Face Gesture Controlled, Computer Vision Based, Robotic Intelligent Device for People with Spinal Injuries, Cerebral Palsy, Amputation, and Muscular Dystrophy

Aryansh Shrivastava

Table of Contents

1. Introduction

People with spinal injuries, cerebral palsy, amputation, muscular dystrophy, etc. have little to no control of their hands or feet, and many times they cannot even speak to express their thoughts. Individuals with such impairments encounter extreme physical, social, and environmental challenges in their daily lives, which, in turn, create impediments to

self-supported living and to opportunities to take part in economic and social aspects of life. To help these people, there is a need of a computer vision based AI system that can interpret their facial gestures in a meaningful way, helping people with disabilities conduct daily activities, enabling them to improve their quality of life and live independently.

The goal of my project is to create a computer vision and AI based system that can interpret the face gestures in an intelligent and meaningful way, helping people with disabilities to do two way written and verbal communication. This system should also use face gestures to control the precise navigation of wheelchairs and other guided robotic devices, which will help their movement. Also, this system should be able to interpret and convert facial gestures into commands which can control home and office gadgets that will help them control the environment around them, such as lighting (on/off), temperature, and sound. The system should also present a similar user interface for all of these activities, and it should be reliable and accurate with quick response time, since it will be used by disabled people. The system should be portable and lightweight, so that it can be taken around for assistive help.

2. **Material**

- Laptop with Built-In or External Camera
- Python 2.7, Python Serial Port Library
- OpenCV, Dlib Libraries, Speech API

For Environment and Navigation Modules

- Arduino Uno, Breadboard, Jumper Wires
- 4 Channel DC 5V Relay Module for Arduino Raspberry Pi, DSP AVR PIC
- ARM K49, Lightbulb, and Fan
- HC-05 6 Pin Wireless Bluetooth RF Transceiver Module

- Arduino C Programming Tool
- Toy Robotic Car with Assembly Instructions

3. Design Constraints

3.1. Facial gestures should be captured in real time using a built-in video camera connected to a laptop or inexpensive portable video camera. Each captured video frame should be interpreted by open source computer vision and AI based network, to identify face, mouth, nose, eyes, or eyebrows in a reliable and accurate way. The interpretation should be very fast, as the video is captured in real time (25-30 fps).

3.2. Identified facial landmarks in each frame should be converted into 2-dimensional coordinates, so that they can be geometrically interpreted into facial gesture signals like mouth open/close, nose movement, eye blink movement, eyebrows up/down etc.

3.3. The face gesture signal mathematical models that are created should be reliable and accurate on all face shapes, sizes, colors, and facial alignments, so that device can be used by anybody. Also, these gestures should be easy to perform.

3.4. The input system must be designed so that it can be superimposed on the real-time face video of the user, so that user can interact with the input system using facial gestures and navigate the system to control different modules. This input system should be easy to use, reliable, fast, and accurate, presenting uniform graphical interface to all controlled modules.

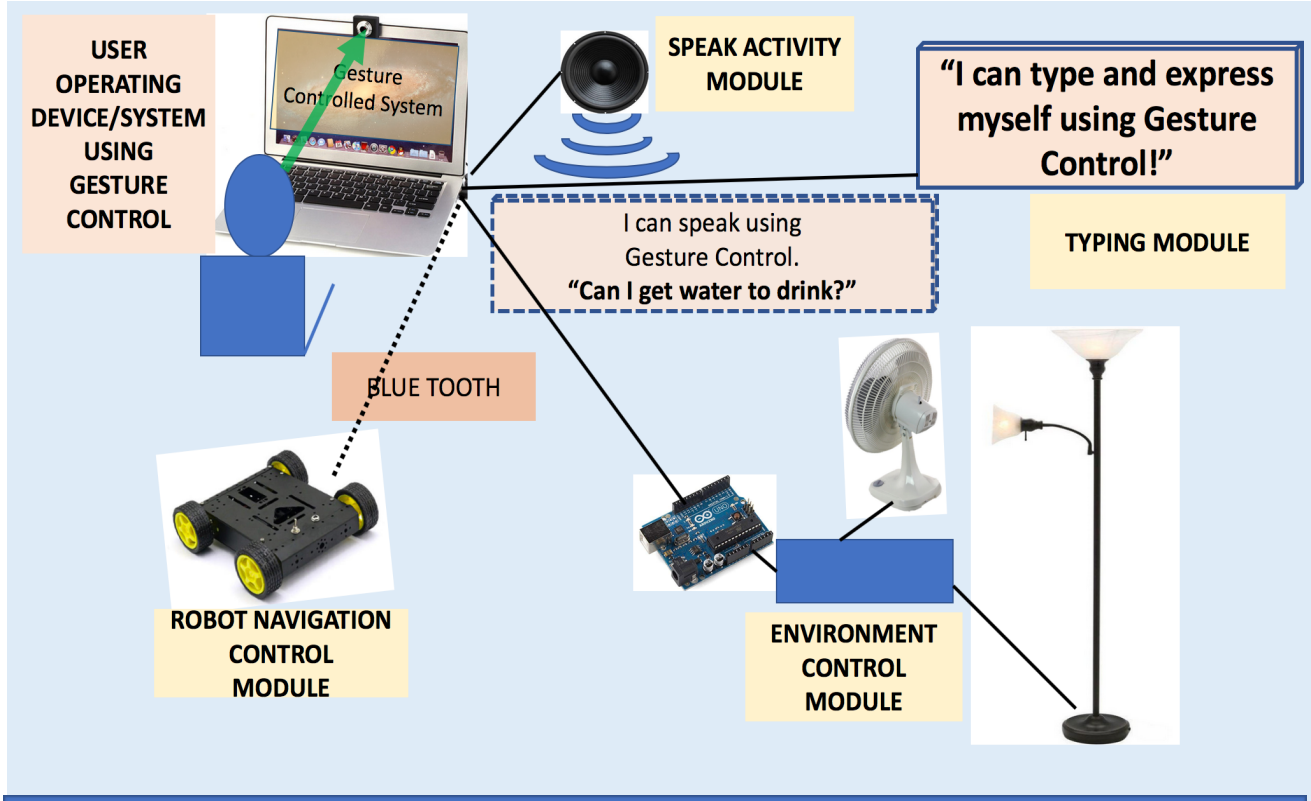
3.5. The speak activity control module should be created so that the user can interact and speak out daily activates which user wants to express.

3.6. The environment control module should be created so that the user can control things like lighting (on/off), temperature, sound, etc.

3.7. The typing module should be created so that the user can express their thoughts by typing which can be spoken out by text to speech software or used to send messages.

3.8. The navigation module should be created so that it can precisely drive and navigate wheelchairs and other guided robotic devices at the will of the user.

3.9. The system should provide a method for integrating new modules in the future like controlling computers and new applications. Also, the system should be portable and lightweight, so that it can be taken around for assistive help; it should also be inexpensive, effective, and open source so that it can be used by the masses. System must be reliable and accurate.



Simplified block diagram of the system, showing person sitting in front of laptop and controlling the different modules using face gestures

4. Methods

4.1. Installing OpenCV libraries, Python virtual environment, Dlib, and other Math libraries.

Programming and using OpenCV to capture the internal video camera and one external video camera for video input.

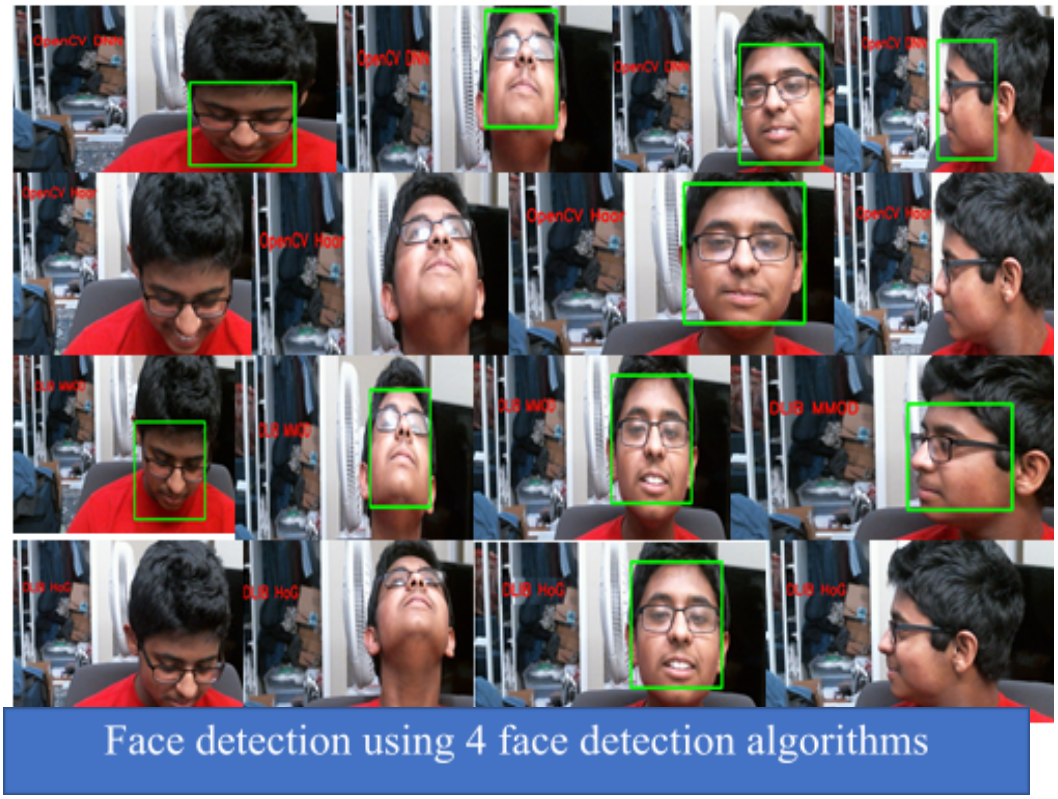
4.2. Writing and testing the code for the four different algorithms of face detection to determine the best one.

4.2.1. FACE DETECTION USING 4 FACE DETECTION ALGORITHIM

4.2.1.1. Four different types of face detection AI networks were tested: OpenCV DNN, OpenCV Haar, Dlib CNN Mmod, and Dlib Hog.

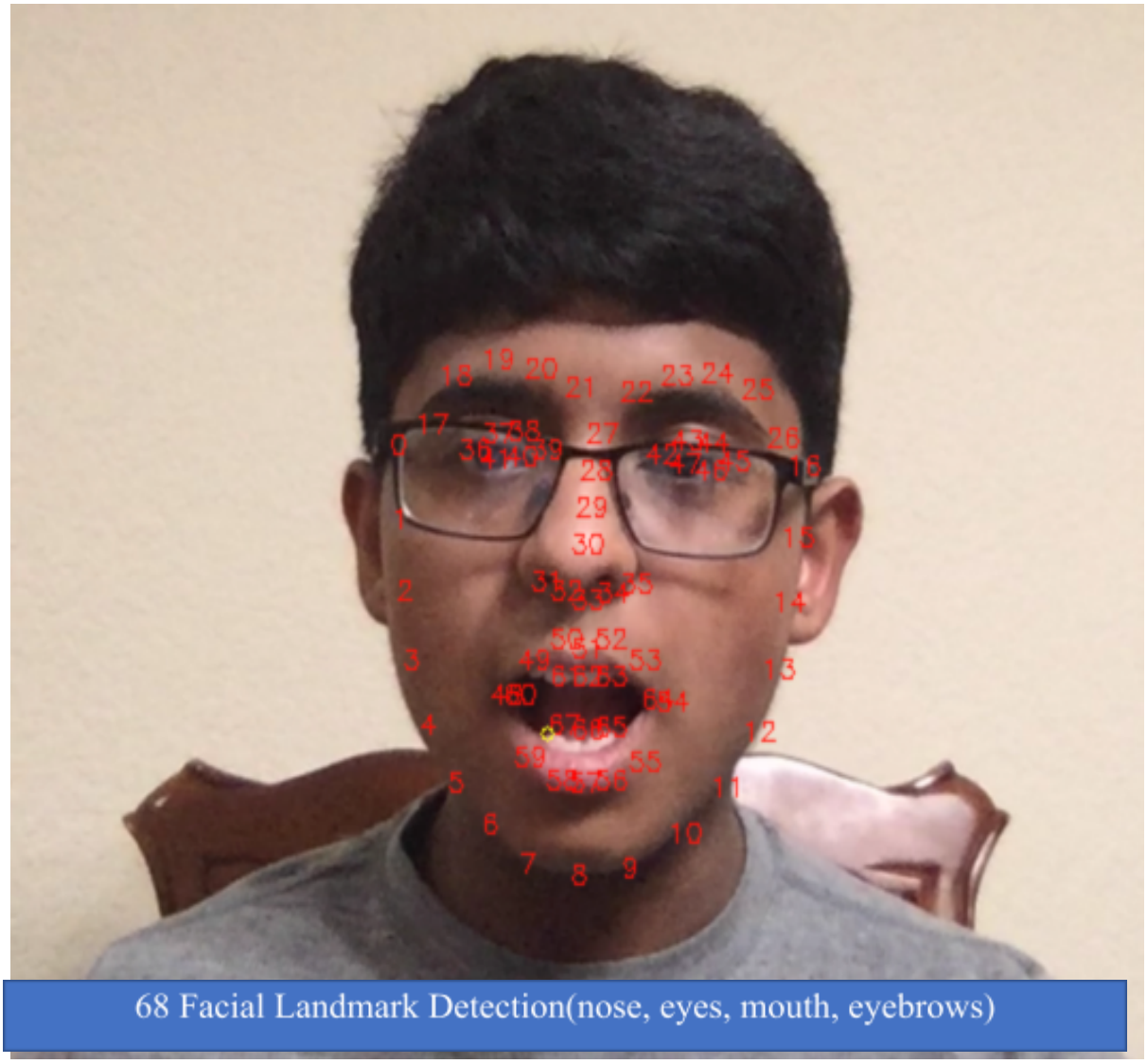
4.2.1.2. The system is going to be used in many different circumstances where users will position their gaze toward the camera at different angles. Facial positioning for face detection was tested for five different orientations center or tilted up, down, left or right. During tests OpenCV DNN and Dlib CNN Mmod models were found to be the most accurate.

