TEAM UNTOUCHABLE

Project Proposal

Non-Touch Gesture Control System

Team Members:
Vincente Nguyen
Swat Vongsay Ramil J Tan
Jonathan Lintag Arjay Villalon
Emil Kako Waseem Akbar

Submitted to: John Kennedy, Richard Lane and Hugh Molesworth

October 11, 2016

Sponsored by:

MORRIS CERULLO WORLD EVANGELISM

SAN DIEGO STATE UNIVERSITY

TABLE OF CONTENTS

INTRODUCTION

ABSTRACT

Gesture control technology is already used in many devices today like the Xbox Kinect and Microsoft Surface. However, these devices still face key issues that limit the user experience. The Microsoft Surface and devices similar to it are limited to a very short range of a couple of inches, and although the Kinect provides the range, it still has delays in response time which can hinder the user experience. The key goal of this project is to design a device that utilizes gesture control technology to recognize users and their gestures at ranges up to two feet with minimal delays in response time.

PROJECT DESCRIPTION

The purpose of this project is to create a device that gives the user the ability to communicate with an application like Google Earth through the use of various gestures. These various gestures or hand motions will allow the user to completely control the Google Earth globe without any touch of the display. The design is based on optical, infrared, and ultrasonic sensing technologies. These sensors will be used to clearly capture the motion of a hand with minimal delays in response time and minimal background noise interference.

SYSTEM DESCRIPTION

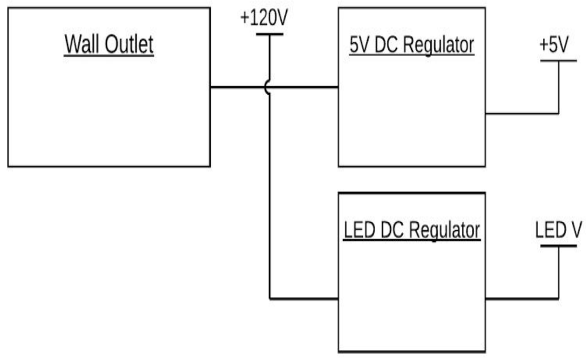
Ultrasonic sensors will be used to capture the distance of a user to the device. Once the user falls into the range of the ultrasonic sensors, an array of powerful infrared LEDs will be turned on to illuminate the hand. These LEDs will be powered by an external power supply connected to a wall socket. The intensity of these light beams provided by the infrared LEDs will allow the hand to appear more illuminated than any other object around it. After the hand has been illuminated, optical sensing will then be used on the Raspberry Pi 3 camera to capture the illumination of this hand. This camera will have a band pass filter that will only pass the infrared band of the light spectrum. Once the camera captures the illumination, image processing is done by the Raspberry Pi 3 by implementing various motion capture and noise removal algorithms to clearly capture the motion of the hand without any delays in response. The device itself will be a long rectangular box with a tinted screen that will house all of these components. This device will be placed at the top of a 40-inch Display.

