SMART WHEELCHAIR FOR PARALYZED

A PROJECT REPORT

Submitted by

KNR18EE024. JASIRA T P

KNR18EE034. NAYANA RAMESHAN

KNR18EE046. SHRAVAN SREEDEEP

KNR18EE048. SOORYA P

to

the APJ Abdul Kalam Technological University

in partial fulfilment of the requirements for the award of the Degree

of

Bachelor of Technology

in

Electrical & Electronics Engineering

Department of Electrical & Electronics Engineering

Government College of Engineering Kannur- 670563

JUNE 2022

SMART WHEELCHAIR FOR PARALYZED

A PROJECT REPORT

Submitted by

KNR18EE024. JASIRA T P

KNR18EE034. NAYANA RAMESHAN

KNR18EE046. SHRAVAN SREEDEEP

KNR18EE048. SOORYA P

under the Supervision of

Dr. C SREEKUMAR

to

the APJ Abdul Kalam Technological University

in partial fulfilment of the requirements for the award of the Degree

of

Bachelor of Technology

in

Electrical & Electronics Engineering

Department of Electrical & Electronics Engineering

Government College of Engineering Kannur- 670563

JUNE 2022

DECLARATION

We undersigned hereby declare that the project report entitled "SMART WHEELCHAIR FOR PARALYZED", submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Dr. C Sreekumar. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: DHARMASALA

Date: 29/06/2022

JASIRA TP

NAYANA RAMESHAN

SHRAVAN SREEDEEP

SOORYA P

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGG.

GOVERNMENT COLLEGE OF ENGINEERING KANNUR-670563

CERTIFICATE

29/06/2022

This is to certify that the Project Report entitled **SMART WHEELCHAIR FOR PARALYZED** submitted by JASIRA T P, NAYANA RAMESHAN, SHRAVAN SREEDEEP, SOORYA P of 8th semester Electrical and Electronics Engineering, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Electrical and Electronics Engineering is a bonafide record of the project work carried out by this project group under my guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Prof. ASOKAN O V	Dr. C SREEKUMAR
(Project Coordinator)	(Supervisor, Head of the Department EEE)

ACKNOWLEDGEMENT

With regards to completion of final year project, we wish to acknowledge the following people who were the source of motivation behind it.

We are grateful to **Dr. V O REJINI** principal, Govt. College of Engineering Kannur, for providing permission and the necessary facilities to complete our project in a systematic way. We would like to express our esteemed gratitude to **Dr. C SREEKUMAR**, Head of Department of Electrical and Electronics Engineering and our guide for providing all the support and guidance throughout the project and also for his valuable suggestions.

We would also like to thank our project coordinator **Prof. ASOKAN O V** for providing insights and pushing us to excel ourselves. We are grateful to **Prof. ANJALI ANAND K** for guiding and supporting us through our initial stages of this project. We would also like to express our sincere gratitude to all teaching and non-teaching staff of Govt. College of Engineering Kannur for their valuable help in the successful completion of our project.

Our acknowledgment would not be complete without acknowledging our gratitude to our beloved parents who have been the pillars of support and constant encouragement during the work of this project. We would also like to express our sincere gratitude to all our friends for their valuable criticism and suggestions. Last but not least, we also want to extend our appreciation to those who could not be mentioned here but have well played their role to inspire us behind the curtain.

JASIRA T P

NAYANA RAMESHAN

SHRAVAN SREEDEEP

SOORYA P

ABSTRACT

Paralyzed stroke patients are unable to normally communicate with their environment. For these patients, the only part of their body that is under their control, in terms of muscular movement, is their eyeballs.

The biggest problem that paralyzed patients face is leading their own life without others support. This include basic day to day operations like switching on basic devices like fan, bulb etc.

An automated working prototype of a smart wheel chair working with a home automation system that can be controlled by eye tracking is implemented in this work. The prototype is designed for the paralysed people with only motor functions for eye movement. This method takes care of surrounding obstructions and decisions are taken accordingly.

Key-words: Eye Blink, Microcontroller, Home Automation, Eye Tracking Techniques, Open CV, Electric Wheelchair, MediaPipe

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

IC	Integrated Circuit
GPIO	General Purpose Input / Output
HDMI	High-Definition Multimedia Interface
LAN	Local Area Network
OpenCV	Open Source Computer Vision
PWM	Pulse Width Modulation
RAM	Random Access Memory
RPI board	Raspberry Pi Board
USB	Universal Serial Bus
VSC	Visual Studio Code

NOTATIONS

g	Acceleration due to gravity, m/s
Q_x	Battery capacity, Ah
χ	Displacement, m
E	Energy, kWh
i_{tot}	Gear transmission ratio
m	Mass, kg
N	Maximum vertical force, N
P	Power, Watts
r_t	Radius of the wheel, m
μ_r	Rolling resistance coefficient
F_r	Rolling resistance, N
t	Time, hours
M_k	Torque on wheel, Nm
T_r	Torque, Nm
V_t	Velocity, m/s
F_z	Vertical force, N
V	Voltage, V
W	Work, Joules

CHAPTER 1
INTRODUCTION

According to the latest report prepared by the World Health Organization and the World Bank, 15 percent of the world's population is disabled. The number of individuals who are paralyzed and thus hooked on others due to loss of self-mobility is growing with the population. The advancement of the wheelchair for incapacitated clients is shockingly late, beginning with the customary physically controlled wheelchairs and progressing to electric wheelchairs. Traditional wheelchair use focuses exclusively on manual use by users, allowing users to still use their hands, excluding those who cannot. So, the idea is to create a system that can track the motion of the eye and move the wheelchair accordingly. The use of power-driven wheelchairs with high navigational intelligence is a great step to integrate severely handicapped and mentally ill people. Different systems are being developed, allowing the end-user to perform safe movements and accomplish some daily life important tasks. The notion is to create an Eye Monitored System that allows wheelchair navigation depending on the movement of the eyes. Eye-tracking is the process of measurement or point of focus (when one is looking) or the movement of an eye relative to the head. Eye trackers are increasingly used for rehabilitative and auxiliary applications (e.g., for wheelchairs, robotic arms, and prostheses control). In this model, we use the image processing system to control the wheelchair. The user's eye movements are transformed into a screen position using a camera without any direct contact. This is an eco-friendly and cost-effective wheelchair that dissipates less power and can be made using minimum resources.

Camera captured the image in real time and analysis the image as input to set the commands for interface the motor driver IC through sending the commands to GPIO pins. The motor driver circuit is used to perform the different operation such as left, right, forward and stop. For the advance level of Image Processing open computer vision (OpenCV) library is used for Face and Eye detection. Google's MediaPipe library is used to find out accurate pupil location detection and tracking of that.

An Eye tracking technique, which capture the image and detects the presents of human face. After detecting the face, it detects area of the eye location on the face detected image, and performs several operations of basic image processing like colour

image to grey conversion, filtering, threshold, pattern matching, noise reduction and circle detection on it.

The Raspberry Pi board is used to perform the control of the complete system operation. Digital Image processing-based output signal sent to the Raspberry Pi board. The Raspberry Pi acquired the data and analyse it. Raspberry Pi sends the control signal to motor driving circuit based on the location of eye pupil. In a Wheelchair two individual motors are embedded on each wheel.

CHAPTER 2
LITERATURE REVIEW

Diksha Raj, Anish Kalra, Sreejith Krishna N, Suhriday Roy, Dr. Shreenath K N, research comprises a wheelchair that works on hand gestures, voice command inputs, app inputs (Keypad), and head movements input for locomotion of the differently abled persons. This project can be upgraded in terms of speed and the lag can be overcome by replacing them with more advanced tools and sensors.[1]

Dola Das, Prottoy Saha and Nawshin Tabassum, proposed in their research that Ultrasonic Sonar sensors measure the distance and velocity of the movable object from the cane[2]. Their system notifies the user using vibration and audio message depending on the direction of the coming objects, distance and velocity. Yi-Chen Lee, Ching-Min Lee [3], research combines Raspberry Pi with the Internet of Things and the foundations of artificial intelligence to develop a real-time smart home surveillance system to improve safety at home. The main method is to connect the Raspberry Pi to a network sharer or computer with a fixed IP and then input it into a computer or mobile phone to achieve remote control.

To provide home safety for the elderly or the challenged people, the proposed system combines a voice control module to improve the user convenience. The authors Muhammad Azlan Alim, Samsul Setumin, Anis Diyana Rosli, Adi Izhar Che Ani, [4] proposed a voice recognition-based intelligent wheelchair system for physically disabled people who are unable to control the wheelchair by their upper and lower limbs. This development employs voice command to controls the movement of the wheelchair in different directions. V. Usha Rani, Dr.J Sridevi, P. Mohan Sai, [5], developed a robot that can move to any location within the range of the network and can be accessed globally from anywhere and as it uses only one camera to secure a large area it is also cost-efficient. At the core of the system lies Raspberry-pi which is responsible for all the operation of the system and the size of the device can be engineered according to the area it is to be used.

Reona Cerejo issued proposed on Arduino circuit. The whole system is controlled by the Arduino. Arduino is a simple microcontroller board and open-source development environment that allows to make functional and creative projects by using

Arduino microcontroller and software. And make the system affordable. And this paper also more concentrate on finds the direction in which eye look using MATLAB frame. Depending upon the location of pupil in these blocks action is performed [6].

Many disabled patients can operate joystick-controlled electric wheelchairs however there are numerous individuals who don't have the prestidigitation required to work a joystick. Additionally, a significant number of them face trouble to avoid impediments. The point of this paper [7] is to carry out a multi-control framework to control the development of the wheelchair by coordinating finger development global positioning framework, a little vocabulary speaker subordinate based word acknowledgment framework and a gathering of observing sensors to keep away from obstructions. Additionally, a joystick control framework is likewise carried out to work with the patients that can utilize the joystick.

Ruzaij et al. introduce a wheelchair controller for Quadriplegia and paralyzed amputee on the basis of their voice or head tilt controller [8]. In [9] "automatic Wheelchair Controlled using Hand gesture", an EMG Sensor, and guide Signal Separation" can be used in this method. A system is designed which uses an IR sensitive camera to identify the gesture shown by the user. The capture images of the gesture are given to the microprocessor which does further processing. The drawback of this method is that it cannot be used by the persons who are suffering from nerve is order and stoke etc.

CHAPTER 3

WHEELCHAIR DESIGN

3.1 TECHNICAL SPECIFICATIONS

During the movement of a wheelchair, there are some resistances that appear. These resistances are variable and depend on various parameters as described below:

- Rolling resistance
- Aerodynamic resistance
- Inertial resistance

In the wheelchair, aerodynamic resistance and inertia resistance are considered insignificant because the accelerations and speeds which are developed on the vehicle are kept to a minimum.