4.2.1.3. The independent variable was the type of network, while the dependent variable was the number of correct facial detections that occurred. All other variables, such as the person tested, the location of the test, and the positioning of the camera were all kept unchanged as controlled variables.

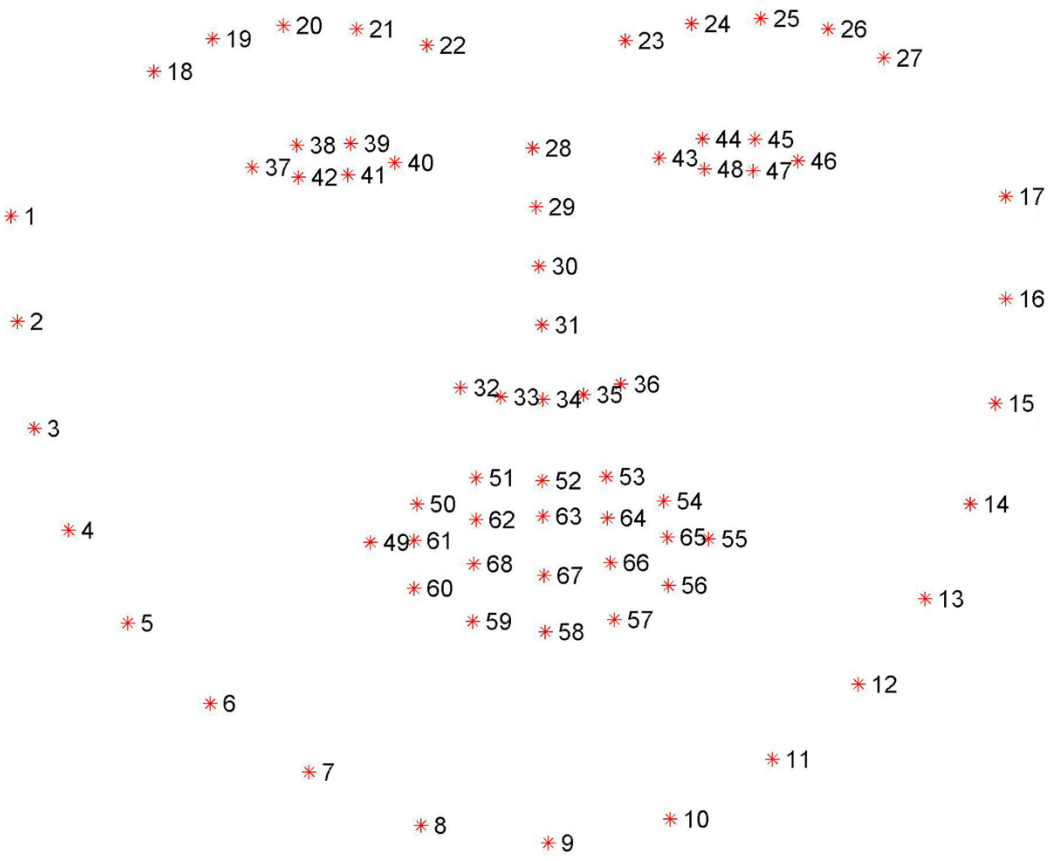


4.3. 68 FACIAL LANDMARK DETECTION ALGORITHIM

For facial landmark detection a trained AI network based on IEEE Conference on Computer Vision and Pattern Recognition Paper One Millisecond Face Alignment with an Ensemble of Regression Trees is used.



68 Facial Landmark Detection(nose, eyes, mouth, eyebrows)



68 Facial Landmarks (Showing nose, eyes, mouth,eyebrows)

4.4. Using the landmarks, to identify the gestures signals which can be mathematically modeled with reliability.

4.4.1. FACE GESTURE SIGNALS

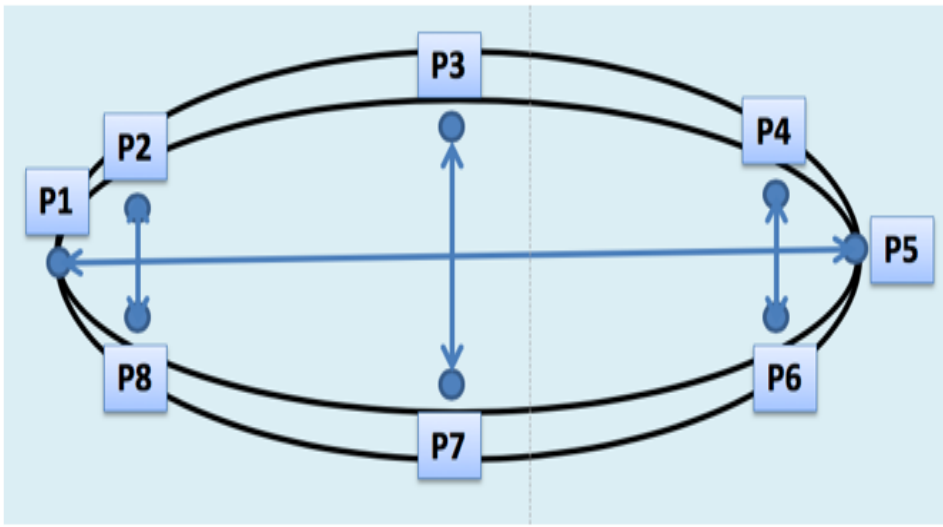
4.4.1.1. I found during experimentation that some face gestures can be identified within one frame by creating geometric ratio thresholds, such as the opening of the mouth, eye blinks, and the raising of the eyebrows when compared with eye.

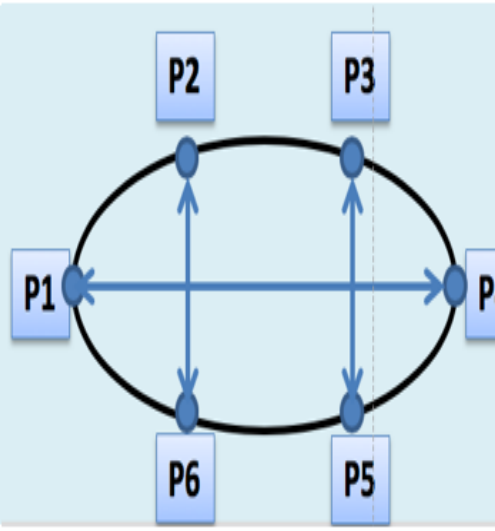
4.4.1.2. Also, there are multi-frame gestures, such as nose movement, eye movement, or face movement; these are very easy to perform, but they are difficult to track.

4.4.1.3. Experimentation and thought revealed that a reliable input system should have one single-frame face gesture to activate the multi-frame gesture, which then can be tracked and used for the input system.

4.4.1.4. As of which gestures to use for single-frame gestures, I rejected the eyebrow raise gesture for now, as it is more intensive to do, and many people might not be able to control or raise their eyebrows past the threshold easily. Contrariwise, eye blinking and the opening and closing of the mouth are more natural gestures that can be performed by the general public.

4.4.1.5. The diagrams and formulas below show exactly how the ratios for the mouth opening and closing, along with the eye blinks, can be calculated using some of the 68 facial landmarks identified by the landmark model.

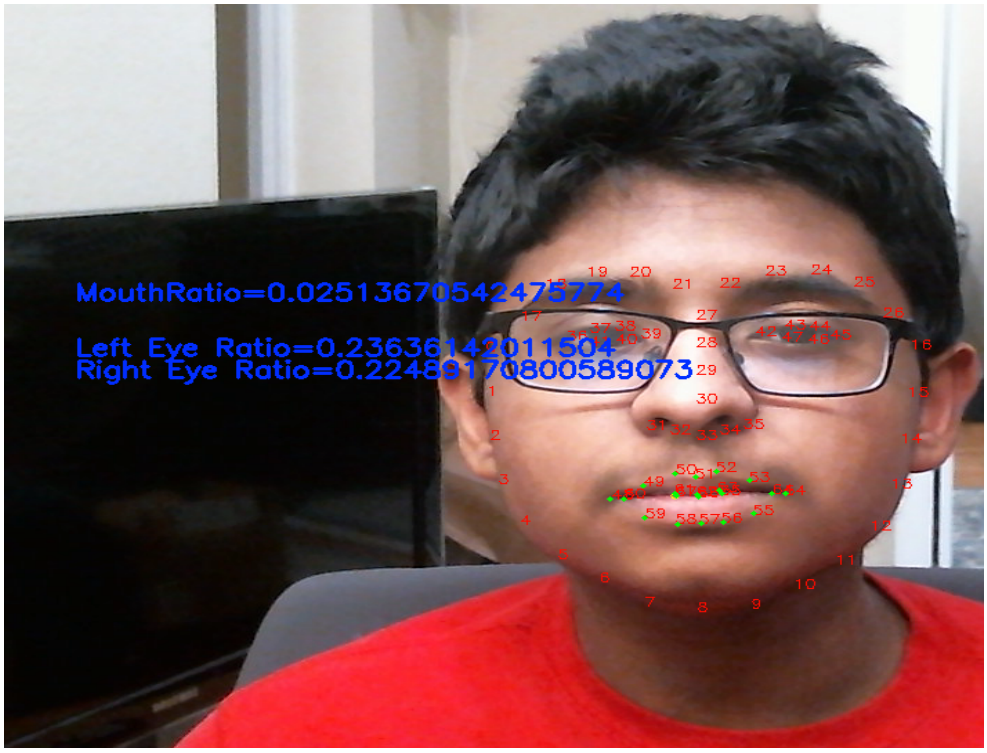




$$\text{Mouth Gesture Signal Ratio (MGR)} = \frac{|p_2 - p_8| + |p_3 - p_7| + |p_4 - p_6|}{3|p_1 - p_5|}$$

$$\text{Eye Gesture Signal Ratio (EGR)} = \frac{|p_2 - p_6| + |p_3 - p_5|}{2|p_1 - p_4|}$$

MOUTH AND EYE GESTURE RATIO



Mouth Ratio shown to increase with open mouth calculated from landmark detection

4.5. Using the face gestures signal to design and fine-tune a reliable input method for assistive actions.

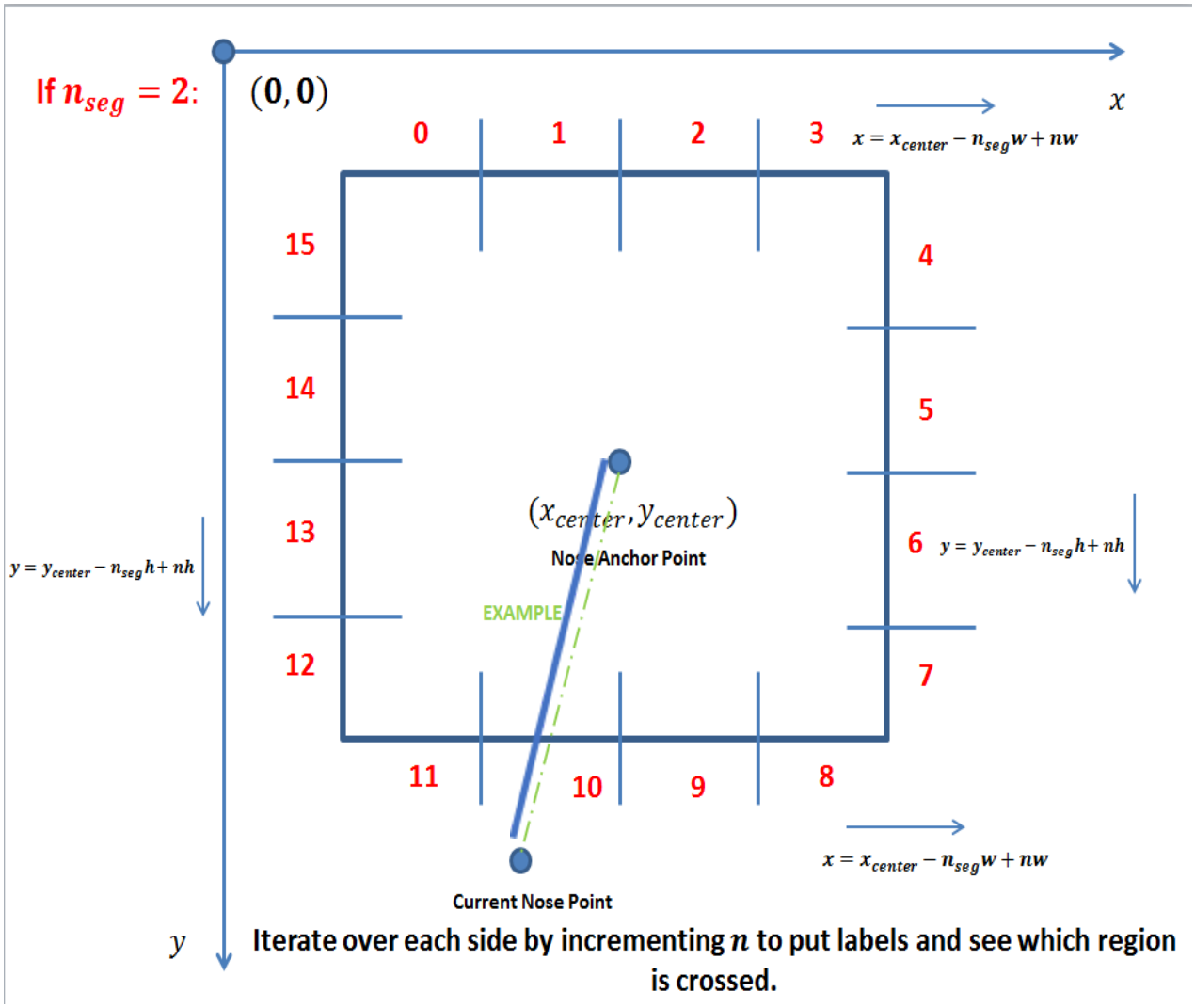
4.5.1. INPUT SYSTEM DESIGN

4.5.1.1. The input system is defined by a visually segmented rectangle that is divided up into different segments for different activity selections. Once the mouth gesture is triggered, the user is able to see the rectangle with the different segments, each of which are identified based on their respective labels which are superimposed on top of the frame on the screen, along with their nose line. These two things are used in the input system which is triggered when the nose line crosses a segment, which is determined mathematically through coordinate geometry by iterating in a clockwise manner around the input rectangle to see where the line crosses. This activity will then be triggered.

