Interim Report

DRAFT OF FINAL REPORT

Programmable Flight Controller

Brendon Camm
B00649176

Lucas Doucette
B00685962

Dylan Humber
B00695554

Submitted: March 27 , 2017

This document was written in LaTeX. It was compiled with pdflatex. The source of the document may be viewed at:
https://github.com/Brendoncamm/SYP/tree/master/Documents/Second%20Semester%20Report

Abstract

The following report summarizes the progress made throughout the Quad Rotor Drone Programmable Flight Controller project and the future recommendations based upon project findings. The goal of the project is to design a programmable quadcopter flight controller that will have the capability of responding to live control inputs and react appropriately to disturbances present. The objective of this report is to summarize the deliverables set forth for the project and to present findings, results and testing procedures. The report also outlines the design process of the project. The Quad Rotor Flight Controller project has three major components; flight and stabilizing simulations, implementation of software, and a GUI interface.

Authorship

- Abstract - Lucas Doucette
- Letter of Transmittal - Brendon Camm
- Simulations - Dylan Humber
- GUI - Brendon Camm
- Physical Implementation - Lucas Doucette
- Conclusions - Dylan Humber
- Future Recommendations - Group
- References - Group
- Appendices - Group

Contents

1 List of Acronyms

1. BLDC - Brushless Direct Current
2. FOV - Field of View
3. GUI - Graphical User Interface
4. PD - Proportional Derivative
5. PI - Proportional Integral
6. PID - Proportional Integral Derivative
7. POC - Proof of Concept
8. PWM - Pulse Width Modulation
9. RF - Radio Frequency
10. USB - Universal Serial Bus

2 Introduction

2.1 Background and Significance

Machine learning is defined as the science of getting computers to act without being explicity asked. This is achieved by the development of computer programs that can teach themselves to grow and change when exposed to different sets of data. There are two traditional types of machine learning algorithms: Batch learning algorithms and on-line machine learning algorithms. Batch learning algorithms require a set of predefined training data that is shaped over the period of time to train the model that the algorithm is running on. On-line learning uses an initial guess model that forms co-variates from that initial guess then passes them through the algorithm to form an evolved model a new set of covariates are formed from the evolved model and then fed back to make a new prediction. The loop runs continuously so that the evolved model is constantly growing and learning to adapt to certain situations. Dr. Rhinelander's research is concerned with on-line machine learning algorithms therefore the drone we are developing will configured to adapt with these algorithms.

Quadrotor drones have been on the rise in popularity in the last several years due to their simplistic mechanical design and many practical uses. The application of these drones vary from a hobbyist flying around their neighbourhood to military

personnel carrying out high risk missions. A video recording device of some sort is generally attached to the drone and the video feed is relayed to a basesation for the operator to gain a field of view (FOV) of an area of interest. Having the capability to have a continuous video feed allows the drone to be used for many practical applications including but not limited to: Traffic condition monitoring and surveillance missions. While these drones are very sophisticated and advanced devices they are are missing one aspect that is very important to further Dr. Rhinelanders research: They are not totally configurable.

As mentioned, Dr.Rhinelanders research is concerned with on-line machine learning algorithms and without a platform that is completely configurable his research would be limited. Before the machine learning algorithms are implemented onto the drone it must first be able to be controlled. This is where we come in, we have been tasked by Dr. Rhinelander to develop a flight controller that recieves control inputs over Wi-Fi. Having a completely open source flight controller will allow for the addition of the machine learning algorithms to the flight controller software so that the drone can learn to partially, and eventually fully fly on it's own and make intelligent decisions.

2.2 Simulations

2.3 Physical Implementation

The physical implementation of the drone software and hardware requires a communication channel between a control input, a base station host, a wireless communication receiving unit, and a host for the controlling system. The objective of the physical implementation is to provide the software and hardware design required to successfully implement a quadrotor drone flight controller. It is desired to have a controller that is capable of sensing disturbances in flight and stabilizing the system based upon the measured disturbances.

The objectives of the physical implementation for the project begin with successfully creating a WiFi communication channel between a controller input and a wireless device placed on the drone hardware. The drone is to have flight sensing capabilities, interfaced with both the on board wireless device and a grounded controlling station. The flight sensing is to include yaw, pitch, roll and altitude measurements with a polling rate sufficient for real time control of the drone. The flight sensing is to be combined with a set point from the wirelessly transmitted control input by a PI or PID loop to control the flight of the drone with stabilizing features. The code written for the control input, communication interface, and finally the controller is to be well documented to ensure ease of use and simplicity for updating and modifying after handing over the final product.

2.4 Graphical User Interface

The Graphical User Interface (GUI) requires a network connection between the base station host it lives on and the Raspberry Pi 3 to recieve the serialized dictionaries containing sensor data and controller information. The objective of the GUI is to provide the user with a medium to access information regarding the current flight, such as, the altitude at which the drone has flown or how the physical controller is configured. It is desired to have a real time plot of the altitude based on the sensor information and the ability to initialize the physical controller.

The objectives for the GUI for the project start with selecting an appropriate development framework that will function across Windows, Mac OS and Unix environments. The GUI is to have the ability to initialize the communications between the base station host and Raspberry Pi 3 to allow for the control inputs to be sent using the base station host as well as receiving the data that is to be displayed on the GUI. The GUI is to be intuitive to avoid any unneccessary confusion with the end user. The code written and any other software used for the GUI will be provided and well documented to ensure that in event that the end user would like to modify anything after receiving the final product, this can be done effortlessly.

3 Methods

3.1 Simulations

3.2 Physical Implementation

3.3 Graphical User Interface

3.3.1 Framework Selection

There is a multitude of frameworks available for graphical user interface development so in order to narrow down the choices the following restrictions were placed:

- Must function on Windows, Mac OS and UNIX environments
- Have the ability to display live plots
- Be compatibile with the physical controller initialization software
- Have well documented libraries for ease of development

With these restrictions placed and further research into various frameworks the selections narrowed down to Qt and PyQt5. Each of these use the Qt framework which is regarded as a reliable, cross-platform GUI development software with extremely well documented libraries which met the major requirements. Although both Qt and PyQt5 use the same framework there are a few key differences.

The most notable difference at first glance is that the two use different languages, Qt uses C++ where PyQt5 uses Python but the biggest difference is in the design suites. When developing using Qt you're able to use the QtCreator design suite which allows the user to design the look of the GUI using click and drag widgets, code the functionality of the widgets then simply compile all within the design suite. When using PyQt5 there isn't a complete design suite, instead the user must design the user interface using QtDesigner and then import the GUI file into the Python script, from here the widgets are coded is relatively the same other than the obvious differences between C++ and Python.

4 Proposed Solutions

4.1 Simulations

4.2 Physical Implementation

4.3 Graphical User Interface

5 Project Results

5.1 Simulations

5.2 Physical Implementation

5.3 Graphical User Interface

6 Discussion

6.1 Simulations

6.2 Physical Implementation

6.3 Graphical User Interface