TÉCNICO LISBOA

Unmanned Aerial Vehicles

M.Sc. in Aerospace Engineering

2022/2023 - First Semester

Motion Control of the Parrot AR.Drone

Laboratory Guide

December 2022

1 Introduction

1.1 Objectives

This laboratory assignment addresses the design, implementation, and analysis of several solutions to the problem of quadrotor motion control. Simulations and experiments will be conducted using the Parrot AR.Drone DevKit to accomplish the following objectives:

1. Design a trajectory tracking controller for a generic quadrotor.
2. Incorporate a adaptation law in the controller to account for unknown disturbances.
3. Implement, using the AR.Drone DevKit, a modified version of the designed controller.
4. Implement, using the AR.Drone DevKit, a simple trajectory planner for the definition of desired trajectories taking the form of straight lines and circles.

1.2 Organization and timeline

There are two kinds of questions: theoretical questions, marked as (T), and laboratory questions, marked as (L). As a guideline, all theoretical questions should be solved before the laboratory sessions, and the simulations should be completed during the sessions. The laboratory component will resort to a matlab software package provided together with this handout.

A single-column report in pdf format with no more than 15 pages, together with the matlab code developed to answer the questions, must be submitted through fenix in the designated dates (check the course's website). Please use the cover page available in the course's website (or similar with the same information) as front page.

1.3 Academic ethics code

All members of the academic community of the University of Lisbon (faculty, researchers, staff members, students, and visitors) are required to uphold high ethical standards. Hence, the report submitted by each group of students must be original and correspond to their actual work.

2 Trajectory Tracking Control

2.1. **(T)** Using the same notation as in previous laboratory handouts, the linear motion of a quadrotor expressed in the inertial reference frame can be written as

$$\ddot{\mathbf{p}} = g\mathbf{e}_3 - R\frac{T}{m}\mathbf{e}_3. \tag{1}$$

Recalling that $R = R_z(\psi)R_y(\theta)R_x(\phi)$, apply a change of coordinates to obtain

$$\ddot{\mathbf{p}} = g\mathbf{e}_3 - R_z(\psi)\mathbf{u}^*, \tag{2}$$

where $\mathbf{u}^*$ denotes a virtual input that will determine the thrust T and references for the roll and pitch angles ϕ_r and θ_r, respectively. Write T, θ_r, and ϕ_r as functions of $\mathbf{u}^*$.

From now on, it is assumed that an inner-loop controller drives $(\phi,\ \theta)$ to $(\phi_r,\ \theta_r)$ and that, for design purposes, the time-scale separation between the outer-loop position dynamics and inner-loop orientation dynamics is sufficiently large to neglect the coupling between the two.

2.2. **(T)** Consider the system described by (2) and let $\mathbf{p}_d(t) \in \mathbb{R}^3$ be a desired trajectory, with known continuous first and second time derivatives denoted by $\dot{\mathbf{p}}_d(t) \in \mathbb{R}^3$ and $\ddot{\mathbf{p}}_d(t) \in \mathbb{R}^3$, respectively. To design a trajectory tracking controller, define the error variables $\tilde{\mathbf{p}} = \mathbf{p} - \mathbf{p}_d$ and $\tilde{\mathbf{v}} = \dot{\mathbf{p}} - \dot{\mathbf{p}}_d$. Define the error system with state

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{\mathbf{p}} \\ \tilde{\mathbf{v}} \end{bmatrix}$$

and write the error dynamics in the form

$$\dot{\tilde{\mathbf{x}}} = A\tilde{\mathbf{x}} + B\mathbf{u}$$

where $\mathbf{u}$ is a new control variable. Write the explicit expression of the input transformation from $\mathbf{u}$ to $\mathbf{u}^*$. In addition, provide the expression for the state-feedback control law $\mathbf{u} = -K\tilde{\mathbf{x}}$ that minimizes the cost function

$$J := \int_0^{+\infty} \left(\tilde{\mathbf{x}}^T(t) Q \tilde{\mathbf{x}}(t) + \rho \left\| \mathbf{u}(t) \right\|^2 \right) dt,$$

where $\rho > 0$ and Q is a positive definite matrix.

