Goose Chaperone

Design Document

Group Number: 17
Clients: Dr. Randall Geiger & Dr. Degang Chen
Advisor: Dr. Randall Geiger

Team Members:
Weston Berg
Zhihao Cao
Alec Morris
Johnson Phan
Woodrow Scott

Team Email: sddec19-17@iastate.edu
Team Website: https://sddec19-17.sd.ece.iastate.edu/
Revised: 04-24-19/V2.0

Table of Contents

List of Figures

1. Introduction

1.1 Acknowledgement

The Goose Chaperone Project team would like to begin this plan by giving our regards to the College of Electrical and Computer Engineering for giving engineering students a great opportunity to create a project while also ensuring our readiness for our prospective careers. We would also like to take this opportunity to thank Dr. Randall Geiger for not only advising us through the design process, but also for being our client and providing the financial backing necessary to create our project.

1.2 Problem and Project Statement

Across North America, golfers, pilots, and homeowners all face a similar pest: *Branta canadensis*. That is, the Canada goose. Whether they are nesting in front lawns, attacking golfers on the greens, or even destroying millions of dollars worth of aviation equipment on the runway, one thing is certain - the current methods of goose prevention are not sufficient.

For this reason, our group has been tasked with finding a way to effectively keep geese out of areas in which our potential clients do not want them. This project will be centered around a autonomous "robot" that will utilize scare tactics of various shapes and sizes in order to accomplish this goal.

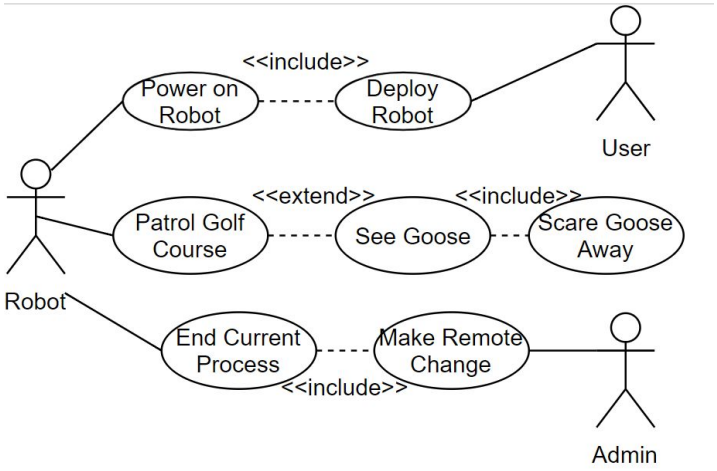

Figure 1: Use Case Diagram

1.3 Operational Environment

While the project will have the ability to work in almost any terrestrial environment, we are initially limiting the scope to just a golf course in Ames. Given the often unpredictable weather in Iowa, we will need to keep in mind both hot and cold temperatures, as well as the constant threat of precipitation and take measures to ensure the safety of our robot.

1.4 Intended Users and Uses

Given the project's nature with an autonomous robot, there will not be many use cases that are not centered around the robot itself. Because of this, we will have to be careful when defining users and use cases to ensure a good design.

We have identified a handful of users that are necessary for the project. They are listed below along with their use cases:

- User - Golf course owner/ employee who will need to deploy robot.
- Robot - Or created robot that will scare and chase geese, patrol, and scan areas.
- Admin - A member of our team that can make remote changes.

1.5 Assumptions and Limitations

Assumptions:

- Materials necessary for the project will be able to be ordered in a timely manner.
- A GPS module will allow for accurate movement within a couple of meters.
- The initial unit will be used only on a golf course.
- Geese will be scared by the methods chosen to frighten them.
- Geese will continuously be scared by said methods.
- The C library chosen will be sufficient to recognize geese.
- The robot will be able to move at speeds greater than 3 MPH.
- The robot's battery will last more than 30 minutes (in use).

Limitations:

- The amount of memory on the microcontroller is expected to be low (< 1 GB), so we will need to use it wisely
- Processing power will also be relatively low, so applications should not be too demanding.
- Overall costs may not exceed $400.
- No AC motors will be used, only direct current.
- No internet access will be used by the robot.
- Project must be completed by December 2019.

