Computergestuurde Regeltechniek
exercise session
Case study : Quadcopter

Oscar Mauricio Agudelo, Bart De Moor
(mauricio.agudelo@esat.kuleuven.be, bart.demoor@esat.kuleuven.be)

February 5, 2016

Problem description

The aim of this exercise session is to control the position and attitude (aircraft orientation relative to the vehicle's center of gravity) of a quadcopter by manipulating the angular velocities of its rotors. You will have to design an LQG control system that drives the quadcopter to several checkpoints as fast as possible, within a room that is 6 meters wide, 6 meters long, and 3 meters high. A checkpoint is declared reached when the quadcopter enters a sphere around it. The radius of this sphere (0.08 m) is the maximum admitted error. In addition, the control system should be robust enough to guarantee that the quadcopter can reach the checkpoints in time when carrying a small payload.

1 Quadcopter dynamics

A quadcopter is an aerial vehicle that is able to hover. It has four identical rotors arranged at the corners of a square body, and its propellers or blades have a fixed angle of attack. Figure 1 shows the diagram of the quadcopter that you will have to control. Notice that the rotors are paired, and each pair rotates in a different direction. Motors 1 and 3 rotate clockwise when looked from above, whereas motors 2 and 4 have a counter-clockwise rotation. When all the motors rotate at the same angular velocity, the torques τ_1, τ_2, τ_3 and τ_4 (these are the counter torques applied to the aircraft as a consequence of the motors rotation) will cancel each other out and the quadcopter will not spin about its z^b-axis ($\dot{\psi} = 0$). The quadcopter will hover when the angular velocities are such that the total thrust ($f_1 + f_2 + f_3 + f_4$) generated by the rotors is equal to the force of gravity.

In order describe the movement of the quadrotor and its attitude, two frames of reference are used, namely, the inertial frame and the body frame (see Figure 1). The inertial frame is defined by the ground, with gravity pointing in the negative z direction. The body frame is defined by the orientation of the quadcopter, with the rotor axes pointing in the positive z^b direction and the arms pointing in the x^b and y^b directions.

The attitude of the quadcopter is determined by three angles, namely, roll-ϕ, pitch-θ and yaw-ψ. The way of changing these angles by playing with the angular velocities of

