Gesture Controlled Robot

ECE 445 Design Document
Arvind Vijaykumar, Qinlun Luan and Bofan Yang
Group 30
TA: David Hanley
3/7 19
1 Introduction

1.1 Objective

Wireless communication systems form the backbone of modern-day human-robot interaction. Namely, wireless remote control methodologies including IR, RF, and network-based technologies such as WiFi and Bluetooth facilitate communication between a client and a robot such that the robot can successfully actualize its desired functionality. The main advantage they possess over wired control is that they provide a much broader range for the robot to interact with its environment. External peripherals (game controllers, smartphones, etc.) are usually required in order to wirelessly transmit data to the robot, however, these wireless physical peripherals still withhold constraints such as the difficulty of using for people with physical disabilities and easy to wear issue for delicate electronic parts. In our project we would like to try and free user of these physical peripherals by introducing a Touchless UI technology based on artificial intelligence, in particular, the use of machine learning methodologies like CNN (Convoluted Neural Networks) for gesture recognition combined with wireless technology such as Bluetooth to form a gesture-based control scheme, could form the backbone of a potential solution to these problems, as our solution only requires one hand to operate and requires no hardware control device, this means people with only one arm can also operate the robot control by our gesture control system, and users no longer need to worry about control device running out of battery or getting less sensitive and accurate due to extensive using. Moreover, our primary objective in this course is to combine these technologies to explore a new and innovative method of robotic control.

We propose the implementation of a robot that can be remotely controlled via hand gestures to serve as a proof-of-concept for a non-peripheral, machine learning based robotics control scheme. We intend on accomplishing this through a software application that can take in a hand gesture displayed to the computer’s camera, identify it, generate a command corresponding to the gesture to the robot, and transmit that command over Bluetooth to control the robot. Our robot’s functionality lies entirely within the domain of motion control; therefore, the input hand gestures will each correspond with a direction of motion for the robot as well as a gesture to cease movement. We intend to utilize a convolutional neural network (CNN) based architecture for our hand gesture recognition system, the details of which will be elucidated upon in the following sections.

1.2 Background
As highlighted above, interfaces used for human-robot interaction often requires using an external device. According to research done by Fujitsu Laboratories\textsuperscript{2}, a touchless user interface like this hand-gesture based control scheme is an effective means by which users can control robots, since hand gestures don’t require external devices and in general are natural control mechanisms that serve as an instinctual means of communication. In addition, there exists a myriad of different gestures we have at our disposal that can be formed and mapped to our robot’s individual functions. In general, the existing research in touchless, specifically, hand gesture recognition based robot control systems often emphasizes on the utilization of external sensors (Leap Motion\textsuperscript{3}, for example), the accuracy of which is often dependent on the distance between the user and the robot, which is not particularly optimal as a remote control mechanism. In addition, the bulk of the processing is done on the robot’s microcontroller itself, meaning that based on the application, the processing unit can be quite costly. In some cases, the systems require the camera module to be mounted to the robot itself for computation, which, again, isn’t optimal for remote control, and the number of gestures utilized by the robot for movement is limited to those that are relatively easier to decode via methods in computer vision\textsuperscript{4}. More complex computer vision and patterns recognition based methods like template matching and the use of feature vectors are also commonly applied to hand gesture recognition robot control systems\textsuperscript{5}. Another main problem of computer vision used in robotic control is the accuracy of the overall motion and according to a smart robotic arm project conducted in Iskenderun Technical University, the use of convolutional neural networks has an 87.8\% overall accuracy\textsuperscript{6}. Thus, the convolutional neural network based model has the potential to mitigate and improve upon the issues seen in existing designs and research.