1.6 Expected End Product and Deliverables

The final delivered project will be a fully-functioning autonomous robot that is able to scare geese away using a number of on-board scare tactics.

The robot will be accompanied by a charging cable that will be used by clients to charge the unit's battery in between uses.

The estimated delivery date for the completed robot will be the end of the fall semester, in December of 2019.

2. Specifications and Analysis

2.1 Proposed Design

For the structure of the robot, we are using polyvinyl chloride or PVC for short. For efficiency and cost, we are using the pipe version instead of sheet as it only requires drilling and cutting. As our design is only intended to scare and chase with no harm to any animal, it's only required to survive the environment. PVC is considerably lighter than most metal and more resistant to corrosion. Its rigid and durable for this project and its pipe form will serve as covering for exposed wires and devices.

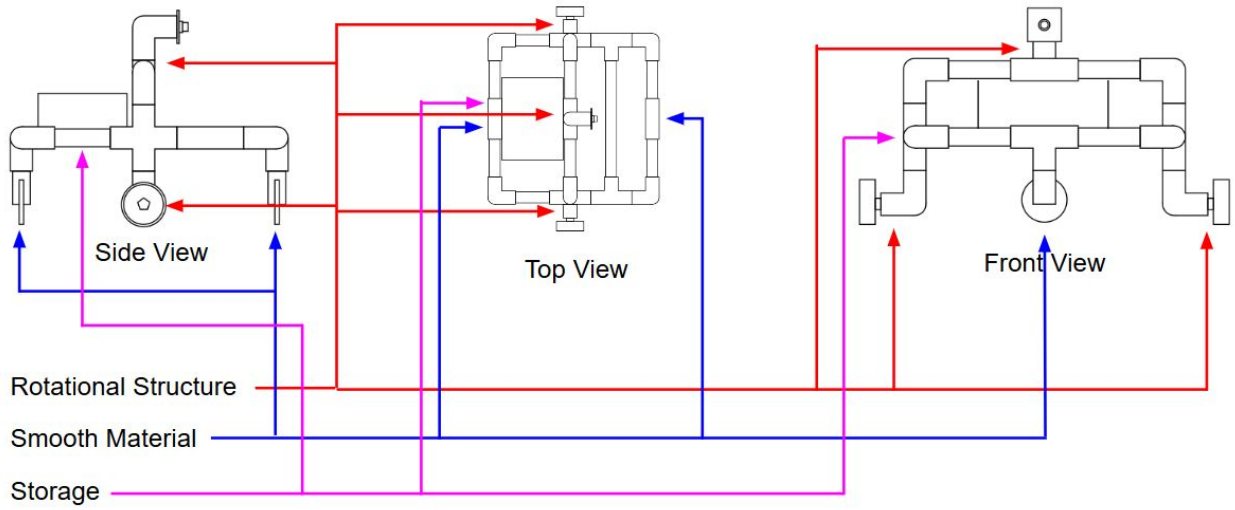

Figure II: Diagram of chassis structure

The Beaglebone Black will be the microcontroller controlling the data flow of the robot. This platform has both the functions required to collect data from the environment and meets the minimum memory needed to store or hold data collected from the environment. All of the following peripherals will connect to and extend the functionality of the BeagleBone Black. For environmental data collection, we are using the Logitech C270 webcam and Ultrasonic Distance

Sensor. The camera is used to identify geese and hazardous objects, such as water or boulders, while the sensors are used to check the distances of objects in the robot's surroundings. Validation of the robot's position will be provided by the Adafruit Breakout GPS module. This is the most compatible GPS for our microcontroller. For robotic movement, we are using brushed, geared DC motors. Brushed DC motors are simple and inexpensive in comparison to brushless which makes brushed ideal for prototyping. A final design for this project may swap for brushless due to improved performance. The motors will also be geared to a medium ratio, likely around 150:1, to trade speed for more torque. The current choice of external battery is lithium ion 2 INR18650-25R 3.7V. These batteries are rechargeable and can therefore be reused.