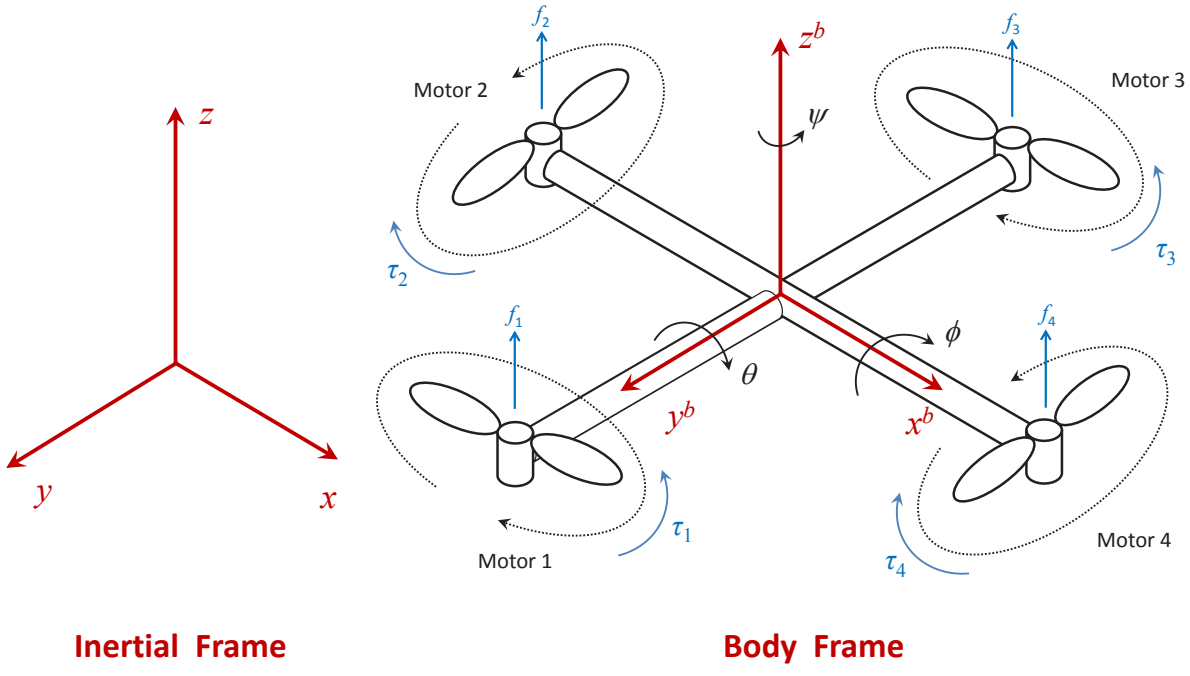

Figure 1: Quadcopter configuration. The roll, pitch and yaw angles are denoted by ϕ, θ and ψ respectively. Motors 1 and 3 rotate clockwise and motors 2 and 4 rotate counter clockwise as indicated by the arrows with black dotted lines. f_1, f_2, f_3 and f_4 are the thrusts generated by the rotors about their centers of rotation. τ_1, τ_2, τ_3 and τ_4 are the torques applied to the aircraft (counter torques) as a consequence of the spinning of the rotors.

the rotors is illustrated in Figure 2. Changes in the roll and pitch angles are accompanied by translational motion. It should be clear that a quadcopter is an underactuated vehicle, since it only has 4 actuators (rotors) for controlling 6 degrees of freedom (three translational, x, y, z and three rotational, ϕ, θ and ψ). Let's denote the angular velocities of the motors by ω_1, ω_2, ω_3 and ω_4, and the angular velocity to which the quadcopter hovers by ω_{hover}. Figure 2 shows some quadcopter motion scenarios which are explained in the following lines:

- The rolling motion corresponds to a rotation of the quadcopter about the x^b-axis. It is obtained when $\omega_2 = \omega_4 = \omega_{\text{hover}}$ and ω_1 and ω_3 are changed. For a positive rolling, we have to set $\omega_1 > \omega_{\text{hover}}$ and $\omega_3 < \omega_{\text{hover}}$. A negative rolling action is produced when we set $\omega_1 < \omega_{\text{hover}}$ and $\omega_3 > \omega_{\text{hover}}$.

- The pitch motion corresponds to a rotation of the quadcopter about the y^b-axis. It is obtained when $\omega_1 = \omega_3 = \omega_{\text{hover}}$ and ω_2 and ω_4 are changed. For a positive pitch, we have to fix $\omega_2 > \omega_{\text{hover}}$ and $\omega_4 < \omega_{\text{hover}}$. A negative pitch action is generated when we fix $\omega_2 < \omega_{\text{hover}}$ and $\omega_4 > \omega_{\text{hover}}$.

- The yaw motion corresponds to a rotation of the quadcopter about the z^b-axis. It is produced by the difference in the torque developed by each pair of rotors. Since the rotors are paired, two create a clockwise torque and two an anticlockwise one; by varying the angular speed of one pair over the other, the net torque applied to