1.3 High Level Requirements

- The vehicle can slowly and smoothly move in the correct direction at a translational velocity of about 10 cm/s based on the corresponding hand gesture, with the system identifying gestures with at least 70\% accuracy.
- Corresponding robot motion must occur within 3 seconds after the user has inputted a valid hand gesture.
- Capable of operating in a room or outdoors with a light intensity no smaller than 1000 Lux and the light source must contain a full light spectrum (natural would be ideal), for a maximum range of no less than 9 meters (ideally with no wall or any other barriers in between the robot and the user)
2. Design

Block Diagram:

- Image input Module
  - Hand Gestures
  - Web Camera

- Image Analysis Module
  - Camera Driver
  - Image Processing Module

- Command Unit
  - Communication Module
  - Classifier Module

- Control Unit
  - Bluetooth Module
  - Microcontroller

- Power Supply
  - 9V Battery
  - Battery Charger
  - Voltage Divider

- Motion Control Unit
  - Motors
  - Motor Control Module

Software Conceptual Chart:

These are the software designs of the project since they required programming and are located in the computer.

The microcontroller is special because it is located on the vehicle but it requires programming in order to fully function.

These are the hardware designs of the project since they are controlled by electrical signals and are located in the vehicle.

- Yellow line: software interconnection
- Blue line: RF signal
- Red line: electrical signal
- Pink line: power line
- Gray line: lines to abstractly stand for how gestures are captured by camera and how the created images connect to the software.
Gesture

Activated Camera

Set Parameter

Stream of data with size 320*240*3

Threshold

Reduce Noise

Stream of data with size 320*240*3

Classifier Module

1st layer 128 fully-connected neurons
2nd layer 8-node softmax layer
Optimization: adam
Loss function: sparse categorical
Metrics: accuracy function

Guess Index of designed gesture

Communication Module

Manipulate Bluetooth in computer to communicate with the Bluetooth component in the vehicle

Signal representing the index
Physical Design:
Block Design

Functional Overview:

- Image Input & Analysis
  - **Hand Gestures:** This block serves as input to our system. The user can input a gesture from a predetermined set of hand gestures that can be used to control our robot. In addition, we require that only the user’s hand could appear before the camera to simplify the design because the skin colors of hands and other parts like face are close to each other and our project aim specifically at classifying hand gestures. It is better if we can use a whiteboard as a background to further improve the performance.
  - The module is a demonstration of input for images and it contains no electrical components so it is not a real module and we skip the R&V part.
  - List of Gesture and corresponding meaning:
    - Move forward:
      ![Move forward gesture image]
    - Stop:
      ![Stop gesture image]
    - Turn left:
- Turn right:

- Move backward:

- Turn left while moving backward:
Turn right while moving backward:

Camera & Camera Driver: The system must acquire a video feed in which the client is able to display hand gestures for processing. For that purpose, the application would require access to the web camera of the user’s computer. The video requires a minimum resolution of 320×240 and will run at 30 FPS which is what we believe the balance between preserving the characteristics of the hand and saving the storage memory because we want the control the movement of the vehicle for at least 10 minutes and 18,000 JPEG images are created during this process. We want the images to contain enough feathers which determine the performance of the built CNN and to save storage at the same time.

| Requirement 1: The size of the images should be 320*240. | Verification: We will verify by the OpenCV built-in instruction frame.shape() to examine the size of each image. |
Requirements:

| Requirement 2: The camera takes in 30 frames (30 images) per second. | Verification: We will verify the frame rate by the instruction call get(CAP_PROP_FPS) to achieve the exact number of frame rate. |

- **Image Processing Module:** This module is required to convert the video with gestures into a stream of pictures by reading and resizing the image, storing one image per frame, removing noises, and creating smooth segmentation. Specifically, light and background invariance are required in order to separate the hand from its environment, regardless of the environment in question. Thresholding would be also be required such that the application is also skin-complexion invariant. Finally, the hand region would be detected and the graphical data such as contour required for the CNN to identify and distinguish the hand gesture would be obtained through various topological and geometric transformations including contour extraction as well as detecting convexity and defects. The stored image along with features would be sent to the computer vision module as input and leverages to complete the classification.

