Rocket Telemetry - Software and Functional Design

Students: Dayja Young, Darnell Richards

Advisors: Dr. Farzad Moazzami, Dr. Richard Dean, Wondimu Zegeye

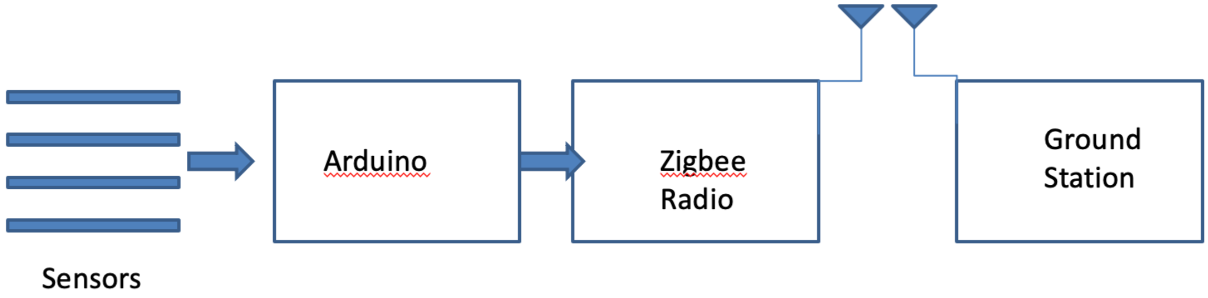
Morgan State University, Electrical and Computer Engineering Department

ABSTRACT

This paper presents a preliminary design for the telemetry package software for Morgan's rocket project. This software will capture and transmit data from accelerometers and gyroscopes to track the trajectory of the rocket after launch. The paper will focus on the software in the design and a simulation of the module to track trajectory. This paper is in conjunction with the "Physical and Function Design" paper as our undergraduate students worked together on the telemetry package. Morgan State has received a $1.6 million aerospace grant that will allow the school to complete a liquid-fuel rocketry lab and to recruit and hire a faculty aerospace leader to create a world-class program in liquid fuel. The school is looking to build and launch a liquid fuel rocket than can reach 150,000 feet by 2022.

INTRODUCTION

Morgan State University's Wireless Network and Security Lab (WiNetS) is under the Electical Engineering Department. The WiNetS lab has a focus on cyber security. We are tasked with creating a rocket telemetry module that provides measurements ranging from altitude, direction, heading, distance and location. The module is outfitted with a micro-controller, two digital radios, a sensor board, and a GNSS (GPS component). The telemetry package will comprise of a ground unit and onboard unit that communicates by radio signals that will relay the information to the ground unit. The program implemented within the Arduino will allow the information to be displayed through the utilization of MatLab.



SOFTWARE MODULES AND FUNCTION

The software comprises of programs for the rocket segment the ground station segment. The software is used to transmit data between the two using the languages Arduino, C++, Python and MatLab. These programs will operate by sending and receiving GPS sentences using a main loop along with a data handling code.

To have a completed telemetry model we must have a a running software. Using the Arduino code that outputs the GPS Sentence. It uses an Arduino IDE Software. The Arduino has limited computational capacity along with limited memory. UART pins use Rx for GPS and Tx for XBee. Latter limits the number of libraries used and limiting variables to avoid stackoverflow. Libraries used are SDFat, SPI, WIRE, NeoHWSerial, L3G, LSM303, and LPS. There are three major functions within the code which include reading the GPS UBX sentence, calculating attitude, and outputting the data. The data is self-contained in 32 bit intervals which makes the data readable by the union of a byte array and a structure. The GPS sentence is an outputted if-else statement.

The other part of the code includes receiving the GNSS, UBX, NAV, and PVT sentence. To read the PVT, the code runs without an interrupt connected to the Rx UART pin. The interrupt code is in the algorithm.

The main loop of this code runs at 50hz, every 20 milliseconds it reads data from the Polou Altimu sensors. The main loop uses the AHRS algorithm to calculate attitude. If the code prints “true” it updates the output sentence and sets the flag to “false”. The system prints a stream of 80 bytes from the structure to the XBee and SD card.

The ground segment consists of a telemetry base station which uses a C++ code. This station records and reads data. The code was developed in a multi-platform IDE for C++ programming. The telemetry station had three main parts which include: receipting and logging off the message, converting data, and plotting vectors and plotting UI. This function logs the sentence into a binary file then passes it onto the vector creating thread. The main thread displays the graphs and options of the UI. This allows users to choose which plot to display. Another code used was Python, to convert binary to CSV code because of built in structure casting and printing. This code allows for easier post processing.

