UGV - Unmanned Ground Vehicle Controlled by Leap Motion
4OI6 Capstone Project Final Report

John Octubre
Computer Engineering IV
McMaster University
Hamilton, ON
johnoctubre7@gmail.com

Nisshanth Navaratnam
Electrical Engineering IV
McMaster University
Hamilton, ON
nissh.n@gmail.com

Nathan Sinnathurai
Electrical Engineering IV
McMaster University
Hamilton, ON
nathan.sinnat@gmail.com

Ragavan Sundaran
Electrical Engineering IV
McMaster University
Hamilton, ON
ragavans45@gmail.com

ABSTRACT

The purpose of this project was to create an unmanned ground vehicle which could be controlled using intuitive hand gestures. The vehicle also provides feedback to the user about the the environment in which its in, by way of two sensors (one proximity and one temperature sensor), and a smartphone which provides a live video feed. This vehicle has a myriad of potential applications, including military reconnaissance, field mine detection, mars exploration and search and rescue.

The brain of our vehicle is the Raspberry Pi, which connects wirelessly to a web server. Our main source of input is the leap motion controller which outputs user hand gesture data to the same server mentioned above. This data is then parsed by the Pi and these motion gestures are then transferred to the Arduino, which acts as a motor controller. Interfacing the three main components (Arduino, Raspberry Pi and Leap), allowed us to successfully develop a proof of concept, for an unmanned ground vehicle which can be controlled using basic hand gestures. The vehicle also provides feedback to the user about the environment in which it in, by way of two sensors (one proximity and one temperature sensor), and a smartphone which provides a live video feed. This vehicle has a myriad of potential applications, including military reconnaissance, field mine detection, mars exploration and search and rescue.

I. INTRODUCTION

The objective of our Capstone project was to design an unmanned ground vehicle (UGV). The UGV is controlled via leap motion controller. We decided to use a leap motion controller for the vehicle, because gesture control is more intuitive and natural when compared to a traditional joystick. In order to achieve this task, we modified an already existing remote control car, to integrate an Arduino board, and a Raspberry Pi. An already existing vehicle was chosen, because this enabled the platform to be more durable and robust.

A smart phone was also used to provide the user with real time video feedback. In our first stage of implementation, our main objective was to have the leap motion interfaced with the RC car via Arduino/Raspberry Pi, and have the car correctly responding to user basic movement gestures. A temperature/pressure sensor was then coupled to the system to provide real time feedback of environmental conditions. In the second stage of implementation, we incorporated a smartphone camera in order to provide video feedback so the user has a live video feed to monitor the environment (this also aids in controlling the vehicle).

In this stage a proximity sensor was also added to the rear of the vehicle in order to sense obstacles, when the vehicle is reversing, and override leap motion control if the vehicle is going to collide with an object. In our final stage of implementation, we attached a servo to the camera, so that the user can control the panning motion of the camera using gestures. We also integrated a Graphical user interface in order to the make the whole experience more user friendly.

II. DESIGN METHODOLOGY

The main approach in design for this project was incremental and modular. This approach was adopted because we were interfacing 3 main components (Arduino, Raspberry Pi and Leap Motion). Each of these devices possess their own intricacies and complexities, and combining them together could potentially magnify complications. Another reason why this approach was taken was due to the fact that all group members had little to no experience in dealing with these devices.

So, in each stage of development, small steps were taken in order to ensure that all potential errors could be identified and rectified. The first basic step was to modify and integrate Arduino into the RC car. The second basic step was to communicate between the Raspberry Pi and the Arduino. The third step was to communicate between the Raspberry Pi and the Leap motion.

These are just a few initial steps outlined to show how the approach was modular. Each step was essentially a building block, in which further blocks can be added. This approach was also important, because the outlined deliverables for this project were outlined in stages.

III. ANALYSIS

A. Gesture Control

The leap motion is designed to sense how one naturally moves ones hands and lets a user control the computer in a whole new way. Pointing, waving, reaching, grabbing, and picking something up and moving are all supported motions. It is capable of tracking all 10 fingers up to 1/100th of a millimeter and has a super wide 150 degree field of view and a z-axis for depth. The goal is to use this device to control car movements naturally with hand gestures. Node JS, JavaScript, and Johnny-Five Library was used because of its power to control microcontrollers such as the Arduino in an intuitive manner.