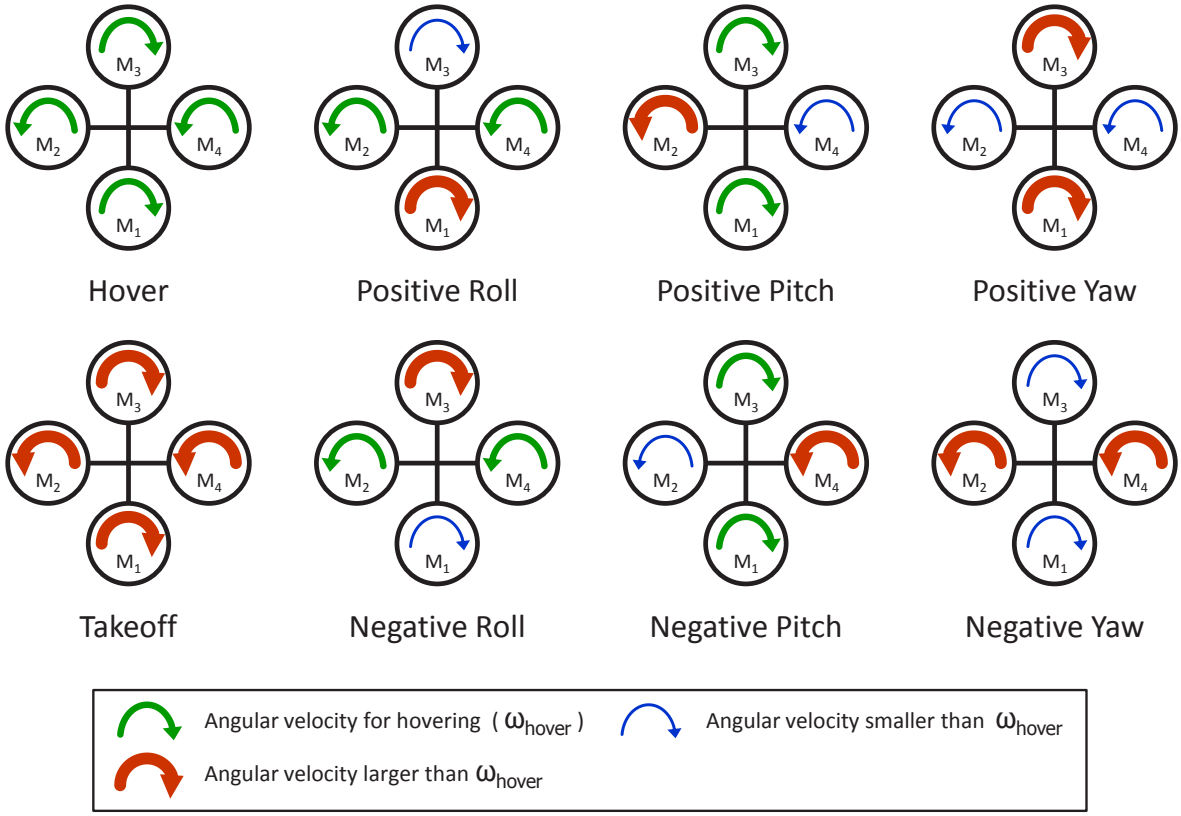

Figure 2: Illustration of the quadcopter motion obtained by varying the angular velocities of its rotors M_1, M_2, M_3 and M_4.

the aircraft changes which results in the yaw motion. For a positive yaw, we have to set $(\omega_1 = \omega_3) > \omega_{\text{hover}}$ and $(\omega_2 = \omega_4) < \omega_{\text{hover}}$. A negative yaw action is achieved when we fix $(\omega_1 = \omega_3) < \omega_{\text{hover}}$ and $(\omega_2 = \omega_4) > \omega_{\text{hover}}$

- The vertical takeoff and landing motions (change in the z-axis) are obtained by equally augmenting or diminishing the angular speed of all motors with respect to ω_{hover}.

2 Quadcopter model

The translational motion of the quadcopter in the inertial frame is described by the following set of equations:

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \boldsymbol{R}\boldsymbol{T^b} + \boldsymbol{F_D} \tag{1}$$

where x, y and z are the coordinates of the position of the quadcopter in the inertial frame, m is the mass of the system, g is the acceleration due to gravity, $\boldsymbol{F_D}$ is the drag force due to air friction, $\boldsymbol{T^b} \in \mathbb{R}^3$ is the thrust vector in the body frame, and $\boldsymbol{R} \in \mathbb{R}^{3\times3}$

is the rotation matrix that relates the body frame with the inertial frame and is defined as follows:

$$\boldsymbol{R} = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi & \sin\psi\sin\phi + \cos\psi\cos\phi\sin\theta \\ \cos\theta\sin\psi & \cos\psi\cos\phi + \sin\psi\sin\theta\sin\phi & \cos\phi\sin\psi\sin\theta - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}.$$

The drag $\boldsymbol{F_D}$ due to air friction is modelled as a force proportional to the linear velocity in each direction,

$$\boldsymbol{F_D} = -k_d \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}.$$

Here, k_d is the air friction coefficient. The thrust f_i generated by the ith rotor is given by the following expression,

$$f_i = k\omega_i^2, \quad \text{for } i = 1, ..., 4,$$

where k is the propeller/rotor lift coefficient and ω_i is the angular velocity of the ith motor. The total thrust $\boldsymbol{T^b}$ generated by the 4 rotors (in the body frame) is given by

$$\boldsymbol{T^b} = \sum_{i=1}^{4} f_i = k \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} \omega_i^2 \end{bmatrix}.$$

In this study it is assumed that the dynamics of the motors is much faster than the one of the quadcopter, and therefore is not taken into account. Since the angular velocity of each rotor is typically proportional to the applied voltage, we have that

$$\omega_i^2 = c_m v_i^2, \quad \text{for } \ i = 1, ..., 4,$$

where c_m is a constant and v is the voltage applied to the rotor.