4.5.1.2. To reduce the jitter of the nose line crossing the segment on the edges multiple times, the program employs the logic that once a line segment is triggered, the absolute Euclidean distance has to reduce below a certain tunable threshold before the next trigger.

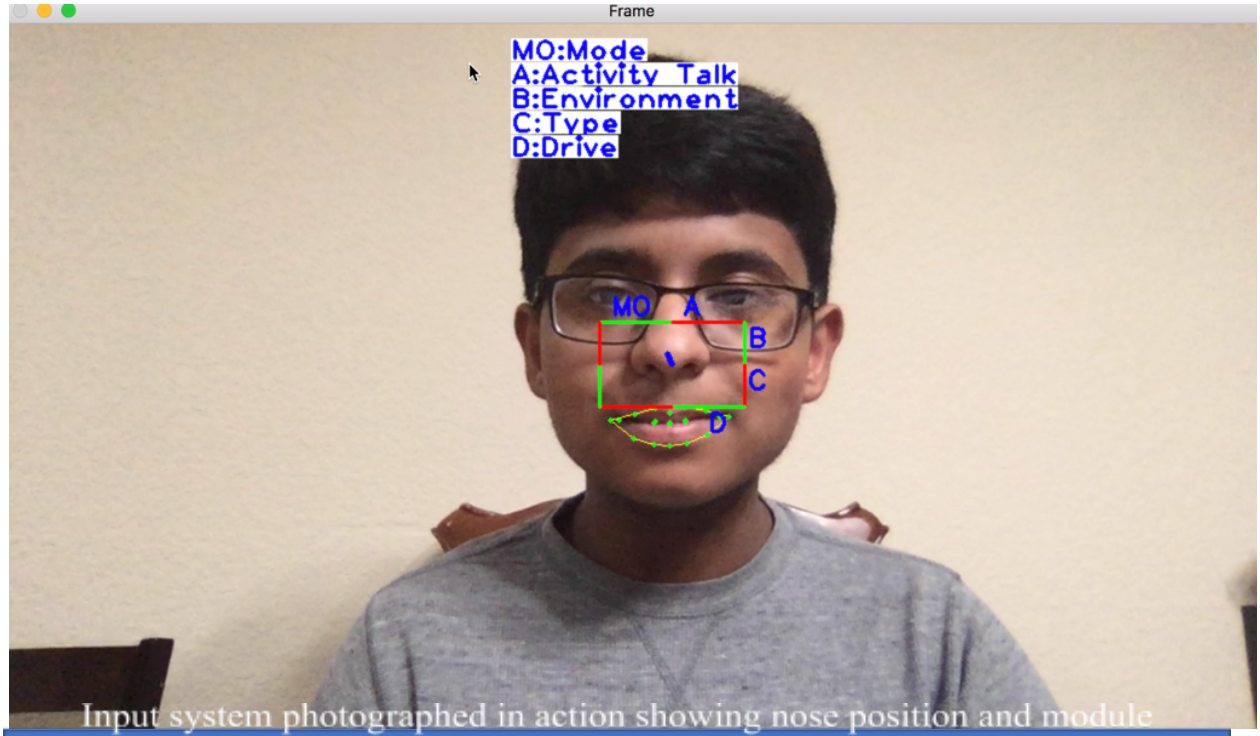
4.5.1.3. Initially, the system is designed to display different modes on the segment line, and each mode corresponds to an activity,

4.5.1.4. The input system is designed so that the number of segments for activity selection can be changed easily. The labels on the segments can change based on the mode selection so that the system is easy to navigate for disabled people.



Input system showing the coordinates of different segments and nose anchor point and current nose position

4.6. Identify and write code for the assistive actions which will be used in this project to show the overall versatility of the system.

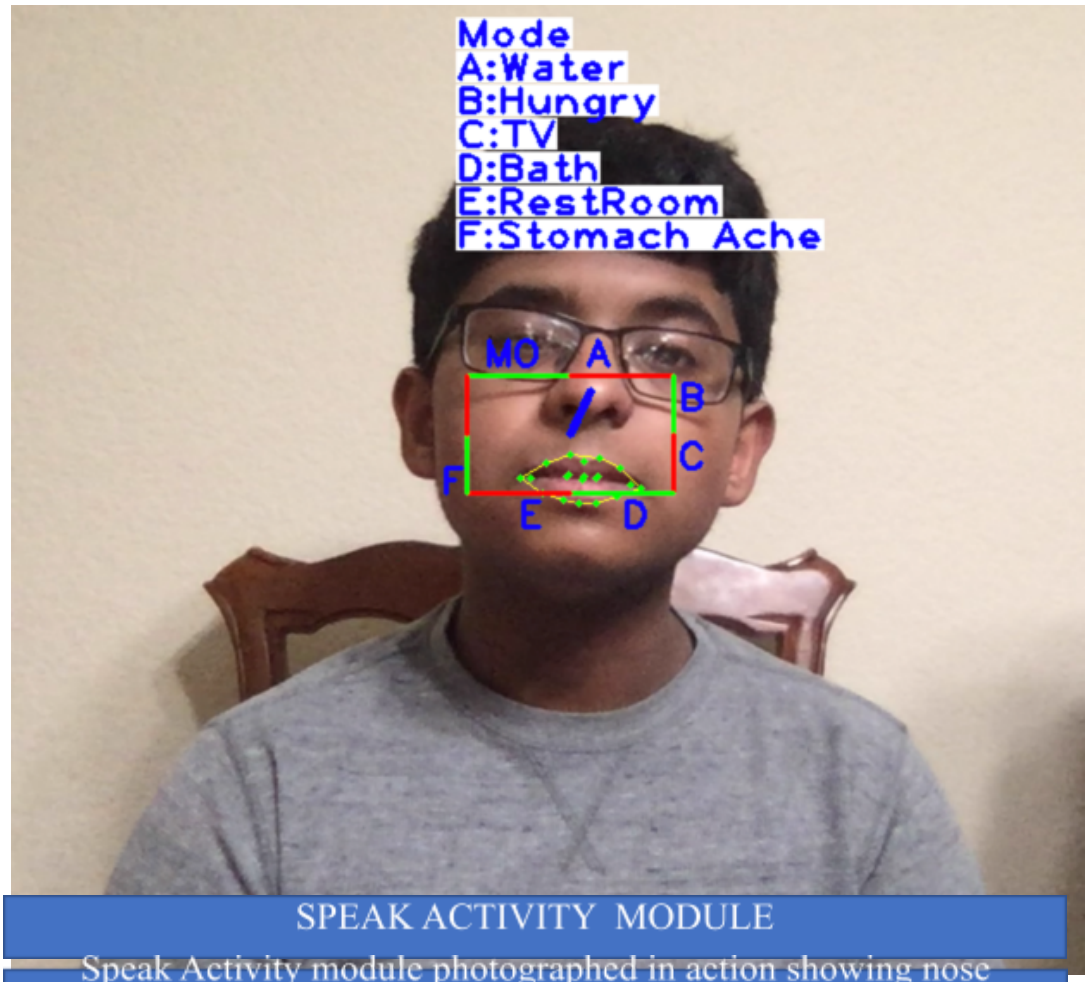


Input system photographed in action showing nose position and module selection; the user can move their face and select the command by touching the module segment from the nose line, which will activate the module

4.6.1. Design and code the verbal communication module using Apple speech API.

4.6.1.1. SPEAK ACTIVITY MODULE

In speak activity module different activity options are listed, such as “I want water” or “I want to go to the bathroom” so that the disabled person can easily communicate their thoughts verbally. These activities are implemented using speech API.



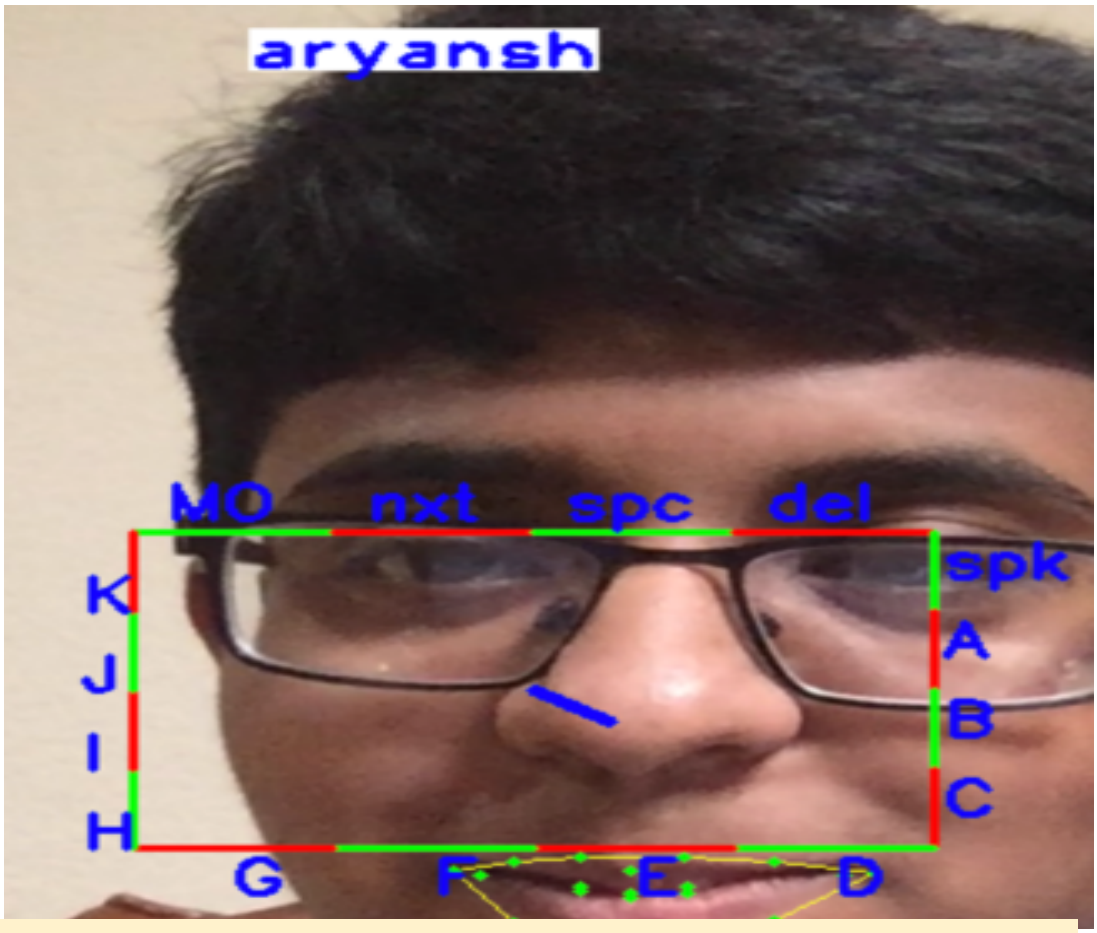
SPEAK ACTIVITY MODULE

Speak Activity module photographed in action showing nose position and command selection, user can move face and select the command by touching the command segment from the nose line which will speak the speech command using speech API.

4.6.2. Design the code to create a very reliable typing system using the same input method.

4.6.2.1. TYPING MODULE

For the typing module, each letter typed is kept inside a string buffer variable which is displayed so that the user can see what they are typing. All 26 letters of the alphabet are included, along with options such as space or backspace to modify the string. If the user wants, they can also choose to communicate this written string using a provided text-to-speech functionality.



Typing Module:

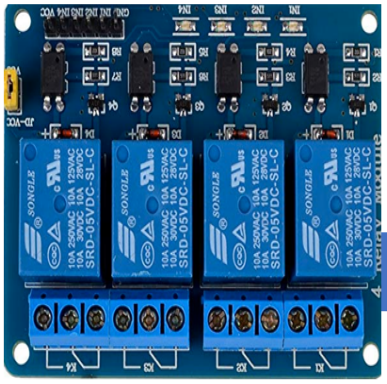
Using my device, I have typed my name just by nose movements.

Typing module photographed in action showing nose position and full typing visual keyboard; user can move their face and select the letter by touching the letter segment from the nose line, which will type the letter in the buffer

4.6.3. Design the environmental control and write code for the Arduino and IOT devices with Bluetooth.

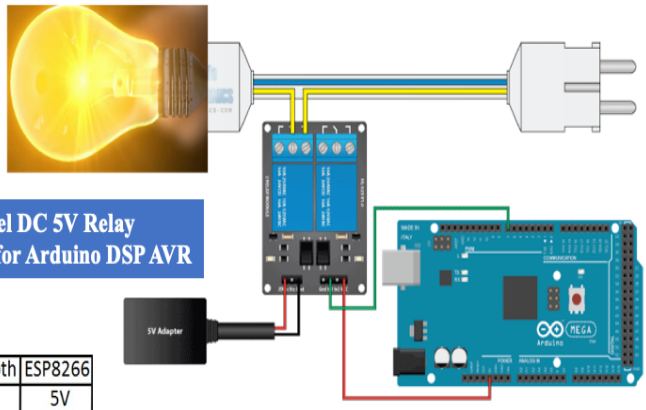
4.6.3.1. ENVIRONMENT MODULE

For the environmental control module, an Arduino microcontroller which is connected to a power relay switch using its digital pins is used; the power relay switch has a light emitting diode (LED) that will make it turn on or off based on how the program sends either 0 or 1, controlled by the user's selections in the visually segmented rectangle input system. When the LED lights up, the fan and other appliances can be turned on.