3.2 CALCULATION OF ROLLING RESISTANCE

When an unformulated wheel is wrapped in unformed ground or, respectively, a fully resilient wheel is wound on a fully developed ground, there is no resistance to the movement. These are the ideal cases, as in reality both the ground and the wheel suffer from deformations. The developed torque which is created equals to:

$$T_r = N*a \qquad \text{---------- } (3.1)$$

where N is the maximum vertical force in Newtons and a is distance from a theoretical axial line of the wheel in meters. To balance this torque, it is necessary to develop a torque opposite to this. This torque equals the product of the pulling force F_r and the dynamic radius of the wheel. The vertical force (F_z) which expresses the mass of a body in strength equals to:

$$F_z = m \times g \qquad \text{---------- } (3.2)$$

where m is the body mass and g is the acceleration of gravity. The rolling resistance F_r is described by the following formula:

$$F_r = F_z * \mu_r \qquad \text{---------- } (3.3)$$

where μ_r is the rolling resistance coefficient which is estimated for the wheelchairs with non-pneumatic wheels on concrete, with the value of $\mu_r = 0.015$. For the calculation of the wheelchair rolling resistance is taken into account the mass of:

- Wheelchair, $m_1 = 7kg$
- User, $m_2 = 100kg$
- Battery, $m_3 = 2\times2.5kg = 5kg$
- Motor, $m_4 = 2\times2.35kg = 4.7kg$
- Auxiliary weight, $m_5 = 200g$

Total weight in kilogram,

$$M= m_1+m_2+m_3+m_4+m_5$$

$$M= 7+100+5+4.7+0.2 = 116.9kg \approx 117kg$$

$$F_z = M \times g \qquad \text{---------- } (3.4)$$

$$F_z = 117 \times 9.81$$

$$F_z = 1147.77\ N$$

$$F_r = F_z \times \mu_r \qquad \text{---------- } (3.5)$$

$$F_r = 1147.77 \times 0.015$$

$$F_r = 17.2\ N$$

When climbing a vehicle due to road inclination, a resulting force from the weight of the vehicle appears (Figure 3.1). This force opposes the movement of the vehicle when it is on the hill.

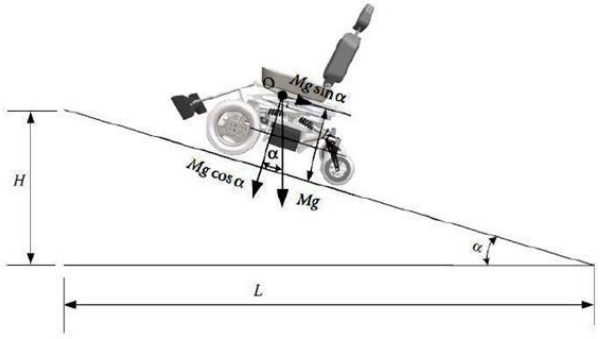


Fig 3.1 Forces in road inclinations.

The ascent resistance F_B is calculated from the following formula:

$$F_B = F_z * \sin\alpha \quad \text{---------- (3.6)}$$

where F_z is vertical force in Newtons and α is the angle of the road inclination. As it is known, the tilt of the road is the ratio of the height H to the distance L (Figure 3.1) and expresses the height to be developed at a distance of 100 km:

$$i = H/L = \tan\alpha \approx \sin\alpha \quad \text{---------- (3.7)}$$

To find the angle α, in order to calculate the ascent resistance, the right formula is the following:

$$\alpha = \tan^{-1}(i) \quad \text{---------- (3.8)}$$

Thus, for a pavement slope of 10^O the vehicle has an ascent resistance equals to:

$$F_B = 1147.77 \times \sin 10 = 199.31\ N \quad \text{---------- (3.9)}$$

Since the forces are opposing to the movement of the vehicle, it is accurate to be summed in order to find the required driving force (traction force) to be applied to the wheels in order to move it.

$$F_{total} = F_r + F_B = 17.2 + 199.31 = 216.51\ N \quad \text{--------- (3.10)}$$

where F_{total} is the total traction force which is the sum of ascent resistance F_B and the rolling resistance F_r . The torque needed on the wheel to overcome these resistors is calculated by the formula:

$$F_t = M_k * i_{tot}/r_t \Rightarrow M_k = F_t * r_t / i_{tot} \quad \text{---------- (3.11)}$$

where M_k is the torque on the wheel in Nm. The r_t the radius of the wheel and i_{tot} its gear transmission ratio from the motor to the wheel. The motors have a diameter of 12inch (308mm) with a working voltage of 24V, 250Watt and a fixed gear transmission ratio of 23:1. The maximum torque that can be produced by the motor according to its operating diagram is 20Nm and is enough to overcome the climb and roll resistors.

$$F_t = M_k \times i_{tot} / r_t$$

$$M_k = F_t \times r_t / i_{tot}$$

$$M_k = 22 \times 0.154 / 23 = 1.5\ Nm$$

3.3 CALCULATION OF REQUIRED POWER

Work results when a force acts upon an object to cause a displacement (or a motion) or, in some instances, to hinder a motion. Three variables are of importance in this definition - force, displacement, and the extent to which the force causes or hinders the displacement. Each of these three variables find their way into the equation for work. That equation is:

$$W=F*\chi \quad \text{---------- (3.12)}$$

where W is the work in Joules, F is force in N and χ is the displacement in meters.

Considering the above mathematical formulas and knowing the velocity it follows that:

$$P = F_t*V_t \quad \text{---------- (3.13)}$$

where P is the power in watts F_t is the traction force in N and V_t is the velocity in m/s. Theoretically, the power given to the vehicle, when it moves at the maximum speed (3.54m/s) it can develop (considering only the rolling resistance):

$$P = 17.7N * 3.54\ m/s = 62\ Watt \quad \text{---------- (3.14)}$$

The needed energy to drive the vehicle for one hour is calculated by the formula:

$$E= P*t => E = 0.062KWh \quad \text{---------- (3.15)}$$

where E is the energy in KWh, P is power in watt and t is the time in hours According to the formula below, the vehicle needs 2.58Ah to move one hour at maximum speed.

$$E = Q_x*V/1000$$

$$Q_x=0.062*1000/24$$

$$0.062*1000/24 = 2.58Ah \quad \text{---------- (3.16)}$$

Where E is the energy needed in Ah, Q_x is the capacity of the battery in Ah and V is the voltage of the battery used in the experiment equal to 24V.

CHAPTER 4

INTERNAL ARCHITECTURE

4.1 INTRODUCTION

In this system it is mandatory to gives the proper power supply to individual components, and the standard power supply should be used for Raspberry Pi, Arduino, camera, sensors, and motors. The Raspberry Pi board is the brain of the wheelchair. Here the modules like camera, motor driver and voice output device are directly connected to the Raspberry Pi board. And Raspberry Pi board is connected to the internet for remote access facility through Wi-Fi. In case, emergency controlling or monitoring the status of wheelchair, will be carried out by accessing the Raspberry Pi board at remote place using web-server (internet). Our system can be divided in to four parts, namely controller, programme, wheelchair and home automation.

4.2 CONTROLLER

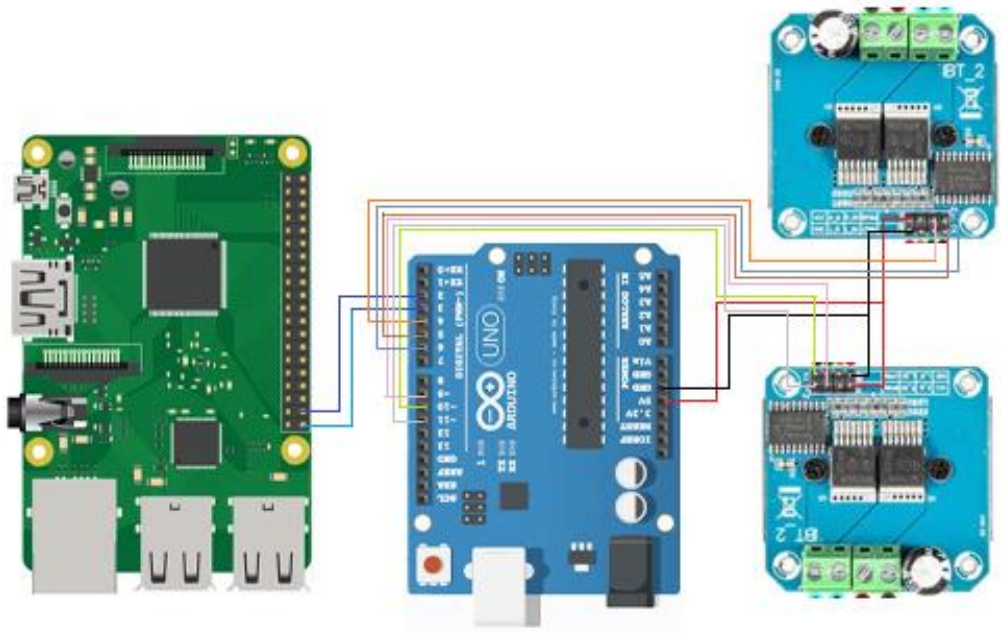


Fig 4.1 Circuit inside controller setup

The circuit diagram of the controller setup is shown in Fig 4.1. Raspberry Pi is the main controller of our prototype. It gives commands to the motor driver circuit, which enables the GPIO pins to perform several operations, such as forward, left, and right on the basis of eye ball movement. The controller arrangement is as shown in Fig 4.2.

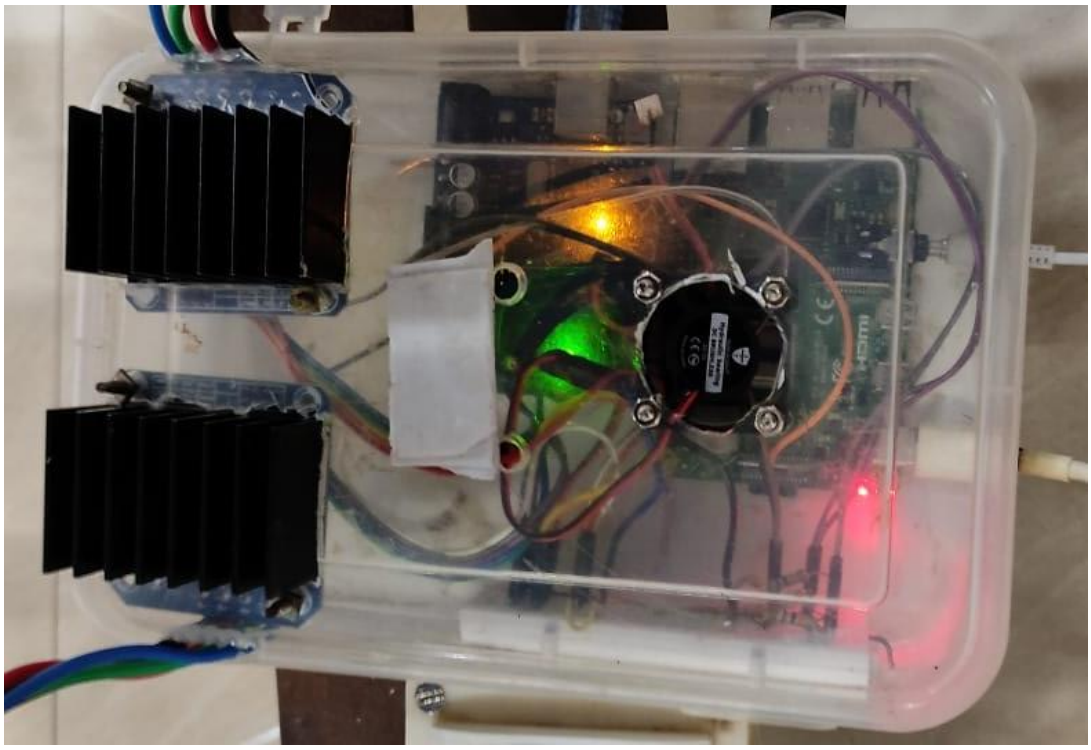


Fig 4.2 Controller unit of wheelchair

For real time video capturing and advanced image processing, a 8 MP web camera is used as shown in Fig 4.3. It is very challenging to detect the exact eye pupil location, so a new image processing technique is used for eye pupil centre detection and tracking, which works based on open computer vision (OpenCV) library. Most of the coding part is done with the help of OpenCV library. Google's MediaPipe library is used to find out accurate pupil location detection and tracking of the same. Python language is used for coding, which is user friendly and helpful to resolve the errors efficiently.

