FRTN65 Modeling and Learning from Data

Laboratory Exercise 3

Learning Quadcopter Flight Dynamics

Department of Automatic Control LTH
Lund University
Fall 2024

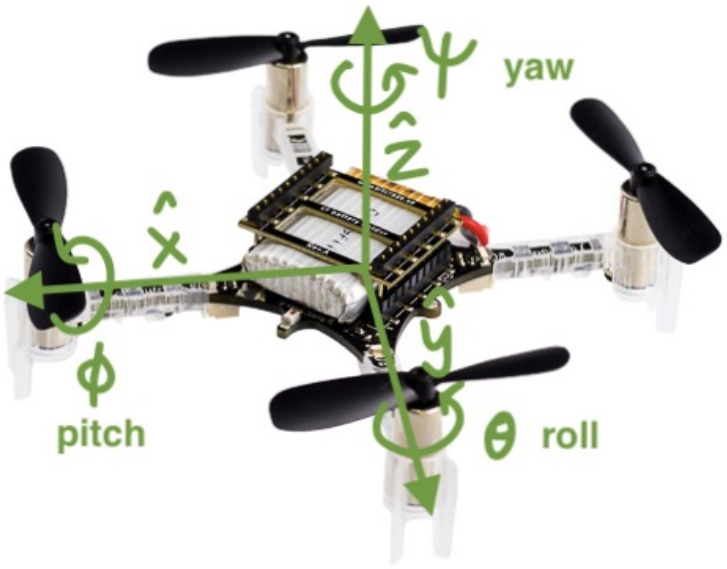

Figure 1 Illustration of the coordinate frame for the quadcopter and the definition of the roll pitch and yaw angles.

Hand in a brief report in PDF format with results and short discussions for each task. Also hand in your Matlab code. You may ask for help and cooperate on the laboration, but the handin should be done individually. (Your are e.g. not allowed to copy material from another persons' report).

1. Introduction

This lab will be carried out using the system identification toolbox in Matlab. Most of the code has been given to you, and you only need to make minor edits.

In the first part of this lab we will estimate the parameters of a quadcopter using open loop experiments. In the second part we will use the estimated parameters to close the loop and study the quadcopter subject to an external disturbance. By estimating a model for this disturbance, we will attempt to synthesize a controller that handles it better than what a controller without the model.

The focus of this lab will be on the estimation, and you can finish it without doing any controller synthesis yourself. However, you are more than welcome to see if you can improve the control design.

The simulations will be carried out in Simulink. You do not need to inspect the simulink models unless you want to. You can access the outputs from the simulink model from the `out` variable in Matlab.

IMPORTANT: The simulink models are designed using Matlab 2020a, and there are some known issues for too old Matlab versions. If you have any problems, we recommend that you upgrade to a more recent Matlab version.

2. Quadcopter modeling

To measure the position of a quadcopter we need three variables x, y, z [m] to describe its position in space , and three variables ϕ, θ, ψ [rad] to describe its orientation in space . In this lab we will use Tait-Bryan angles to describe the orientation of the quadcopter. We also need to describe the angular speed of the four rotors, $\mathbf{\Omega} = [\Omega_1, \Omega_2, \Omega_3, \Omega_4]$ [rad/s].

2.1 Tait-Bryan Angles

The three angles (ϕ, θ, ψ) describe the rotation around the x, y, and z axes respectively.

ϕ is the rotation around the x axis, often called the pitch. θ is the rotation around the y axis, often called the roll. Finally, ψ is the rotation around the z axis, often called the yaw. For an illustration see Fig. 1.

Later in this lab we will try to make the quadcopter move in the y direction. Note that this requires a (negative) rotation about the x axis.

2.2 Coordinate Frames

To describe the dynamics of the quadcopter it is often useful to introduce a coordinate frame for the quadcopter, $\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}$, see Figure 1 for an illustration. The mapping between this coordinate frame and the global coordinate frame is then described by the Tait-Bryan angles. We will throughout this lab only use small angles, which allows us to use only the global coordinate frame.