4 Channel DC 5V Relay Module for Arduino DSP AVR

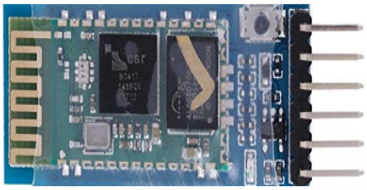
https://www.amazon.com/gp/product/B01BACQF1Y



Bluetooth	ESP8266
VCC	5V
GND	GND
TXD	RX
RXD	TX

Bulb controlled via Arduino listening on blue tooth and 4 Channel DC 5V Relay

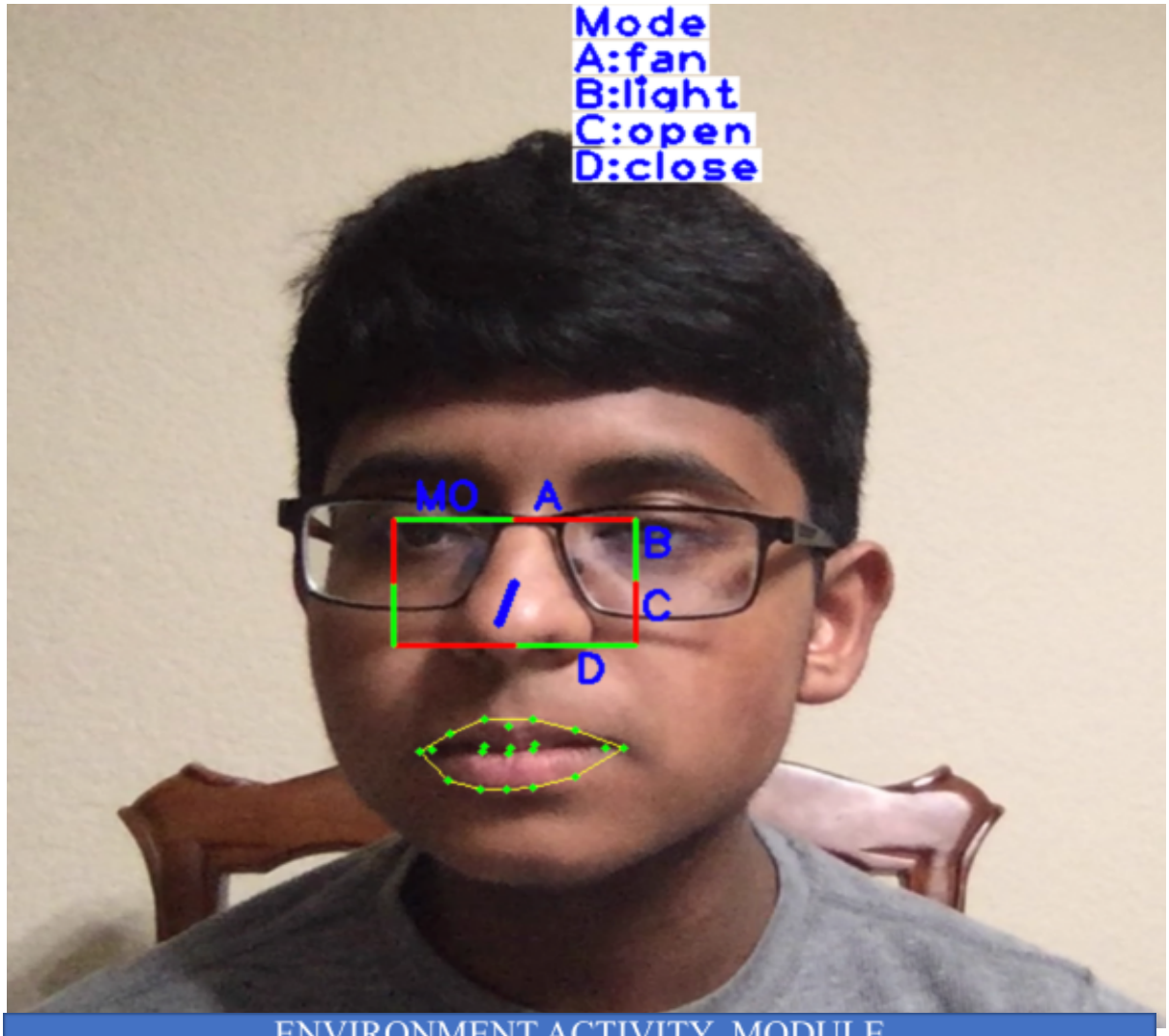
https://howtomechatronics.com/tutorials/arduino/control-high-voltage-devices-arduino-relay-tutorial/



https://www.amazon.com/gp/product/B01BACQF1Y

HC-05 6 Pin Wireless Bluetooth RF Transceiver Module Serial BT Module

Circuit Diagram representation of environment module where bulb is connected to 4 channel DC relay controlled by Arduino which is listening for on/off command relayed over Bluetooth from the input environment control module



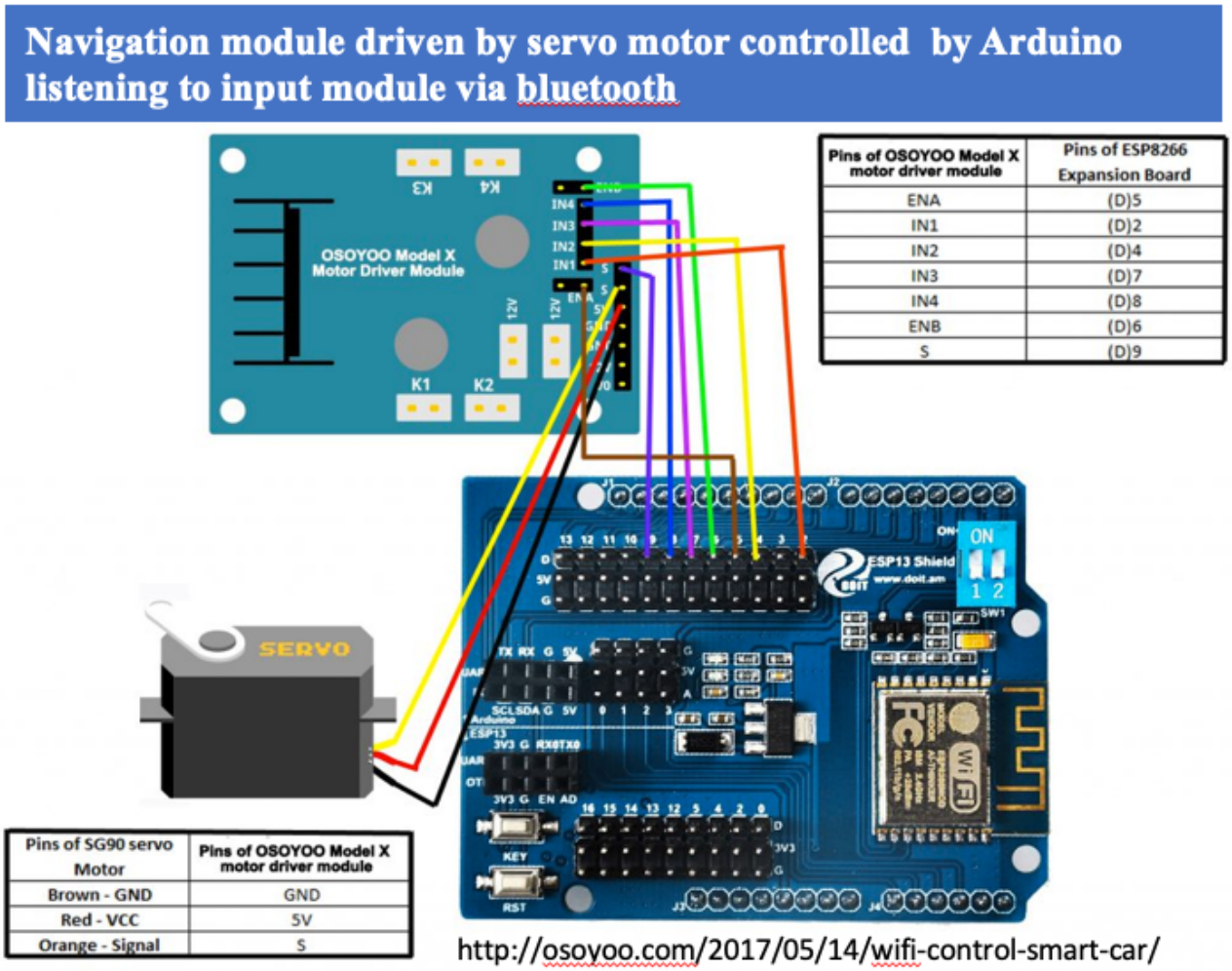
ENVIRONMENT ACTIVITY MODULE

Environment module photographed in action showing nose position and fan open and close, light open and close; user can move face select the device segment first and then action segment

4.6.4. Design a prototype model which can be easily implemented into the wheelchair for drive maneuverability.

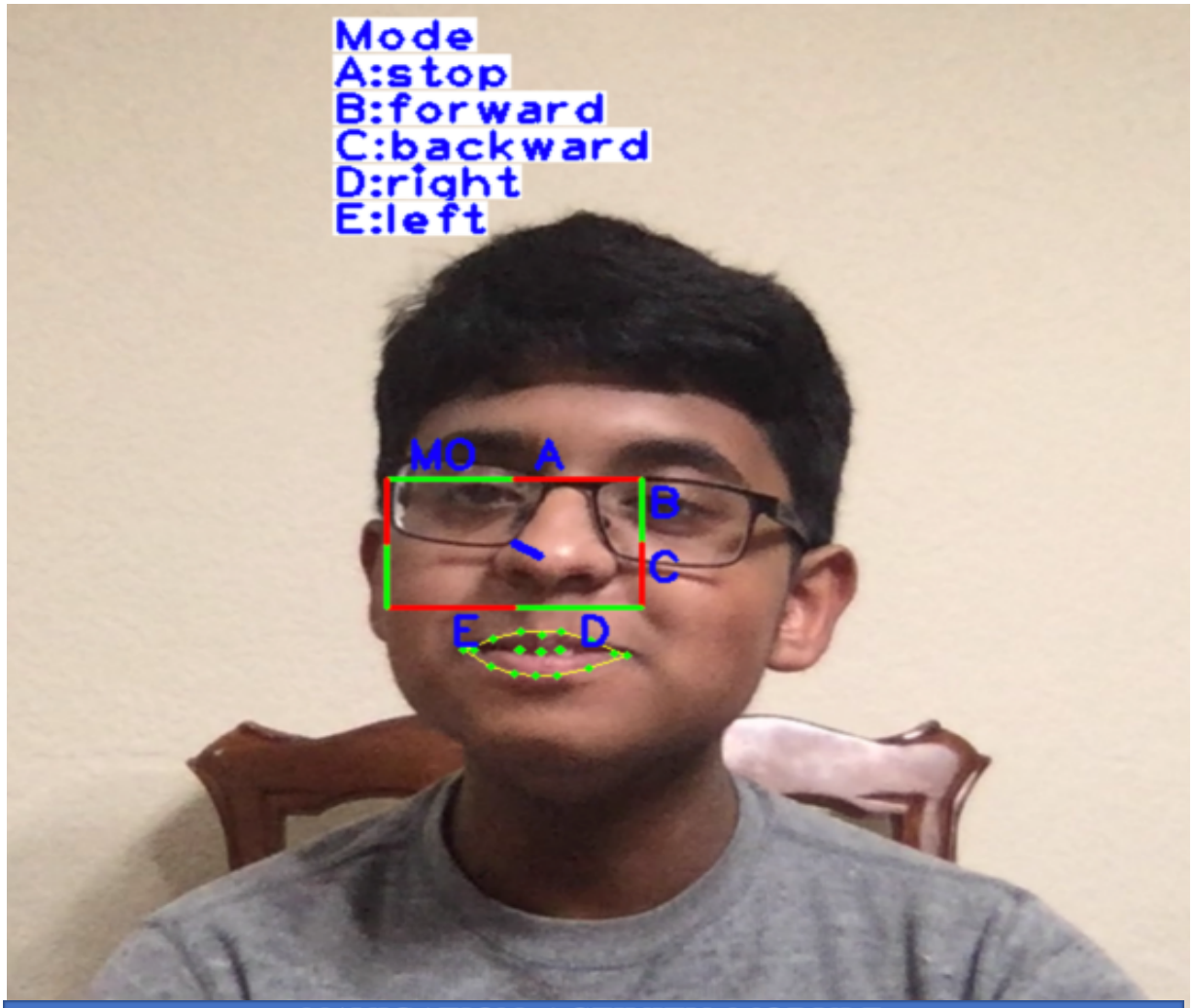
4.6.4.1. NAVIGATION MODULE

Similarly, for the intelligent machine movement control module, the user input is sent over Bluetooth to the Arduino and to servo motors. The user can then use this facility to give commands to motors to travel in different directions, enabling wheelchairs or robotic cars to move accordingly.



Pins of OSOYOO Model X motor driver module	Pins of ESP8266 Expansion Board
ENA	(D)5
IN1	(D)2
IN2	(D)4
IN3	(D)7
IN4	(D)8
ENB	(D)6
S	(D)9

Pins of SG90 servo Motor	Pins of OSOYOO Model X motor driver module
Brown - GND	GND
Red - VCC	5V
Orange - Signal	S