The final code used was a MatLab data handling code. MatLab was used for figure plotting. The code takes inputs from a comma separated value file (CSV), extracts data, and applies factors to get measured parameters. The code also obtains time vectors for GPS and sensor data values. This script was used for post processing and analyzing data. (Caiado 30)

RESULTS

Here are results from simulation based off Newtonian Physics.

MATH

Distance vector in X and Y dimensions

$D_x(t)= x_0 + V_{xo}(t) +.5A_x(t)^2$

$D_y(t)= y_0 + V_{yo}(t) +.5A_y(t)^2$

Where Xo, Yo, Vxo, Vyo are initial values set to zero

Add effect of Drag

$Fd= pACV^2/2$

Where P = density of medium, A = cross section area, C = drag coef

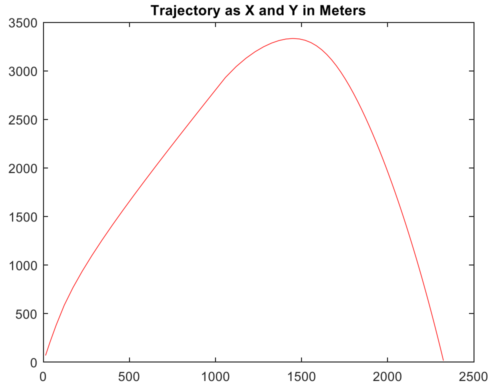
P= 1.23 Kg/M2   for air

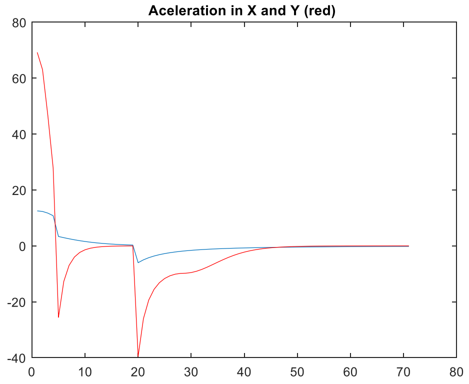
A= 2 sq meters (~ foot diameter rocket)

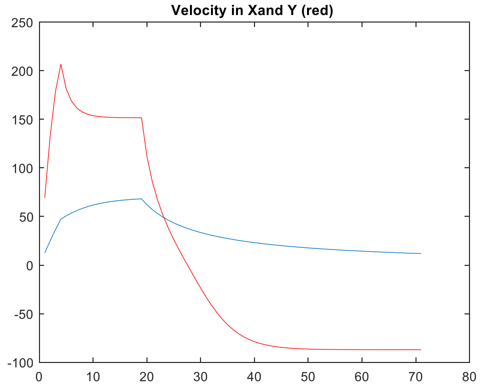
C= .1 for sharp pointed shape

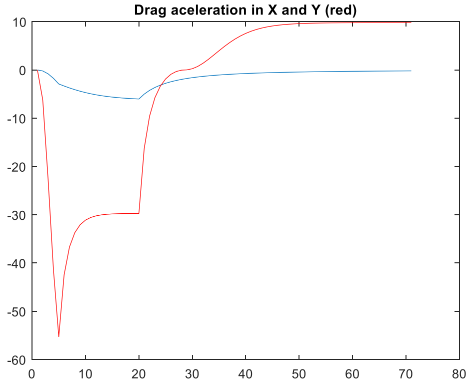
A = Acceleration = Force (F)/Mass(M)

Force = Fp (Propulsion) – Fg(gravity) – Fd (Drag)


Trajectory as X and Y in Meters


Aceleration in X and Y (red)


Velocity in Xand Y (red)


Drag aceleration in X and Y (red)

CONCLUSIONS AND FUTURE WORK

The module is curated with tried and tested methods with the intentions of making modification specific to the rocket's requirements. Progression has been pushed back this past year due limitation of access to parts and labs, however the setup with establishing communications with the Arduino and radios have been successful. By late august or early September a fully functioning module will be operational.

REFERENCES

Clavel, Guilherme and Perestrelo Caiado. "Development of an Instrumentation System for Small Rockets." (2016).

MorganStateU. (n.d.). *Rocketry program*. Morgan State University. https://www.morgan.edu/research_and_economic_development/cross_school_research_programs/centers/rocketry_program_.html.

Key words: *Rocket Telemetry, software design, functional design*