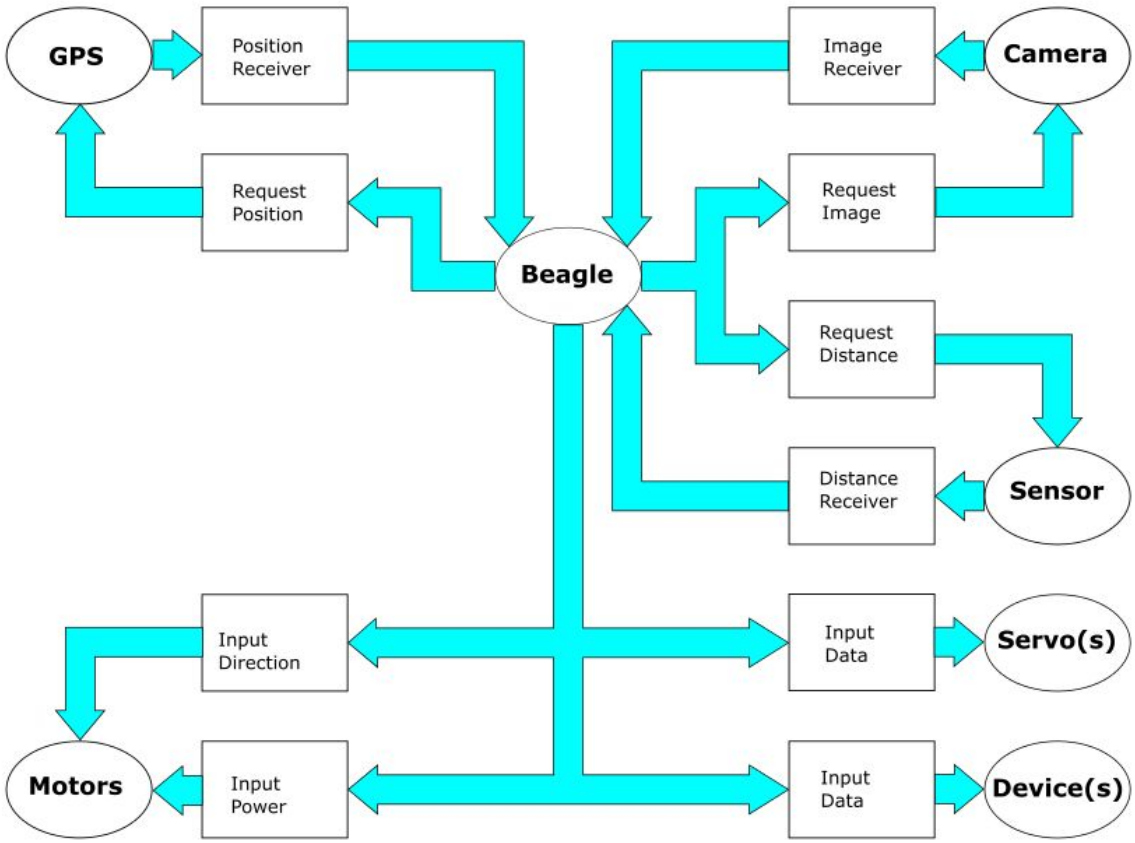

Figure III: Block diagram of peripheral component connections

The software environment of the robot will be relatively simple. Natively, the BeagleBone Black runs a distribution of Debian as its operating system. Debian, a 'Unix-like OS', will provide a standardized framework for the rest of the robot's software to run in. The robot's mainloop will be implemented in C. C exposes the low-level interfaces needed for configuration, operation, and monitoring of the various external components that will be attached to the microcontroller. Interfacing with these components should be straightforward thanks to the structure provided by Debian. Two software libraries, Tensorflow and OpenCV, will be utilized for image processing. Tensorflow is useful for picture categorization and determining if certain objects are present within and image. Tensorflow needs to be run within a VM. Most likely we will use the 'light'

version of Tensorflow to save space since we only need basic functionality. The categorized image data from Tensorflow will be fed into OpenCV. OpenCV will allow the robot to determine where the identified objects are in relation to itself. This will be essential for accurate deployment of goose deterrence methods and safely avoiding hazardous objects.