| Requirement: The contour, the number of fingers, the convexity, and the defects should match the gesture inputted by the user. | Verification: By design of our and the library we use to process the image, the contour and vertices would be shown in real time on the hand gesture user made and be displayed on our APP’s user interface.
and we observe whether there are some parts of hand disappear in the output or other parts show up where they should be. Specifically, we will use cv2.imshow() function to demonstrate each frame of the images and a group member compare whether the number of fingers, the convexity, and the defects match the designed gestures listed above. The contour examination is highly subjective so it is used as an optional test depending on the conditions. |
• Command Unit
  ○ Accuracy (true positive rate): Before we start, we first want to define what we call accuracy here, since it is the most important criteria to judge whether the algorithm is successful or not. As mentioned above, we classify each image per frame and 30 frames per second. Thus, the accuracy is calculated by the number of images with correct classification over 30 and the correct rate should be at least 70%.
  ○ Classifier Module: This module determines the hand gesture based on the input data from the previous module. The input images could be divided into two data set: the train set and the test set.
    ■ Training Set: Specifically, we would first record a one-minute video of the designed gestures with a web camera and manually classify these images to make sure they contain exactly what we want. Then the images with features gained from image-processing module would be used to train set for the neural network to learn from.
    ■ Test Set: After the train is completed, we would use the previous modules to get continuous real-time images with features and the neural network would classify the images based on input data.
    ■ Model: The model contains two layers: the 1st layer 1280 fully-connected neurons, and the 2nd layer 8-node soft-max layer. We make the design based on the size of the image (320x240) and the 8-node soft-max layer is used to classify 8 types of gestures. (Move forward/backward, turn left/right, turn left/right while moving backward, stay and unknown).
    ■ Optimization: adam which is the most commonly used in recent designs.
    ■ Loss function: sparse categorical. This function is used because it is designed for the neuron network with exclusive classes, i.e., each gesture can only have one accurate index.
    ■ Metrics: accuracy function.
    ■ Once a gesture is identified at the acceptable degree of accuracy (70%), the next step would be to map it to the corresponding direction of movement. We intend to have seven gestures to correspond to seven movement functions: moving straight, turning left, turning right, stopping, moving straight in reverse, reversing left, and reversing right.

| Requirement 1: Accuracy must be at least 70%. | Verification 1: One of our group members would hold his hand and pose one of the |
seven designed hand gestures before the camera for 5s to get 150 test images. We require at least 105 of them should be classified correctly. A helper UI function will be constructed and it takes in an index and return the accuracy. Specifically, it halts the whole program and set the variables called accuracy and times to 0 until we type in the index of gesture we aim to show and pose the gesture before the camera at the same time. Each time an image is classified, the helper function adds 1 to the variable times and compares the result with the input index; if the two value are the same, the variable accuracy is added by 1/150, else nothing changes with accuracy. When the variable times is equal to 150, return accuracy and suspends the classifier program. The function would only be used for test purpose, and it needs to be disabled when combined with other modules and used in the whole project. We will run this function for at least 6 times and we need to make sure all of them have the accuracy higher than 70%.

Requirement 2: All classifications of the 30 images in the same second need to be done within 1 second. Specifically, each time

| Verification 2: At the same time of using the helper function above, a physical timer (timer function of iPhone) would be set to 5s and the |
after the five-second gesture demo, we allow at most 5 seconds to get the number of accurately classified images.

helper function must return the output within the time limit.

○ **Communication Module:** At a high level, this module is reminiscent of a switch box. The module receives a signal informing it that a valid gesture has been identified by the CNN classifier, and based on that gesture, the module sends the corresponding control signals over Bluetooth to our control unit. The module would also be responsible for receiving acknowledgment signals sent as feedback from the microcontroller ensuring that the gesture has been registered and mapped to its equivalent direction of movement.