2.3. **(T)** Consider the closed-loop system $\dot{\tilde{\mathbf{x}}} = (A - BK)\tilde{\mathbf{x}}$, where $K = \frac{1}{\rho}B'P$ and $P > 0$ is the solution of the Riccati equation $A'P + PA - \frac{1}{\rho}PBB'P + Q = 0$. Show that the time derivative of the Lyapunov function

$$V_1 = \tilde{\mathbf{x}}^T P \tilde{\mathbf{x}}$$

along the solutions of the closed-loop system is given by

$$\dot{V}_1 = -\tilde{\mathbf{x}}^T Q^* \tilde{\mathbf{x}},$$

where Q^* is a positive definite matrix. Draw conclusions about the stability of the equilibrium point $\tilde{\mathbf{x}} = \mathbf{0}$.

2.4. **(T)** Suppose now that the quadrotor linear dynamics is described by

$$\ddot{\mathbf{p}} = g\mathbf{e}_3 - R_z(\psi)\mathbf{u}^* + \mathbf{w}, \tag{3}$$

where $\mathbf{w} \in \mathbb{R}^3$ is an unknown disturbance (possibly a wind disturbance) and that an estimate $\hat{\mathbf{w}}$ is used in the outer-loop control law $\mathbf{u}^*$ such that

$$\mathbf{u}^* = R_z(-\psi)(K\tilde{\mathbf{x}} + g\mathbf{e}_3 + \hat{\mathbf{w}} - \ddot{\mathbf{p}}_d). \tag{4}$$

Using (4) and defining the estimation error $\tilde{\mathbf{w}} = \mathbf{w} - \hat{\mathbf{w}}$, show that the new closed-loop error dynamics is given by

$$\dot{\tilde{\mathbf{x}}} = (A - BK)\,\tilde{\mathbf{x}} + B\tilde{\mathbf{w}}.$$

2.5. **(T)** Assuming that $\mathbf{w}$ is constant, consider the problem of designing an adaptive controller that simultaneously estimates $\mathbf{w}$ and guarantees convergence to the desired trajectory. For that purpose, consider the Lyapunov function

$$V_2 = V_1 + \frac{1}{k_w}\|\tilde{\mathbf{w}}\|^2,$$

where $k_w > 0$ is a tuning gain. Propose an adaptation law for $\dot{\hat{\mathbf{w}}}$ that guarantees that $\dot{V}_2$ is negative semi-definite. Note: recall that $\mathbf{w}$ is constant but unknown.

2.6. **(T)** Show that the origin of the system with augmented state $\boldsymbol{\xi} = \begin{bmatrix}\tilde{\mathbf{x}}^T & \tilde{\mathbf{w}}^T\end{bmatrix}^T$ is globally asymptotically stable.

2.7. **(T)** Show that the resulting closed-loop system can be represented in the form of the block diagram shown in Fig. 1. Write the expression for $\bar{K}(s)$ and discuss the connection between the proposed controller and a PID controller.

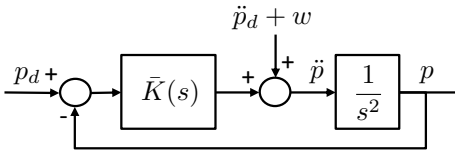

Figure 1: Closed-loop system for trajectory tracking.

2.8. **(T)** The proposed trajectory tracking controller has been designed independently from the yaw angle dynamics, highlighting the fact that, for quadrotor vehicles, the yaw angle is not constrained by the linear dynamics. Assume that the yaw motion is simply described by a first order system, not necessarily a pure integrator. Design a feedback controller that guarantees tracking of a constant reference with zero steady-state error.