While it is convenient to have the linear equations of motion in the inertial frame, the rotational equations of motion are useful to us in the body frame, so that we can express rotations about the center of the quadcopter instead of about the inertial center. To this end we can use the Euler's equations for rigid body dynamics, which are defined as follows:

$$\boldsymbol{I\dot{\omega}} + \boldsymbol{\omega} \times (\boldsymbol{I\omega}) = \boldsymbol{\tau} \tag{2}$$

where "$\times$" denotes cross product, $\boldsymbol{I} \in \mathbb{R}^{3\times3}$ is the inertia matrix, $\boldsymbol{\omega} = [\omega_x \ \omega_y \ \omega_z]^T$ is the angular velocity vector, and $\boldsymbol{\tau} = [\tau_\phi \ \tau_\theta \ \tau_\psi]^T$ is the vector of external torques.

We can model the quadcopter as two thin uniform rods crossed at the origin with a point mass (motor) at each end. This results in a diagonal inertia matrix of the following form:

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix},$$

where I_{xx}, I_{yy} and I_{zz} are the moments of inertia of the quadcopter about the x^b, y^b and z^b axes respectively. After computing the cross product, equation (2) reduces to:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} - \begin{bmatrix} (I_{yy} - I_{zz})\,\omega_y\omega_z \\ (I_{zz} - I_{xx})\,\omega_x\omega_z \\ (I_{xx} - I_{yy})\,\omega_x\omega_y \end{bmatrix}. \tag{3}$$

The roll τ_ϕ and pitch τ_θ torques are derived from standard mechanics as follows:

$$\tau_\phi = L\left(f_1 - f_3\right) = Lk\left(\omega_1^2 - \omega_3^2\right) = Lkc_m\left(v_1^2 - v_3^2\right)$$
$$\tau_\theta = L\left(f_2 - f_4\right) = Lk\left(\omega_2^2 - \omega_4^2\right) = Lkc_m\left(v_2^2 - v_4^2\right),$$

where L is the distance between the rotor and the quadcopter center (radius). As it was discussed earlier, all the rotors apply torques to the aircraft about its z^b-axis while they rotate. In order to have an angular acceleration about the z^b-axis, the total torque generated by the rotors has to overcome the drag forces. The total torque about the z^b-axis, that is the yaw τ_ψ torque is given by the following equation:

$$\tau_\psi = b\left(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2\right) = bc_m\left(v_1^2 - v_2^2 + v_3^2 - v_4^2\right)$$

where b is the propellers drag coefficient.

The roll, pitch and yaw rates are related to the components of the angular velocity vector by means of the following expression:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \boldsymbol{Q}\boldsymbol{\omega} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{4}$$

where $\boldsymbol{Q}$ is a projection matrix.

Finally, from equations (1), (3), and (4) we can write down the the entire model of the quadcopter in a state space form as follows:

$$\dot{x} = v_x \tag{5}$$
$$\dot{y} = v_y \tag{6}$$
$$\dot{z} = v_z \tag{7}$$
$$\dot{v}_x = -\frac{k_d}{m}v_x + \frac{kc_m}{m}\Big(\sin\psi\sin\phi + \cos\psi\cos\phi\sin\theta\Big)\Big(v_1^2 + v_2^2 + v_3^2 + v_4^2\Big) \tag{8}$$
$$\dot{v}_y = -\frac{k_d}{m}v_y + \frac{kc_m}{m}\Big(\cos\phi\sin\psi\sin\theta - \cos\psi\sin\phi\Big)\Big(v_1^2 + v_2^2 + v_3^2 + v_4^2\Big) \tag{9}$$
$$\dot{v}_z = -\frac{k_d}{m}v_z - g + \frac{kc_m}{m}\Big(\cos\theta\cos\phi\Big)\Big(v_1^2 + v_2^2 + v_3^2 + v_4^2\Big) \tag{10}$$
$$\dot{\phi} = \omega_x + \omega_y(\sin\phi\tan\theta) + \omega_z(\cos\phi\tan\theta) \tag{11}$$
$$\dot{\theta} = \omega_y\cos\phi - \omega_z\sin\phi \tag{12}$$
$$\dot{\psi} = \frac{\sin\phi}{\cos\theta}\omega_y + \frac{\cos\phi}{\cos\theta}\omega_z \tag{13}$$
$$\dot{\omega}_x = \frac{Lkc_m}{I_{xx}}\Big(v_1^2 - v_3^2\Big) - \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right)\omega_y\omega_z \tag{14}$$
$$\dot{\omega}_y = \frac{Lkc_m}{I_{yy}}\Big(v_2^2 - v_4^2\Big) - \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right)\omega_x\omega_z \tag{15}$$
$$\dot{\omega}_z = \frac{bc_m}{I_{zz}}\Big(v_1^2 - v_2^2 + v_3^2 - v_4^2\Big) - \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right)\omega_x\omega_y. \tag{16}$$