First camera module will start to capture the images. For the face detection MediaPipe library is used. After detection of proper face, it will be trying to detect the eye inside the face region of interest. And again, MediaPipe library is used to detect eye. It will draw the rectangular box over the eye. Now, the main target is to detects the eye pupil and define its centre points. There are several image processing operations performed in system, such as blur Image, colour conversion, thresholding, filtering, edge detection and Hough transform is used. For circle detection Hough transform method is used. By the usage of USB webcam, images are captured on Raspberry Pi. OpenCV library is installed in Raspberry Pi memory.

The camera module mounted in the wheelchairs arm so that it captures images of face for eye detection. The distance of camera from the face is more than 30 cm so that the position of camera doesn't affect field of vision of the user. The camera module has two-degree freedom of movement. It is adjusted before the use of wheelchair so that the camera aligns with the face for capture of images.

Fig 4.3 Webcam arrangement in wheelchair

4.2.1 Raspberry Pi 4

Raspberry Pi 4 Model B is the latest product in the popular Raspberry Pi range of computers is shown in the Fig 4.4. It offers ground-breaking increases in processor speed, multimedia performance, memory, and connectivity compared to the prior-generation Raspberry Pi 3 Model B+, while retaining backwards compatibility and similar power consumption. For the end user, Raspberry Pi 4 Model B provides desktop performance comparable to entry-level x86 PC systems. This includes a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro-HDMI ports, hardware video decodes at up to 4Kp60, up to 8GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0, and PoE capability.

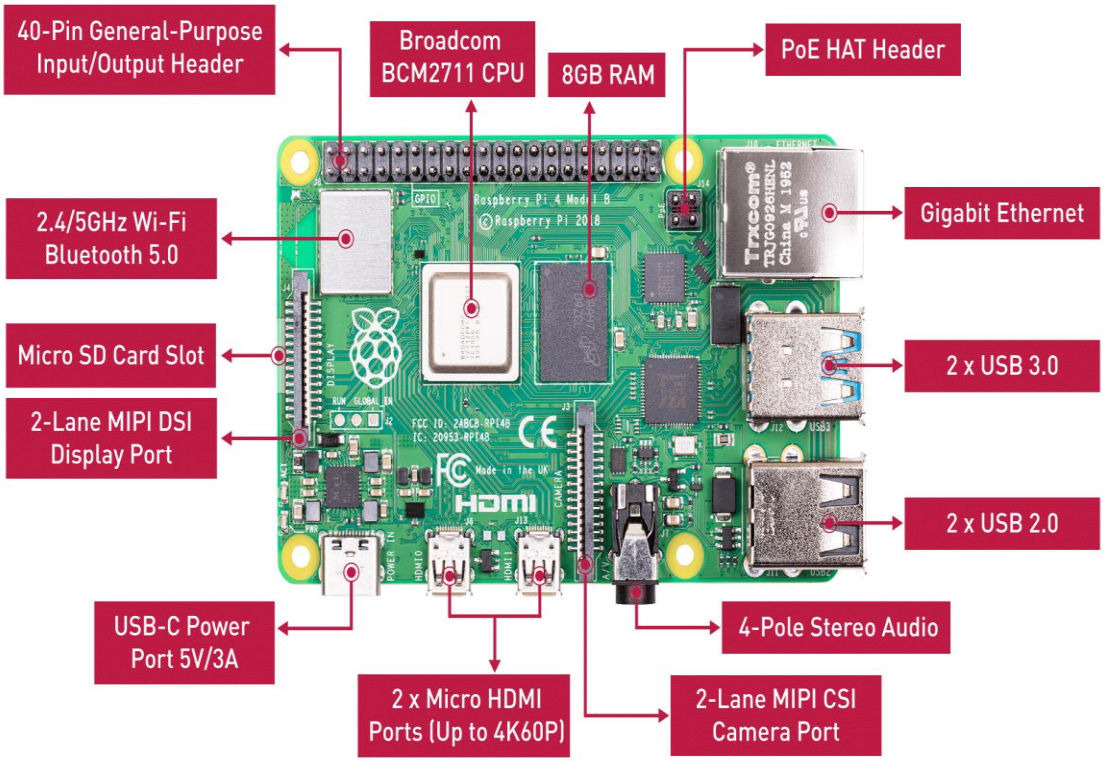


Fig 4.4 Raspberry Pi 4

4.2.2 Webcam

The webcam is a video camera as shown in Fig 4.5 that feeds or streams an image or video in real time to the Raspberry Pi 4 in order to track the eye movements. The Webcam is connected to Raspberry Pi 4 using USB port.

Fig 4.5 Webcam

4.3 ALGORITHM

The basic function of this program is eye tracking and detection of eye movement. To detect the location of eye pupil Haar cascade algorithm is used this technique comprises of several stages which are implemented to find the eye movement and also for face and eye detection, colour switching, object tracking, Hough transform, edge detection and motion detection.

Initially system captures image by making use of webcam. First step is the algorithm accurately detects the face of the user. According to the algorithm the system represents the user face in specific area of indicated image. Several process of image processing techniques is performed for eye pupil tracking. Primarily camera module captures image, face is detected using MediaPipe library which use Haar cascade algorithm. Once the face is detected it detects eye inside the face region rectangular boxes are drawn across eye once it is detected. Now detection of eye pupil and definition of pupil centre point is need for doing this several image processing techniques like colour conversion/switching, edge detection, blur image, filtering, Hough transform and thresholding is implemented, for detecting circular shapes Haugh transform is implemented, with the help of webcam, images are captured and transferred to Raspberry Pi. OpenCV library is installed to Raspberry Pi which supports the image processing technique.

To find out the pupil centre point of the Eye, we followed some steps:

4.3.1 Face Detection and Eye Detection

For the face detection and eye detection the OpenCV library is used directly. A system camera detects the face of user. Once it will be detected, system finds the eye location and marks the eye region using MediaPipe library. And system accurately detect both the eyes based on the proper distance of the each other.

4.3.2 BGR To Gray Conversion

A very next operation of the image colour convention to reduce the system delay time. The Image frame size should be low, because the processor cannot process the image frames in run time condition. So, by using the BGR to GRAY conversion a coloured image converted into grey image.

4.3.3 Features Detection and Blurring Image

The Gaussian blur filter is used for blurring the image. Which helps to detect the exact edges of specific area of the cropped image. Features is nothing but it found some special pattern on image which is unique, based on it will make a pattern.

4.3.4 Edge Detection

A canny Edge detection and corner edge detection algorithm is applied for determine the soft edges of the image. To set the proper threshold value it will allowed easy to recognize rectangles or circle presented in Image.

4.3.5 Eye Tracking

To track the Eye movements, we use projection function algorithm was used, where the coordinates system points the eye centre point location. In Fig 4.6 indicates the eye pupil location with respective coordinate's system graph.

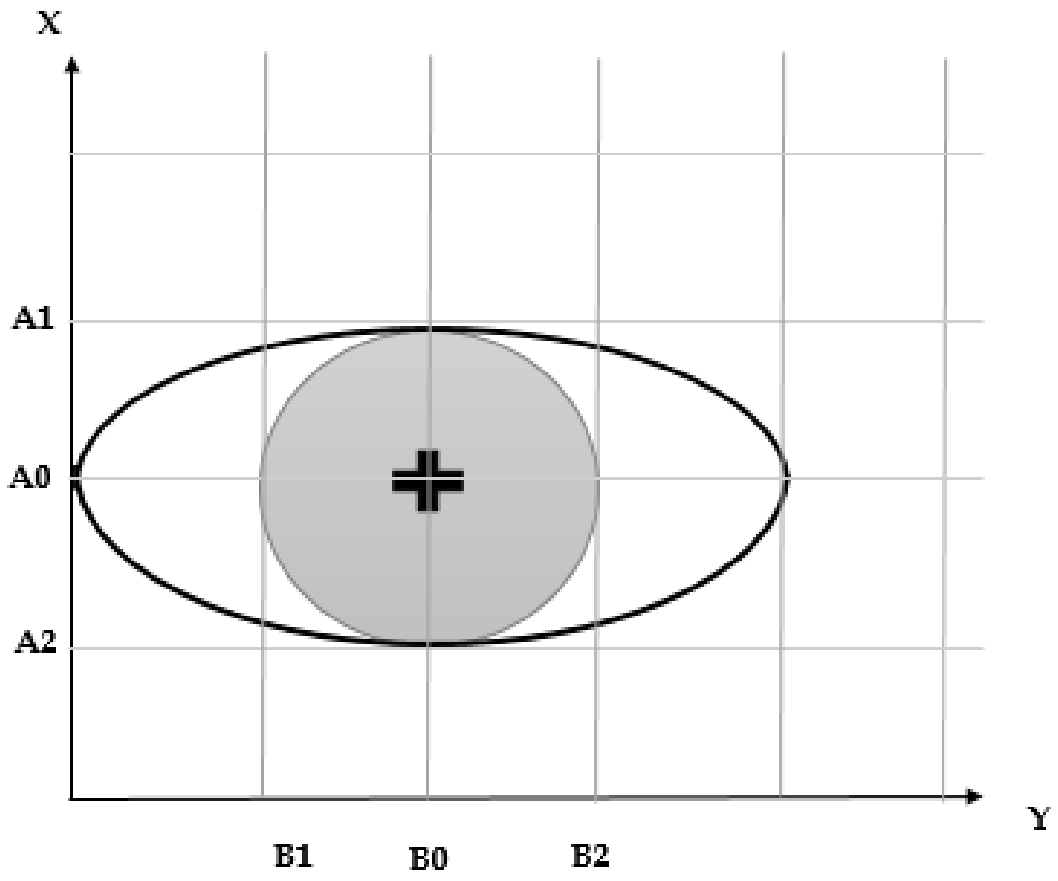


Fig 4.6 Coordinates system with respective eye position

4.4 WHEELCHAIR

An ordinary foldable wheelchair is used for the working model. It is then converted to electric wheelchair with the help of 250W 24V DC motors, BTS7960 motor drivers, and 12V 7.2Ah rechargeable lithium-ion battery. The motor drives the wheels with the help of chain and sprocket arrangement as shown in Fig 4.7.