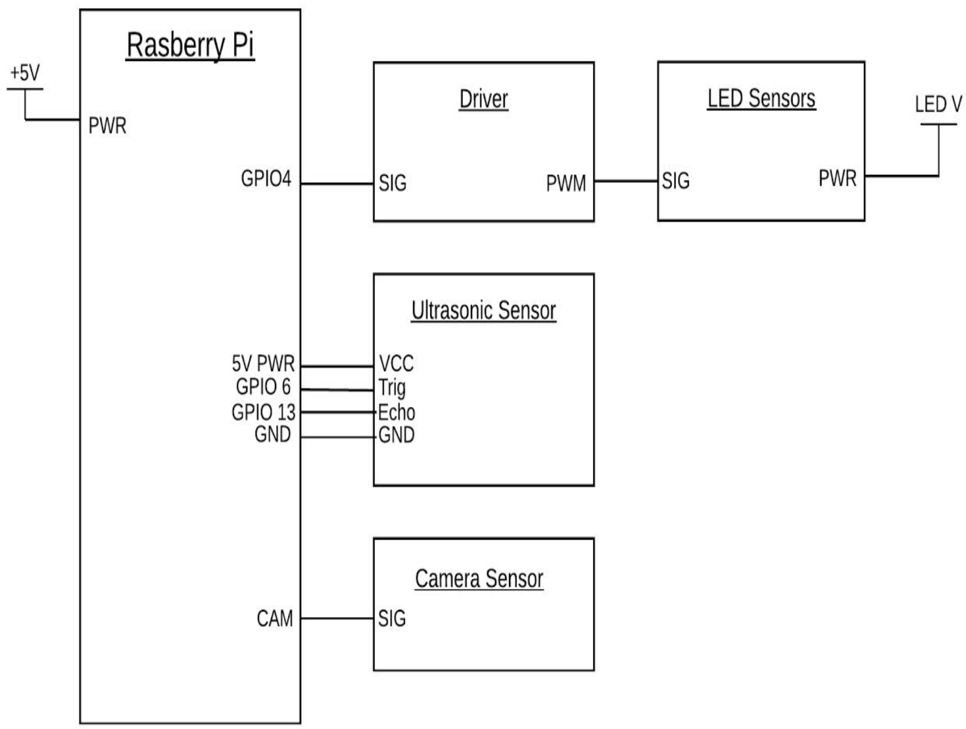
DETAILED USER EXPERIENCE

A user will begin their experience by approaching a large 40 inch display running Google Earth. Once the user falls into the range of the ultrasonic proximity sensors, the display interface will ask the user to calibrate the system by placing their hand in a specific region in front of the screen. Then the user will be shown all of the various gestures in a quick tutorial. The tutorial will interface with the user with the following commands. First a swipe left will be shown to spin the globe to the left. A swipe right to spin the globe to the right. Upward and downward motions to spin the globe up and down. Inward and outward motions to allow the user to zoom in and out of the globe. Once the tutorial is complete the user will then be able to completely control Google Earth on their own with the gestures provided by the tutorial.