Notice that in this model we have not considered the ground effect. This means that in principle we can have negative values for the z coordinate of the quadcopter. If we

Table 1: Parameters of the Quadcopter model

Parameter	Symbol	Value	Unit
Mass of the quadcopter	m	0.5	kg
Radius of the quadcopter	L	0.25	m
Propeller lift coefficient	k	$3 \cdot 10^{-6}$	$\mathrm{N\,s^2}$
Propeller drag coefficient	b	$1 \cdot 10^{-7}$	$\mathrm{N\,m\,s^2}$
Acceleration of gravity	g	9.81	$\mathrm{m/s^2}$
Air friction coefficient	k_d	0.25	kg/s
Quadcopter inertia about the x^b-axis	I_{xx}	$5 \cdot 10^{-3}$	$\mathrm{kg\ m^2}$
Quadcopter inertia about the y^b-axis	I_{yy}	$5 \cdot 10^{-3}$	$\mathrm{kg\ m^2}$
Quadcopter inertia about the z^b-axis	I_{zz}	$1 \cdot 10^{-2}$	$\mathrm{kg\ m^2}$
Motor constant	c_m	$1 \cdot 10^4$	$\mathrm{v^{-2}s^{-2}}$

denote the state vector of the system by $\boldsymbol{x} \in \mathbb{R}^{12} = [x\ y\ z\ v_x\ v_y\ v_z\ \phi\ \theta\ \psi\ \omega_x\ \omega_y\ \omega_z]^T$, and we define the input and output vectors as $\boldsymbol{u} \in \mathbb{R}^4 = [v_1^2\ v_2^2\ v_3^2\ v_4^2]^T$ and $\boldsymbol{y} \in \mathbb{R}^6 = [x\ y\ z\ \phi\ \theta\ \psi\ ]^T$ respectively, we can write down the model of the quacopter in a more compact way,

$$\dot{\boldsymbol{x}} = \boldsymbol{\xi}(\boldsymbol{x}, \boldsymbol{u}) \tag{17}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \tag{18}$$

where $\boldsymbol{C} \in \mathbb{R}^{6\times 12}$ is the output matrix and $\boldsymbol{\xi}$ is a nonlinear vector function defined by (5)-(16). Notice that the the input vector is in terms of the **squared voltages** of the rotors, and therefore **your control system should compute** v_1^2, v_2^2 v_3^2 and v_4^2 instead of v_1, v_2 v_3 and v_4. The maximum voltage that can be applied to the motors is 10 v. To make sure that this constraint is not violated, the circuitry within the quadcopter incorporates a clipping mechanism. Observe that negative voltages would change the spinning direction of the motors and therefore would invalid the model that has been derived. Based on the previous considerations, it is clear that the input constraints of the system are given by

$$0\ \mathrm{v}^2 \leq u_1, u_2, u_3, u_4 \leq 100\ \mathrm{v}^2 \tag{19}$$

where $u_1 = v_1^2$, $u_2 = v_2^2$, $u_3 = v_3^2$ and $u_4 = v_4^2$. In the present setup, the attitude (ϕ, θ, ψ) of the quadcopter is computed from the readings of a gyroscope and an accelerometer installed in the aircraft. The position $(x,\ y,\ z)$ of the vehicle is determined by a set of cameras installed within the room.

3 Available tools

In order to simplify the control design process, you have at your disposal the following tools:

- A *.mat file `references_xx.mat` (where `xx` corresponds to your group number) that contains the sequence of references for the x, y, and z coordinates of the quadcopter (*references*), the maximum simulation time ($Tmax$) and the initial state of the system (*x0_quadcopter*). You have to load this file first before performing any simulation with Simulink. You can always check which variables are in the workspace by typing the Matlab command `whos`.

