Lab 5 - Tuning PID controllers

In your final project you will likely be adding some additional sensor to the crazyflie. This additional weight will affect the crazyflie flight characteristics and may cause your crazyflie to perform worse than before. One way to compensate for this is to tune the PID controllers to match the new model. In this lab you will tune your PID controllers while simulating additional weight by taping a nickel to the top of your crazyflie's battery with double sided tape (Take care not to short circuit your board with the nickel).

One Approach

One approach to this problem would be to determine what PID parameters to adjust (as is the crazyflie has 20), update these values wirelessly using the pc-client, manually fly the crazyflie and "eyeball" how good it is. You would then land it and reason about if the change was better and why and then repeat. This would be a difficult method which isn't likely to produce effective results.

Problems - To do this properly we need to eliminate user error.

1. Any change in the amount of thrust applied by the user during flight test will change how the crazyflie performs from one trial to the next.
2. Stabilization happens very quickly and most of the time it is not visible to the human eye.
3. Even if we tuned only a single PID, manually iterating through all permutations of its 3 parameters within a given range would be difficult. This would require a lot time to keep track of what has changed, what should change next, and set all new variables via keyboard at each iteration.

A Better approach

Part1: Design For Test

To overcome problem 1, we will be modifying the pc-client to "auto pilot" your flight test. We provide you a basic implementation of a "AiController" class which will override inputs from the joystick and perform an automated test (see below for details). Take some time to think of the best way to test your PID controllers performance. You will need to improve this class to match your design.

Part2 - Design a "Goodness" function

To overcome problem 2, you will need to implement some sort of function to evaluate how well the PID performed. One obvious solution is to keep a sum of the total error of the controller over the test flight. The crazyflie would need to coordinate start and stop times with the PC-Client and return the total error after stop signal received.

Part3 - Design a PID Tuner

Addressing problem 3, you will need to research approaches on tuning a PID and write some code to automate this process. Typically, the tuner would determine new values for the desired PID controller and update them on the crazyflie. The new values performance would be exercised using Part1 and the results would be returned by Part2. The tuner would then evaluate the results and determine the next trails PID values accordingly, then repeat until some criteria is met. You can implement this tuner in the PC-Client code or in the Crazyflie firmware.

An Even Better approach?

Each team should consider the problem independently and criticize both approaches mentioned above. If you think there is a better method you must get approval from Dr. Zhong or the TA before completing your lab.

Provided Code

We provide you a AiController class for the PC-Client for download or you can clone the modified project. See the code comments for setup instructions. This class is a dropin replacement for the pygamereader.py and provides the ability to override any of the joysticks output with artificial intelligence (auto pilot). The AiController commandeers the "Exit" button and re-purposes it as an enable/disable switch for the AiController. Be sure to have this disable button and the kill button on your input device at the ready in case things go awry.

This class provides a basic takeoff, hover, land and delay loop. You will need to gradually increase the maxThrust parameter to an appropriate value so that it hovers just above ground level. If needed you will still have control over the roll/pitch/yaw. You can always turn off the AI by pressing the "exit" button, and pressing again to re-enable. Also be sure to have the "kill" button ready in especially when testing new functions.

This class also provides a function to update parameters on the crazyflie for you to use with your PID tuner.

Safety

When using any auto pilot functionality, do it in an enclosed space where everyone including you yourself wears eye protection.

What to Turn in

Your website should contain a report PDF, Final PID values in a plain text file, and a YouTube Link as described below. These files will be downloaded at the deadline. Any missing items will be counted late.

6/10 points - For each of the three sections, you will need to justify your design, discuss your implementation, and explain your results. You will need to include relevant code snippets in your report directly. While doing so be sure to maintain code indention and syntax highlighting for increased readability.

3/10 points - In a plain text file place your final PID values in the exact format shown below (replace the defaults with your values).

```
self.cfParams = {
   'pid_rate.pitch_kp': 70.0,
   'pid_rate.pitch_kd': 0.0,
   'pid_rate.pitch_ki': 0.0,
   'pid_rate.roll_kp': 70.0,
   'pid_rate.roll_kd': 0.0,
   'pid_rate.roll_ki': 0.0,
   'pid_rate.yaw_kp': 50.0,
   'pid_rate.yaw_kd': 0.0,
   'pid_rate.yaw_ki': 25.0,
   'pid_attitude.pitch_kp': 3.5,
   'pid_attitude.pitch_kd': 0.0,
   'pid_attitude.pitch_ki': 2.0,
   'pid_attitude.roll_kp': 3.5,
   'pid_attitude.roll_kd': 0.0,
   'pid_attitude.roll_ki': 2.0,
   'pid_attitude.yaw_kp': 0.0,
   'pid_attitude.yaw_kd': 0.0,
   'pid_attitude.yaw_ki': 0.0,
   'sensorfusion6.kp': 0.800000011921,
   'sensorfusion6.ki': 0.00200000009499,
   'imu_acc_lpf.factor': 32 }
```

Your results will be evaluated autonomously as follows:

1. 1 pt for maintaining basic hover. This should pass with default values :)
2. 1 point for maintaining flight during basic maneuvers (roll/pitch/yaw).
3. 1 point for top %50 of class with best "goodness" during the two evaluations above. The goodness function will be similar to approach described in lab

1/10 points - A YouTube video demonstrating your flight test in action (part1) with a nickle tapped on top of your crazyflie.

Looking Forward

During the final lab competition, your crazyflie's ability to stay within a ring will be tested by a similar AiController. You might want to use/expand this lab for your final lab to test autonomously keeping the crazyflie in a ring. A well turned PID controller will help keep the crazyflie in place. You might want to include additional PID Controllers in your firmware.