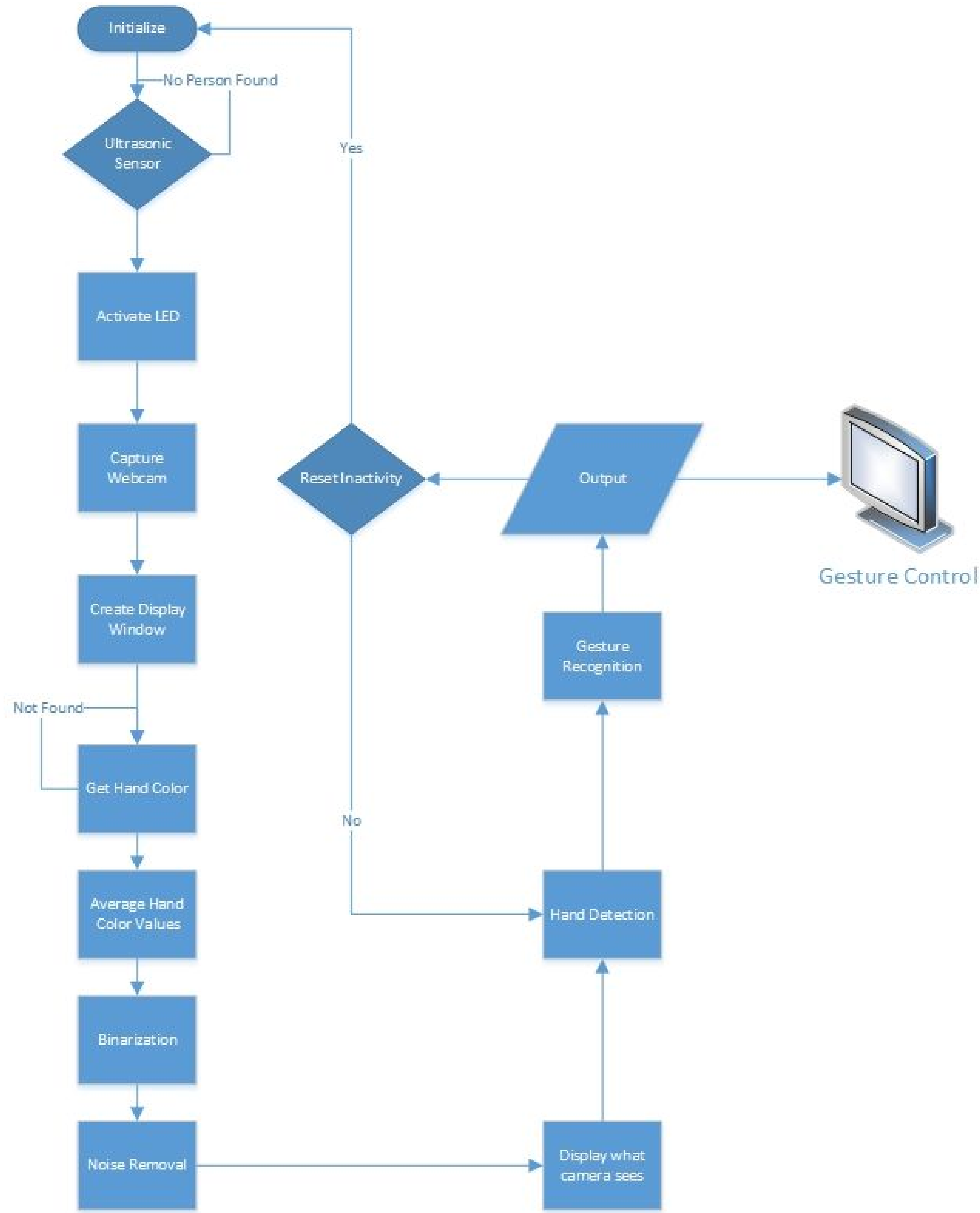
DESIGN

HARDWARE BLOCK DIAGRAM



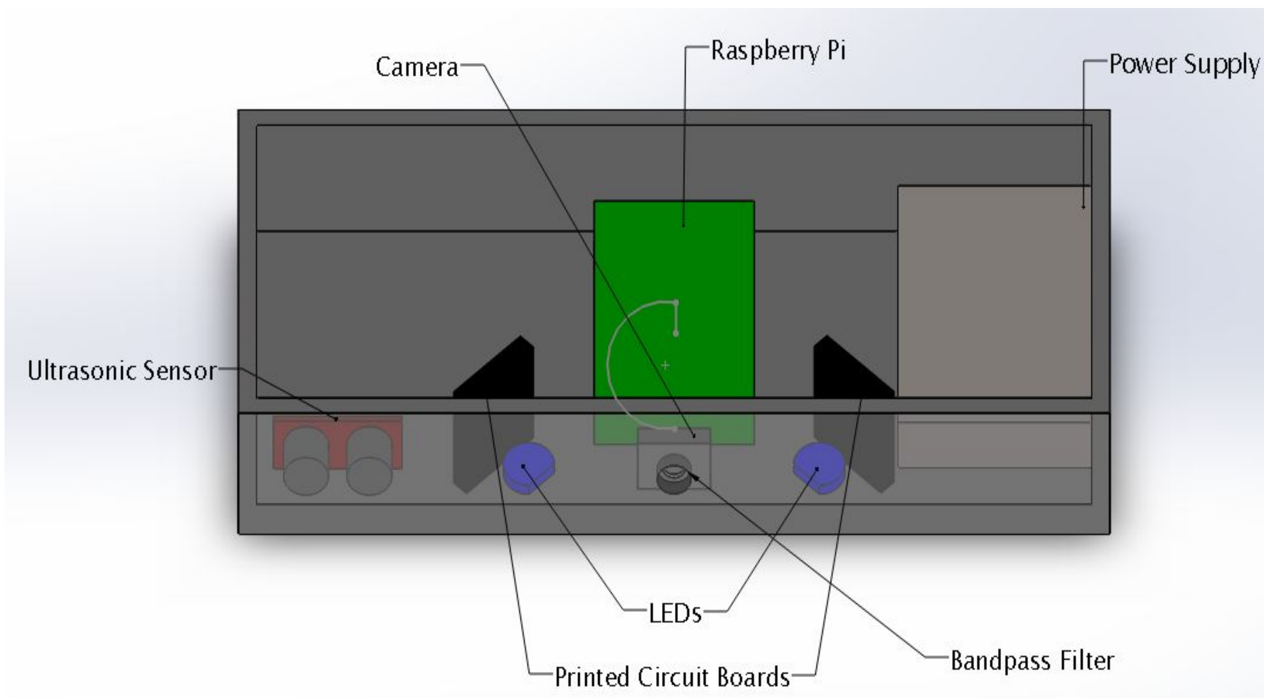


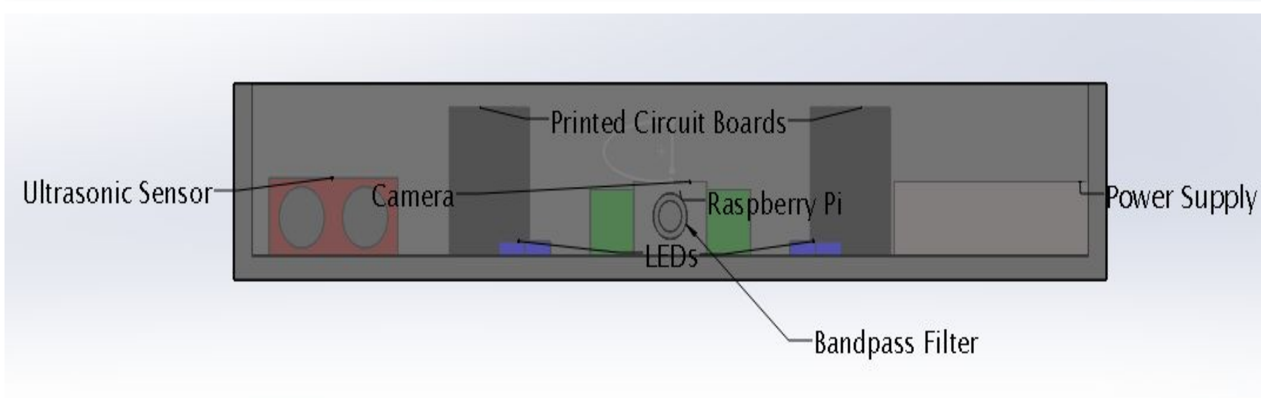
SOFTWARE FLOW DIAGRAM



The overall software interfaces with two main components, the camera and infrared LEDs. These all rely on the ultrasonic sensors signal to alert the rest of the components whether to turn on or remain idle. When the system activates we first start off with camera looking for a hand for color and light calibration purposes. These values are compared between frames and pixels allowing us to achieve software-based noise removal. In addition to this, the noise removal algorithm controls the rate of our LED illumination which allows us to have hardware-based noise reduction when a bandpass filter is placed in front of the camera. Incorporating both noise removal techniques, we can more easily detect just the hand allowing us to achieve seamless gesture recognition.

SOLIDWORKS RENDERINGS





PERFORMANCE SPECIFICATIONS

Raspberry Pi 3:

The Pi 3 will run code in C++ to do image recognition that will recognize hand-motion gestures with noise removal algorithm. We chose the Raspberry Pi 3 because it is an affordable microcontroller with powerful system specs. The Raspberry Pi 3 family also has two different camera modules that work seamlessly with the microcontroller. The Raspberry Pi 3 runs at 5V and its GPIO pins provide proper output voltage (3.3 or 5 V).

Raspberry Pi Camera:

The camera will capture images in a certain proximity to be processed on the Raspberry Pi 3. The Pi camera module provides us with good images that we can work with at an affordable price. The camera module also provides a decent resolution and comes without an IR filter allowing the camera to detect IR light.

IR LED:

The LEDs will illuminate the the focus point (of person) for the camera to focus on and eliminate background distractions (e.g. other people, people in motion, etc.). The LEDs will most likely be connected in series and will have a 3.3 forward voltage drop across each LED. We may also use a separate microcontroller to act as a fail safe switch to the LEDs by using pulse width modulation.

Power Supply:

The Raspberry Pi 3 will not be able to supply enough voltage for the IR LEDs so they will require an external power supply. We will have to use an external power supply connected to a wall outlet to provide the proper voltages. We will use a DC voltage regulator to give us around 12V-30V depending on the optimal number of LEDs.