- A Simulink file `template_quadcopter.slx` containing the nonlinear model of the quadcopter and a block that imports the reference trajectory from the workspace of Matlab. The input of the quadcopter block is a vector containing the squared voltages of the rotors (denoted by `u(t)`) and the outputs of this block are the entire state vector (denoted by `x(t)`) and the measured variables (denoted by `y(t)`), to which some measurement noise has been added.

- A protected Matlab function `generate_report.p` for visualizing and evaluating the simulation results after a Simulink run. It plots the trajectory followed by the quadcopter (blue line) within the room as well as the checkpoints (red circles), the evolution in time of the inputs or control actions (with indication of the input limits), the evolution of the angles ψ, θ and ψ of the quadcopter, and the evolution of the x, y and z coordinates together with their setpoints and maximum allowed deviations (0.08 m). This function also gives a simulation report indicating the number of checkpoints reached, and the time that the aircraft has spent traveling from one checkpoint to the next one. The quadcopter has a time limit of 7 seconds to reach each checkpoint. For simulations where it is assumed that the entire state vector is available (no observer), you should type in the command window `generate_report(0)`. For the case where the state vector is estimated by an observer, you should type `generate_report(1)`.

Remarks:

- Do not bother about the file `quadcopter_sfunction.p`. It is used for implementing the nonlinear model of the quadcopter in Simulink. Additionally, the file `icon_quad.png` provides an image mask for the Simulink block representing the aircraft.

- Make sure that the folder where you downloaded all the files is part of the Matlab path (or working directory).

4 Assignment

4.1 Linearization

In order to design an LQR or an LQG controller, it is necessary to have a linear approximation of the nonlinear model around an operating point. In this exercise such operating point, which is also an equilibrium point, is the one of a stationary flight (the quadcopter is in hover).

- Determine the linearization/equilibrium point of the vehicle for $\phi = \theta = \psi = 0$ and $x = y = z = 0$.

- Linearize the nonlinear model (5)-(16) about the equilibrium point. Do not forget to provide both the symbolic expression and the numerical values of the matrices describing the resulting linear model,

$$\Delta\dot{\boldsymbol{x}} = \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}\Delta\boldsymbol{u}$$
$$\Delta\boldsymbol{y} = \boldsymbol{C}\Delta\boldsymbol{x}$$

  where $\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}^*$, $\Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}^*$ and $\Delta\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{y}^*$ are the deviation variables and $\boldsymbol{x}^*$, $\boldsymbol{u}^*$ and $\boldsymbol{y}^*$ are the values of $\boldsymbol{x}$, $\boldsymbol{u}$ and $\boldsymbol{y}$ at the operating point.

Note: It is always a good idea to verify how good your linear model is in approximating the nonlinear model. To this end you can apply a small step to the inputs of the models and compare their respective outputs. We suggest you to apply the following step signal:

$$\Delta u_1 = \Delta u_2 = \Delta u_3 = \Delta u_4 = \begin{cases} t < 1, & 0 \\ t \geq 1, & 5 \end{cases}$$

and simulate both models for 5 seconds. Do not forget to perform the proper conversions between the deviation variables ($\Delta\boldsymbol{x}$, $\Delta\boldsymbol{u}$, $\Delta\boldsymbol{y}$) and the original variables ($\boldsymbol{x}$, $\boldsymbol{u}$, $\boldsymbol{y}$) at the moment of using the models. The previous test also serves to check whether you make a mistake or not when you derived the linear model.

4.2 Discretization

- Choose a discretization rule (e.g., zero-order hold, Euler's rule, bilinear transformation - also called tustin's rule, etc.) and find a discrete-time state-space model for the quadcopter. Use a sampling time of $T_s = 0.05$ s. Provide the numerical values of the matrices describing the discrete-time model.

- Is the discrete-time model stable?, what are the poles?, is it controllable?, is it observable?, is it stabilisable?, is it detectable?, is it minimal?, are there transmission zeros?

Remark: The discrete-time model derived here, is the model that you have to use for designing the control schemes of the following subsections.