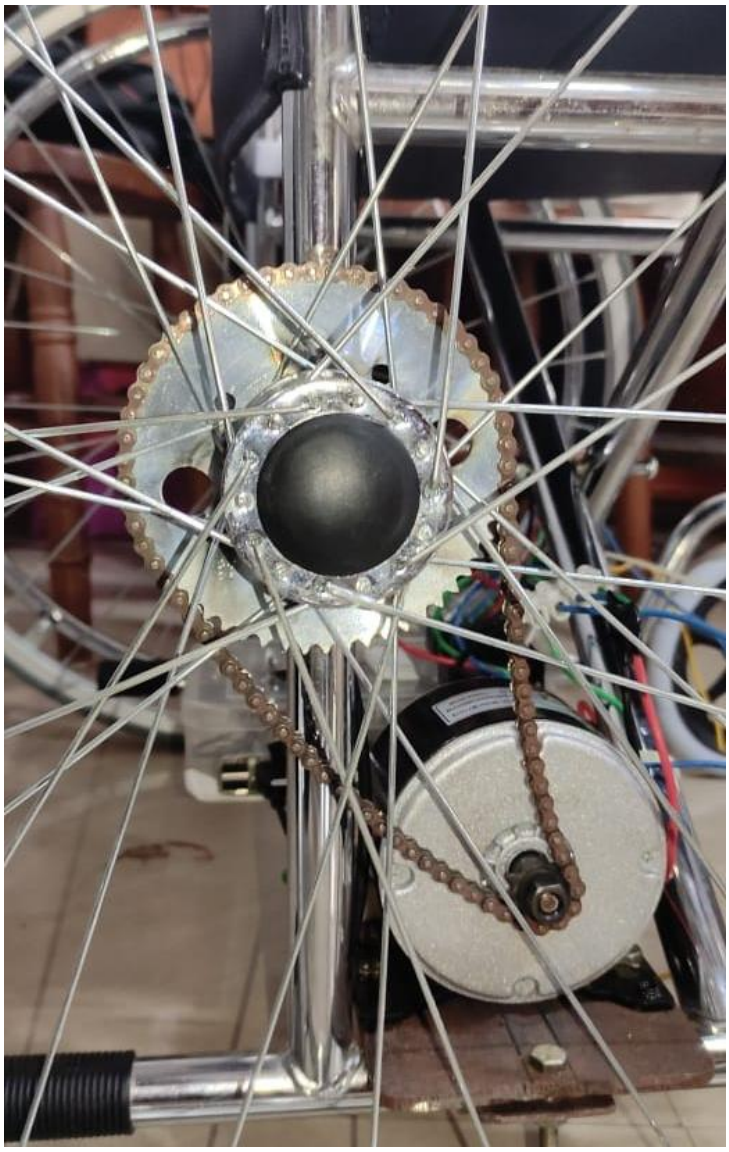


Fig 4.7 Chain and sprocket

The complete arrangement of the converted wheelchair is shown in Fig 4.8 and Fig 4.9. A platform is laid across the wheelchair to fix the motors and controller setup in them. The sprocket is brazed into both the wheels and is connected to the motor as shown in Fig 4.7. The controller setup is attached to the platform using bolts and nuts. The folding capability of the wheel chair is lost after the conversion to electric wheelchair.

Fig 4.8 Side view of smart wheelchair

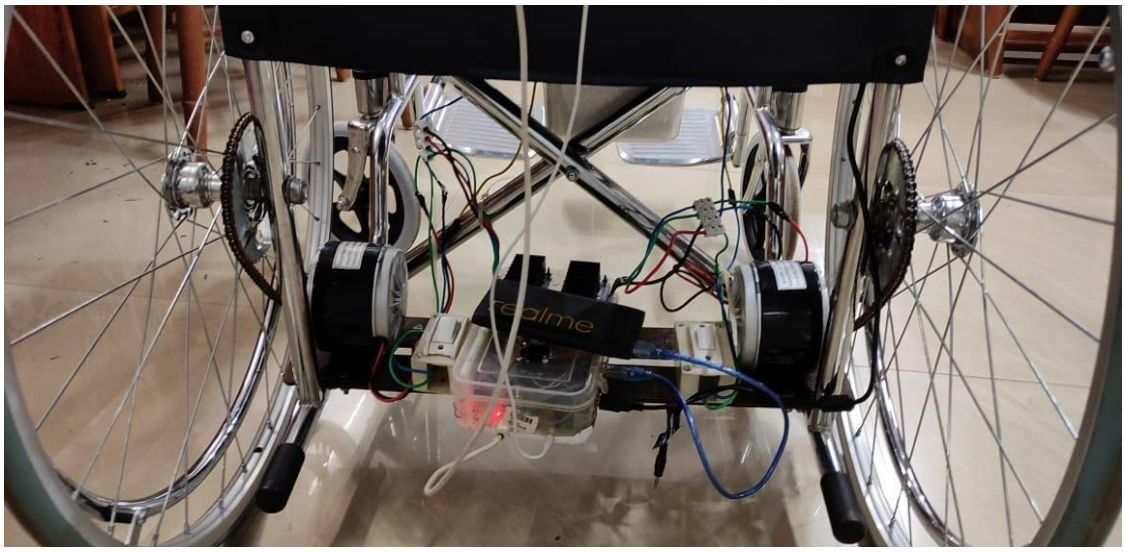


Fig 4.9 Conversion setup

The specification and working of each component are provided below.

4.4.1 Wheelchair

The manual wheelchair as shown in Fig 4.10 and Fig 4.11, is a solution for transporting old age and differently-abled people/patients from one place to another. The chair is easy to fold, unfold and move manually either by pushing from handles or by moving the wheels forwards. Due to practical difficulties for the conversion of wheel chair to electric wheelchair, the folding mechanism is neglected and cannot be relied on the converted wheelchair.

Fig 4.10 Side view of wheelchair

Fig 4.11 Front view of wheelchair

4.4.2 DC Motor

MY1016 250W 24V 2650RPM DC motor has 11-tooth, 4-bolt mounting bracket (threaded M6) on the base. As this is a DC motor it is capable of rotation in either the clockwise or counter clockwise direction by just reversing the battery polarity to the motor and can be speed controlled. It has an operating power of 250W, operating voltage 24V, rated current in the range of 9A to 11A and having a speed of 2650rpm. This motor is shown by Fig 4.12 given below.

Fig 4.12 DC Motor

4.4.3 Lithium Ion Battery

The Figure 4.13 shows 12V 7.2Ah rechargeable lithium-ion battery. Each battery weigh up to 2.4kg. The battery is capable of providing power to the motors to carry a load up to 100 kg for 20 minutes.

Fig 4.13 Batteries

4.4.4 H-Bridge High-Power Stepper Motor Driver Module

The Double BTS7960 43A H-Bridge High-Power Stepper Motor Driver Module as shown in Fig 4.14 is a fully integrated high current H bridge for motor drive applications using the BTS7960 high current half bridge. It has the PWM capability of up to 25 kHz combined with active freewheeling and Switched mode current limitation for reduced power dissipation in overcurrent.

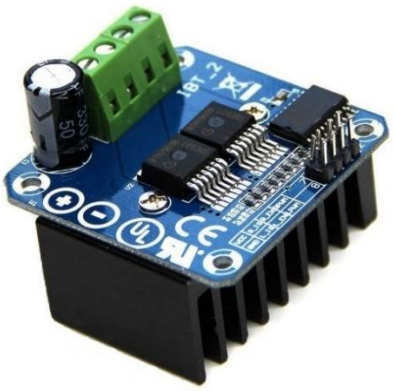


Fig 4.14 Motor Driver Module

Arduino Uno provides the necessary PWM and polarity signals for speed and direction control of the motors.

4.4.5 Cooling Fan for Raspberry Pi

A 5V Cooling Fan finds its use as an exhaust fan on the Raspberry Pi case as well as 3D printers. It maintains the temperature of the extruder, which results in smooth printing. The cooling fan operates on 0.2A 5V DC power supply which can be easily powered from RPI board. It has a 2 pin JST connector to easily connect to the printer controller system, and provide hassle-free operation. The fan is made of plastic and is light-weight. It is noiseless as well. Cooling fan is shown in Fig 4.15.

Fig 4.15 Cooling Fan

The fan is connected to the 5V power supply from Arduino UNO and is powered on when the system turns on.

4.5 HOME AUTOMATION

The circuit for the home automation system is shown in Fig 4.16. The 3-pin plug provided in the circuit is used to connect the adaptor of the Node MCU. The system can be controlled both using eye tracking and with the help of the webserver hosted in the Node MCU in its local network. The interface of the webpage in the webserver is shown in Fig 4.17.

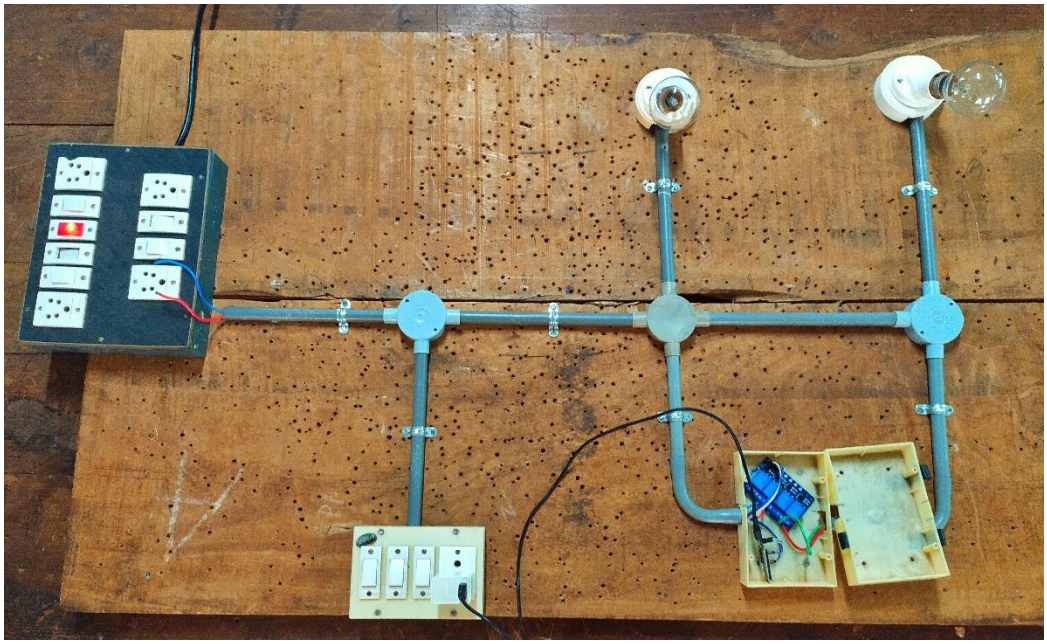


Fig 4.16 Circuit for home automation

Fig 4.17 shows the webpage hosted in Node MCU over LAN. The webpage sends API calls to the Node MCU server so that it can control different pins of the Node MCU. By setting the value of each pin HIGH or LOW with the help of API calls, we can turn ON or turn OFF the relay circuit connected with the respective pins. This in turn help us to control the devices connected with the relay module, which in in this case two light bulbs. The same API call is made by the Raspberry Pi eye controller program. When the mode for a respective light bulb is selected, looking left will initiate an API call to set the pin in the Node MCU to HIGH value. Looking right in the same mode, initiates an API call to set the same pin in Node MCU to LOW value and thus turns the respective bulb ON or OFF. Each mode controls a device that are assigned to their modes by the eye tracking program.