Ultrasonic Sensors:

The ultrasonic sensors will be triggering often enough to determine if a person is near the gesture control system. This will provide us with a way to determine if someone is in range to determine if the other sensors should turn on. The sensor will have a threshold range to only turn on the other sensors when a person is in a set range. The Ultrasonic sensors will run at 5V and provide us a range of over 3-5 feet.

TESTING AND VERIFICATION

TESTING PROCEDURES

Software Requirements:

1. Camera control
2. Ultrasonic sensor input
3. Hand detection
4. LED illumination switch
5. Gesture recognition
6. User interface

The most important component of our project is the algorithm that controls hand detection and translates that into gesture control. We plan to test our code in chunks as they are created and in the end create an overall algorithm test. The brain of our code is the useage of the OpenCV (Open Computer Vision) library which gives us access to a lot of powerful computer vision related functions. This allows us to capture specific colors and detect the differences between colors that the camera sees effectively giving us hand recognition and background detection. After we successfully detect the hand we can create gesture recognition and control based on the data that we get from the hand detection algorithm.

To test our software we must test that our algorithm detects the hand successfully. The OpenCV library allows us to draw an outline of the hand and fingers using both a contour and convex algorithm. If the algorithm successfully sees the hand, it will project an outline onto our display window. If there is a lot of background noise, our hand outline will be spastic and will draw contours on objects that we do not want. We've observed that this often happens to objects that are similar in color to the hand.

This problem is alleviated when we introduce a noise removal algorithm (in our case, a background subtraction algorithm and binarization). The way the algorithm works is the software will first ask the user to calibrate their hand color by placing it in front of the camera so that it can take multiple points of the hand and average the values. It will then compare the frames that are captured by the camera and subtract out colors that are not similar to the background. This process will remove the majority of the background and convert our image to black and white. To test this we can visually inspect the output, which should be a black and white image that should only display the hand illuminated as white with everything else as black. After some testing we can see that the face is still illuminated and occasionally confuses the face as the hand because they are similar in color.

Afterwards we have to test if our microcontroller will fire our LEDs at very specific times so we can introduce a physical noise removal system. We can test this by making LEDs blink at specific times using the Raspberry Pi 3's timer and interrupt service. Afterwards we can test our LEDs to make them fire at a specific rate when certain conditions are met.

Finally, we want to maintain longevity of the system by not having the system on unnecessarily. We can do this by using an ultrasonic sensor to detect if a person is in front of our device or not, allowing the system to activate only when a person is detected. We will also have an inactivity timer so the system can shut itself off to prevent wasteful power useage and needless IR illumination.

Hardware Requirements:

1. LED sensors
2. Ultrasonic sensor
3. Camera

When testing the hardware, our team will be testing every part individually in order to become familiar with the components and to make sure each component meets our standards.

In order to test the infrared LEDs we will need to have the Raspberry Pi 3 connected to an NPN transistor, which will act like a switch. We will use a DC power supply to provide 3.3 volts with a constant current of 2 amps. The Raspberry Pi 3 will drive a voltage to the gate that will quickly turn on the LEDs and then turn them back off. This will repeat for a constant amount of time.

The Ultrasonic sensor will be tested using the GPIO pins from the Raspberry Pi 3. The sensor will be powered by a 5V pin on the Raspberry Pi 3 while the trigger pulse input connection and the echo pins on the sensor will be connected to GPIO pins. The Raspberry Pi 3 will send an input signal to trig, then the sensor will wait until it receives a pulse back from someone standing in its range. The sensor will then return data on how far the person is standing. We want the user to be standing about 3-5 feet away before the system turns on.

The Raspberry Pi 3 camera is designed to work very seamlessly with the Pi so testing it should be relatively simple. As long as the camera is able to capture images that are adequate for our system, we will be able to move our focus to the rest of the design. Of course, we would like to apply a bandpass filter to the camera that only accepts wavelengths of the specific color of our LEDs. This will help reduce background noise even before our software process our images.