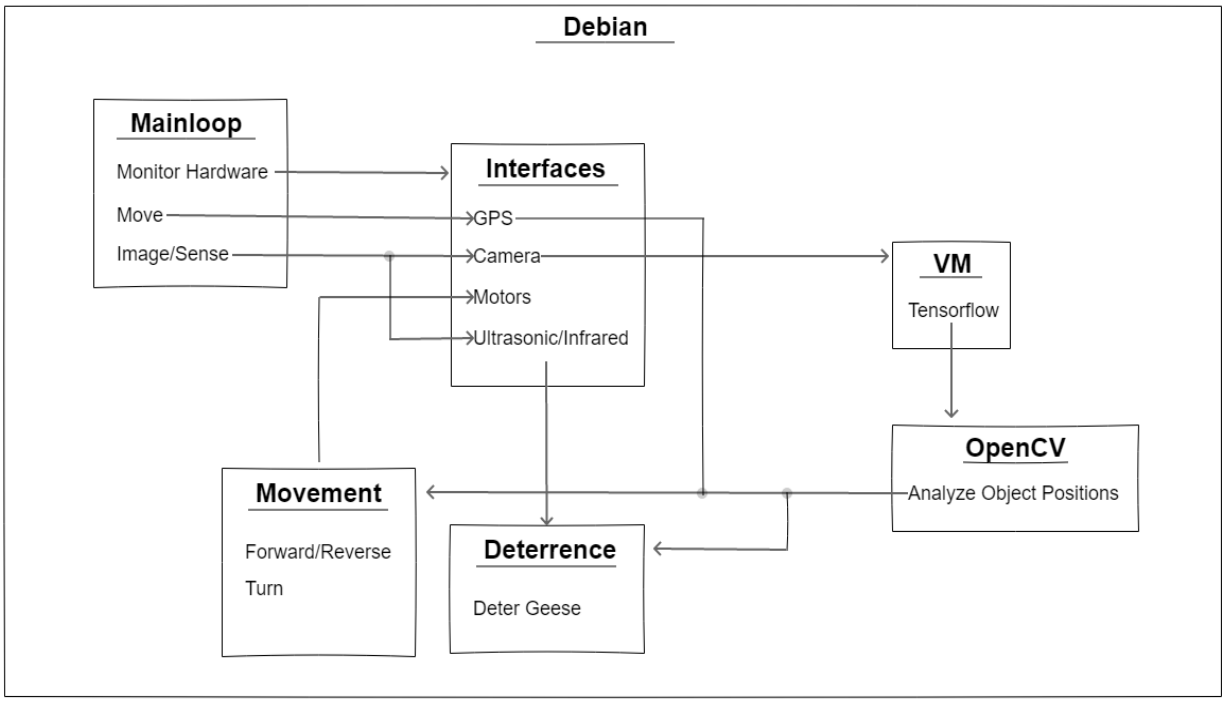

Figure IV: Block diagram of software environment

Our design should follow the functional and nonfunctional requirements below:

Functional Requirements

- The product must meet several requirements involving both software and hardware, including their ability to integrate and function cooperatively.
- Physically, the machine must be capable of movement by means of motorized wheels. They must be able to carry the robot reliably, and efficiently in terms of energy consumption. Additionally, they must be damaging to the course.
- Additional motor requirements include an actuator of some form to be used in startling and/or moving geese away from certain areas.
- A functioning logic board must be integrated and programmed to process data from sensors including GPS, optical sensors (camera), ultrasonic, and infrared. This data then must be used to drive the hardware components.
- The sensors applied must be capable of recognizing geese from periodically captured images, and determine the distance and orientation of the robot in relation to identified targets. Sensors must also be capable of notifying the logic board of obstacles and hazards.

- Due to the high possibility the coverage area is very large. The machine must be able to determine a pattern to follow as it patrols, scan, and analyze in specific locations to cover the specific area.

Non-functional requirements

- The current proposed board, the Beaglebone Black, will process updates though the camera and sensor at least once per second.
- The board will pair with the GPS module at least once per second to provide an accurate location
- The board will update movement speed by coordinating the motors at least 10 times per second.
- Internal logic platform-independent where possible and scalable
    - Platform-independence for future proofing
    - Scalability between different sized prototypes
- Robot can continuously operate for at least 60 minutes
    - Proof of concept not market ready design
- GPS Module frequency should be 5hz or greater
    - Allows for updates after 2 feet of movement.
- Keep cost within $400 budget
- Abide by relevant laws

2.2 Design Analysis

As of this moment, the vast majority of ours group's work has been centered around both research and making initial design decisions for our first prototypes. The research is separated into several branches: microcontroller, chassis, motor, sensor, camera, and gps.