3 Implementation and simulation using the AR.Drone DevKit

An updated version of the DevKit is provided for this laboratory. The following changes were introduced:

- The starting script is now called `start_here_CTRL.m`. Under the options `(1) Simulation` and `(2) Wifi-Control`, a new suboption called `(4) Trajectory Tracking` is available, which opens `ARDroneTTSim.slx` (for simulation) or `ARDroneTT.slx` (for wifi-control).
- Compared with previous versions, the model contains two new blocks: `desired_trajectory` and `tt_controller` (the latter one inside the subsystem `Outer-loop Controller`). The inputs and outputs of these blocks are given but the content is partially empty. To implement and simulate the trajectory tracking controller, both blocks need to be filled in the form of Matlab Functions.

3.1. **(L)** Recall that $\mathbf{u}^*$ defines a virtual input and the thrust input T and the references ϕ_r and θ_r can be written as functions of $\mathbf{u}^*$ to be used as inputs for the inner-loop controllers, together with ψ_r. In contrast to the model and inner-loop control structure described in Section 2, the model provided in the AR.Drone DevKit has as inputs: 1) the references ϕ_r and θ_r, 2) the vertical velocity reference w_r, and 3) the yaw rate reference $\dot{\psi}_r$.

To implement the controller and adapt it to the available inputs, keep the expressions for the roll angle, pitch angle, and yaw rate references. To simplify the design for the vertical channel, all the trajectories are assumed to belong to the horizontal plane, the roll and pitch angles are very small, and a simple proportional controller is implemented, as given by

$$w_r = -k_w \left(z - z_d\right).$$

3.2. **(L)** Focusing now on the generation of the desired trajectory, fill the block `desired_trajectory` such that $\mathbf{p}_d(t)$ defines a horizontal straight line trajectory (with a suitable constant height), to be described with a speed of 0.1 m/s and a constant yaw angle. The initial desired position should match the initial position of the quadrotor when the tracking is started. Do not forget that the first and second time derivates of $\mathbf{p}_d(t)$ are also required.

3.3. **(L)** Test the implemented controller and check whether or not tracking is accomplished. Tune the gains if necessary and comment on the achieved performance.

3.4. **(L)** Consider now the tracking of a circular trajectory on the horizontal plane. Make the necessary changes in `desired_trajectory` to generate a circular trajectory with a radius of 1 m, to be described with an angular rate of 0.1 rad/s and a constant yaw angle. Again, the initial desired position should match the initial position of the quadrotor when the tracking is started.

3.5. **(L)** Test the implemented controller with the new desired trajectory and check whether or not tracking is accomplished. Comment on the achieved performance.

3.6. **(L)** Considering once more the circular trajectory and in an attempt to obtain zero tracking error, change the desired yaw angle so that the quadrotor always points towards the center of the circumference. Simulate the system with this new trajectory and compare the results obtained for this case with those obtained in the previous question. Is the proposed change sufficient to achieve zero tracking error? If not, suggest a change in the controller to accomplish this objective.

3.7. **(L)** As mentioned earlier, to avoid large excursions in actuation, the block `desired_trajectory` assigns the current x and y position of the drone to the initial desired x and y positions whenever the reference commands are enabled. Alternative approaches can be adopted to avoid such large excursions. Consider the following one: apply the necessary changes to the block `desired_trajectory` to implement a line of sight scheme for straight line path-following, involving expressions for the desired yaw angle ψ_d and desired velocity components $\dot{x}_d$ and $\dot{y}_d$ given by

$$\psi_d = \arctan(-(y - y_d)/\Delta), \tag{5}$$
$$\dot{x}_d = V\cos(\psi_d), \tag{6}$$
$$\dot{y}_d = V\sin(\psi_d), \tag{7}$$

respectively. Choose adequate values for the scalars Δ and V and discuss the effect of these parameters on the convergence to the path.