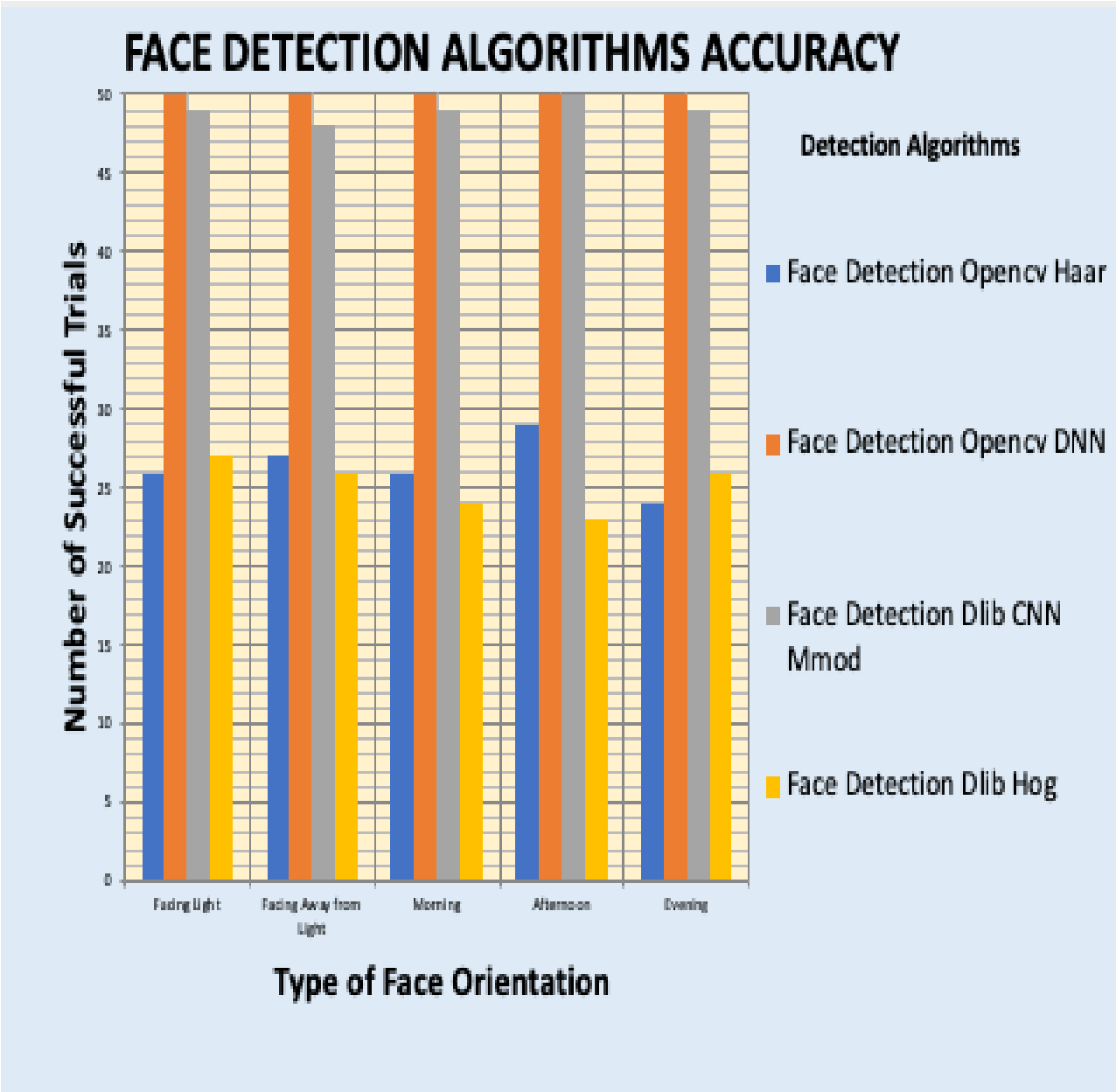


NAVIGATION ACTIVITY MODULE

Navigation module photographed in action showing nose position and navigation commands for the robotic car like stop, move forward, backward, turn right/left, stop; Same concept can be used to drive a wheelchair

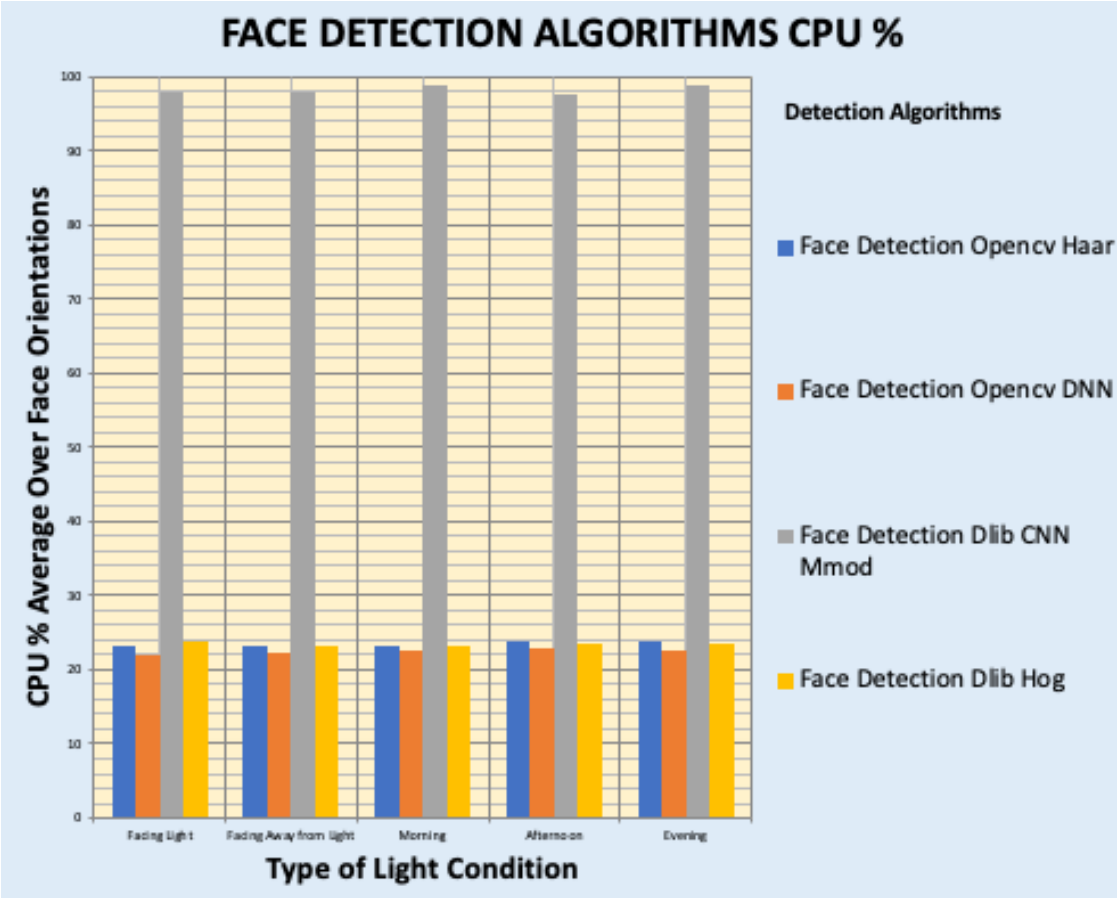
5. Results

5.1. A prerequisite of face landmark detection is accurate face detection, so I tested four different algorithms of face detection to select the best one suited for my application, meaning that it should be able to detect faces under different face alignments and light conditions. Hence, five face positions (center, aligned up, aligned down, aligned right, aligned left) were tested 10 times for each of the four algorithms (Haar Cascade in OpenCV, DNN in OpenCV, CNN Mmod in Dlib, HoG in Dlib) in five different light conditions. One point was given for correct detection and zero points were given for no detection; all other variables like the face, the distance from the camera, etc. were all kept the same. Then, a bar graph was plotted, and face detection with OpenCV DNN and Dlib CNN Mmod was found to be most accurate.

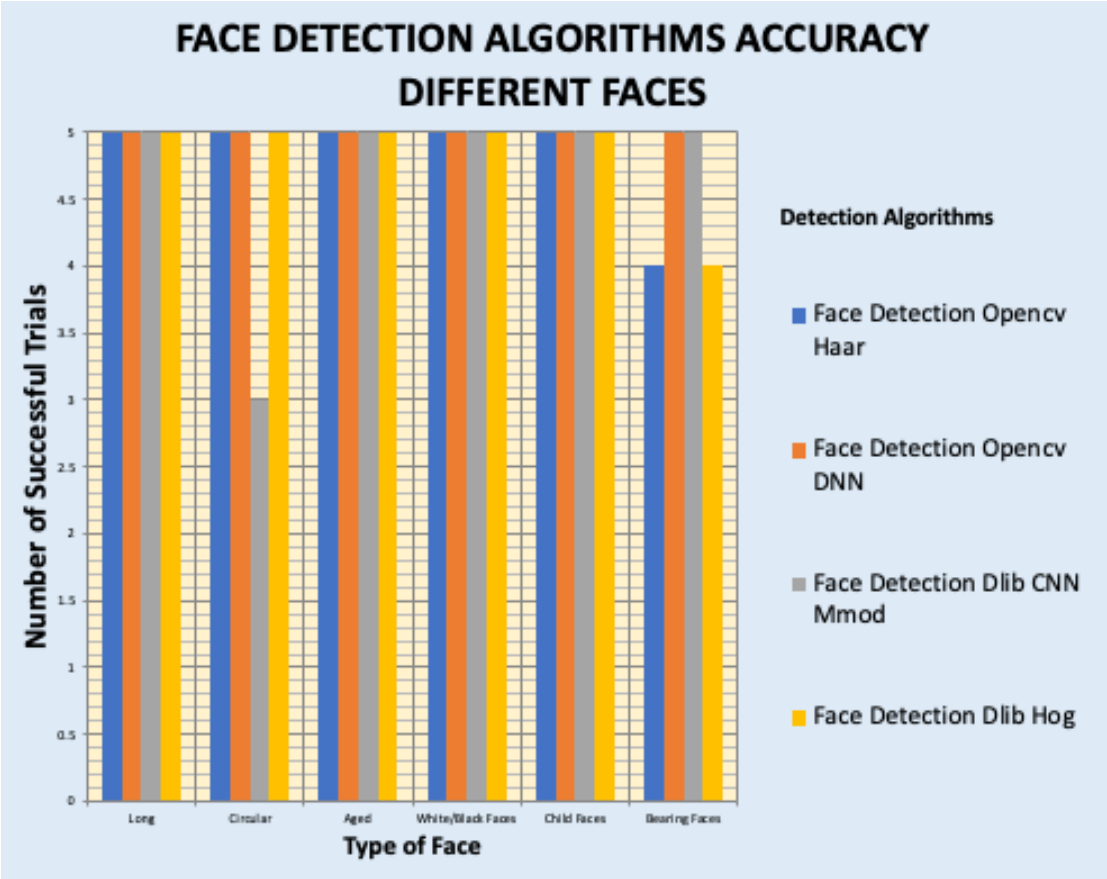


5.2. For the above test, %CPU utilization was also noted, and it was averaged for each light condition over the different algorithm. This is done because my application needs to recognize faces in real-time video at around 25-30 fps. I found that, for face detection,

the Dlib CNN Mmod used 98% CPU, making the video lag; in OpenCV DNN, CPU utilization did not go over 23%.

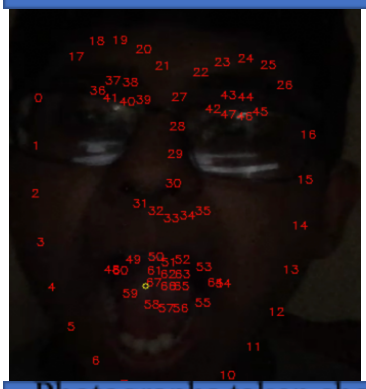


5.3. These four face detection algorithms were also tested for different face types, i.e. long, circular, aged, mix of white/black faces (two white, two black, and whitish), child faces, and bearded faces. These face images were downloaded from the internet. For circular faces, CNN Mmod could not recognize one of the faces, and for bearded faces, OpenCV Haar and Dlib HoG could not identify one face.

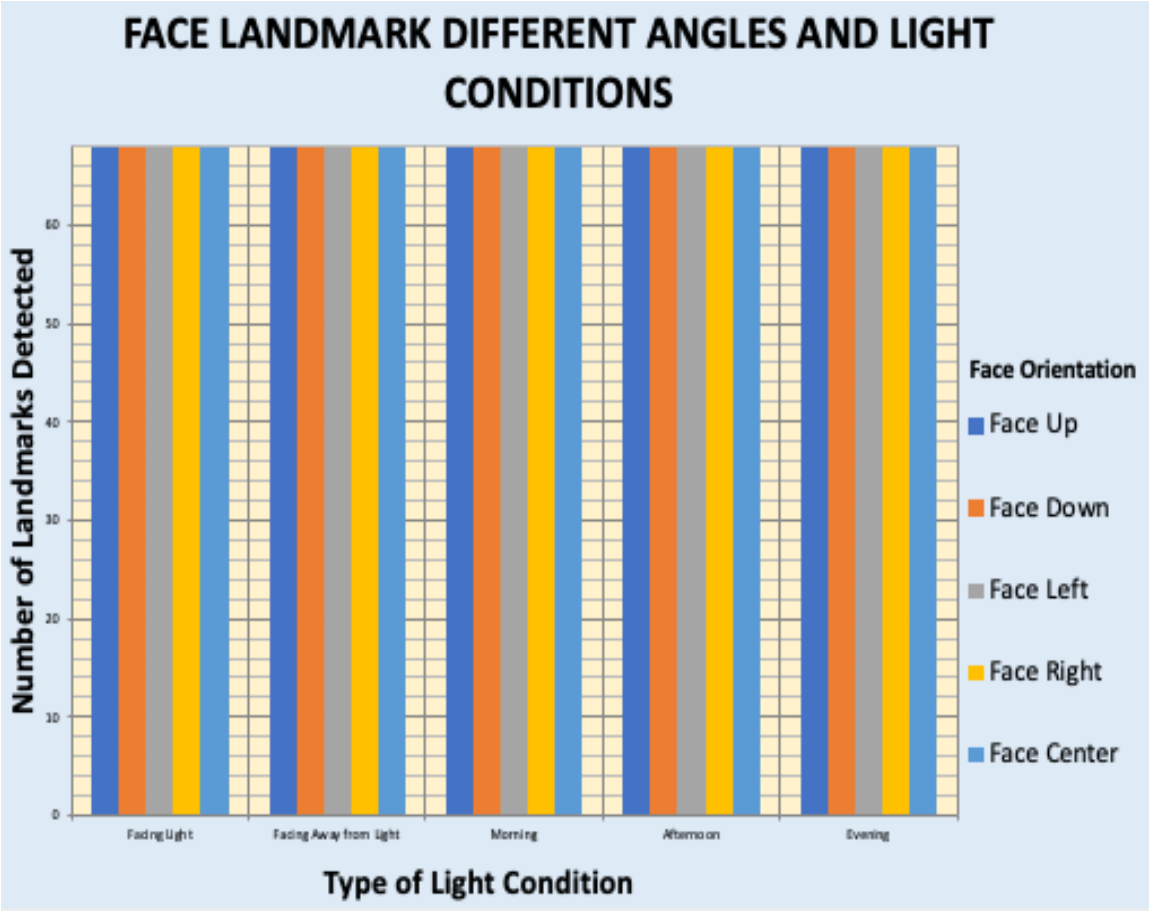


5.4. Based on the above test, OpenCV DNN was chosen for the face detection. Then, a test was done for the 68 landmark detection AI network to validate 68 landmark detection under different light condition and different face angles, and to see if all the landmarks are identified. The result was that all 68 landmarks were identified for all light conditions and face angles. This test was done because landmark points are used in the geometric calculation of face gesture signals.

