CNT Capstone Project

RC Black Box

Michal Szmaj & Joel Roy

Proposal

We, Michal Szmaj and Joel Roy, are submitting this project proposal for consideration by the CNT department. After a few revisions we are confident in the project we are proposing: an *RC Black Box* (RC: Remote Controlled). It will be produced using an *Arduino Uno,* a *Raspberry Pi Zero,* and several sensors that can measure altitude, temperature, and several other metrics discussed below.

After some research we are confident that this device has merit in the real world as we intend on creating a device that can not only measure metrics for an *RC* device such as a drone, but, can also provide the same - or similar - metrics for larger machinery such as a car. We intend on using an *Arduino Uno* for the metric capture device and a *Raspberry Pi Zero* in order to transform the data and allow wired (USB) and wireless (WLAN) interfaces for the user to collect the data during and after recording.

The metrics will be collected using the *Arduino Uno* which will collect the data using three sensors: a GPS module, a *BMP180* (which collects barometric pressure, temperature, and altitude), and an *ADXL335* (which collects x, y, and z axes and speed). The device will be powered by a *3.7V Lithium Ion Polymer battery* and converted into usable energy by an *Adafruit PowerBoost 1000 Charger*.

The low level code - which involves data collection on the *Arduino Uno* - will be written in *C* and the *Arduino C* library. The *Raspberry Pi Zero* will be running *Raspbian OS* and any low level code written on this device will be written in *C* as well. If time allows, we intend on making a *GUI* (Graphical User Interface) for users which will most likely be written in *Python* using *PyQt5* (possibly *PyQt4* if *PyQt5* isn't applicable). However, we are looking into the possibility of writing the *GUI* application in *C++* and *Qt4/Qt5* instead as this would provide both of us some experience and knowledge in *C++*.

In order to meet the deadline we are producing the device in small, defined increments. First we will start with the basic device that can collect the metrics defined below, then we will incorporate a battery to power the device. After that we will create an interface for the user to interpret the data. The final part of the project, if time allows, is to create a wireless receiver device that can read the metrics in real time — this could be used in emergency scenarios in which the device is lost or out of range and data needs to be transferred wirelessly.

Research & Scope

Elements that require research/in scope:

- Learn how to power the *Raspberry Pi Zero* and the *Arduino Uno* using a *Lithium Ion Polymer* battery and the *PowerBoost Charger* breakout. This will involve getting a long enough charge for the device to be used and calculating charge remaining.
- Interfacing requirements for the *Adafruit Ultimate GPS Featherwing* Breakout board, *BMP180* sensor, and the *PowerBoost Charger* using the *Arduino Uno.*
- Learn how to design and 3D print a case for the device that will be as light and small as possible.
- Learn how to configure the *Raspberry Pi Zero* in order to interface with a receiver and/or a computer to transfer data.
- Code that will need to developed in order to interface between the *Arduino Uno* and the *Raspberry Pi Zero* (most likely done in C).
- Learn *Python* and *PyQt5* (or *C++* and *Qt4/Qt5*) in order to create a *GUI* application for reading and interpreting data by the user.

Elements that are out of scope:

- Testing on other machinery than what we have at our disposal (two drones, car).
- Creating a printed circuit board in order to make the device smaller and utilize space.
- Fine tuning the battery life of the device.
- Acquiring the most accurate/efficient parts for the device.

Parts

Elements that are currently available:

- *Arduino Uno* single board micro-controller
- *ADXL335* Accelerometer
- *Adafruit Ultimate GPS Featherwing* Breakout board ~$55
- *BMP180* Barometric Pressure, Temperature, and Altitude sensor ~$13.20
- *PowerBoost 1000 Charger* for converting 3.7V battery to 5V power source ~$27
- *Tiny Breadboard* ~$6

Elements not currently not available:

- *Raspberry Pi Zero* ~$20

Schedule

We propose the following timeline to ensure that the project is completed in time:

Week	Activity
1: January 2nd - 6th	Project selection and research. Select a project and figure out the necessary hardware and software that will be required in order to complete this project.
2: January 9th - 13th	
3: January 16th - 20th	Project Proposal due. Parts should arrive this week. Start assembling the device and start planning the software architecture.
4: January 23rd - 27th	With device assembled, start work on 3D printing the case. Start preliminary programming in order to get metrics from hardware. Start research on interfacing between *Arduino Uno* and *Raspberry Pi Zero.*
5: January 30th - February 3rd	Finish programming the *Arduino Uno* and start interfacing with the *Raspberry Pi Zero*. Start research on *Python/PyQt* or *C++/Qt*
6: February 6th - 10th	Progress Report #1 finished and handed in. Finish physical build of the device (add LED battery indicators).
7: February 13th - 17th	Start research on how to power the device with a *Lithium Ion Polymer* battery. Continue programming.
X: Reading Week	Reading Week; no classes.
8: February 27th - March 3rd	Research ways to calculate remaining battery change. Implement battery in device. Continue programming.
9: March 6th - 10th	Continue battery implementation and coding. Start testing if possible.
10: March 13th - 17th	Progress Report #2 finished and handed in. Start/Continue testing.
11: March 20th - 24th	Start GUI application for data retrieval. Continue testing. Start TPS Report and operator's manual.
12: March 27th - 31st	Finish coding for GUI and metric collection. Finish battery implementation.
13: April 3rd - 7th	Start implementation of wireless interfacing between device and receiver. TPS Report due, operator's manual due.
14: April 10th - 14th	Project Complete. Technical report, working demo, in-class dry run.
15: April 17th - 21st	Project handed in for grading and final presentations.

This project is an attempt to gain a better understanding of hardware and software that didn't get a chance to work with in school. We gain experience in *Arduino/Raspberry Pi* software and hardware along with learning and gaining experience with two languages we didn't use in school: *Python* and *C++*. We will also gain knowledge on subjects that aren't necessarily part of computer engineering: wireless connectivity and battery management.