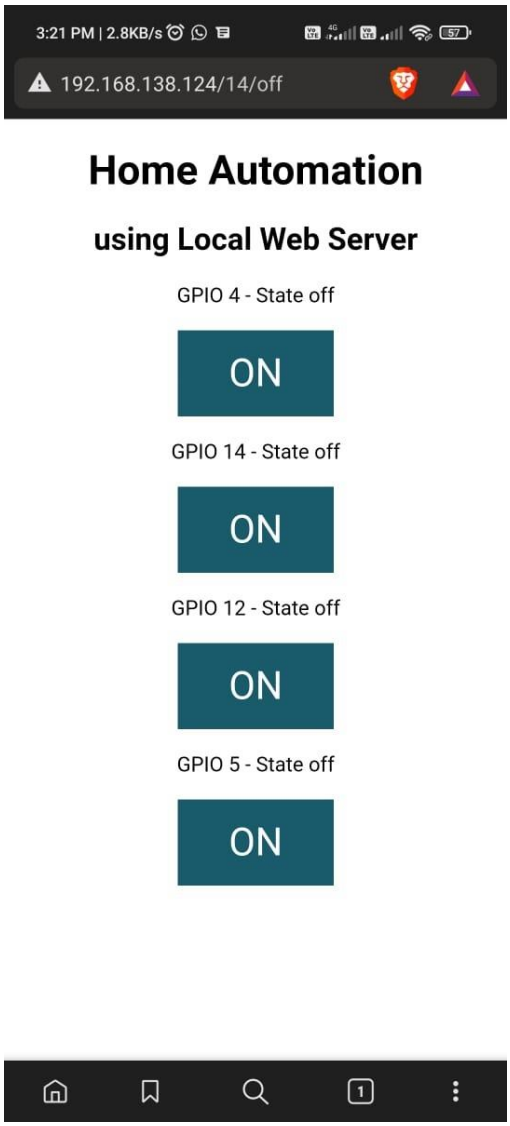


Fig 4.17 Webserver interface for home automation

4.5.1 Node MCU

ESP8266 Node MCU has powerful on-board processing and storage capabilities that allow it to be integrated with the sensors and other application-specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, and the entire solution, including the front-end module, is designed to occupy minimal PCB area.

This Wi-Fi development board already embeds in its board all the necessary components for the ESP8266 (ESP-12E) to program and upload code. It has a built-in USB to serial chip upload codes, 3.3V regulator, and logic level converter circuit so you can immediately upload codes and connect your circuits. Node MCU is shown in Fig 4.18.

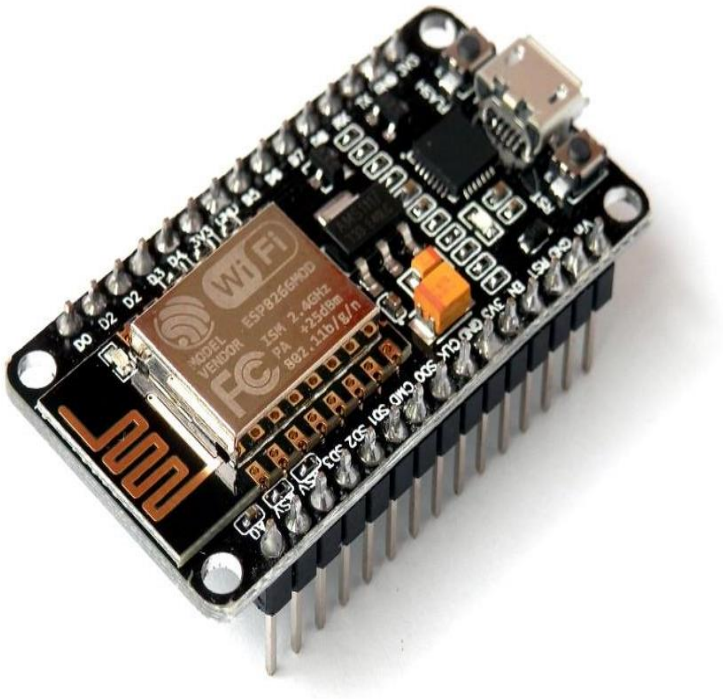


Fig 4.18 Node MCU

It also runs a webserver to control the relay circuit with the devices in the same local network. This server is used for control of home automation system over internet through computers, phones and using eye-controlled program.

4.5.2 Four Channel Relay

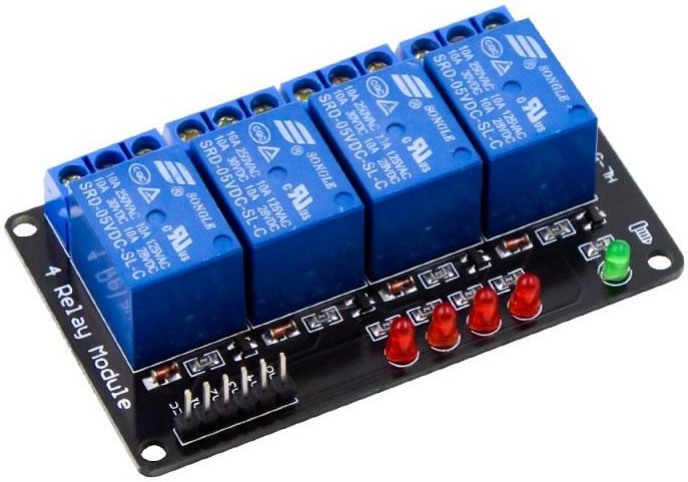


Fig 4.19 Four Channel Relay Circuit

The Figure 4.19 shows 4-Channel Relay interface board and each one needs 15-20mA Driver Current. It can be controlled directly by Micro-controller with 5V input voltage.

CHAPTER 5

SIMULATION

5.1 INTRODUCTION

The smart wheelchair was simulated using proteus software. Since eye tracking program and proteus scripting program for wheelchair model control cannot be controlled together in the proteus, the eye tracking program was executed separately in Visual Studio Code editor and its output is send to program in proteus using virtual serial connector. The arrangement of the simulation model is shown in Fig 5.1.

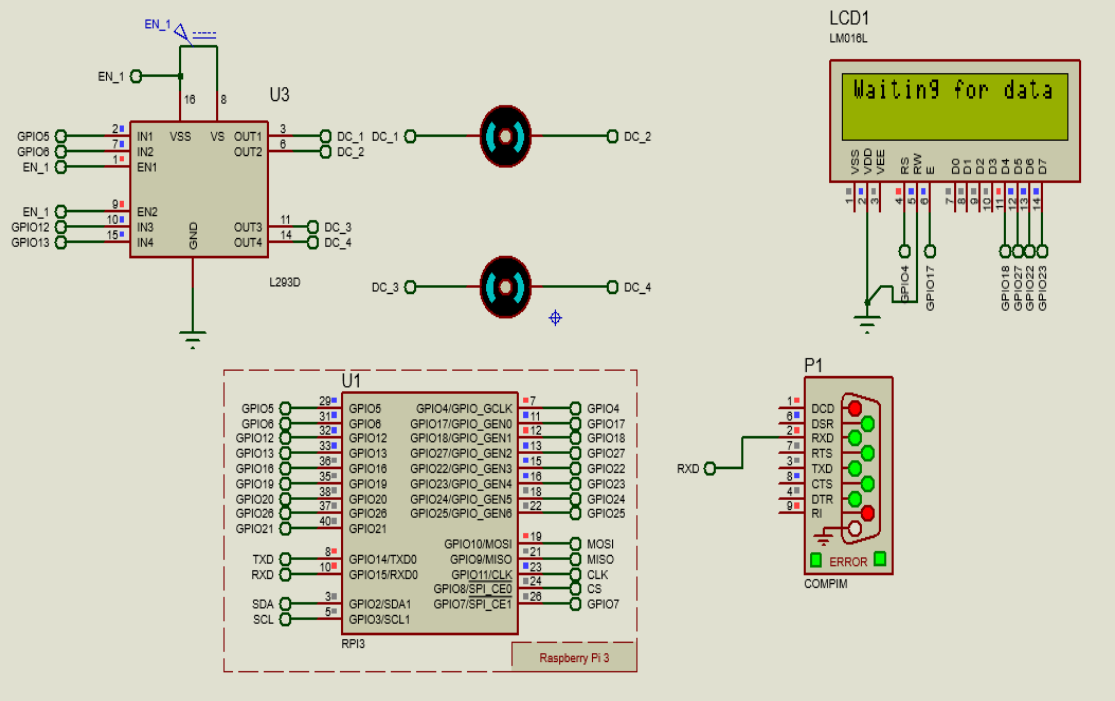


Fig 5.1 Simulation Model of Smart Wheelchair

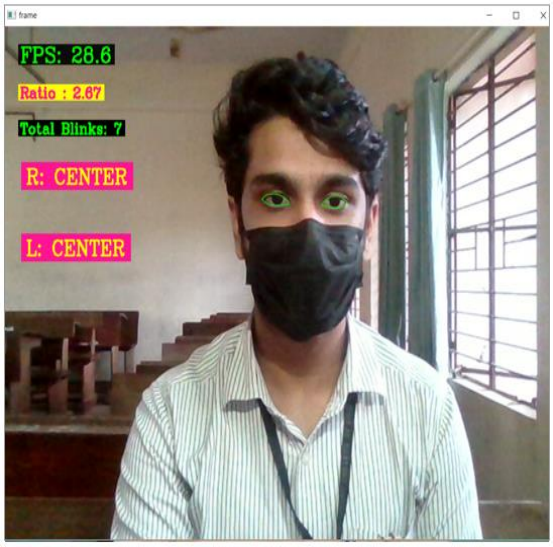
In simulation, 2 DC motors of 24V is used with motor driver circuit for controlling the motors. The signals from the eye tracking program are transferred to the Raspberry Pi 4 model of proteus design suite. To identify the control signals and LCD monitor is connected to the Raspberry Pi 4.