Photographs taken showing that 68 landmark detection happens accurately at different light conditions and face angles

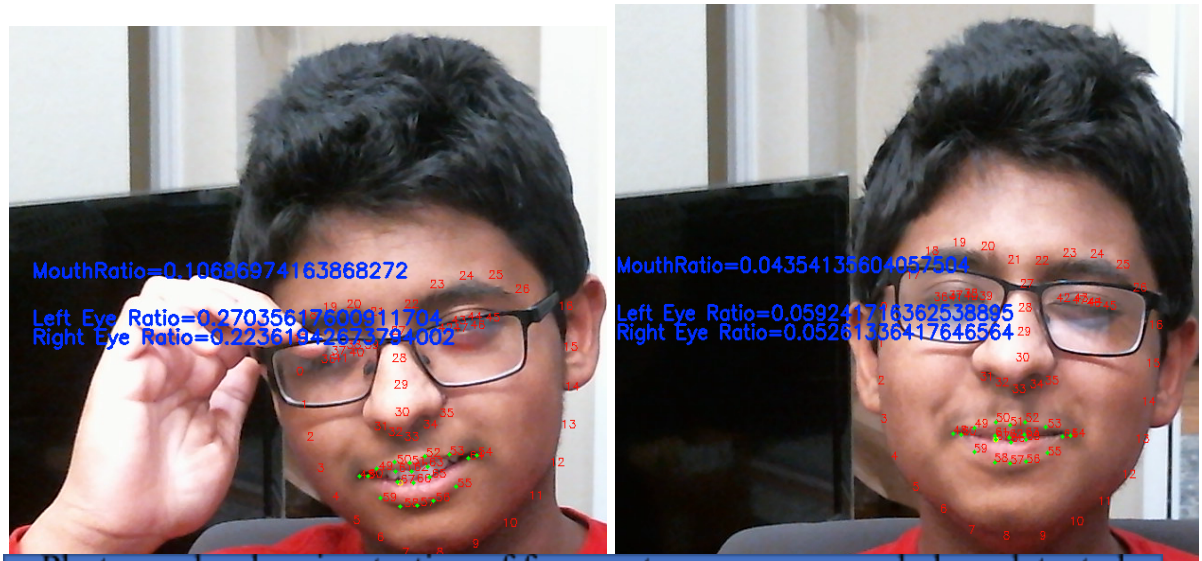


Photographs taken showing 68 landmark detection also occurs accurately in the darkness accuracy

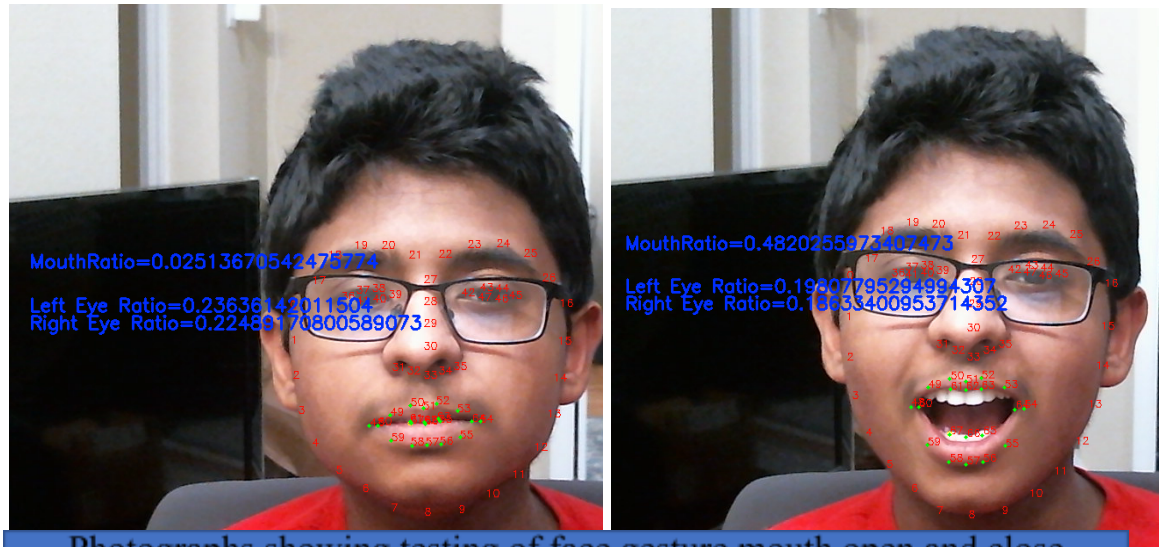


5.5. To activate the input system, a choice had to be made between mouth open/close gesture and the eye blink gesture to understand which one could be used as a face gesture signal. There are fluctuations for the values of these ratios in few frames and if the variation of the ratio for open/close for a gesture is higher than fluctuations then the face gesture signal can be identified over a period of frames as a signal. To do this, I plotted the variation of ratio values for both the mouth open/close ratio and the eye open/close ratio and fluctuations. It was found that if the mouth ratio is more than .2

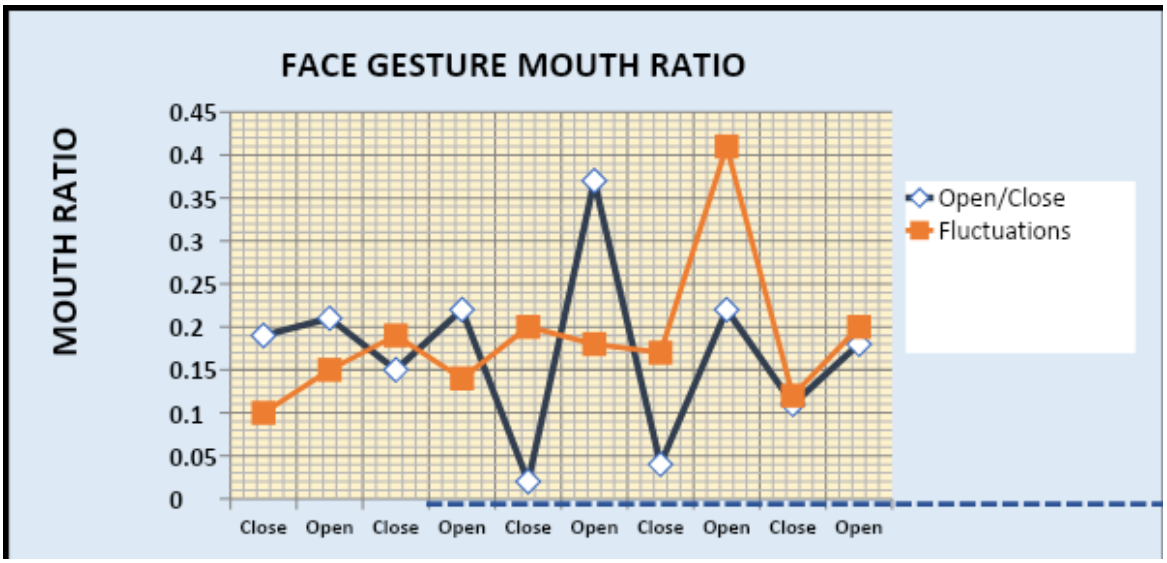
then the mouth open signal can be clearly identified. For eye ratio (open and close) and fluctuations, there was no clear demarcation, so clearly the mouth ratio is better.

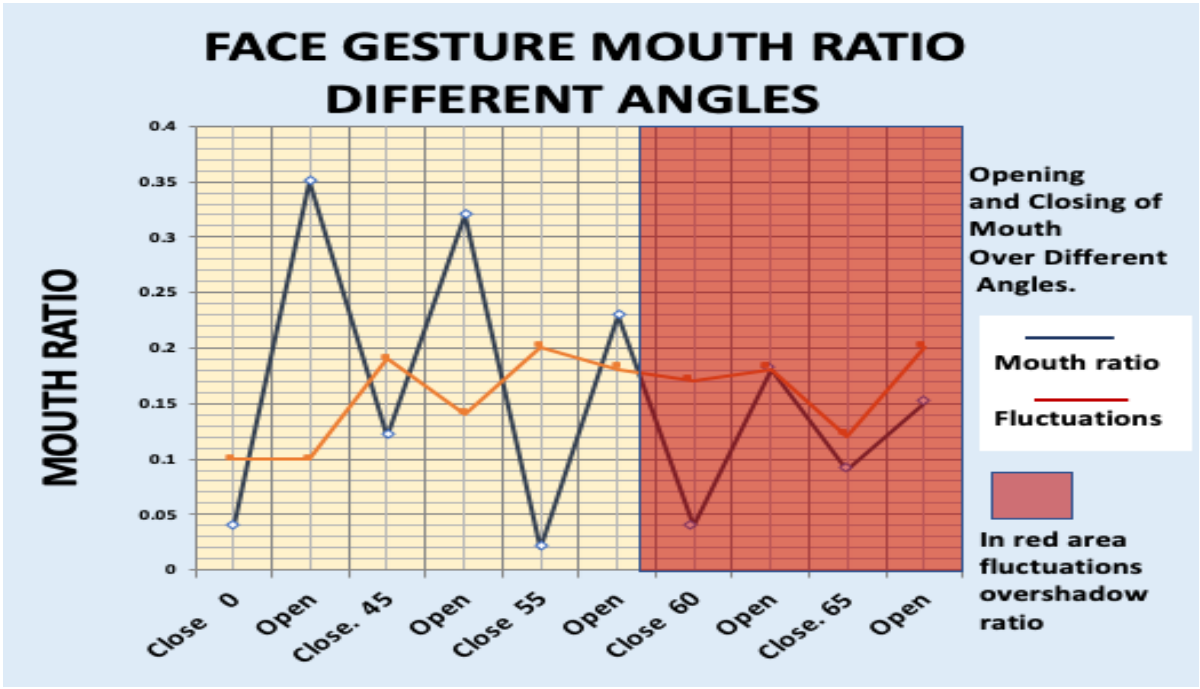


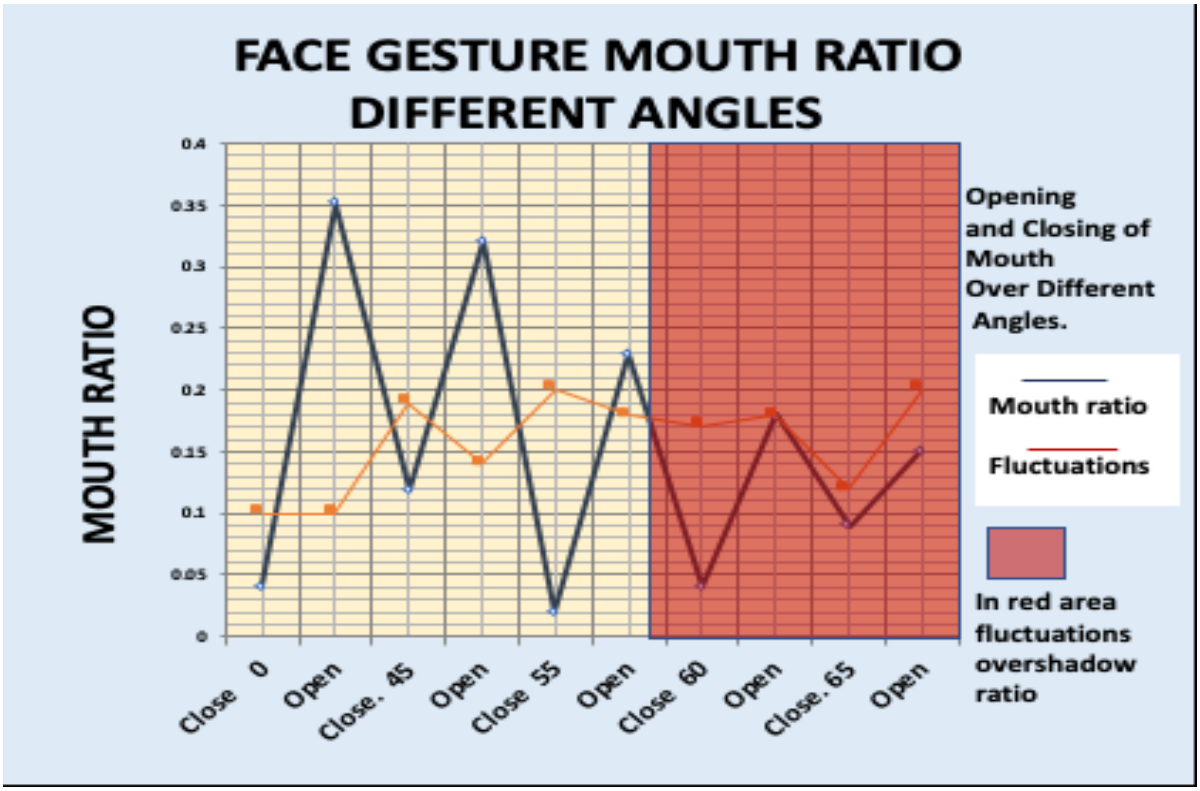
Photographs showing testing of face gesture eye open and close detected by my algorithm and how the programmed ratios change



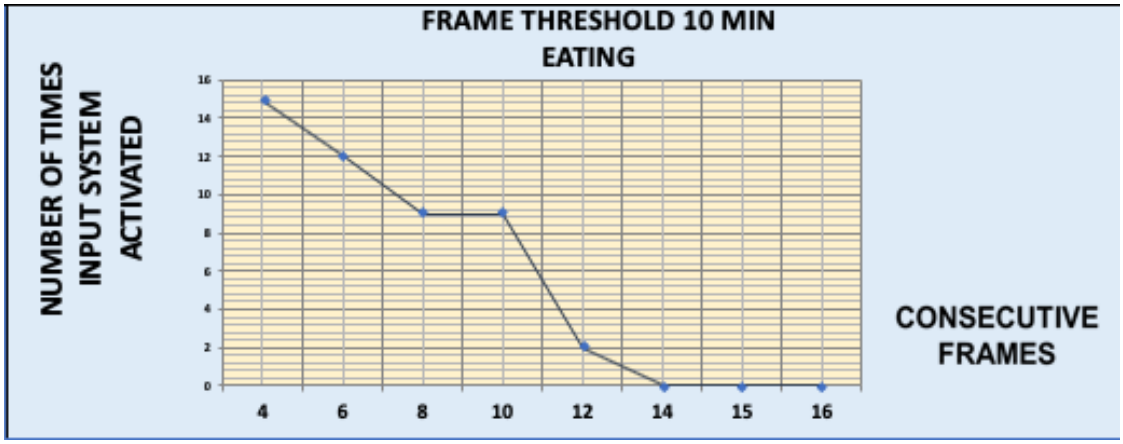
Photographs showing testing of face gesture mouth open and close detected by my algorithm and how the programmed ratios change