Our initial microcontroller was meant to be the TI MSP430 but it lacked the necessary memory to store the data collected from the camera and sensors. After some consideration, we decided to use the Beaglebone Black as it meets the requirement for analog functions and large memory.

The initial research into a custom or compatible chassis was cancelled due to the nature of the project. Instead, the idea moved to using EI Tech components to construct the chassis needed for the robot. The components are bought separately allowing the chassis to be remodeled and since it is a metal type component, it is durable and long lasting for testing in the outside environment. Although durable and moddable, it is exposed to undesirable elements. It does not provide cover for electronic devices implemented. Other forms of protection like plastic wrapping will be required to cover from elements like rain.

The current motor, camera, sensor, and gps components have been selected based on cheap, compatibility, and meets our minimum function for this project. As it is yet to be tested, we cannot be certain of the reliability and function of the components. Should it prove to be ineffective, we will need to replace the components better ones.

The design pattern codes for autonomous robot has been decided for the robot to move to specific locations in the coverage area and analyze the surrounding for goose. As our project is

design to cover a large area, it would not be recommended to move through the entire area itself as it would be time consuming. By creating a design pattern codes for designated locations, the robot will patrol to all locations set by buyer/client. This is far more quick and efficient method. The implementation of the codes will require complex coding and modification.

3. Testing and Implementation

This device is focused on recognizing targets, sweeping through locations on motors, and using distance detection sensors in order to function. As such, each of these functions outline the tests required the device.

The motor systems may be remotely controlled at first to ensure that each motor coordinates properly. This involves ensuring that when the robot is directed to move forwards in a straight line that each motor spins in the correct direction and amplitude. When turning, the machine must be measured to rotate and move at the correct angle based on input.

Image recognition will largely be tested outside of the machine at first, as this is a time consuming process that will require many iterations of testing and refinement. Tests will be done manually, as test images are fed into the software and visually confirmed by a member of the team until the software can accurately determine geese. After this, the software may be loaded onto the beagleboard and tested in its environment to ensure that it functions correctly on a separate platform.

Distance sensors will largely be tested directly on the device, with results reported over USB or some other medium. Physical measurements of distance will be measured conventionally to ensure accurate measurements by the system. This process will be done with both the ultrasonic and infrared sensors.

Currently, as we have yet to order an necessary hardware, there is nothing to test outside of certain softwares.

The image recognition software, Tensorflow, has been under some work in order to set up a testing environment. However, due to problems with mismatched Python version requirements, there have been no significant tests at this time.

The test plan have 2 types. First type is components tests which test on robot motors, analog sensors, image processing, GPS, scare technique and goose identification. These tests is testing each components could work functionally. And then we move to the second type of test: prototype test. This test is testing components compatibility. Microcontrollers testing with the tensorflow to make sure microcontrollers could control each parts of robot. Navigating test with GPS to make sure robot understand its positions. Robot movement test with motor and wheels.

Safety test to make sure our robot do not damage people or the environments around. Behavior test is a evaluation of all tests in order to have some feedback to improve the robot.

3.1 Interface Specifications

Several types of hardware/software interfacing will be needed in order to test the components of our project. Tests will be necessary to ensure the proper function of hardware and software. For hardware, these tests will confirm inputs match outputs. For software, the accuracy of algorithms will be tested.

The different kinds of hardware that will be need tested are: motors, Sonar/IR sensors, camera, heat sensor, and microcontroller. A multimeter can be used to ensure that proper voltage and current levels.

The main software algorithm needing testing in this project is the logic for identifying geese from an image. Since this image processing will be obtained through a machine-learning algorithm we will be able to test the accuracy via datasets of images containing and not containing geese.There are several standards applicable to our project listed below. It will also be important to standardize code syntax for uniformity and clarity and interfaces for easy to use communication protocols between components.