5.2 SIMULATION RESULTS

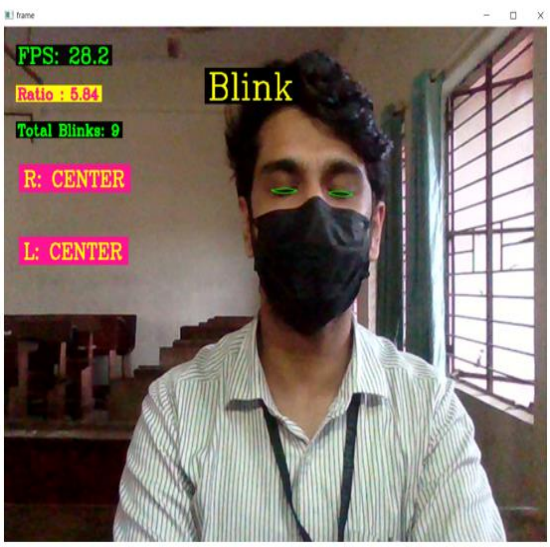
The results of the smart wheelchair and eye tracking models are provided below. The model was programmed in python language.

5.2.1 Eye Tracking Model

The result for eye tracking program is shown below.

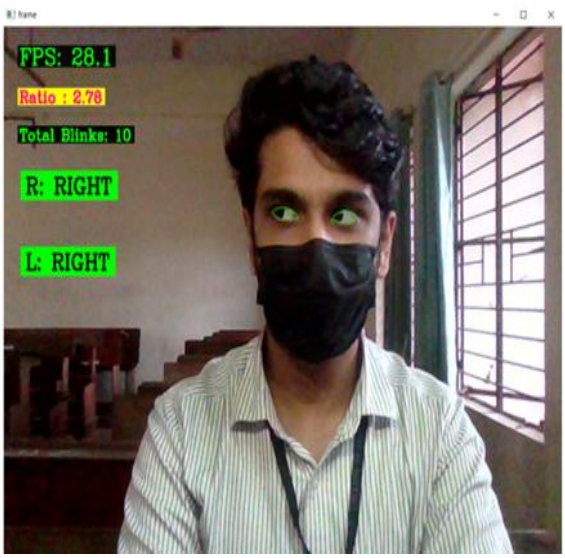


a. Centre



b. Closed

c. Left



d. Right

Fig 5.2 Eye Tracking Results.

In Fig 5.2, when the user directly looks at the camera, the program detects eye positions as centre. On looking left of the camera, the program detects eye positions as left and when looking right of the camera the program detects eye position as right. On blinking, it is displayed as blink and the count of number of blinks are shown.

The program was able to detect eye pupil positions. The accuracy of the program varied according to the lighting conditions of its surroundings.

5.2.2 Wheelchair Model Simulation Results

The simulation was conducted using proteus software, Microsoft visual studio code and virtual serial monitor and the result for smart wheelchair model using proteus design suite is shown below. The results confirm the functionality of the program for eye tracking and wheelchair control.

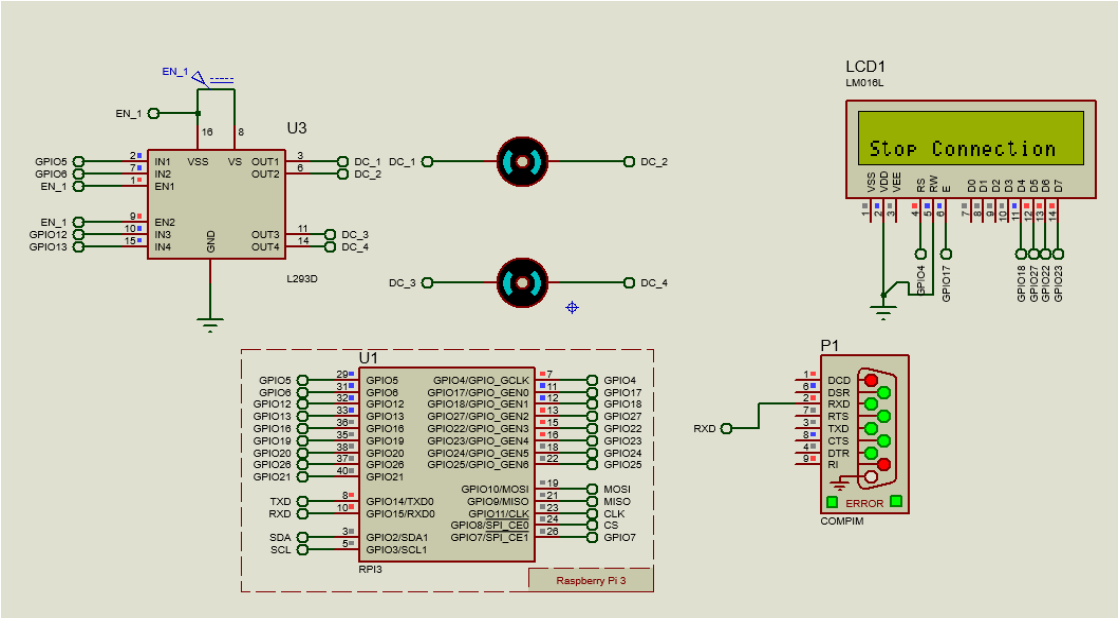


Fig 5.3 When Eye pupil are centred.

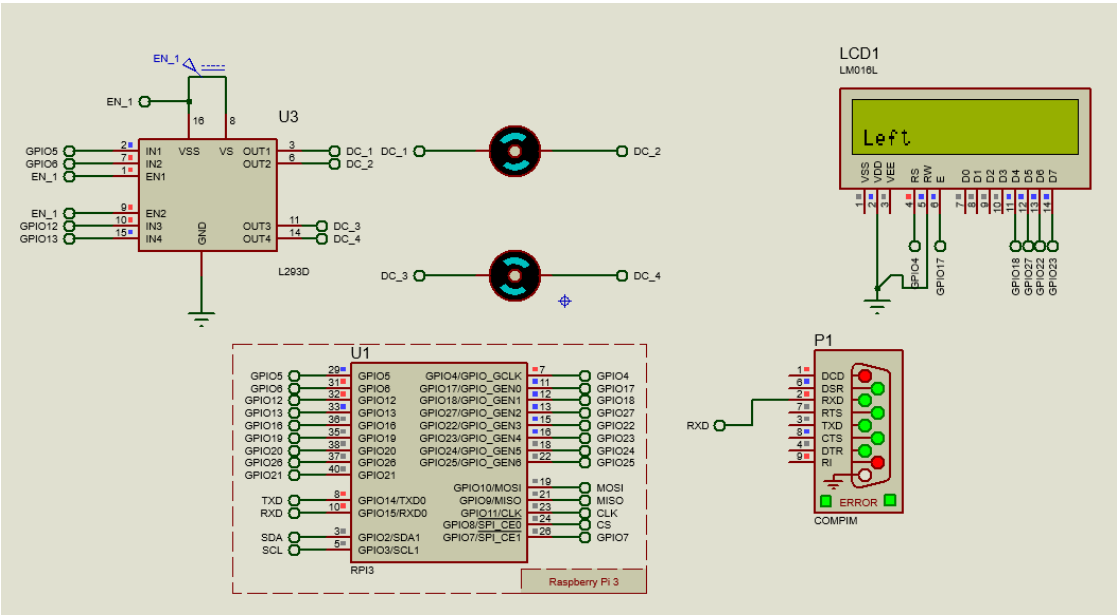


Fig 5.4 When Eye pupil are to the left.

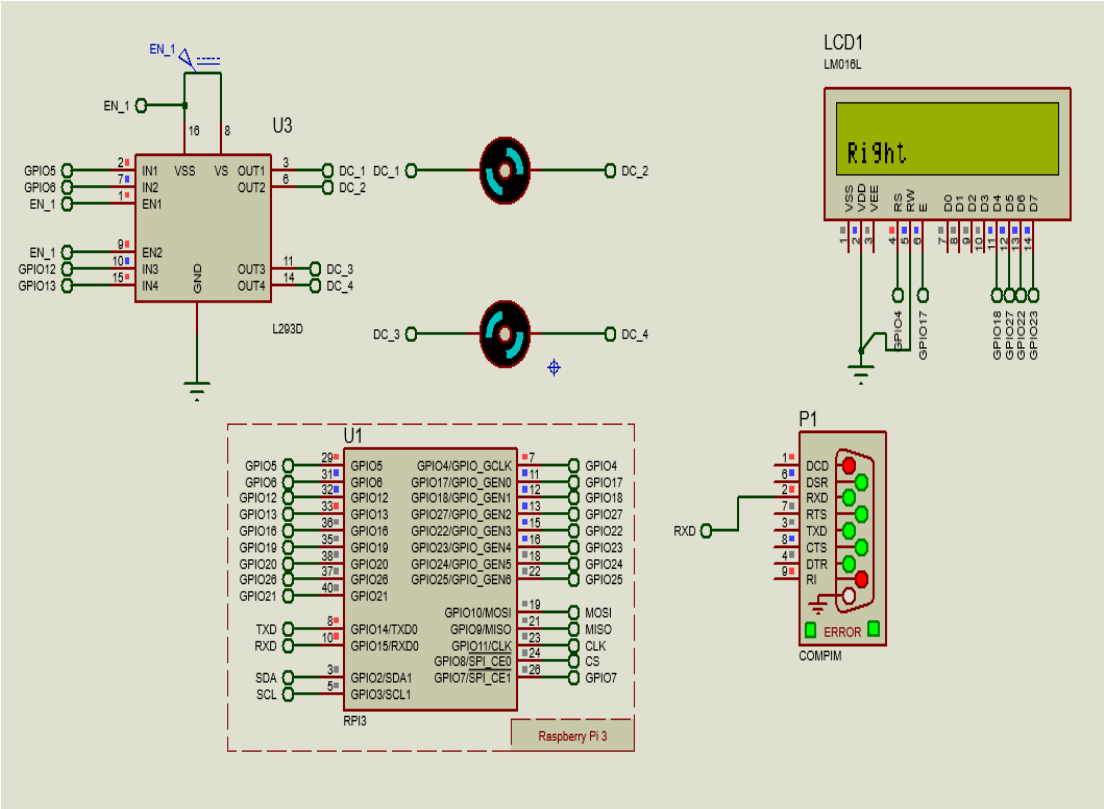


Fig 5.5 When Eye pupil are to the right.

