WCC ARLISS 2016 Status Report

Damion Rosbrugh

April 24, 2016

Introduction

The following is a report on the most up to date status of the WCC quadcopter project intended for entry in the 2016 ARLISS competition. Currently we have an assembled quadcopter which has jumped out of my hands with great enthusiasm but still requires a fair amount of tweeking before it will be ready for flight.

All files currently associated with this project; including navigation program, flight test program, and budget/flight analysis can be found on github at the following link. https://github.com/dhaead/Arliss_2016

The areas of this project currently being discussed are:

The airframe

consists of: frame,motors,esc's,power distribution,micro controller,batteries

The navigation system

consists of: micro controller, IMU, GPS, navigation algorithm, data aquistion, data processing

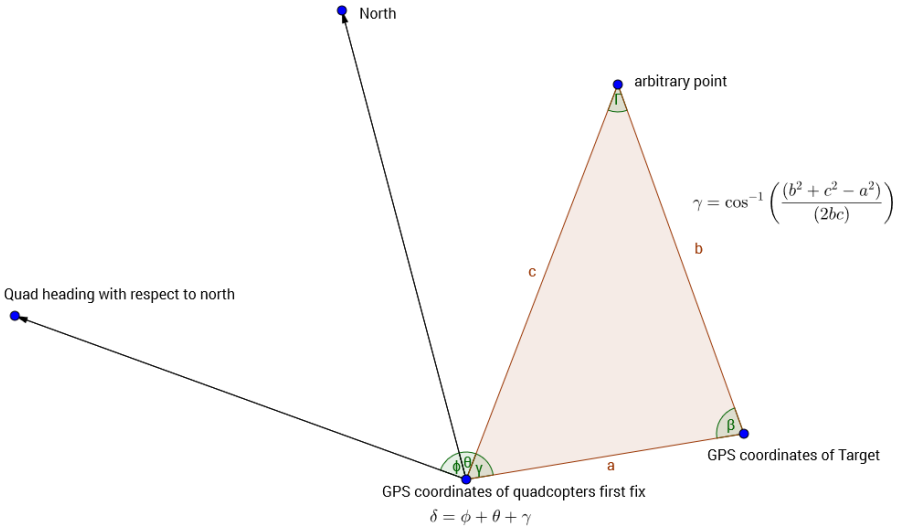
Airframe

Initial proof of concept flight tests for the airframe have been so far unsuccessful. Due to a malfunction with our main micro controller (Adafruit feather M0 basic proto) it was necessary to utilize a different microcontroller which happened to be on hand to do testing with the airframe. Our current airframe electronics package is the arduino atmega 2560 which signals the ESC's to turn on the motors. We are using a 1.3 A 30C lipo battery 11.1 V. I have implemented an 11 second test flight program which immediatly arms the ESC's then waits 7 seconds and goes to full throttle for 2 seconds then throttles down to 3/4 throttle for 2 seconds then shuts off. I can see that the quad is trying to lift but it's not quite getting there. I believe that our quad is too heavy. I am trying to find ways to lighten our load however I think the current frame that we have is the bulk of the trimmable weight. We either need a store bought frame or a 3-D printed frame which will be lighter than the bass wood frame I put together. There may be other factors which affect the quads ability to lift and I will be looking in to those as well. In addition, after analysis of our quads 30C lipo battery pack, I've found that this pack is not ideal for our purposes as it is not capable of delivering the full load of current being demanded from the motors. Further analysis is needed to determine the optimal battery to suit our needs.

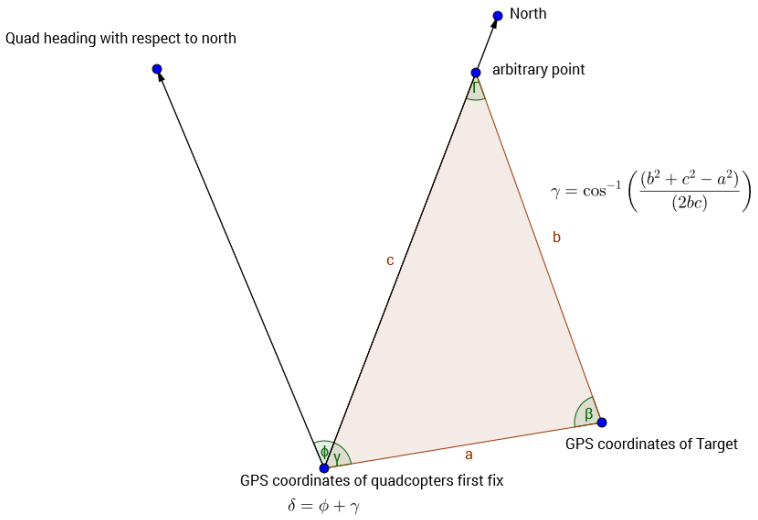
I have begun to implement code which should allow us to control the quads roll, pitch, and yaw. This will in turn lead to the programming necesarry for the microcontroller to send translational and rotation commands to the motors during the descent.

Navigation

We made some progress as far as our navigation system is concerned. I have wired up the IMU and the GPS to the Adafruit feather M0 basic proto microcontroller, and began doing distance calculations and developing and implementing the algorithm that will be used to navigate to the target. The start of the mission will be triggered by a photocell which will activate upon exit from the rocket. The algorithm to be implemented and tested will utilize the law of cosines and prior knowledge of the test area to simplify the math. The idea will be to take a gps fix of the quads initial position after ejection from the rocket. This fix combined with the coordinates of our target as well as an arbitrary third set of coordinates will give us the triangle we need to calculate the distances between all three of these points.