○

| Requirement: 1. The module can establish a connection with the robot’s Bluetooth module. 2. The module could generate the desired control signal based on the user input for our robot’s control chip. | Verification: For the first requirement, since the HC-06 Bluetooth module we’ll be using has test command that makes the Bluetooth return an OK signal, using that will let us know is a connection is established. For the second requirement, HC-06 sends the return signal containing the signal just received, which can be used to verify that the correct signal has been transmitted. |
Hardware Schematics:

- **Control Unit**
  - **Microcontroller**: This module is responsible for controlling the speed and the direction of movement on our robot. We intend to use a PIC16F977a microcontroller since it supports functionality for both Bluetooth module operation and simple DC motor control. Interfacing with the Bluetooth module simply requires connecting the module with the UART-based Tx and Rx ports on the PIC controller. The output registers on the controller as well as the PWM outputs are fed directly into the motor control module, as can be seen from the schematic.

| Requirement: Microcontroller circuit environment must be able to functionally interact with the Bluetooth and motor control modules. Namely, the data processing performed by the microcontroller should be within the three second runtime limit of the system. |
| Verification: As a preliminary procedure, we would have to perform a basic “Hello World” style test for the microcontroller alone by setting up the basic circuit environment (i.e. just the microcontroller component from the schematic above) and writing test code that allows it to toggle an LED. This would ensure the microcontroller’s proper functionality and allow us to proceed in implementing our... |
In addition, we would need to interface the microcontroller with the Bluetooth module (HC-06) and motor control module (L298N) individually before connecting both components together. First, using the microcontroller and Bluetooth module alone, we would toggle an LED via Bluetooth control. An acknowledgment code would be displayed on the source device to ensure the transmission’s success.

The motor control testing environment would only include the microcontroller, L298N, and motors. This would allow us to manually test motor control parameters such as PWM duty cycles to set the speed each motor and ensure that the microcontroller can control the IC.

Finally, using debugging tools in the MPLAB IDE, we can determine the runtime of the motor control testing environment. Given that the full runtime is approximately the sum of the CNN gesture recognition time, the transmission time, and the PIC processing time, finding the time of each individual subsystem and determining the sum can provide a decent approximation of the total processing time of the entire system.

- **Bluetooth Module:** This module is responsible for facilitating communication between hardware and software. It is mounted on the vehicle and wirelessly receives the aforementioned control signals over Bluetooth to be sent to the microcontroller. We intend on using an HC-06
Bluetooth module since it operates within our 9-meter threshold and is relatively simple to interface with our microcontroller.

| Requirement: Must successfully transmit data over a 2.4GHz frequency band at a baud rate of 9600 bps within a 9 meter radius. All pins on the module must be set at the proper voltages to ensure successful communication. | Verification: By setting up the Bluetooth module connected to our microcontroller and pairing it with our laptop, we can use a serial communication debugger to set the necessary input parameters. For the HC-06, in particular, the Vcc and Tx pins can support 5V inputs as they have internal regulation; however, the Rx pin only supports a 3.3V logic level, and thus, a voltage divider is required between it and the Tx port on the microcontroller. The 9 meter coverage is intrinsic to the module itself; but the range limitations can be manually determined from the test outlined in the previous verification in which we toggle an LED over Bluetooth by measuring the maximum distance between the transmitting device and the module while doing the test. |

- Power Supply
  - **Batteries:** This module would consist of a single rechargeable 9V 550mAh lithium-ion battery with the charger integrated on the vehicle for powering the motor control IC mentioned in the following section. Since the IC also takes an optional 5 V input and has an on-chip 5 V voltage regulator, we can use the 5 V input port on the IC as an output to power our microcontroller.

| Requirement: Module should be able to output the required voltages to all parts of the circuit, including the 5V microcontroller power requirements, as well as the required voltages for the | Verification: Verify output voltages at different parts of the circuit using a multimeter. |
• Motion Control Unit