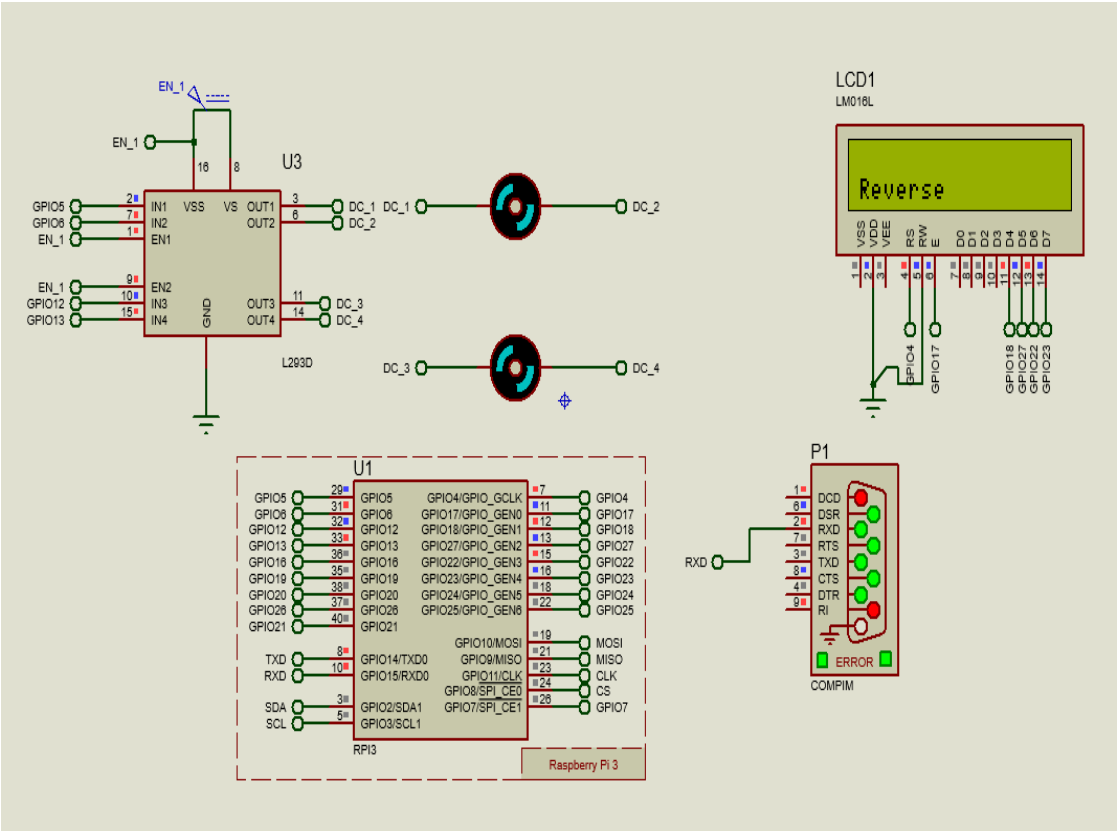


Fig 5.6 When Eye pupil are down.

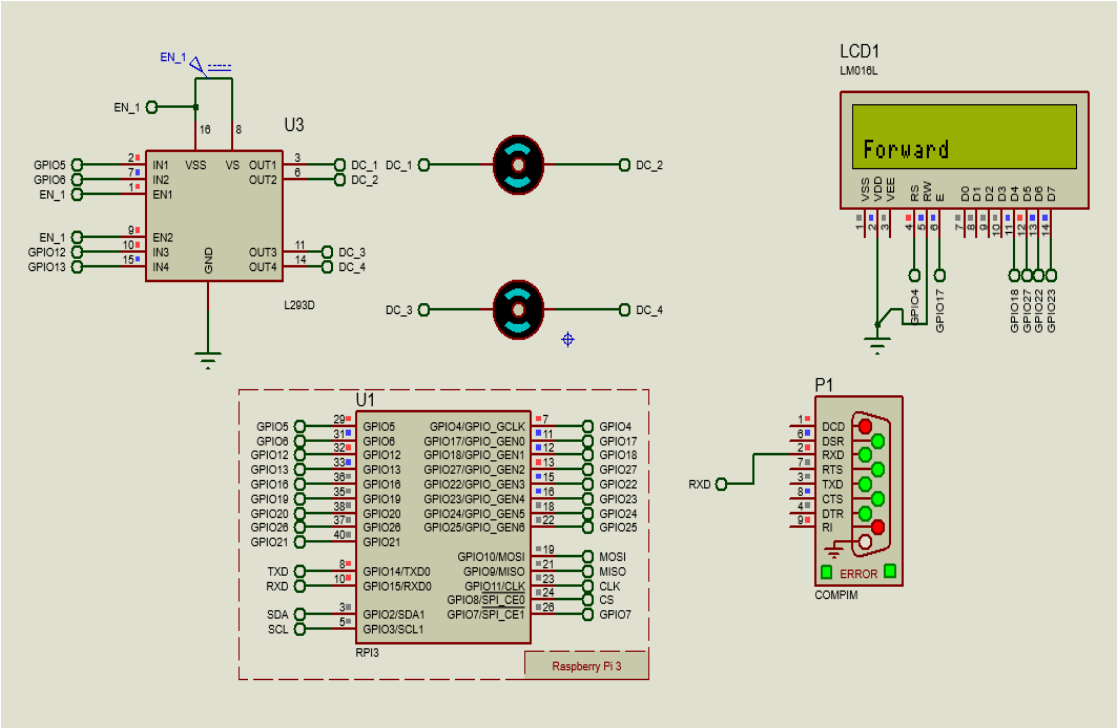


Fig 5.7 When Eye pupil are in up position.

The eye tracking program is controlled by the python program separately. When the position of the pupils is detected, it is sent to the Raspberry Pi model in the simulation model and appropriate motor functions are carried out in it. The motor driver is directly interfaced with Raspberry Pi and control signals are produced by them to control both motors as needed and display the direction of rotation of wheelchair.

CHAPTER 6

PROTOTYPE

6.1 INTRODUCTION

In this model as shown in Fig 6.1, a wheelchair prototype is designed and implemented that contains Raspberry Pi, Arduino, Node MCU, Relay module, battery, motor etc. Raspberry Pi is programmed for three modes. Voice output is implemented in Raspberry Pi, in order to identify the mode selected. A web interface is hosted in Node MCU in order to control switches in real time other than eye control.

Fig 6.1 Front view of smart wheelchair

A normal wheelchair is converted to electric wheelchair with the help of two 250W DC motors and by coupling the motors with the help of sprockets and chains. The motors are the directly connected with the BTS7960 drivers in the controller setup. A 12V rechargeable battery is connected with each BTS7960 drivers through a switch.

Other than the battery and motors, a USB camera mounted with the arm of wheelchair using a camera mount is connected to the controller setup. An earphone is also connected to the Raspberry Pi 4 inside the controller setup. Both the Raspberry Pi 4 and Arduino UNO module is powered using a 5V 2.5A power supply from a power bank.

CHAPTER 7

WORKING OF THE PROTOTYPE

7.1 INTRODUCTION

The main control unit of the wheelchair is Raspberry Pi. The working of the prototype can be divided into 2 parts. That are the wheelchair working and home automation system.

7.2 HOME AUTOMATION SYSTEM

The home automation system is controlled using Node MCU and Raspberry Pi. A server is hosted in the Node MCU over the LAN. By using different API calls different devices connected over the LAN can control various output pins of Node MCU.

When we select the option to turn on switch 1 in the webpage hosted by Node MCU using a smartphone or personal computer, an API call is made by the device to the Node MCU to energise the pin corresponding to the pin controlling the first device, in this case a bulb. When the pin output is set to HIGH after the API call, the relay connected with the pin turns on and the bulb is turned on. The same is repeated with the other switch for controlling.

7.3 WHEELCHAIR

The wheelchair is controlled using Raspberry Pi 4 and Arduino Uno. The Raspberry Pi 4 runs the program for eye position detector and detects the position of eye along with no of blinks using the camera.

Table 7.1 Truth Table for Arduino control

Input 1	Input 2	Output
0	0	Stop
0	1	Left
1	0	Right
1	1	Forward

The program is divided into 3 modes, in which each mode is selected based on number of blinks. The maximum number of blinks is set as 9 after which the count resets. When the count is at 2 the switch 1 mode is selected in which the program calls an API to turn on corresponding switch in the Node MCU. Next mode is at blink 4 where switch is controlled as that of mode 1.

Table 7.2 Truth Table for wheelchair movement

Motor 1		Motor 2		Direction
Input 1	Input 2	Input 1	Input 2	
0	0	0	0	Stop
1	0	1	0	Forward
1	0	0	1	Left
0	1	1	0	Right

The truth table for movement of wheelchair and its corresponding GPIO pin values are shown in Table 7.1. When the blink count is 6, wheelchair mode is selected. Here when eye position is UP, GPIO 21 and GPIO 22 in Raspberry Pi is set as HIGH. Thus, the Arduino Uno connected to Raspberry Pi gets both the input as HIGH. This in turn sets both motors to rotate clockwise which in turn moves the wheelchair forward. When the eye position is at LEFT, one pin is set HIGH and other LOW and the opposite for RIGHT. This in turn rotates one wheel clockwise and other anticlockwise according to the direction of movement of wheelchair.

As shown in Table 7.2, after receiving signals from the Raspberry Pi, the Arduino UNO generates control signals to control the movement of wheel chair. When all the outputs are kept low, the wheelchair is in stop position by not rotating the wheels. When input 1 of both motor 1 and motor 2 are in HIGH value, both wheels rotate in clockwise direction which lead to forward movement. The wheel chair rotates left or right based on the motor's direction of rotation.

CHAPTER 8

CONTROL OF THE PROTOTYPE

8.1 INTRODUCTION

The proposed system has the following three modes of operation on the basis of number of blinks made by the eye. They are listed below.

8.2 MODE 1: BULB 1

Mode 1 gets selected at 2 blinks. When this mode is selected, and automated text to speech audio output is played in the voice output device which reads out the text “Mode 1 selected” and will help the user to identify the mode selected. In this mode, we will be able to control the first device which is a bulb in this prototype. When the eyeball moves towards the left side, the device will be turned ON as shown in Fig 8.1, and similarly when it moves to the right side, the device will be turned OFF. Similarly, any other device that can be turned on and off with the help of a switch can be connected to the relay circuit and operated with the help of the prototype.

Fig 8.1 Mode 1 of eye control of smart wheelchair

8.3 MODE 2: BULB 2

Mode 2 gets selected at 4 blinks. When this mode is selected, and automated text to speech audio output is played in the voice output device which reads out the text “Mode 2 selected” and will help the user to identify the mode selected. In this mode, we will be able to control the second device which is another bulb. When the eyeball moves towards the left side, the device will be turned ON as shown in Fig 8.2, and similarly when it moves to the right side, the device will be turned OFF and just as in Mode 1, any other device that can be turned on and off with the help of a switch can be connected to the relay circuit and operated with the help of the prototype.

Fig 8.2 Mode 2 of eye control of smart wheelchair

8.4 MODE 3: WHEELCHAIR

Mode 3 gets selected at 6 blinks and till 9 blinks. When this mode is selected, and automated text to speech audio output is played in the voice output device which reads out the text “Mode 3 selected, wheelchair control is turned on” and will help the user to identify the mode selected. In this mode we will be able to control the wheelchair. When

the user looks up, both the motors rotate in the clockwise direction and the wheelchair moves forward as shown in Fig 8.3.