B. Hardware

The hardware components were chosen and designed with careful analysis of intuitive and ease of use manipulation. An RC car was stripped down and the H-Bridge was analyzed to see the signal points of contact. Using the remote control that it came with, it was possible to identify the solder points that allow control of the forward, backward, left, and right motion of the car. A type of controller was chosen to send signals high and low to the motion contact points and this type of controller should be intuitively programed.

The Arduino was chosen as the microcontroller for this job and it was tested by connecting pins from the Arduino to the contact points found on the H-Bridge of the RC car. With some initial tests such as a movement sequence of forward, backward, left, right using native Arduino code, the result of the motions desired were met. The servo to move the pan motion of the smartphone attached to the car was also controlled by the Arduino.

The IR Proximity sensor that handles the disablement of the back motion when an object that is too close is detected was also handled by the Arduino. The results were as desired in earlier assumptions for controlling all hardware.

C. Communication

JavaScript was chosen because of a library named Johnny-Five that allowed simple and quick communication to transmit data to the microcontroller. It was very intuitive and easily adjustable. The motivation to design the project with JavaScript in mind came from the growing technology in JavaScript and Node JS that allowed easy communication between the Leap Motion and the Raspberry Pi that controlled the Arduino. The Leap Motion was initially tested by using Java, however due to the easier method of capturing.

IV. IMPLEMENTATIONS

A. High Level Concept

The user will use the Leap Motion as the controller for the remote control car. The idea is for the car to be controlled wirelessly through WiFi. The RC car was modified and equipped with an Arduino Microcontroller and a Raspberry Pi to handle the communication and control between the Laptop and the vehicle. Sensor attachments were connected to the Arduino and the Raspberry Pi to feedback and provide information to the user. The main idea of this project is visualized below.

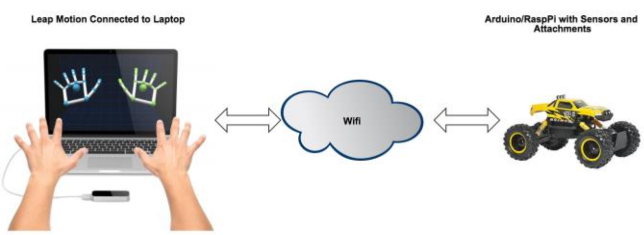
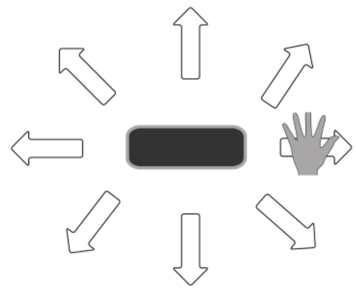
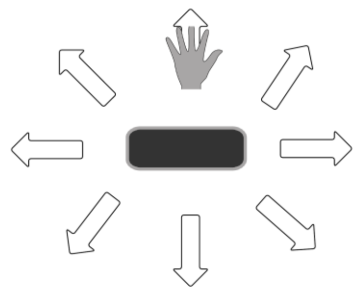
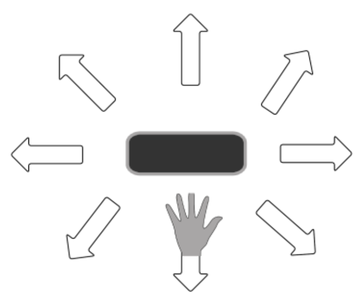


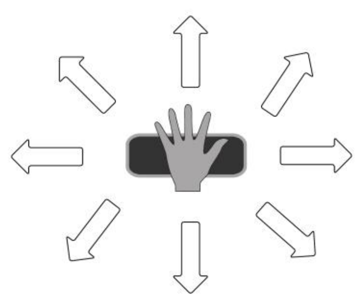
Figure 1- Concept Usage

The hand gesture motions represent the movements that are supported for the application that has been written. This creates a natural motion and movement for the user to interact with in an intuitive manner. Hand in the middle is a rest zone and the car will not movie, the forward, backward, left right, north east, and etc. all move accordingly to where the hand is moving.









B. Hardware Design

The hardware components for this project can be broken down into two areas, the users' stationary control and the RC Car. The users' stationary control only consists of a laptop that is running all the necessary software and a Leap Motion controller to provide a 3D motion and gesture control input to the laptop.

Figure 2- Control Station

The RC Car is the other area of hardware components and can be broken down even further to show the implementation and design of control. The RC car has been broken down and modified to gain access and control with the following diagram.

