We will consider the same controllers as in Section 5.3. Use the weighting matrices and code from **Task 8**. This means that you will have integral action for the elevation but only proportional control for the travel angle λ.

Task 10 (LQ control of travel angle) Apply a constant elevation reference and a square wave reference of amplitude $\pm 10°$ for λ. Try out different settings for the weights related to λ and ρ, let the weights related to elevation stay as they are. Can you find any good settings? Is there a steady state error?

Extend the A and B matrices such that there are integral states for both ε and λ. Remember to do the necessary changes to Q_x as well. You should now split the resulting K matrix as:

```
K = lqr(A,B,Q_x,Q_u);
K_I = K(:,7:8);
K = K(:,1:6);
```

Task 11 (LQR control of travel angle) Start with a conservative value and try to find a good weighting matrix. How does the resulting system behave?

Task 12 (Robustness) You should now evaluate the robustness of your angle and travel control. Instead of using the physical weight, give the system a gentle push with your hand. How does the system react?

5.5   Travel Speed Control (if time permits)

In these final experiments, you will use `lab_speed.mdl` to control the process. Make sure to follow the procedure explained in Section 5.1 and respect the warnings from Section 5.2.

We will now consider ε and $\dot{\lambda}$ as system outputs. Given a constant set point for $\dot{\lambda}$, λ will in turn grow unboundedly. Thus we will have to remove any trace of λ (not $\dot{\lambda}$!) from our model. Modify A, B, C and Q_x accordingly.

Task 13 (Feedback control of travel speed) Start out with a base voltage and settings as the ones you used in **Task 8**, with the modifications above added of course. Use a constant elevation of $0°$ and set a constant speed reference of $10°/s$. Try out different weights for $\dot{\lambda}$ using varied speed set points and record the system performance.

Task 14 (LQR control of travel speed, with integral action) Add integral action to $\dot{\lambda}$, how does the system perform now? Change the reference speed.

Task 15 (Varying travel reference speed) Using your preferred settings from the previous task, apply a square wave to the speed reference such that the $\dot{\lambda}$ reference switches between say $15°/s$ and $30°/s$.

Task 16 (Varying both reference travel speed and elevation) Using the same reference for $\dot{\lambda}$ as in the previous task, add a square wave to the ε reference. Let the square wave have amplitude $10°$. Observe the system behaviour, are you satisfied with its performance?

References

[1] O Wigström, *3-DOF Helicopter Laboratory Session*, Chalmers University of Technology, Göteborg, Sweden, 2014.

[2] J Apkarian, M Lévis, C Fulford *LABORATORY GUIDE: 3 DOF Helicopter Experiment for MATLAB / Simulink ® Users*, Quanser, 2012.

[3] https://se.mathworks.com/help/slcontrol/ug/linearize-simulink-model.html

[4] K J Åström, R Murray *Feedback Systems: An Introduction for Scientists and Engineers*