4.3 LQR control

The control goal is to drive the quadcopter to several checkpoints as fast as possible (tracking problem). Furthermore, the control system should be capable of driving the aircraft to the checkpoints when carrying a payload of 0.1 kg. A checkpoint is declared reached when the quadcopter enters a sphere around it. The radius of this sphere, which is 0.08 m, is the maximum admitted error. The quadcopter is confined to a room that is 6 meters wide, 6 meters long, and 3 meters high. **In this section it is assumed that the entire state-vector is available**. So, use the proper output of the quadcoper block in Simulink. Do not forget that for generating a simulation report, you have to type in the command window `generate_report(0)` after a Simulink simulation run.

- Experiments **without payload**. Initially make sure that the payload parameter of the quadcopter block is set to **0 kg** (to access the parameters window double-click the quadcopter block).
  - Implement a full-state feedback controller ($\boldsymbol{N}_x$, $\boldsymbol{N}_u$, $\boldsymbol{K}$) based on an LQR kind of feedback.
  - Implement an LQR controller with integral action. Keep in mind that we are interested in tracking only the x, y and z coordinates of the vehicle. So, set the number of integrators accordingly. Verify the controllability of the augmented system.
  - Briefly describe the design process, indicating how you obtained the weighting matrices for both control schemes.
  - Discuss the simulation results. Which control strategy performs better?
- Experiments **with payload**. Adding a payload or cargo to the quadcopter can be seen as a disturbance that affects the dynamics of the aircraft. Now, make sure that the payload parameter of the quadcopter block is set to **0.1 kg**.
  - Keep the weighting matrices previously found, and simulate again both control systems.
  - What is the effect of the payload on the performance of the controllers? which control system is more robust? Discuss. You can change the tuning of your controllers if they become unstable.
  - What are the fundamental differences between the full-state feedback controller and the integral controller?

Remark: It is extremely important to set the "sample time property" of the Simulink blocks (gains, discrete-time blocks, etc.) that are part of your controller to T_s.

4.4 LQG control

For this section you have to use the LQR controller with integral action that you designed in Section 4.3. Additionally, **the assumption that the entire state-vector is available is NOT valid anymore**. Therefore your control system only has at its disposal the measurements given by the sensors. Do not forget that for generating a simulation report, you have to type in the command window `generate_report(1)` after a Simulink simulation run.

- Design and implement a Kalman Filter for estimating the entire state-vector. The the variance of the measurement noise for x, y, z is $2.5 \cdot 10^{-5}$ and for ϕ, θ, ψ is $7.57 \cdot 10^{-5}$. Briefly discuss how you fixed the covariance matrix of the process noise and the covariance matrix of the measurement noise. **Note:** In the discrete-time stochastic model of the system (see page 174 of the course notes), assume $\boldsymbol{B}_1 = \boldsymbol{I} \in \mathbb{R}^{12\times 12}$, the identity matrix.
- Carry out a simulation with a payload of 0 kg. What is the effect of adding a Kalman filter on the overall performance of the control system? Discuss the results.

- Perform a simulation with a payload of 0.1 kg, and compare the results with the ones obtained in the previous section. Discuss.

5 Deliverables

We expect from you:

- A written report (a hard copy and a digital copy).
- All the Simulink and Matlab files that allow us to reproduce your results. Also provide a document where you briefly explain the purpose of each of them.

Remarks about the report: The report should be written preferably **in English**, and it should address every single point of Section 4. Your report should describe the technical aspects of the designed controllers, and clearly it should state which design specifications were met and which were not (and why they could not be achieved).

Not only the technical correctness of your report is evaluated but also the quality and presentation. So, make sure that everything is clear and well explained. For example, make sure that you label every axis of every plot, that your figures include legends if necessary, that in the text you discuss or address what is presented in every figure of the manuscript, that the text is free of typos and well redacted, etc. Do not forget to include the Simulink diagrams that you have created. If a Simulink diagram contains subsystems, you should also present the contents of these subsystems (excluding the Simulink blocks that were given to you). In addition, your report must be self-contained. Do not refer the reader to your Matlab files. Keep in mind that we only ask you for your Matlab files for verification purposes.

6 Examination

At the end of the semester you will have an oral exam, where you will have to defend your design choices, explain technical details, answer some general questions, etc. Further information (dates of examination, how the points are distributed between exercises and practicum, etc.) can be found in the webpage of the Computergestuurde Regeltechniek course in Toledo.