BENCHMARKS

1. Our device will be capable of detecting if a person is nearby and perform hand detection.
2. The bandpass filter and infrared LEDs will only allow a specific light wavelength to be seen to achieve noise reduction.
3. The algorithm will work in conjunction with the hardware noise reduction to also get a software based noise reduction.
4. We will be able to detect different positions of the hand and count how many fingers are up.
5. Gesture control will be achieved based on hand position and movement.

PROJECT MANAGEMENT

PROJECT PLAN

I. Non-Technical Responsibilities
   1. Vincente Nguyen - Project Manager
   2. Swat Vongsay - Repo and Doc Manager
   3. Ramil J. Tan - Webmaster
   4. Jonathan Lintag - Parts Manager
   5. Arjay Villalon - PowerPoint Manager
   6. Emil Kako - Editor
   7. Waseem Akbar - Secretary

II. Technical Responsibilities

   We have divided our team up into two smaller groups to allow us to concentrate on working on the software and the hardware of the system.

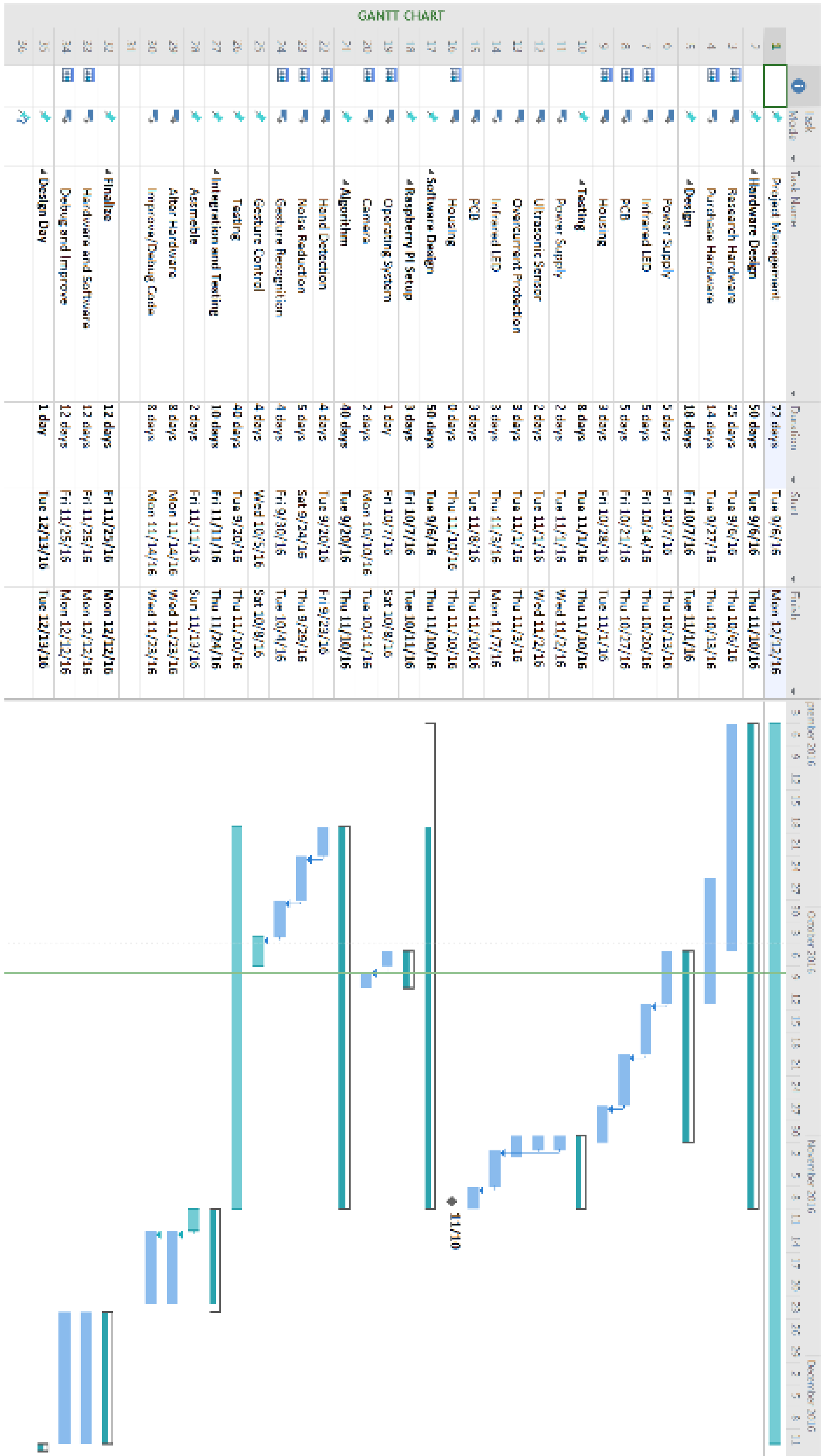
Hardware Team

The hardware team is responsible for selecting the LED sensors, the bandpass filter, and the ultrasonic sensors. They are also in charge of designing the PCB board, power supply, and the enclosure.

Software Team

The software team is responsible for writing the code that interacts with the Raspberry Pi 3. This includes the code that will drive the LEDs and code that will allow the Raspberry Pi 3 camera module to do gesture control.

III. Gantt Chart



ID	Task Name	Duration	Start	Finish
1	Project Management	72 days	Tue 9/6/16	Mon 12/12/16
2	Hardware Design	50 days	Tue 9/6/16	Thu 11/10/16
3	Research Hardware	25 days	Tue 9/6/16	Thu 10/6/16
4	Purchase Hardware	14 days	Tue 9/27/16	Thu 10/13/16