  ○ **Motor Control:** We plan to use the L298N, a dual H-bridge based motor control integrated circuit in order to facilitate this module's operation. It requires a 7-35 V input in order to drive the motors, as well as the signals from the microcontroller. Each motor consists of an enable input as well as two additional inputs each (IN1+IN2 and IN3+IN4, respectively) to control for the direction of rotation (forward or reverse). The polarity of one input must be the inverse of its corresponding input in order to prevent short-circuiting in the H-bridge. Additionally, can apply a pulse width modulated signal to the enable input, as the duty cycle of each signal determines the speed of the motors, allowing for the vehicle to turn.

  | Requirement: This unit must functionally allow for the microcontroller to operate on the motors, including PWM parameters for speed control and motor direction. Parameters must be specified to facilitate smooth and natural robot motion at around 10 cm/s. | Verification: As specified in the microcontroller R&V block, this requirement can be verified in the motor control testing environment. In order to ensure that the robot moves slowly and smoothly without any jarring motions that could potentially damage it, the PWM signal sent from the microcontroller to the enable inputs of the motors must be set with a small enough duty cycle to achieve the optimal speed. Moreover, we can verify the current required to meet our 10 cm/s requirement through modifying PWM control values. |

  ○ **Motors:** Simple DC motors will suffice for this project. We intend to use ones that have an operational voltage range of 4.5-9 V to account for our supply voltage input.

  | Requirement: Power supply input voltage and voltages delivered directly from the L298N should trigger motor operation. Must also | Verification: Aside from the previous test which would verify operation with the L298N, we plan to utilize a DC power supply with |

be designed around a 10 cm/s operation speed. the motors alone to verify the operational voltage range. Aside from the PWM based control for speed adjustment, the speed can also be further decreased from the normal loaded speed of the motor through the use of gears attached to the end of the motor. Assuming a 60 mm diameter wheel, a 10 cm/s operational velocity would approximately correspond 30 rotations per minute, which will likely be achieved just using PWM control, but if the necessity arises, gear-based adjustment can be a good asset to further decreasing the speed to the desired value.

Tolerance Analysis:

The first important tolerance factor is the accuracy. In the high-level requirements, we specify that the for 70% times, the vehicle would move in the direction of the given gesture. The number 70% is calculated based on the true positive rate required in our classifier module. However, the hardware components and the bluetooth module might sometimes fail. Therefore, we think we also allow at most 10% failure rate of these modules due to either mechanical or electrical reasons. Thus, the lowest accepted accuracy rate is 70% * (100% - 10%) = 63%.

One important tolerance factor we need to consider would be the file size for the images we’re going to use for the machine learning part. The image shouldn’t be too big such that it takes days for the machines learning program to process them all, nor should it be too small that it becomes hard to make out a gesture in the picture. We’ve designed our CNN network, and it’s running platform to be powerful enough to handle 28.8KB of files per second, we get this number by doing the calculation 320*240/8 *30, 320*240 is the photo resolution, 8 is 8 bit per byte and 30 is the frame rate(frame per second), since we’ll be generating our pictures through clipping each frame of a 30fps video of a hand waving, this could save us a lot of time in finding the ideal pictures for gesture machine learning. This data process rate allows us to process 3000 pictures in less than 2 minutes, making the debugging of our CNN much faster.
Another tolerance factor we need to consider would be the lighting condition of the operating environment of our system. Not having gone through any stress system to out CNN algorithm we do not know what would be the lower or higher boundary of the light condition. We want to design our project to first operate in an ideal environment with sufficient lighting (1000 Lux condition in a ECE 445 meeting room size room or a overcasted daylight condition). However after experimenting cameras on several popular models (Macbook pro 2012 and after, Dell Inspiron etc), we’ve found that a hand gesture could remain visible even when the indoor (a bedroom with size less than 9 square meters) light condition drops to below 50 lux. As long as our preprocessing unit could recognize a hand gesture in a picture taken by the camera there should be no reason for the CNN algorithm to fail to recognize it, and we already know that OpenCV library can support gesture recognition in room lit by natural daylight, so the acceptable lighting range we expect would be much wider than our current “well-lit room” requirement.