2.3 Dynamics

From blade element theory, each rotor i with rotor speed Ω_i generates a force

$$f_i \approx k_i \Omega_i^2 \tag{1}$$

in the positive $\hat{\mathbf{z}}$ direction. It also generates a torque

$$\tau_i \approx b_i \Omega_i^2 + I_M \dot{\Omega}_i,$$

around the $\hat{\mathbf{z}}$ axis. However, the effect of the acceleration term $I_M \dot{\Omega}_i$ is typically negligible and we ignore it in this lab. The system is symmetric in the $\hat{\mathbf{x}}$ $\hat{\mathbf{y}}$ plane, where each motor is l meter from the center of gravity. We assume that all rotors have the same characteristics, i.e. $k_i = k$ and $b_i = b$ $\forall i$.

We will use the plus configuration, which means that the $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ is chosen as in Figure 1. The resulting thrust force T and torque τ in the body coordinate system is given by

$$\hat{\mathbf{T}} = \begin{bmatrix} 0 \\ 0 \\ k\sum_{i=1}^{4}\Omega_i^2 \end{bmatrix}, \quad \hat{\tau}_{\mathcal{B}}^{+} = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} kl(-\Omega_2^2 + \Omega_4^2) \\ kl(-\Omega_1^2 + \Omega_3^2) \\ b(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{bmatrix}$$

The inertia in the body coordinate system is given by

$$\hat{\mathbf{I}} = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{bmatrix}$$

To simplify the modeling we will assume that only one of the three angles (ϕ, θ, ψ) is non zero, and attempt to control it in one single positional dimension

3. Estimating k

We start be estimating k. The mass of the quadcopter can easily be measured by weighing it. If all the rotors have the same rotor speeds, the torque on the quadcopter will be zero. The dynamics of the quadcopter can then be simplified significantly, and modeled as a double integrator, using Newtons second law,

$$m\ddot{z} = F = 4 \cdot k\Omega^2 - mg,$$

where $\Omega_i = \Omega$ for all rotors.

Throughout this lab we assume that we have access to the acceleration and the position of the quadcopter. (Knowing the position typically requires a camera system).

By letting $u = 4\Omega^2$ we can rewrite the expression as

$$\underbrace{\ddot{z} + g}_{\zeta} = \frac{k}{m}u$$

We can calculate ζ by measuring $\ddot{z}$ and knowing the gravity constant g. We now want to identify the process $\zeta = \frac{k}{m}u$. This gives an estimate on the form $\zeta = K_p u$, from which we can find our estimate of k as $k_{est} = K_p \cdot m$.

Task 1: Open the file `identify_k.m` and construct a suitable input trajectory $\Omega[t]$ that can be used to verify (1) by comparing the three models

$$f_i = k\Omega$$
$$f_i = k\Omega^2$$
$$f_i = k\Omega^3$$

and estimate k for the case of $f_i = k_i\Omega^2$. Compare it to the true value k, which can be found in the matlab workspace.

Hints:

- You can run a segment of the code at a time by selecting it and pressing ctrl+enter or cmd+enter depending on your OS.
- The parts that you are supposed to write yourself has been indicated by a `%TODO` comment.

4. Estimating I_3 and b

By setting $\Omega_1 = \Omega_3$ and $\Omega_2 = \Omega_4$, the resulting torque profile will be such that there is only a torque around the z-axis. The equation of motion for the yaw angle ψ is given by

$$\tau_\psi = I_3 \ddot{\psi},$$

where the torque τ_ψ is given by

$$\tau_\psi = b_i(-2\Omega_1^2 + 2\Omega_2^2).$$

With a gyro we can measure the yaw rate $\dot{\psi}$. Let $u = (-2\Omega_1^2 + 2\Omega_2^2)$ then the dynamics for $\dot{\psi}$ are given by

$$\frac{d}{dt}\dot{\psi} = \frac{b}{I_3}u,$$

which is just an integrator.

Task 2: Open the file `identify_b.m` and construct an input trajectory for $\Omega[t]$ so that the quadcopter has constant z position and changes yaw angle. The script switches between two different inputs, where Ω_1 and Ω_2 switches values. The number of switches depends on the segments parameter.

Let $\Omega_2 = c\Omega_1$. For a given value of c find the value of Ω_1 so that the quadcopter hovers,

$$mg = k(2\Omega_1^2 + 2\Omega_2^2)$$

Then chose a suitable value of c and the number of segments so that the yaw angle stays in the interval $(-\pi, \pi)$.

Why can this experiment not be used to find both b and I_3?

Assume that I_3 has been estimated by some other method. Estimate b by choosing a suitable process model.