5	Design	18 days	Fri 10/7/16	Tue 11/1/16
6	Power Supply	5 days	Fri 10/7/16	Thu 10/13/16
7	Infrared LED	5 days	Fri 10/14/16	Thu 10/20/16
8	PCB	5 days	Fri 10/21/16	Thu 10/27/16
9	Housing	3 days	Fri 10/28/16	Tue 11/1/16
10	Testing	8 days	Tue 11/1/16	Thu 11/10/16
11	Power Supply	2 days	Tue 11/1/16	Wed 11/2/16
12	Ultrasonic Sensor	2 days	Tue 11/1/16	Wed 11/2/16
13	Overcurrent Protection	3 days	Tue 11/1/16	Thu 11/3/16
14	Infrared LED	3 days	Thu 11/3/16	Mon 11/7/16
15	PCB	3 days	Tue 11/8/16	Thu 11/10/16
16	Housing	0 days	Thu 11/10/16	Thu 11/10/16
17	Software Design	50 days	Tue 9/6/16	Thu 11/10/16
18	Raspberry Pi Setup	3 days	Fri 10/7/16	Tue 10/11/16
19	Operating System	1 day	Fri 10/7/16	Sat 10/8/16
20	Camera	2 days	Mon 10/10/16	Tue 10/11/16
21	Algorithm	40 days	Tue 9/20/16	Thu 11/10/16
22	Hand Detection	4 days	Tue 9/20/16	Fri 9/23/16
23	Noise Reduction	5 days	Sat 9/24/16	Thu 9/29/16
24	Gesture Recognition	4 days	Fri 9/30/16	Tue 10/4/16
25	Gesture Control	4 days	Wed 10/5/16	Sat 10/8/16
26	Testing	40 days	Tue 9/20/16	Thu 11/10/16
27	Integration and Testing	10 days	Fri 11/11/16	Thu 11/24/16
28	Assemble	2 days	Fri 11/11/16	Sun 11/13/16
29	Alter Hardware	8 days	Mon 11/14/16	Wed 11/23/16
30	Improve/Debug Code	8 days	Mon 11/14/16	Wed 11/23/16
32	Finalize	12 days	Fri 11/25/16	Mon 12/12/16
33	Hardware and Software	12 days	Fri 11/25/16	Mon 12/12/16
34	Debug and Improve	12 days	Fri 11/25/16	Mon 12/12/16
35	Design Day	1 day	Tue 12/13/16	Tue 12/13/16

MILESTONES

10/25/16: Perform gestures and have the Raspberry PI 3's camera detect and perform gesture control.

11/10/16: Finish all hardware design and software algorithms.

11/24/16: Fully integrate both software algorithms and hardware into a single project.