Since our choice for the third point is arbitrary and we have an onboard compass, we should pick a point known to be north of the approximate ejection region(the estimated flight corridor of the rocket). Because we can calculate our angle off of north using the onboard compass, and if we know the distances between the three gps coordinate pairs and we choose our third point to be north, we can use the law of cosines to determine the angle between north and our target. This added to the angle between our body axis and north, will give us the angle the quad must rotate in order to be heading toward the target. In the figure this is the angle δ.



Summary

Overall our current status is that we have an operational if not flight capable airframe. The navigation circuit has been built and is in working order, though more work needs to be done implementing the navigation algorithm.

Analysis

Below is a graph of the thrust to weight ratio of the combined efforts of our motor as a function of load weight colored by Throttle Percentage. This is usefull information because in order to maintain flight the quad must have a thrust to weight ratio of at least 2. From the graph we can see that our quad should weigh no more than 680 grams in order to have a thrust to weight ratio of 2 or more for 60 percent of full throttle.

```
library(reshape2)
library(ggplot2)
```

Thrust To Weight Calculations

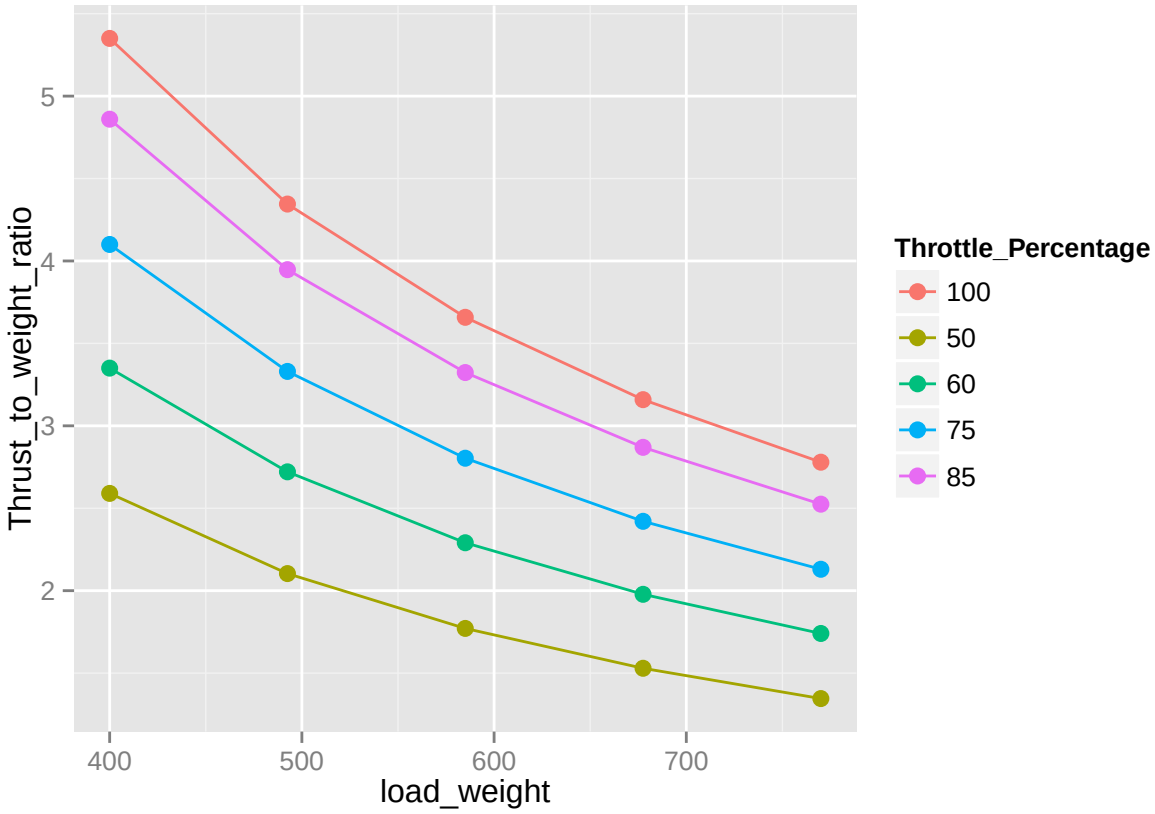
```
num_motor<-4
load_weight<-seq(400,770,by=370/4)
total_motor_thrust<-num_motor*cbind(259/load_weight,335/load_weight,410/load_weight,486/load_weight,535/
amps.per.motor<-c(3.9,5.6,7.3,9.4,11.1)
watts.per.motor<-c(43,62,81,104,123)
percentage.per.motor<-c(50,60,75,85,100)
total_motor_thrust<-data.frame(total_motor_thrust)
tmt<-data.frame(cbind(load_weight,total_motor_thrust))
names(tmt)<-c("load_weight",percentage.per.motor)

time<-seq(1,60,by=1)


bat<-data.frame(cbind(percentage.per.motor,4*amps.per.motor,60*((amps.per.motor*4/.7)/1.3)))
names(bat)<-c("Throttle_Percentage","Total_Current","Bat")


tmt<-reshape(tmt,
  varying = c("50", "60", "75", "85", "100"),
  v.names = "Thrust_to_weight_ratio",
  timevar = "Throttle_Percentage",
  times = c("50", "60", "75", "85", "100"),
  idvar = c("load_weight"),
  direction = "long")

ggplot(tmt, aes(x=load_weight, y = Thrust_to_weight_ratio)) +
geom_line(aes(colour=Throttle_Percentage, group=Throttle_Percentage)) + geom_point(aes(colour=Throttle_
```


Thrust_to_weight_ratio
5
4
3
2
400
500
600
700
load_weight
Throttle_Percentage
100
50
60
75
85