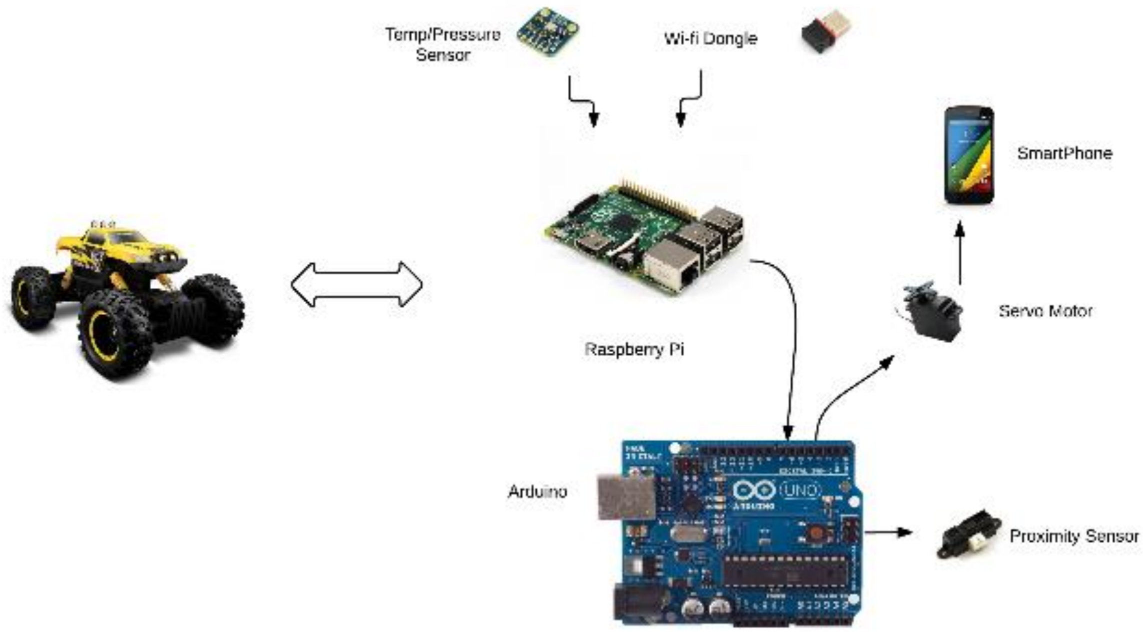
In order to communicate with the vehicle, a Raspberry Pi with a WiFi dongle has been used. This Raspberry Pi is also necessary for attachments such as a temperature sensor. For live video feed that will be transmitted, a smartphone has been used and connected to the same network. The Arduino is not good enough for image processing or transmission, so the smartphone will be used for this application. The Raspberry Pi will also be able to send control signals to the Arduino and receive raw data from the Arduino's proximity sensor attachments.

The Arduino will be used to control the RC car's motors and also be connected to sensor's that will feed data back to the Raspberry Pi. In order to communicate and interface with the vehicle motors, some experimenting and analysis has been done on the vehicle.

The objective is to tear apart the RC car and analyze and test the existing circuitry of the car in order to find out how the servo's connected to the board are interfaced together. Experimentation was necessary with the existing PCB on the RC car to discover the different types of control signals such as left turn, right turn, forward, and backward movements. A servo has been attached to the front of the car where the smartphone is held. There has been a gesture that was implemented to control the degree and angles of the pan angles of the camera. Jumper wires can then be connected to the servo control points on the PCB and can be connected to the Arduino. The vehicle itself is powered by 6 AA batteries (battery compartment built in). A rechargeable 9V battery powers the Arduino board. The Raspberry Pi is powered by a Portable Rechargeable USB Power supply. This power supply has a power capacity of 3200mAh, with an output of 5.25 V at 2A.

C. Software Design

The Leap Motion Controller's software can broadcast the information for tracking in the format of JSON using Web Socket output. The Leap Motion API can communicate with web browser-based applications in languages such as JavaScript. Because of this, the decision was made to use Leap Motion's supported framework LeapJS to communicate between the Raspberry Pi and the Laptop. The Raspberry Pi will also be running a web application using the Johnny-Five library which is an open source programming framework for JavaScript.