The final tolerance factor pertains to our hardware components. In general, we need to ensure that proper voltages are distributed to every part of the system. We intend on accomplishing this through simulating the circuit in the MPLAB X IDE. This shouldn’t be a major problem considering that there are only three major components that require voltage regulation, namely, the motor driver IC, the microcontroller, and the motors. The L298N motor driver IC specifically is required to provide a 5V output to power our microcontroller the and our Bluetooth module (We may need extra voltage regulator since the HC-06 bluetooth module take in 3V power input). The onboard 5V voltage regulator allows us to accomplish this provided that the input voltage isn’t 12V or greater, a criterion which we have already met. An issue could potentially occur if both pins required to operate either motor’s respective H bridge from the motor driver IC receive a digital 1 since this would short circuit the bridge and destroy the motor. Thus, we must ensure that the polarities are always reversed for every required direction of movement.

3. Cost and Schedule

Cost Analysis:

<table>
<thead>
<tr>
<th></th>
<th>Prototype</th>
<th>Bulk Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Input &amp; Analysis Unit</td>
<td>Free</td>
<td>Free</td>
</tr>
<tr>
<td>Command Unit</td>
<td>Free</td>
<td>Free</td>
</tr>
<tr>
<td>PIC16F877A Control unit</td>
<td>$10</td>
<td>$3 or below</td>
</tr>
</tbody>
</table>
The fixed development cost of our group is estimated to be $50 per hour, 10 hours per week for 3 people and the project required a semester (16 weeks). Thus, the cost should be $50*10*16*3*2.5 = $60,000.

The Image Input & Analysis Unit and the Command Unit both utilize open source software, so they’re free.

The Control Unit requires one of the provided PIC16F877A microcontrollers as well as an HC-06 Bluetooth module, the latter of which is around $10 on Amazon.

For the power supply, a 1-pack 9 V battery also costs around $10 on Amazon.

For the motor control portion, the L298N costs about $7 on Amazon. The motors themselves are sold by Adafruit and cost around $1.95 each.

All required circuit elements such as wires, resistors, capacitors, etc. as well as the cart body itself can be acquired for free using the resources at our disposal in the ECEB.
Altogether, the cost will be roughly $40.9, although some of the estimates we found may be higher than expected.

The total cost is estimated to be $6,040.9.