5. Estimating I_1 and I_2

Due to the symmetry of the quadcopter it holds that $I_1 = I_2$ and it is enough to find one of them.

The equation of motion for the ϕ angle is given by

$$\tau_\phi = I_1 \ddot{\phi}$$

where for small angles, the torque τ_ϕ is given by

$$\tau_\phi = kl(-\Omega_2^2 + \Omega_4^2).$$

It is also important that ϕ stays quite small so that the quadcopter is stable.

If $\Omega_1 = \Omega_3$ and $-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2 = 0$ there will only be a torque on ϕ.

Let $u = (-\Omega_2^2 + \Omega_4^2)$, then $\dot{\phi}$ will satisfy

$$\frac{d}{dt}\dot{\phi} = \frac{kl}{I_1}u$$

which again is just an integrator.

Task 3: Open the file `identify_I1.m` and design an input sequence $\mathbf{\Omega}$ for estimating I_1. Start by finding Ω_H such that if all $\Omega_i = \Omega_H$ then the quadcopter hovers. Let $\Omega_1 = \Omega_3 = \Omega_H$ and $\Omega_4 = c\Omega_2$. Then it must hold that $\Omega_2^2(1+c^2) = 2\Omega_H^2$ for the quadcopter to hover. For any c find Ω_2. Then choose a suitable c and number of switches. Finally construct a suitable `iddata` object and choose a suitable model structure.

Hint: You can look at the code for the previous task if you need inspiration.

6. Closing the Loop

With the parameters estimated it is now easy to derive controllers to control the quadcopter. To keep it simple we will only control the quadcopter in two dimensions. Thus we only need to consider one rotation, in this case we start by designing one controller for the height z of the quadcopter and one for control of the y position. This require us to control the pitch angle ϕ

For the z controller we have the dynamics

$$m\ddot{z} = -mg + \cos(\phi) \cdot (T) \approx (T - mg)$$

for small angles ϕ. We design a controller that gives a thrust reference ΔT, which is the thrust relative to the thrust that keeps the quadcopter hovering. This gives the controller structure in Figure 2.

The y controller is more advanced. The dynamics, assuming constant thrust $T = mg$, are given by

$$m\ddot{y} = -mg\sin\phi \approx -mg\phi$$

for small angles. The dynamics for ϕ are, as already mentioned,

$$I_1\ddot{\phi} = \tau_\phi.$$

We start by controlling the y position using a cascaded control structure, where an outer PID controller takes the position error and gives an angle reference ϕ_{ref}. The inner controller is a PD controller which takes the angle error ϕ_e and gives a torque reference τ_ϕ. See Figure 3 for an illustration. The wind force present in the figure will be considered in the next section. For a description on how to map the wanted thrust T and torque τ_ϕ, see the appendix.