Relevant Standards for the project include the following:

- IEEE 1008-1987: Standard for Software Unit Testing
    - *Description* - This standard is about unit testing of software or firmware systems. It describes methods with which unit testing should be carried out and how. The standard helps developers to write useful, relevant unit tests to ensure the functionality is true to its intended use.
    - *Application* - The software written to fulfill project requirements will need testing to prove the functionality of software units. Writing units tests based on the standard will lend to consistency across tests and validation of testing methods.
- IEEE 1625-2004: Standard for Rechargeable Batteries for Portable Computing
    - *Description* - This standard provides a framework for the creation of lithium-ion and lithium-ion polymer batteries. This includes the standardization of interfaces between batteries and their consumers. This standard helps assure reliable power to portable computing systems.
    - *Application* - Since one of our deliverables is a mobile computing system we will need ‘mobile’ power. Lithium-ion batteries fit the bill perfectly and a knowledge of their design and how to interface with them will be necessary.

3.2 Hardware and software

Various different kinds of hardware and software will likely be used for the test interfacing.

Hardware:

- Multimeter

At the very least a multimeter will be used to test the physical connections between the microcontroller and the external components. The multimeter will be used to record voltage and current levels over the connections. These recorded values can then be checked against technical specifications to confirm safe usage.

Software:

- Machine learning testing tools/software libraries
- Unit testing libraries

Testing sets for the machine learning algorithm will be used to validate the integrity of the machine learner's output.

Unit testing libraries will be used to confirm the overall logic of software components are behaving correctly. These components could include functionality such as wheel movement, sensor utilization, and course planning/tracking.

3.3 Functional Testing

At this point in time, there are no existing tests. However, we will soon be purchasing our components, and will then begin the process of doing small proof-of-concept tests for each component. After this, we will integrate them one-by one to the larger whole. For each piece added, standardized regression tests will be done to ensure the new component did not affect other pieces of the system.

Functional testing will include the following:

- Robot Motors testing
- Analog Sensors testing
- Image Processing testing
- GPS testing
- Scare Technique testing
- Goose Identification testing
- Compatibility testing
- Microcontrollers and tensorflow testing
- Navigation testing
- Safety testing
- Behavior testing

3.4 Non-Functional Testing

Testing must also be done to ensure that the project is viable in accordance with our non-functional requirements seen in the second section of this report. This should be done in the proof of concept stage, so that we are able to tell if each component is viable. There should be one test for each non-functional requirement that needs to pass initially in the proof of concept tests as well as in each regression test.

3.5 Process

This process can be seen in the below flow diagram:

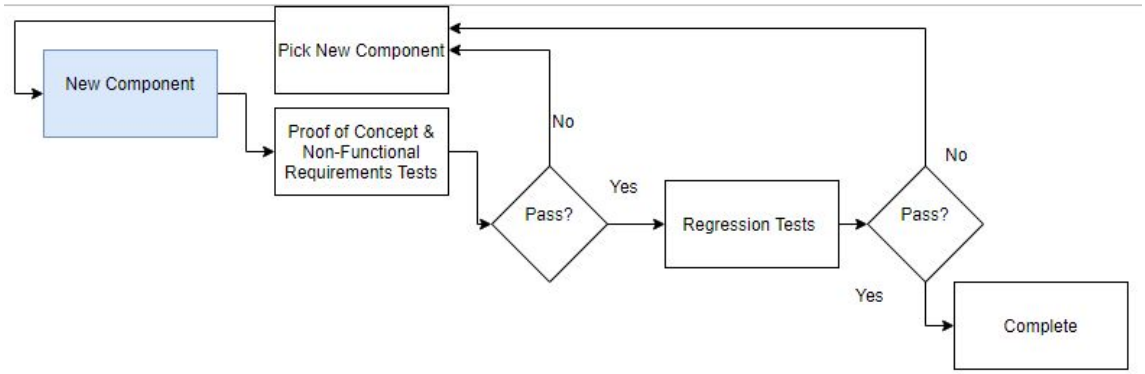

Figure V: Flow Diagram

As described in the above sections, the testing process will look something like the above diagram. Starting with proofs of concept, then moving into regression testing, the components will finally be accepted and added to the main system once completing this process.