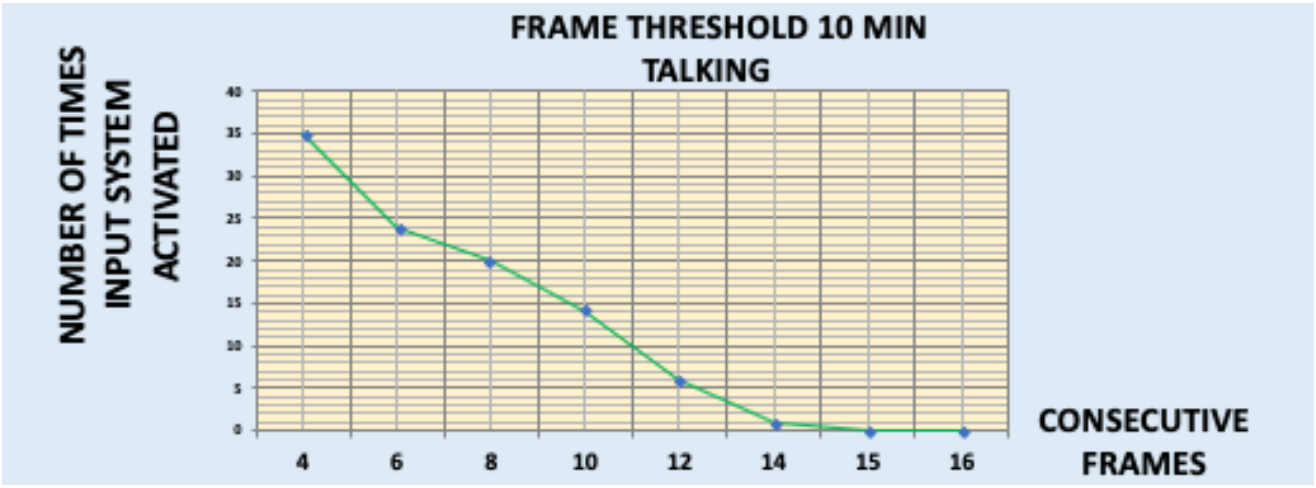

FACE GESTURE MOUTH RATIO
MOUTH RATIO
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Close
Open
Close
Open
Close
Open
Close
Open
Close
Open
Open/Close
Fluctuations


FACE GESTURE MOUTH RATIO
DIFFERENT ANGLES
MOUTH RATIO
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Close 0
Open
Close. 45
Open
Close 55
Open
Close 60
Open
Close. 65
Open
Opening and Closing of Mouth Over Different Angles.
Mouth ratio
Fluctuations
In red area fluctuations overshadow ratio



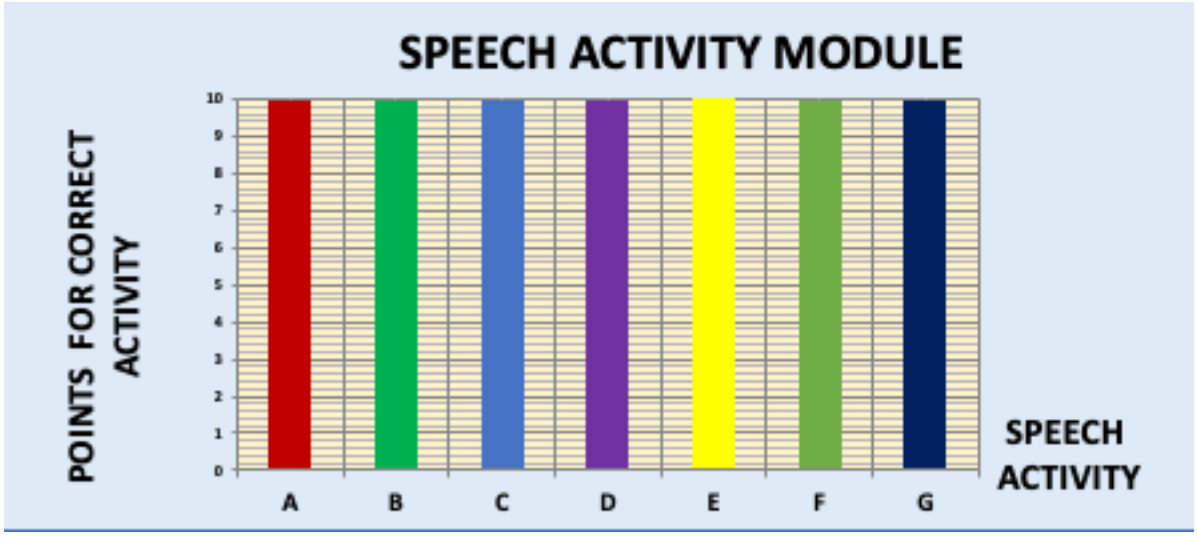
5.6. The input system should be activated only when it is intended to be activated by face gesture, not just by ratio fluctuation or other normal opening/closing of the mouth when person is eating, or for other normal movements like sneezing, or face expressions. So for doing this, user needs to keep the mouth gesture signal for a continuous number of frames in which mouth threshold is detected, signaling user intention of activating the system, too high of a value is also not good, as it will delay the trigger of the input system. This test was done for 10 minutes at different thresholds while eating and talking. The value of 15 frames was found to be good.

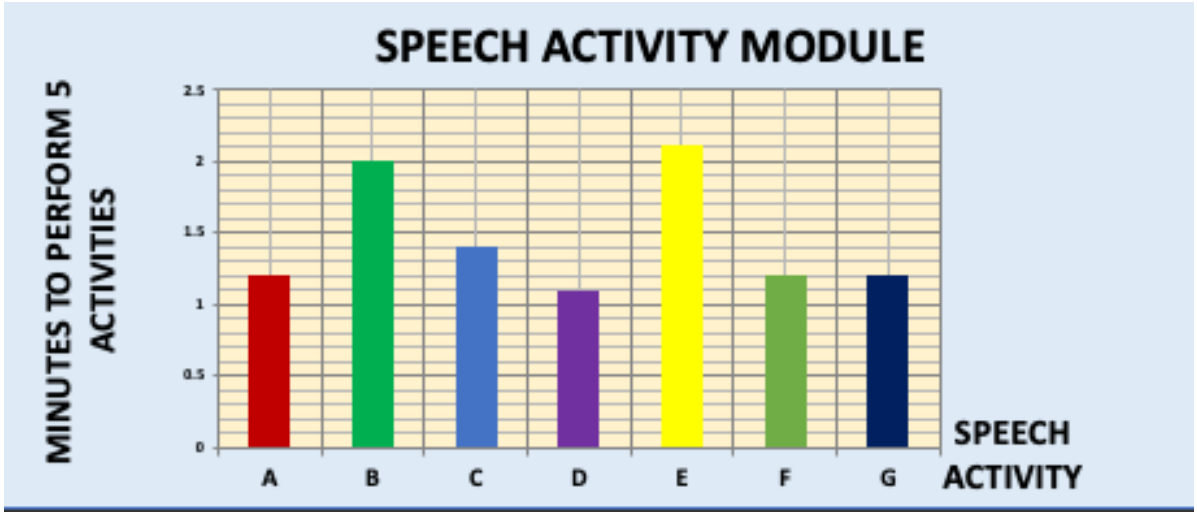




5.7. The Speech Module was tested seven times. Five random speech tasks were given, and the task was started by unlocking the input system and navigating to the speak module, performing each task using gesture control. Also, after completing the task, the user had to come back to the mode selection screen and lock the system. Different light conditions/backdrops were used for performing this task. The time was noted for completing the task. The number of mistakes that were done was also noted on a point

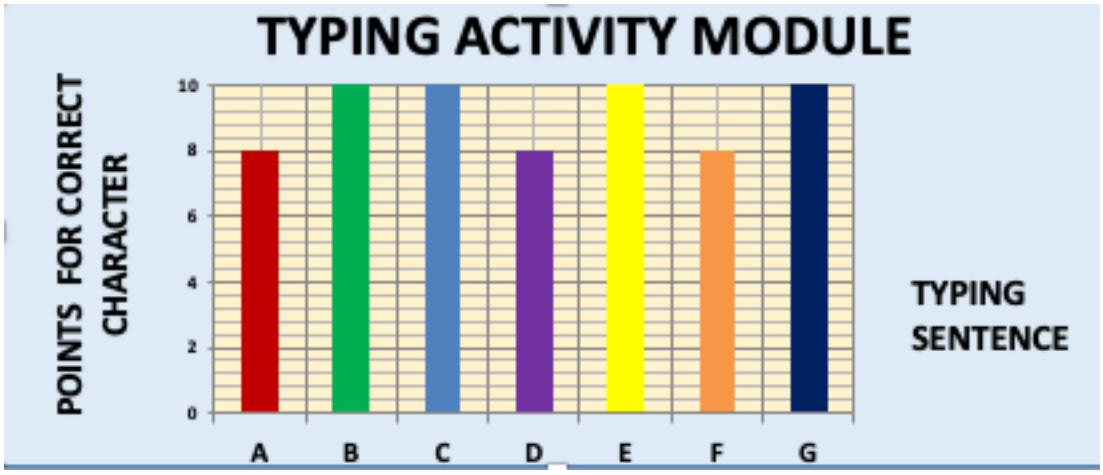
system; with each mistake, two points were taken off, so five mistakes would result in zero points. The user could also get a negative number of points.

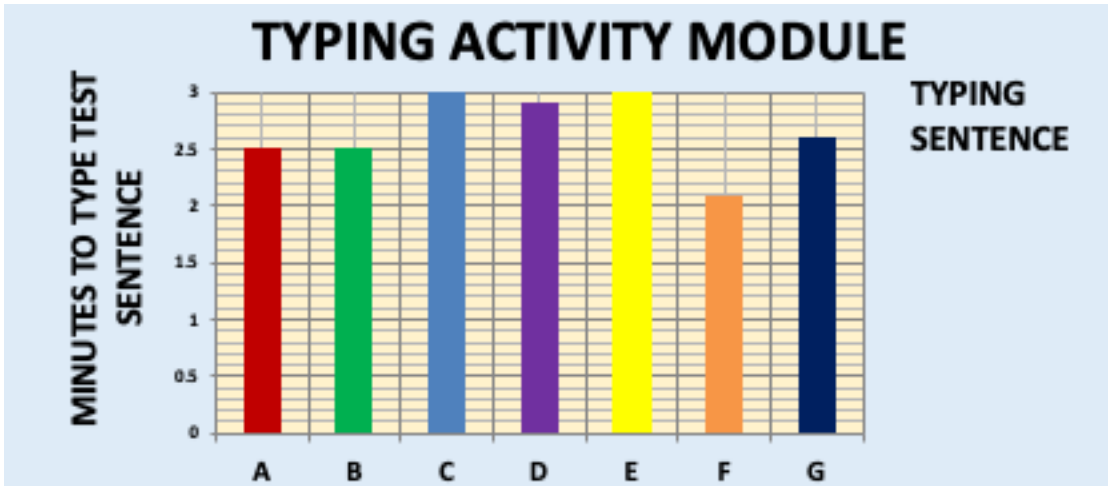




5.8. The Typing Module was also tested seven times. The user was asked to unlock the input system and navigate to the typing module and then type seven randomly chosen sentences. Using gesture controls, after completing the task, the user had to come out

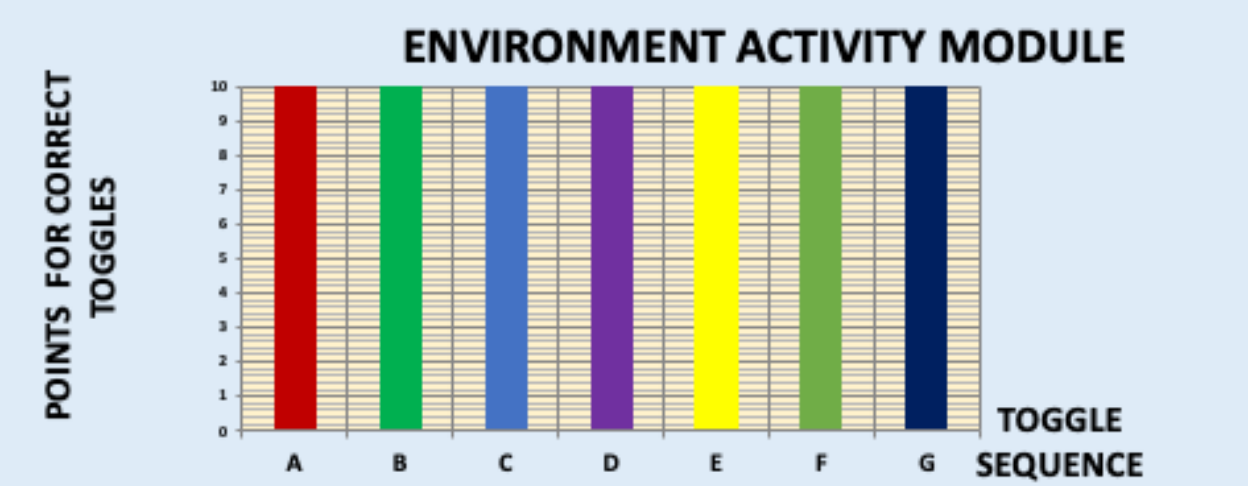
by locking the system. Different light conditions/backdrops were used for performing this task. Time was noted for completing the task. How many mistakes were done was also noted on a point system; with each mistake, two points were cut, so five mistakes would result in 0 marks. The user could also get negative marks.