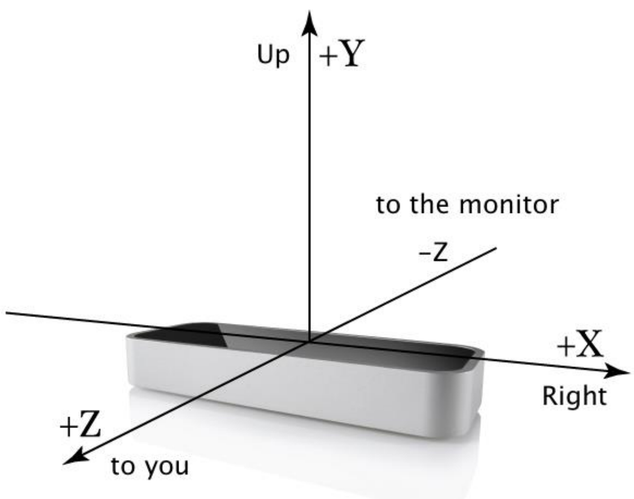


Figure 4 - Leap Motion

Figure 5- Leap Motion Break Down

V. SUSTAINABILITY AND SOCIAL IMPACT

Social: If implemented on a large scale, our vehicle system will greatly benefit society, as it would allow people to remotely monitor or explore dangerous environments. The remote operation of the UGV ensures that operators do not need to be directly in harm's way. This will also allow humans to explore regions that were too hazardous to explore in the past. The UGV has temperature and pressure sensors integrated which feeds data back to the user allowing exploration of different atmospheres. Another application that was considered for our project was a rescue bot. A rescue bot would benefit society by helping to save a person or people in difficult situations such as being lost and in need of directions.

Economic: This project would cost a lot of money to be implemented on a large scale, as powerful motors and a robust durable chassis would be necessary. If, let's say, that the military, government, or NASA decided to use the UGV, then communication systems will need to be more reliable to ensure that connection is never lost to the UGV. Therefore, the cost of implementing this can easily increase depending on the application its used for. One way to counter these high costs would be to mass produce the UGV and save money by negotiating the price of components in large scale purchases.

Environmental: The UGV requires use of batteries and PCB circuit boards, which will be detrimental to the environment at the end of their life cycles. One way to rectify the battery issue would be to use rechargeable batteries where possible. Another solution would be to not use batteries, but to install solar-powered cells that will use solar energy to power the UGV.

VI. TEAM ORGANIZATION

Tentative Task Assignment (decided in beginning stages of project):

Task	Assignees
Hardware	Nathan, Nissh
Software (RaspPi/Arduino)	John, Raga
Software (Leap Motion)	All
Testing	All

Final Task Assignment:

Task	Assignees
Hardware	Nathan Sinnathurai, Nisshanth Navaratnam
Software (RaspPi/Arduino)	John Octubre, Ragavan Sundaran
Software (Leap Motion gesture for movement)	Nathan Sinnathurai, Nisshanth Navaratnam
Software (Leap Motion gesture for headlight)	John Octubre, Nisshanth Navaratnam
Software (Leap Motion gesture for servo control)	John Octubre, Ragavan Sundaran
Video Feed	John Octubre, Nisshanth Navaratnam
GUI implementaion	John Octubre, Nathan Sinnathurai
Sensor Integration (ultrasonic, tempature, and pressure)	Ragavan Sundaran, Nathan Sinnathurai
Testing	All

VII. CONCLUSIONS

Our group was overall successful in achieving the goals of our capstone project. Initially, our main goal was to build an unmanned ground vehicle (UGV) with motion gesture based remote control. However as we progressed, we added more features to our UGV such as video feedback to the user, temperature and pressure sensor, and etc.

These were the criterion for our silver and gold deliverables, and we were successfully able to achieve them. Primarily due to time constraints, the one criteria we were unable to incorporate into the UGV was a graphical user interface to make controlling the UGV more convenient.

There are many types of applications that our project may be used for (as mentioned in the "sustainability and social impact" section of this report). Although the concept behind our UGV is not yet ready for the market, whatever we were able to accomplish is proof that using motion gesture to control a vehicle is attainable. For future development, one suggestion is to add speed control. A lack of speed control makes it difficult for the use to control vehicular motion effectively.

VIII. REFERENCES

BMP180 Barometric Pressure/Temperature/Altitude Sensor- 5V ready ID: 1603 - Adafruit Industries, Unique & fun DIY electronics and kits. (n.d.). Retrieved from http://www.adafruit.com/product/1603

leapmotion/leapjs GitHub. (n.d.). Retrieved from https://github.com/leapmotion/leapjs

rwaldron/johnny-five GitHub. (n.d.). Retrieved from https://github.com/rwaldron/johnny-five/

websockets/ws GitHub. (n.d.). Retrieved from https://github.com/websockets/ws