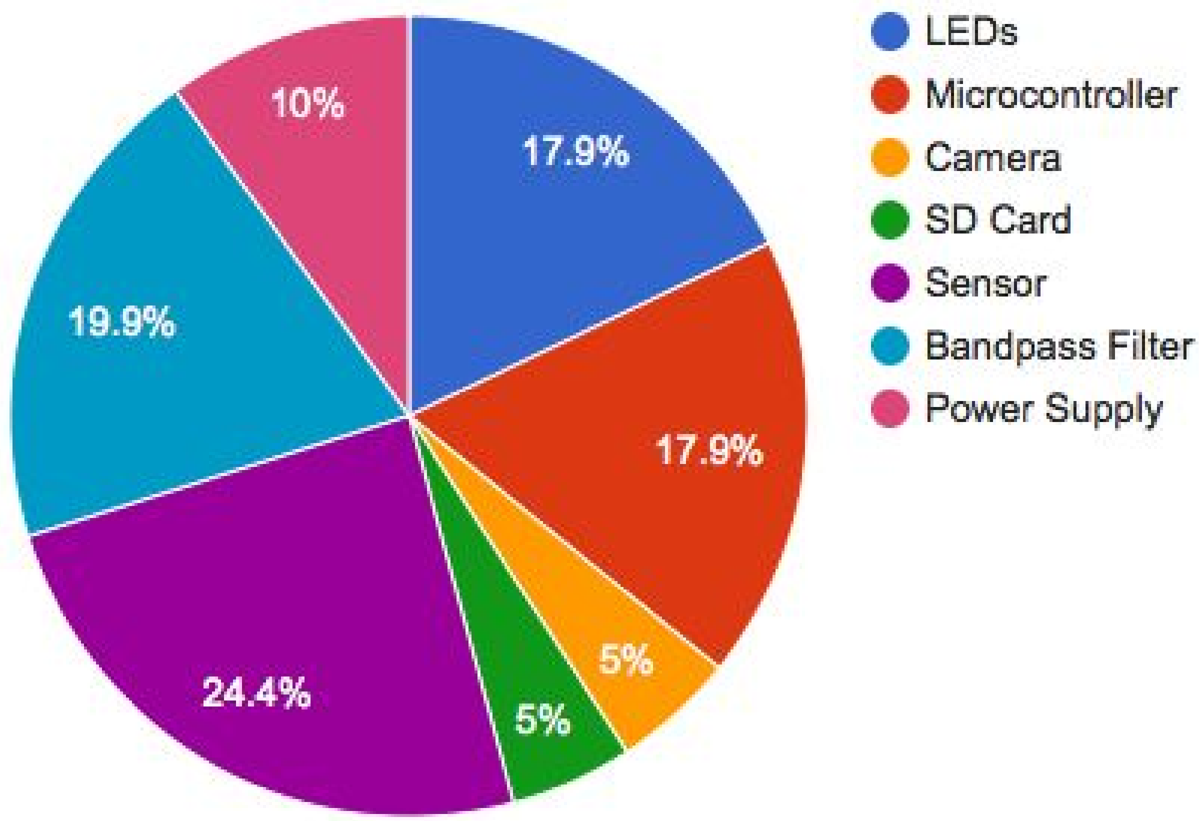
12/10/16: Project will be completed and be able to perform gesture control.

12/13/16: Perform at design day.

BUDGET

COST ANALYSIS

We have a funding of $1,500 to meet the requirements for our project. We expect to use a majority of the funds on the microcontroller, bandpass filter, and LEDs. We have used some of the funding towards testing multiple sensors in order to find the best possible solution to our project. We have allocated some of our budget towards possible replacements parts and new components that may need to be added to our system in the future.



TEAM UNTOUCHABLE

Our Mission:

Develop a Non-Touch Gesture Control System

Our control system uses optical, infrared and ultrasonic sensors to accurately recognize and track a user's hand gestures.

Team Members:

Vincente Nguyen

Waseem Akbar   Emil Kako

Jonathan Lintag   Ramil J Tan

Arjay Villalon   Swat Vongsay

Sponsored by:

MORRIS CERULLO WORLD EVANGELISM

SAN DIEGO STATE UNIVERSITY