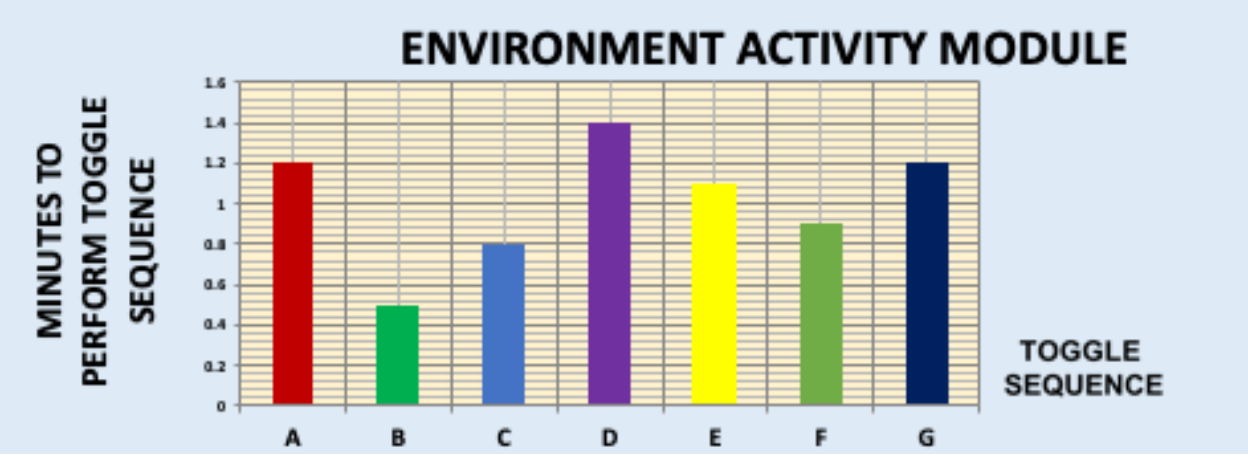




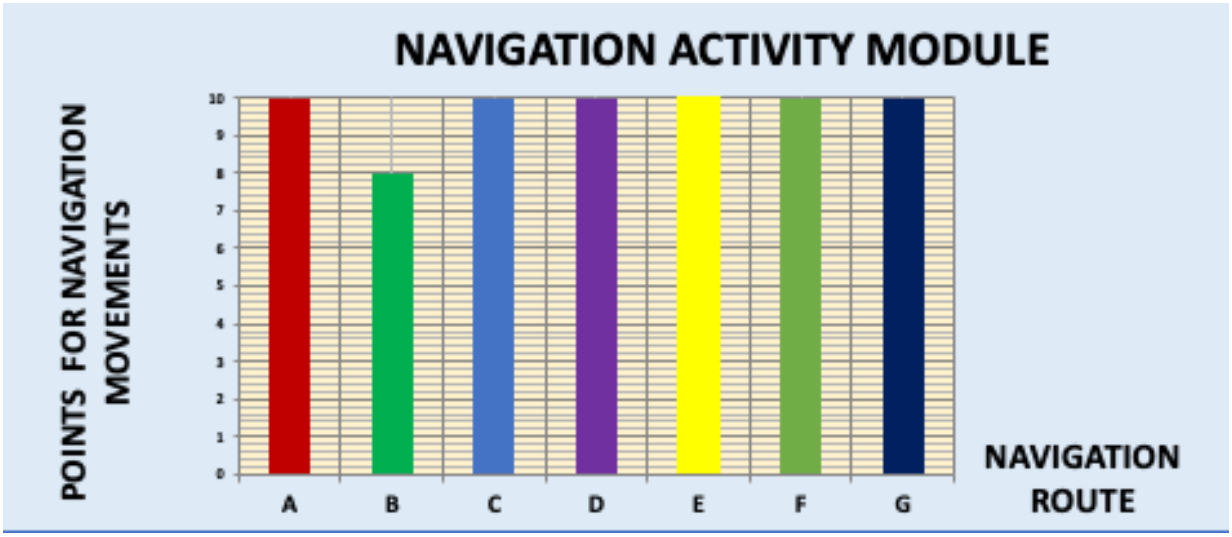
5.9. The Environment Module was tested seven times. The user was asked to unlock the input system and navigate to the environment module, and then perform seven different sequences of toggling the light and fan on/off using gesture controls. After

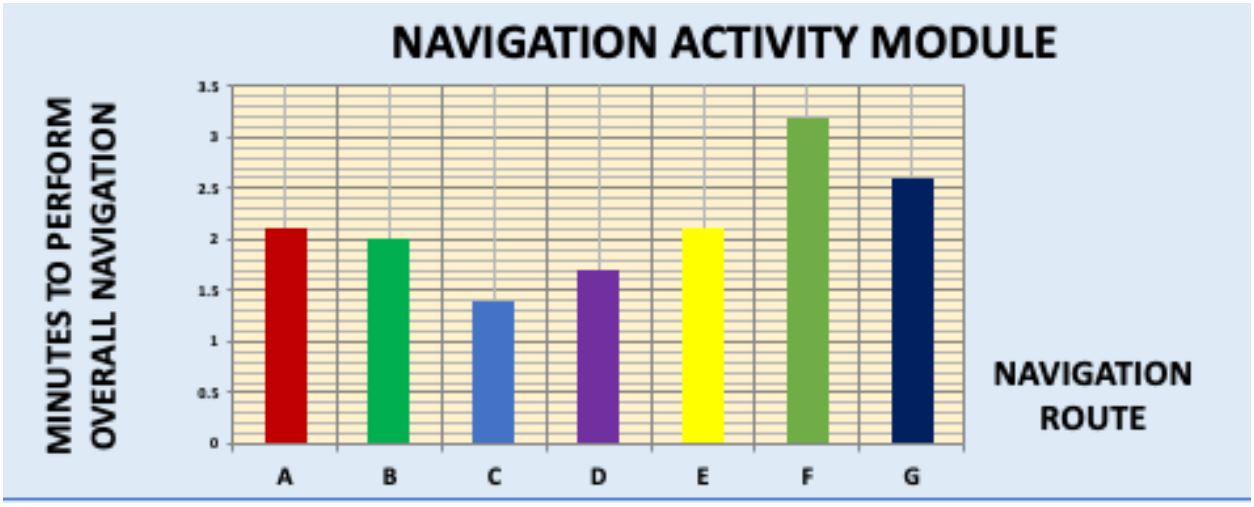
completing the task, the user had to come out by locking the system back. Different light conditions/backdrops were used for performing this task. The time was noted for completing the task. The number of mistakes that were done were also noted on a point system; with each mistake, two points were taken off, so five mistakes would result in zero points. The user could also get a negative number of points.





5.10. Similarly, the Navigation Module was tested seven times. The user was asked to unlock the input system and navigate to the navigation module, and then start the toy car and navigate it in a rectangle using gesture controls.





6. How System Works

6.1. When the system is started, the face recognition and landmark models are trained using the training data. Then, the system gets the video capture handle using the OpenCV library; the system gets each frame from the video capture, which is a bitmap.

6.2. The frame bitmap is then passed into the trained facial recognition network, which identifies the coordinates of the face rectangle.

6.3. Then, the trained facial landmark network which will identify the facial landmarks is given the whole frame bitmap, along coordinates of the facial rectangle. The network identifies the 68 facial landmark points in that rectangle, which signify facial features classified into categories like the nose, eyes, mouth, eyebrows, and chin.

6.4. A facial gesture signal can be mathematically defined as any sort of change in a facial landmark position; in particular, I have identified them by ratios. The program detects that the mouth is open (the mouth ratio exceeds a certain tunable threshold) for 15 consecutive frames.

6.5. As soon as the mouth gesture is activated, it will initialize one constant variable as the initial nose center coordinate which will serve as an anchor point for the activity selection.

6.6. In subsequent triggered frames, the program gets the instantaneous nose center coordinate and plots a line between the initial nose center coordinate and this coordinate to give the user feedback of the nose movement, which they will use to navigate through the input system.

6.7. Along with the frozen initial nose coordinate, the program will also draw rectangle around the nose. This rectangle will be broken up into different segments, establishing an effective feedback mechanism system.

6.8. Activity inputs can be triggered in the rectangle navigation system by mathematically determining when the nose line crosses a segment.

6.9. Then, in order to take the next input, the system waits for the absolute Euclidean distance between the current nose point and the initial frozen nose center coordinate to be less than a certain tunable threshold before the next trigger.

6.10. Labels signifying the corresponding activities are also printed for each segment of the visual activity selection rectangle, and the typing module also displays the words that are being typed by the user's nose movement, which are saved by a string buffer variable in the program.

6.11. In speak activity module input selected by user is spoken using speech api.

6.12. For environmental control, there are two parts. One part of the program runs inside the Arduino which has a Bluetooth receiver and is listening for signals from the other part of the program which is controlled by input system. When it sends a 1 using the Python serial library through Bluetooth, an LED light is turned on, which triggers the fan on, etc.

6.13. Similar technique is used for navigation module.

7. Engineering Design Decisions

7.1. A big challenge faced was finding a way to use face gestures in a reliable and easy to learn way to take inputs. For this, I have used a single-frame mouth gesture based on the aspect ratio threshold, along with a navigable visually segmented rectangle to take in activity inputs. The system is designed to be extensible if the user wants to integrate more activity modules. In order to select various activities, the multi-frame nose gesture is used, with an initialized nose anchor point and a locking mechanism. This is reliable, as it uses the mathematics of Euclidean coordinate geometry to trigger inputs, as detailed further in the procedure.

7.2. Because there are many algorithms available for face detection, I wanted to use an algorithm that could reliably detect different types of faces at different angles and light conditions; however, the algorithm should also be fast in its calculations. Moreover, lag must be minimized, as the camera captures real-time binary bitmap frames, instead of just using a single frame. After experimentation, it was found that the OpenCV DNN model was the best for face detection, as it demonstrated speed, reliability, and accuracy throughout each test.

7.3. Another challenge faced was determining a precise, reliable way to identify face gestures for people with different facial shapes. For example, some people may have a bigger nose and a smaller mouth. The goal is for the system to perform accurately, independent of these differences. To solve this, I employed a novel landmark ratio detection mechanism for gestures, which correlates each gesture with the surpassing of

a specific ratio threshold. Because only ratios are used in the process, if a person has a small mouth, his or her mouth open ratio will be similar to that of a person with a bigger mouth. For example, the mouth gesture is activated if the mouth is open wide enough to surpass the threshold ratio for 15 consecutive frames. For the nose, I use the nose center position as opposed to a landmark ratio. After experimenting, I decided not to use eye ratios because someone might be wearing glass, an eye gesture may be hard to perform for some people, and the change in eye ratio was found to be harder to detect.

7.4. The next challenge was that the face landmark trained network takes in the input in a certain way, using a Numpy array to gather the facial landmarks. It was necessary to find a way to get the face rectangle landmarks from the Numpy array and create a compatible, easy-to-use data structure. This was done by breaking down the Numpy array into different arrays for different facial features, making it easier to get the information mandatory to compute gesture ratios.

7.5. Another challenge I faced was finding a way to remove jitters at the edge of the thresholds, so that the duplicate events are not triggered. This was done by adding a Euclidean distance threshold for the nose, so that, in order to initiate two activities consecutively, the user must bring their nose to a certain proximity from the central anchor. Through testing, this was found to be a reliable method to prevent duplicated inputs.

7.6. There can be random fluctuations in the ratios, also during the daytime, but those fluctuations will happen for one to five frames and drop down. By setting the frame

threshold to 15, the system filters out the random fluctuations from the input method. Similarly, as the face angle with the camera has an extreme divergence for different angles. For example, if the face is positioned too high or low, then the ratios can also result in random fluctuations which can stay over a couple of frames. I have tested that at 15 frames, all of these random fluctuations are filtered 99% of the time; however, it is still possible that, when you are moving your face downward, the fluctuations can increase over 15 frames, triggering the system. To safeguard this, a password sequence needs to be inputted before all of the modules can be activated. If somebody puts a password incorrectly once, perhaps because of random fluctuations, then the system will deactivate back

8. Conclusion

Overall, the device is found to be very accurate and reliable, easy to use for all individuals, as demonstrated through the numerous tests done on the device. Also I have designed a very unique input system of navigable, segmented rectangle mechanism which works with nose and mouth gesture to take input very reliably, fast and intuitively. It will be very beneficial in improving the quality of life of disabled people as it fulfills their needs of communicating to the external world using speech and typing as well as making them independent by using the device for robotic navigation and wheel chair. The device is very extensible, as it can be extended to cover more activities with pluggable modules with reusable code. The total cost

of the device is $88, divided up into $20 for an Arduino Uno, $8 for the power relay switch, $25 for a camera to use for capturing frames, $10 for the Bluetooth module, and $15 for a computer Bluetooth USB. The rest of the system is open source.

9. Future work

In the future, I plan to extend the device by adding face identification, so that the device can work well in isolating the activities of multiple users. The overall segmented rectangle input system can also be implemented on a tablet, making it portable and easy to access over many different geographical locations. I also plan to add different languages, enabling it to be used by disabled people that do not understand English, based on their Native language. The overall software will also be put on a website portal so that it can be made open source. Currently I am in the process of talking to professors: I have emailed them about publishing the input model so that it can be used in the future. I will also contact caregiver organizations so that, if they want, I can assemble it, load the software, etc. As the number of modules increase, the cost will also increase, so they can select how many modules they want to use. I also plan to try the system using a wheelchair.

10. References

Kazemi, V., & Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2014.241

11. Bibliography

Hung, G. (2020, February 11). L706077/DNN-Face-Recognition-Papers. Retrieved February 12, 2020, from GitHub website: https://github.com/L706077/DNN-Face-Recognition-Papers

Detecting Faces (Viola Jones Algorithm) - Computerphile. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=uEJ71VlUmMQ

Face Alignment - How MSQRD, Snow, Snapchat works - Part 1/2. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=SLLZr9IjPcE&t=865s

OpenCV library. (2019). Retrieved from Opencv.org website: https://opencv.org/

OpenCV TUTORIAL with Python Series #1 - Basic Grayscale - 1. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=YY9f-6u2Q_c&list=PLEsfXFp6DpzRyxnU-vfs3vk-61Wpt7bOS&t=0s&index=9

HOW TO: Bluetooth Arduino connecting to Python on desktop. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=3tcn496oxnk&t=515s

Control High Voltage Devices – Arduino Relay Tutorial. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=LLFQ8sBWc80

ArduinoBasics: 4 Channel 5V Relay Module Tutorial. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=ic0PXlG36d4

IoT Power Relay. (n.d.). Retrieved February 12, 2020, from www.youtube.com website: https://www.youtube.com/watch?v=4R-kp6ghQZc