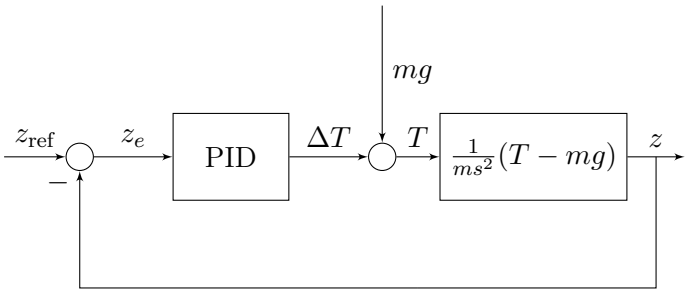

Figure 2 Illustration of the control structure for the z controller

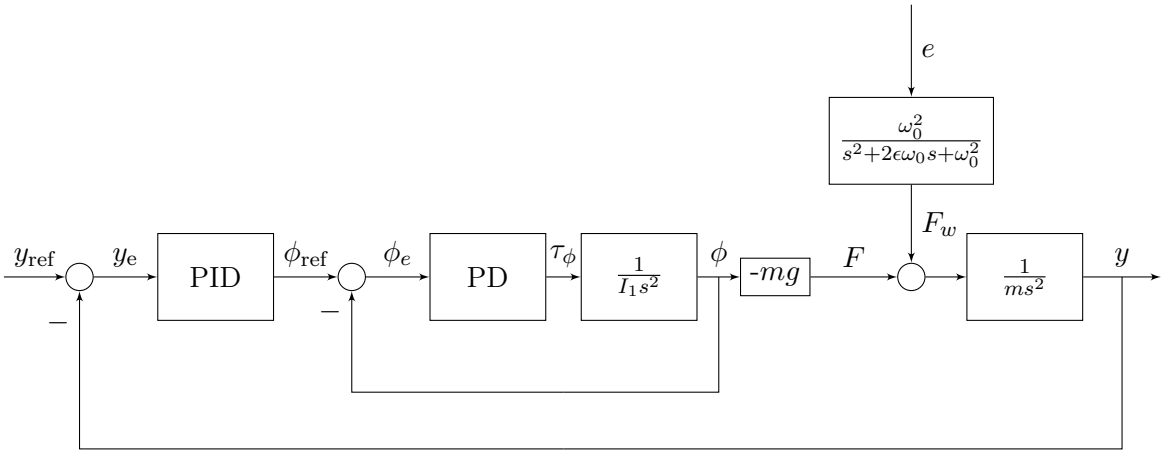

Figure 3 Illustration of the control structure for the y controller. There is a wind disturbance driven by white noise e, which we will try to estimate.

Task 4: Open the file `test_pid.m` and evaluate it. Make sure that the controllers work. The goal is to move 0.1m in the y direction while maintaining the z level. You can edit the file `design_pid.m` try your own design if you want to.

Note that this simulates the system without the wind disturbance.

7. Estimating a Disturbance Model

We now consider the case when there is a wind acting on the quadcopter, resulting in a force in the y direction, as depicted in Figure 3. The goal is to make the quadcopter stay in the origin. We measure the acceleration $\ddot{y}$ which satisfies $m\ddot{y} = F$ where the total force is given by $F = F_w - mg\sin\phi \approx F_w - mg\phi$. As we can measure ϕ we can thus extract the resulting wind force F_w.

Task 5: Estimate an ARMA model for the wind disturbance using the accelerometer to directly calculate the the wind force. Most of the code you need can be found in `identify_wind.m`. Try different values for the decimation factor R and enter the correct model structure for the ARMA estimation.

Compare your results to the true denominator which can be found in `wind.denum` and also using the code supplied for plotting the bode diagram. For the control design in the next step it is only necessary that the estimation of the denominator is good.

At the end of the script you can plot the position of the quadcopter.

8. Using the Disturbance Model in a control design

We will now design an LQG controller and a Kalman filter with a noise model to better handle the disturbance. It is OK if you have never heard about these things before, all you have to do is insert your estimation of the disturbance model and inspect the result!

We can model the dynamics of the y position, including the external disturbance as follows. Introduce the state ξ, where $\xi_1 = \dot{\phi}, \xi_2 = \phi, \xi_3 = \dot{y}, \xi_4 = y$ and ξ_5 and ξ_6 are states for describing the external disturbance. The dynamics for ξ is then given by

$$\begin{aligned}
\dot{\xi_1} &= \tau_\phi / I_1 \\
\dot{\xi_2} &= \xi_1 \\
\dot{\xi_3} &= -\xi_2 g + \xi_6 / m \\
\dot{\xi_4} &= \xi_3 \\
\dot{\xi_5} &= -2\omega_0 \epsilon \xi_5 - \omega_0^2 \xi_6 + e(t)\omega_0^2 \\
\dot{\xi_6} &= \xi_5
\end{aligned}$$

Task 6: Insert your values for the disturbance parameters ω_0 and ϵ into `design_LQG.m`, run the control synthesis and run the simulation to verify that the controller improves the performance. Note that the figure axis might differ from when you only used PID controllers.

Appendix

We consider the problem to find Ω_i given references ΔT and τ_ϕ. We start by finding the Ω_T so that if all $\Omega_i = \Omega_T$ then the wanted thrust is achieved. This gives

$$4k\Omega_T^2 = mg + \Delta T \Rightarrow \Omega_T = \sqrt{\frac{mg + \Delta T}{4k}}$$

Now let $\Omega_1 = \Omega_3 = \Omega_T$. Note that Ω_2 and Ω_4 must satisfy

$$\begin{aligned}
kl(-\Omega_2^2 + \Omega_4^2) &= \tau_\phi \\
\Omega_2^2 + \Omega_4^2 &= 2\Omega_T^2
\end{aligned}$$

This is a linear system of equations in Ω_2^2 and Ω_4^2 which can always be solved.