AE483 Lab Manual: Week #2
Sensor Data Collection and Analysis

T. Bretl

September 9, 2019

1 Goal

Last week, you collected sensor data from IMU and mocap during a short manual flight. Your goal this week is to collect the data that will allow you to answer four questions in particular:

- At what rate does the on-board code run?

  Any controller you design will be implemented with on-board code, using the function `lab()` in `lab.c` of the SDK (Appendix A). This function is called at a fixed rate, some number of times per second. It is really important to know this rate. You need to estimate it in some way from your data.

- To what extent do IMU and mocap agree on orientation?

  Last week, you saw that the motion capture system measures the orientation of the drone in terms of yaw, pitch, and roll angles. It turns out that the on-board code also estimates the orientation, by filtering the IMU measurements. You will likely discover that the on-board code runs at a higher rate than the off-board code, so it would be nice to use the estimate of orientation from the IMU for control. In order to decide if this is a good idea, you need to verify in some way from your data that the IMU and mocap agree (or that they do not agree) on orientation.

- Does the IMU measure angular velocity or angular rates?

  As you know, the components of angular velocity are, in general, **not the same** as the angular rates (i.e., the time derivatives of yaw, pitch, and roll angles). The IMU measures one or the other—it is really important to be sure which one. You have seen ways in which the on-board and off-board code are ambiguous on this point, for example variously referring to the same measurement as `gyro_x` and `angvel_roll`, and so forth. You need to use your data in some way to justify a conclusion about what the IMU is measuring. Note that you can estimate the angular rates (for sure) with a finite-difference approximation, using the yaw, pitch, and roll measurements from the motion capture system.

- What causes mocap to lose track of the drone and what happens then?

  You may have noticed that the motion capture system can lose track of the drone. This is bad—it would likely cause any controller you implemented to crash. To convince yourself that this is possible, move the drone slowly away from the center of the room, and watch it disappear from the motion capture interface application. You need to use your data to identify and describe, in some way, the conditions under which mocap will **not** lose track of the drone and will continue to provide accurate measurements of position and orientation. (Are there places that the drone cannot go? Does it matter if people are between the drone and the cameras? Is mocap more likely to fail if the drone is flying quickly? Does the accuracy vary with position and orientation? These are all examples of questions you might ask.)

You will be responsible for writing a report that answers these four questions and for submitting this report (as a group) no later than 11:59PM on Friday, September 20, 2019. Details will be posted to Slack. You will do the following things in lab today to prepare for writing this report:

- Change what data are being logged (Section 2).
- Collect at least two sets of data, one while flying the drone and the other while moving the drone around in your hands (Section 3).
- Visualize your data in graphic simulation with MATLAB (Section 4).
- Choose a representative for the AE483 Congress (Section 5).

Section 6 has a summary of in-lab deliverables. **Make sure to save your flight data for later analysis.** It is highly recommended that you take video of all flights, in addition to logging data with the ground-station.

2 Change what data are logged

Last week, the ground-station logged the following things to a file:

- time,
- angular velocity and acceleration measurements from IMU,
- position and orientation (yaw, pitch roll) measurements from mocap.

Follow the instructions in Appendix D of the lab manual to log the following things instead:

- time,
- a count of how many times the function `lab()` (in `lab.c`) is called by the on-board code,
- orientation (yaw, pitch, roll) and angular velocity measurements from IMU,
- position and orientation (yaw, pitch roll) measurements from mocap.

Show your TA the file with extension `.csv` that has these data from at least one flight.

3 Collect at least two sets of data

What data do you need in order to answer the four questions in Section 1? Think carefully, make a plan, and collect at least two sets of data. At least one dataset should be collected while flying, and at least one dataset should be collected while moving the drone around by hand. You may want to collect a lot more data than this—it's up to you, depending on what you think you need. At minimum, you will likely want to move the drone by hand through large changes in orientation (what happens if you flip it upside down? what happens if you spin it around over and over?) and large changes in position (what happens if it moves back and forth, up and down, in spirals, near and far from cameras, behind cameras, etc.—where are the boundaries of the "mocap workspace"?). Describe your plans for data collection to your TA and show your TA the resulting `.csv` files.

4 Visualize your data in graphic simulation

Show a 3D animation of the drone's flight in MATLAB, given the measurements of position and orientation in the coordinates of the room frame from mocap. You can do this by adding a small amount of code to `lab1_visualize.m`, available on the course website (along with `quadmodel.mat` and `mocapmodel.mat`). In particular, you might do the following:

- Create a MATLAB script `lab1.m` that opens a data file, that extracts the time, position, and orientation measurements from this file, and that calls `lab1_visualize.m` with the parsed data.
- Add code to `lab1_visualize.m` to apply coordinate transformation. You are given points that describe the shape of things—of the drone, and of a frame attached to the drone—in the coordinates of the body frame. You need to describe these points in the coordinates of the room frame. Here is a common pattern for coordinate transformation in MATLAB:

```
% p_in1 is a 3xN matrix that describes a set of points, one in each column,
%       in the coordinates of frame 1
%
% p_in0 is a 3xN matrix that will describe these same points, one in each
%       column, in the coordinates of frame 0
for i = 1:size(p_in1, 2)
        p_in0(:, i) = o_1in0 + R_1in0 * p_in1(:, i);
end
```

Note that `lab1_visualize.m` will even create and save a movie for you to share! Show this movie to your TA. (It may be helpful also to show the video that you took of the real flight.)

5 Choose a representative

Building community is important to me. To help build community in our class this semester, I would like to form a group of representatives—an "AE483 Congress," as you called it on our first day of lecture. There will be one elected representative from each lab section. This representative will have the following responsibilities, at minimum:

- Know the names of everyone in your section.
- Be attentive to your colleagues (e.g., bring their needs to the attention of course staff, work with course staff to meet these needs and to improve the course over time).
- Meet weekly as a Congress with course staff for one hour over snacks.

Talk about this as a full lab section. Give the name of a representative to your TA. Describe the process by which you chose this representative to your TA.

6 Summary of in-lab deliverables

You should have done the following things in lab:

1. Show the TA that you changed what data are being logged.

2. Show the TA that you collected at least two sets of data (that can plausibly be used to answer the four questions in Section 1).

3. Show the TA that you can visualize your data in graphic simulation.

4. As a full lab section, give the TA the name of a representative for the AE483 Congress.

Details of report requirements and of the submission process will be posted to Slack.