AE 483 Lab 4: Modeling and Simulation of the Crazyflie 2.1

Linyi Hou[[1]](#footnote-1)

University of Illinois at Urbana-Champaign, Urbana, IL, 61820, United States

The PID controller of the Crazyflie quadcopter is manually tuned to minimize flight time along two predetermined flight paths. Tuning strategies are discussed and demonstrated. Two configurations are developed that improve flight time by over 15% and 33% for the two paths, respectively. Ideal flight times are discussed, and possible improvements are identified, which include reduction in oscillation and synchronization of rise times along multiple axes.

# Nomenclature

= position in inertial frame

= velocity in inertial frame

= attitude in inertial frame

= attitude rate in body frame

= torque in body frame

= thrust in body frame

= state vector

= time derivative of state vector

= control input

= moment of inertia matrix

= rotation matrix about x-axis

= rotation matrix about y-axis

= rotation matrix about z-axis

= Tait-Bryan rotation matrix

= mass

= gravity,

= desired state

= state error

= measured state

= PID proportional gain

= PID integral gain

= PID derivative gain

# Introduction

The Crazyflie 2.1 is a small and versatile quadcopter with open-source firmware produced by bitcraze. In previous lab reports, features of the Crazyflie’s attitude and positioning system have been examined, and the effects of tuning the proportional-integral-derivative (PID) controller were explored. In addition, physical flights were conducted to evaluate the performance of the quadcopter, and the position PID controller was optimized to minimize flight time.

However, PID tuning with the physical quadcopter may be time-consuming. The testing duration is limited by the battery capacity of the quadcopter, and recharge times are significantly longer than flight times. Crashes can damage motor and cause inconsistencies in flight results, while replacement parts such as propellers, though inexpensive, might not be readily available. In contrast, PID tuning via software simulations is highly desirable as it can mitigate all of the above constraints.

In this report, a MATLAB simulation of the Crazyflie will be developed. First, a physics simulation will be created by solving the equations of motion (EOMs). Next, the Crazyflie firmware will be emulated to recreate the PID controller. Then, perturbations and noise models will be implemented to reflect physical flight conditions. Finally, the trajectory from a physical flight test will be compared against the fully developed simulation to validate its accuracy.

# Simulation Development

## Equations of Motion

The quadcopter has six degrees of freedom, meaning it can translate and rotate along any axis in 3D space. As a result, the states of the quadcopter include position, velocity, attitude, and attitude rate. The state vector is given as:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

The control input to the quadcopter, , comprises of the net thrust produced by all four rotors, as well as the body torques imparted by the thrust difference between motors and the motor torques. The control input vector is then

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

The EOMs of the quadcopter are found by taking the time derivative of the state and applying the control inputs:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

To begin, it is trivial to show the time derivatives of position and velocity:

|  |  |  |
| --- | --- | --- |
|  | ; | (4) |

where the rotation matrix is the z-y-x Tait-Bryan rotation sequence defined in Equations (5) and (6).

|  |  |  |
| --- | --- | --- |
|  |  | (5) |
|  | , , | (6) |

To obtain the attitude and attitude rate derivatives, some additional algebra is required. The attitude is given as z-y-x Tait-Bryan angles, so can be obtained by converting , which is in the body frame, to the inertial frame. The equation for the inverse operation, converting attitude rate from the inertial frame to body frame, is as follows [1]:

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

The body to inertial frame transformation is then easily found:

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

The time derivative of attitude rate in the body frame is found by taking the derivative of the moment equation, accounting for the Coriolis effect in the body frame. This yields the torque applied:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

Finally, the above can then be rearranged to find in Equation (10), thus completing the derivation of the EOMs.

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

## Cascaded PID Controller

The Crazyflie 2.1 uses a cascaded PID controller to convert vehicle and target state knowledge to motor commands. A detailed schematic describing the Crazyflie’s firmware implementation of the cascaded PID controller is shown in Figure 1. The simulation developed in this report will model all four PID controllers as well as the motor power distribution as shown in Figure 1, whereas the state estimator and the sensor measurements will be represented by random perturbations applied to ground truth. Implementation details are discussed in the next section.

Diagram

Description automatically generated

Figure 1: The cascaded PID controller implemented in Crazyflie 2.1 [2].

## Crazyflie Firmware Emulation

The Crazyflie’s firmware is extensively documented on GitHub [3]. To emulate the firmware,

## Hardware and Perturbation Modeling

# Flight Data Comparison

Table 1: Summary of position and hover requirements for flight plans A and B.

|  |  |  |  |
| --- | --- | --- | --- |
| Waypoint # | Waypoint Position [x, y, z] (m) | Hover Time (s) | |
| Flight Plan A | Flight Plan B |
| 1 | [0.0, 0.0, 0.5] | 1 | 1 |
| 2 | [1.0, 0.0, 1.0] | 1 | 0 |
| 3 | [1.0, 1.0, 0.5] | 1 | 0 |
| 4 | [0.0, 1.0, 1.0] | 1 | 0 |
| 5 | [0.0, 0.0, 0.5] | 1 | 1 |

Table 2: PID controller gains for flight plans A and B and corresponding flight times.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Flight Plan | , , | , , | , , | Flight Time (s) |
| A | [2.2, 0.0, -0.05] | [2.2, 0.0, -0.05] | [2.6, 0.4, 0.05] | 18.80 |
| B | [3.3, 0.1, 0.1] | [3.6, 0.1, 0.1] | [1.9, 0.2, 0.1] | 10.10 |

# Results

## Best Controller Gains and Flight Times

## Optimality of Controller Design

# Conclusion

# References

|  |  |
| --- | --- |
| [1] | G. P. Subramanian, "Nonlinear Control Strategies for Quadrotors and Cubesats," University of Illinois at Urbana-Champaign, Department of Aerospace Engineering, Urbana, IL, 2015. |
| [2] | bitcraze, "Controllers in the Crazyflie," 2020. [Online]. Available: https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/functional-areas/sensor-to-control/controllers/#cascaded-pid-controller. [Accessed 6 December 2020]. |
| [3] | bitcraze, "Crazyflie Firmware GitHub Repository," bitcraze, 2020. [Online]. Available: https://github.com/bitcraze/crazyflie-firmware/tree/master/src. [Accessed 11 December 2020]. |
| [4] | bitcraze, "Crazyflie 2.1," 3 October 2020. [Online]. Available: https://www.bitcraze.io/products/crazyflie-2-1/. |
| [5] | K. Ang, G. Chong and Y. Li, "PID Control System Analysis, Design, and Technology," *IEEE Transactions on Control Systems Technology,* vol. 13, pp. 559-576, 2007. |
| [6] | AE 483 Course Staff, "Compass2g - Lab 2 Week 1," October 2020. [Online]. [Accessed 8 November 2020]. |
| [7] | AE 483 Course Staff, "Compass2g - Lab 2," 29 October 2020. [Online]. [Accessed 2 December 2020]. |

1. Undergraduate Student, Department of Aerospace Engineering, AIAA Student Member. [↑](#footnote-ref-1)