**Schedule:**

<table>
<thead>
<tr>
<th>Date</th>
<th>Arvind</th>
<th>Bofan</th>
<th>Qinlun</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Feb 18-22</strong></td>
<td>Work on Design Documents</td>
<td>Work on Design Documents</td>
<td>Work on Design Documents</td>
</tr>
<tr>
<td><strong>Feb 25 - Mar 1</strong></td>
<td>Finalize decisions on hardware components</td>
<td>Work on setting up the work environment</td>
<td>Finish tutorials of the Tensorflow and Keras</td>
</tr>
<tr>
<td><strong>Mar 4-8</strong></td>
<td>Collaborate on CNN model setup, set up microcontroller circuit</td>
<td>Establish a preliminary connection between the robot and the laptop</td>
<td>The basic model of CNN</td>
</tr>
<tr>
<td></td>
<td>including the Bluetooth module and verify Bluetooth module functionality</td>
<td>(able to transmit on/off signal would be a good start)</td>
<td></td>
</tr>
<tr>
<td><strong>Mar 11-15</strong></td>
<td>Finish designing hardware model including the motor control IC and test</td>
<td>Work with Qinlun to test transmitting command outputted by CNN to the</td>
<td>The first version of functioning CNN (work with one gesture)</td>
</tr>
<tr>
<td></td>
<td>the interaction between the HCI system and the robot</td>
<td>robot</td>
<td></td>
</tr>
<tr>
<td><strong>Mar 16-24</strong></td>
<td>Collaborate with implementing more gestures in the CNN system</td>
<td>If success in the previous week, test with more gestures, if not, debug.</td>
<td>The second version of CNN (expand to other gestures)</td>
</tr>
<tr>
<td><strong>Mar 25-29</strong></td>
<td>Complete embedded systems component by integrating the other gestures</td>
<td>Signal transmission part should be up and functional this week</td>
<td>Coordinate with other modules</td>
</tr>
<tr>
<td></td>
<td>into the equivalent hardware commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Apr 1-5</strong></td>
<td>Collaborate on the comprehensive debugging between hardware and</td>
<td>Comprehensive test our work thus far (Test transfer control signals</td>
<td>Debug the software</td>
</tr>
<tr>
<td></td>
<td>software</td>
<td>generated by all the gestures our</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CNN could recognize now)</td>
<td>Reserved for possible setbacks in previous weeks, if all goes well, help other members</td>
<td>Improvement on the CNN to achieve better performance</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>Apr 8-12</strong></td>
<td>Reserved for possible setbacks in previous weeks, if all goes well, help other members</td>
<td>Reserved for possible setbacks in previous weeks, if all goes well, help other members</td>
<td>Improvement on the CNN to achieve better performance</td>
</tr>
<tr>
<td><strong>Apr 15-19</strong></td>
<td>Reserved for possible setbacks in previous weeks, if all goes well, help other members</td>
<td>Reserved for possible setbacks in previous weeks, if all goes well, help other members</td>
<td>Debug the overall system</td>
</tr>
<tr>
<td><strong>Apr 22-26</strong></td>
<td>Work on the report</td>
<td>Work on the report</td>
<td>Work on the report</td>
</tr>
<tr>
<td><strong>Apr 29-May 3</strong></td>
<td>Finish up and prepare for the demo</td>
<td>Finish up and prepare for the demo</td>
<td>Finish up and prepare for the demo</td>
</tr>
</tbody>
</table>

### 4. Discussion of Ethics and Safety:

The main ethical concern in our project is data safety. We need a large training set consisting of clear hand gestures for our CNN to identify relevant features and use them to distinguish between gestures. Some of these photos may contain surrounding information that may be considered sensitive by their owners. It is our responsibility to obtain open source training data that is viable to utilize for this project.

Another possible security issue is the lithium-ion based battery we will be using to power our motor and control unit. Since extreme temperature, humidity and overcharging could all cause the battery to break, and in some extreme cases, to explode. However since we designed for our project to operate in the indoor environment, as long as we don’t put the batter in undesirable position (like right next to the motor), we shouldn’t need to worry too much about it malfunctioning.

Given how our project would require the use of many open sourced code resources (such as OpenCV and CNN libraries), it would be our responsibility to use these resources in a way that does not breach the IEEE code of ethic #4 and #7 [6]. Specifically, there should be a clear list and discussion of all the external code resources we used for our project in our final report and lab notebook, marking what and where did we use them for.
In addition, as we detailed in our original project pitch, we’re modeling our robot based on the cart from ECE 110 lab and as such, there shouldn’t be any severe safety issues in the design. While power dissipation is an inevitability in motor control based robotics, we need to ensure that our specified voltage and currents don’t exceed the thresholds of the motors and the cart design itself should have all components placed and shielded optimally to avoid damage in the case of an accident, including the motors, microcontroller, Bluetooth module, and power strip. Also, locomotive robots, in general, need to keep several design considerations in mind, including terrain understanding, limitations in speed and size, the materials used in the design, and operating distance from the client.

5. Citations:


Using Computer Vision,” *Elektronika ir Elektrotechnika*, vol. 21, no. 6, 2015.