AUVE Lab - Quadrotor control

Damien SIX
damien.six@ls2n.fr

Released: December 2, 2022

1 Objectives of the lab

The purpose of this lab is to perform a stabilized flight of a quadrotor using a cascaded control law. The quadrotor is simulated using GAZEBO.

The report for this lab have to be done using LaTeX. You already have been provided templates in the AMORO lab.

Report should detail you work on the control strategy and explain clearly your achievements (with curves, tables,...).

You also have to provide you uavcontrol.cpp file such that it can be compiled and tested by the examiner.

To start the lab clone the lab_auve folder in your ros2 workspace, similarly to what you did in the AMORO lab. You can build your workspace with the command "colcon build".

Once the workspace is built, you can check that everything is ready to go using "ros2 launch lab_auve uav_simulation.launch.py". Two launch files are provided for this lab.

- lab_auve uav_simulation.launch.py simulates the quadrotor.
- lab_auve uav_simulation_fixed.launch.py simulates the drone attached to a ball joint in its center. This simulation helps you to test your attitude control law in a first step. Keep in mind that this trick modifies quadrotor dynamics and so the gains that you may obtain with this simulation won't be valid for the unconstrained drone.

Once the simulation is launched, you can open the project with qtcreator. You can then run or debug the project directly from qtcreator. You should only modify the file "uavcontrol.cpp" for the purpose of this lab.

Gazebo must be launched for the project to run properly. Also, closing Gazebo ends the program.

In the context of this lab, the vectors and quaternions are represented using the Eigen library.
https://eigen.tuxfamily.org
https://eigen.tuxfamily.org/dox/classEigen_1_1Quaternion.html

2 Description of the system

The quadrotor is described by Fig. 1. The axis of the body frame are such that $\mathbf{x}_b$ is pointing forward and $\mathbf{z}_b$ up; $\mathbf{y}_b$ is then pointing left. The useful variables for

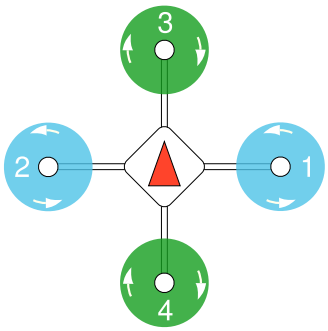

Figure 1: Scheme of the quadrotor (source: docs.px4.io)

this lab are the following (and recalled at the beginning of the "uavcontrol.cpp" file).

- Arm length : 0.17 m
- Rotor thrust gain: k_t = 5.5e-6 N per rpm^2
- Rotor drag gain: k_d = 3.299e-7 Nm per rpm^2
- Drone mass: 1 kg

3 Writing a control law for a quadorotor

In the context of this lab, we assume that you have a precise measure of the quadrotor configuration in 6D, and of the associated velocities. You can control the quadrotor via the rotation speed of each of the four motors (in rpm).

In you cpp implementation, you can use

- uav.getPosition() to get the 3D position of the drone
- uav.getorientation() to get the orientation of the drone as a unit quaternion.
  If you do not know what a quaternion is, please ask the professor!!!.
  You can also read https://danceswithcode.net/engineeringnotes/quaternions/quaternions.html
- uav.getLinearVelocity() or uav.getAngularVelocity() to respectively get the linear and the angular velocity of the drone.

In order to establish a control law, you must follow those three step:

- Write a mixer matrix to compute torques and thrust force of the quadrotor, given a motor rotation speed.
- Write an attitude controller, using a control law based on quaternions.
- Write a position controller.

At each step, please check carefully by all mean at you disposal that your solution works. Plot some curves and save them for your report! It is useless to go to the following step if you are not 100% sure that everything is working so far.

3.1 Mixer

To compute the mixer matrix you must use the motor thrust and drag gains in relation to the motor rotation speed. Those equations have been covered during the course but be careful, your system is not parameterized exactly in the same way.

Be smart and find a way to test you mixer matrix.

3.2 Attitude controller

In a similar way to the attitude control law for Euler angles, it is possible to use a PD controller to reach a desired orientation using the drone torques. The control law will however use the quaternion error to reach this goal.

Given a desired orientation $\mathbf{q}_d$ and a current orientation $\mathbf{q}$ the quaternion error is defined by

$$\tilde{\mathbf{q}} = \mathbf{q}^{-1}\mathbf{q}_d$$

Keep in mind that we are talking about quaternion multiplication and inverse.

Then, the following control law can be used to converge toward a desired orientation

$$\tau = -\mathbf{K}_d\mathbf{w} + \mathbf{K}_p \text{sign}(\tilde{\mathbf{q}}_r)\tilde{\mathbf{q}}_i$$

where

- τ is the drone input torque
- $\mathbf{w}$ its current angular velocity (body frame)
- $\tilde{\mathbf{q}}_r$ is the real scalar part of the quaternion error
- $\tilde{\mathbf{q}}_i$ is the imaginary part of the quaternion error, represented as a vector
- $\mathbf{K}_p$ and $\mathbf{K}_d$ are diagonal gain matrices

Your gains should be tuned manually. Do not spend to much time to tune the fixed model, use it only to check that your control law converges. Use the free model to further tune your gains (again, you have to think a way of how to do that).

3.3 Position controller

For the position control, we will use a simple PD control with compensation of gravity.

$$\mathbf{f} = \mathbf{K}_d(\mathbf{v}_d - \mathbf{v}) + \mathbf{K}_p(\mathbf{p}_d - \mathbf{p}) - m\mathbf{g} \tag{1}$$

where

- $\mathbf{f}$ is the desired force provided by the drone
- $\mathbf{p}_d$ and $\mathbf{p}$ are respectively the desired and current position of the drone
- $\mathbf{v}_d$ and $\mathbf{v}$ are respectively the desired and current velocity of the drone
- m is the mass of the drone and $\mathbf{g}$ the gravity vector

From the desired force vector $\mathbf{f}$ you can extract a desired thrust (sent to the mixer) and orientation (sent to the orientation controller).

Once your controller is working properly for a fixed point, you can go further with the tracking of a trajectory, or try to cancel the steady-state error, or provide some feed-forward to the control laws. Try things, and document them in your report!