3.6 Results

****Testing phase has yet to begin, but this is where results will go in the future****

4. Closing Material

4.1 Conclusion

So far, we have researched all necessary components for our project. We have specific components selected for the project and an iteration code plan for the project. The current objective is to buy the necessary materials to begin construction and testing of the autonomous robot.

Our main goal is to construct the autonomous robot and implement all necessary functions. Our team will use components to build the robot model and modify it in the testing stage should it not meet the requirements. The coding implementation will be done in iteration design with the main focus on patrolling, collecting, analyzing, and sequence of actions. As we do not currently have the microcontroller or components, we cannot test or validate the iteration designs or begin construction until we receive the materials.

4.2 References

[1]"Beagleboard:BeagleBoneBlack - eLinux.org", *Elinux.org*, 2018. [Online]. Available: https://elinux.org/Beagleboard:BeagleBoneBlack. [Accessed: 27- Mar- 2019].

[2]"Maestro GPS Receiver Datasheet", *Farnell.com*, 2019. [Online]. Available: http://www.farnell.com/datasheets/1681426.pdf. [Accessed: 27- Mar- 2019].

[3]"SPECIFICATION OF PRODUCT Lithium-ion rechargeable cell for power tools", *Vruzend.com*, 2014. [Online]. Available: https://vruzend.com/wp-content/uploads/2017/09/SAMSUNG-INR18650-25R.pdf. [Accessed: 27- Mar- 2019].

[4]"TensorFlow", *TensorFlow*, 2019. [Online]. Available: https://www.tensorflow.org/. [Accessed: 27- Mar- 2019].

4.3 Appendices

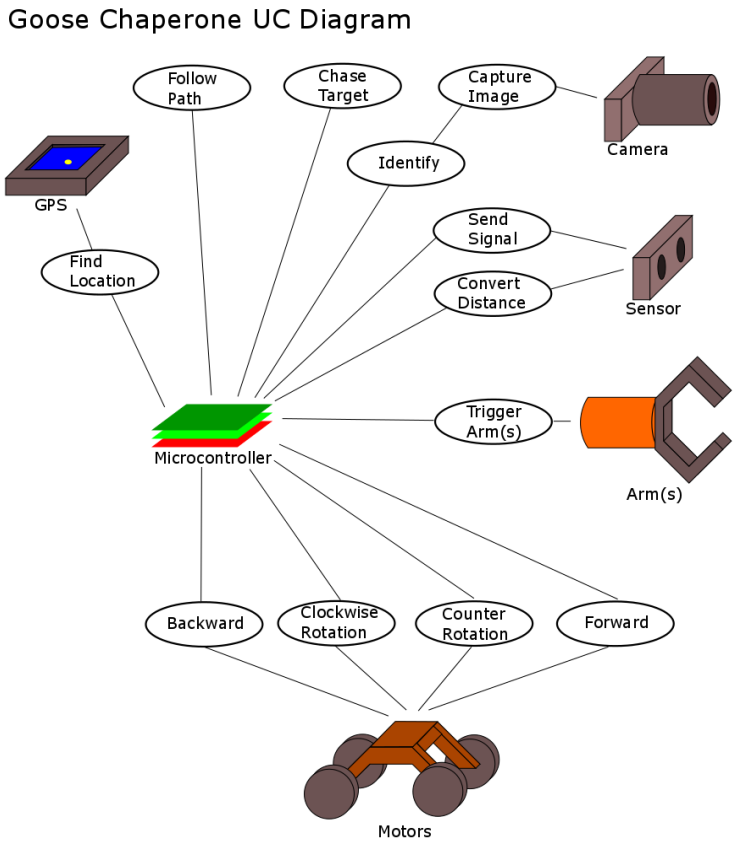

Figure VI: Inner Robot Use Case

The microcontroller sends data to move the motors. It also sends data to move the arms. It sends data to trigger sensor to send and receive frequency to find if there is anything within range. It receives image from camera to be processed for geese. Should it find a geese, it will follow a chase pattern. Otherwise it will follow a set path. The gps determines it way back.