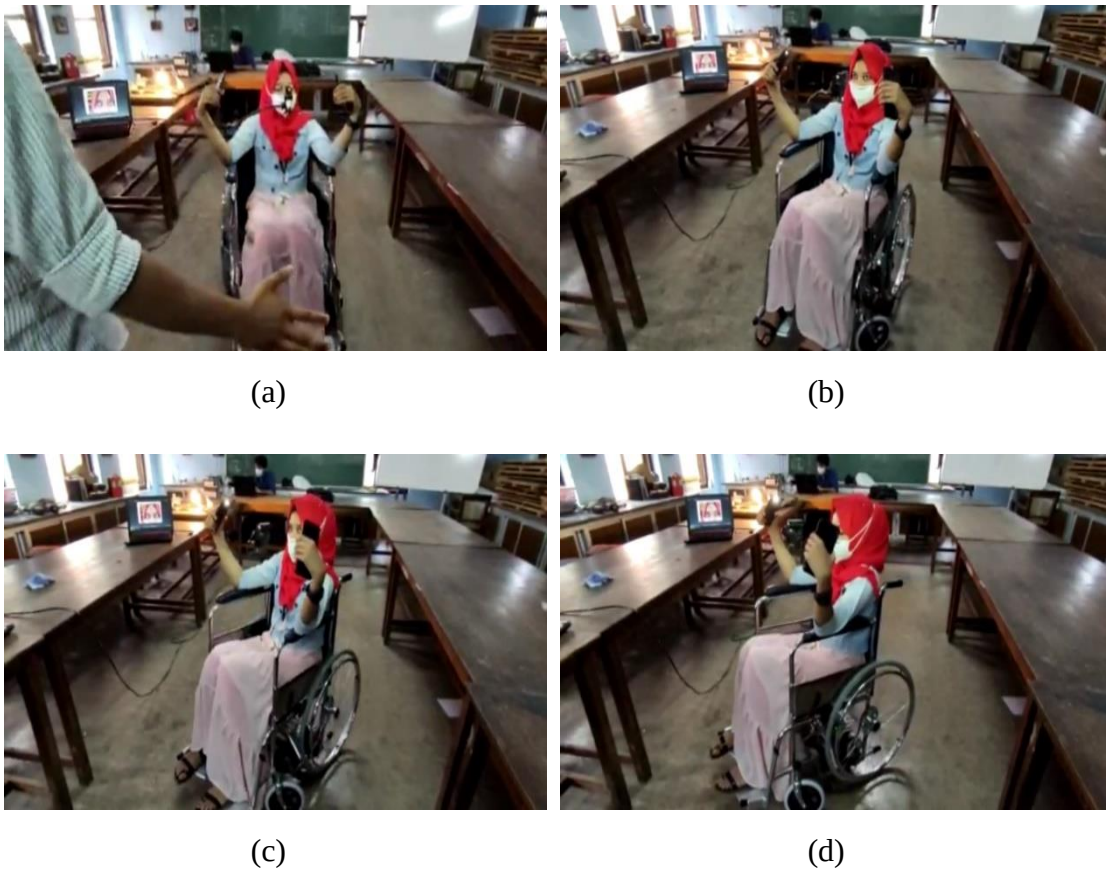
(a) (b)

Fig 8.3 Forward movement of smart wheelchair

(a) (b)

(c) (d)

Fig 8.4 Left rotation of smart wheelchair



(a) (b)

(c) (d)

Fig 8.5 Right rotation of smart wheelchair

To move the wheelchair in left direction the user has to look to left side, then its left motor will rotate in anticlockwise direction and the right motor rotates in clockwise direction and the wheelchair moves in left direction as shown in Fig 8.4.

To move the wheelchair in right direction, the user has to look to right side, then its right motor rotates in anticlockwise direction and the left motor moves in clockwise direction, so the wheelchair moves in right direction as shown in Fig 8.5.

When the user looks up or centre, both the motor gets turned ‘off’ and the wheelchair stops. After the blink counts above 9 blinks, the count resets and starts counting again from zero where the whole process is carried out again.

CHAPTER 9

CONCLUSION

Paralyzed stroke patients are unable to normally communicate with their environment. The biggest problem that paralyzed patients face is leading their own lives without the assistance of others. As a solution to this problem, a smart wheel chair working with a home automation system that can be controlled by eye tracking is implemented.

The eye tracking model is developed in python language using OpenCV and MediaPipe libraries. Eye tracking technique, captures the image and detects the presence of human face. It detects the location of the eye on the face and conducts basic image processing operations such as colour image to grey conversion, filtering, threshold, pattern matching, noise reduction, and circle detection on it after recognizing the face.

Raspberry Pi 4 is used as the base controller. The Raspberry Pi board is utilized to control the entire operation of the system. It generates signals based on digital image processing. The data is collected and analysed using the Raspberry Pi. Based on the location of the eye pupil, the Raspberry Pi sends a control signal to the Arduino which in turn sends the appropriate signals motor driving circuit. A normal wheelchair is modified to electric wheelchair with two separate motors that are placed in each wheel. The motors are coupled to the wheelchair using chain and sprocket. Different modes selected on the basis of blink count is used to control home automation system and the wheelchair movement.

REFERENCE

[1]**Diksha Raj, Anish Kalra, Sreejith Krishna N, Suhriday Roy, Dr. Shreenath K N**, "Wagon-The Smart Wheelchair", *International Journal of Emerging Technologies and Innovative Research*, vol. 8, Issue 8, pp. b343-b349, August 2021

[2]**D. Das, P. Saha and N. Tabassum,** "Real Time Distance and Mobility Detection of Obstacles Using a Smart Multisensor Framework for Visually Impaired People", *2020 IEEE Region 10 Symposium (TENSYMP)*, pp. 590-593, doi: 10.1109/TENSYMP50017.2020.9230880, 2020

[3]**Y. C. Lee and C. M. Lee,** "Real-Time Smart Home Surveillance System of Based on Raspberry Pi", *2020 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*, pp. 72-74, doi: 10.1109/ECICE50847.2020.9301929, 2020

[4]**Muhammad Azlan Alim, Samsul Setumin, Anis Diyana Rosli, Adi Izhar Che Ani**, "Development of a Voice-controlled Intelligent Wheelchair System using Raspberry Pi", *International Conference on Applied Sciences, Information and Technology*, vol. 846, 2021

[5]**V. U. Rani, J. Sridevi and P. M. Sai,** "Web Controlled Raspberry Pi Robot Surveillance", *2021 International Conference on Sustainable Energy and Future Electric Transportation (SEFET)*, pp. 1-5, doi: 10.1109/SeFet48154.2021.9375666, 2021

[6]**Reona Cerejo, Valentine Correia and Neil Pereira, "**Eye Controlled Wheelchair based on Arduino Circuit", *International Journal of Technical Research and Applications*, vol. 3, Issue 6, pp. 94-98, 2015

[7]**Fahad Wallam and Muhammad Asif**, "Dynamic Finger Movement Tracking and Voice Commands Based Smart Wheelchair", *International Journal of Computer and Electrical Engineering*, vol. 3, pp. 497-502 ISSN: 1793-8163 doi:10.7763/IJCEE.2011.V3.368, 2021

[8]**Al-Okby, Mohammed & Neubert, Sebastian & Stoll, Norbert & Thurow, Kerstin.** "Low-cost Hybrid Wheelchair Controller for Quadriplegias and Paralysis Patients", *Advances in Science, Technology and Engineering Systems Journal*, pp. 687-694, doi:10.25046/aj020388, 2017

[9]**Luis A. Rivera, Guilherme N. DeSouza**, "An Automatic Wheelchair Controlled using Hand Gestures", *IEEE, University of Missouri*, April 2010.

[10]**F. Klaib, N. O. Alsrehin, W. Y. Melhem, H. O. Bashtawi,and A. A. Magableh,** "Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and internet of things technologies", *Expert Systems with Applications*, vol. 166, pp. 114037–114174, 2021

[11]**P. A. Punde, M. E. Jadhav and R. R. Manza**, "A study of eye tracking technology and its applications", *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*, pp. 86-90, doi: 10.1109/ICISIM.2017.8122153, 2017

[12]**Tan Kian Hou, Yagasena and Chelladurai,** "Arduino based voice controlled wheelchair", *First International Conference on Emerging Electrical Energy, Electronics and Computing Technologies*, vol. 1432, doi:10.1088/1742-6596/1432/1/012064, 2019

[13]**M. H. A. Sibai and S. A. Manap**, "A study on smart wheelchair systems", International Journal of Engineering Technology and Sciences, vol. 4, pp. 25–35, doi:10.15282/ijets.4.2015.1.4.1033, 2015

[14]**Deepak Kumar, Reetu Malhotra, S.R. Sharma, "**Design and Construction of a Smart Wheelchair", *Procedia Computer Science*, vol. 172, pp. 302-307, doi:10.1016/j.procs.2020.05.048, 2020

[15]**G. Boustany, A. El Deen Itani, R. Youssef, O. Chami and Z. O. Abu-Faraj,** "Design and development of a rehabilitative eye-tracking based home automation system," *2016 3rd Middle East Conference on Biomedical Engineering (MECBME)*, pp. 30-33, doi: 10.1109/MECBME.2016.7745401, 2016

[16]**Joseph K George, Subin K B, Arun Jose, Hima T**, "Eye Controlled Home-Automation For Disables", *IOSR Journal of Electrical and Electronics Engineering*, pp. 06-08, 2017

[17]**Deepak Kumar Lodhi, Prakshi Vats, Addala Varun, Prashant Solanki, Ritakshi Gupta1, Manoj Kumar Pandey, Rajat Butola**, "Smart Electronic Wheelchair Using Arduino and Bluetooth Module", *International Journal of Computer Science and Mobile Computing*, vol. 5, pp. 433-438, 2016

[18]**Airi Ishizuka, Ayanori Yorozu and Masaki Takahashi**, "Driving Control of a Powered Wheelchair Considering Uncertainty of Gaze Input in an Unknown Environment", *Applied Sciences* , doi: 10.3390/app8020267, 2018

[19]**M. A. Eid, N. Giakoumidis and A. El Saddik,** "A Novel Eye-Gaze-Controlled Wheelchair System for Navigating Unknown Environments: Case Study with a

Person with ALS", *in IEEE Access*, vol. 4, pp. 558-573, doi: 10.1109/ACCESS.2016.2520093, 2016

[20]**Mohammed Hayyan Al Sibai and Sulastri Abdul Manap**, "A Study on Smart Wheelchair Systems", *International Journal of Engineering Technology and Sciences*, doi:10.15282/ijets.4.2015.1.4.1033, 2015

APPENDIX

ALGORITHMS

A1: Algorithm for Raspberry Pi

```
from pickle import FALSE

from urllib import request

import cv2 as cv

import mediapipe as mp

import time

import math

import numpy as np

import requests

import RPi.GPIO as GPIO

import pyttsx3


# variables

frame_counter =0

CLOSED_TIME=0

CEF_COUNTER =0

TOTAL_BLINKS =0

FORWARDCHAIR = 0

STOPCHAIR = 0

RIGHTCHAIR = 0

LEFTCHAIR = 0
```

```
# colors

# values =(blue, green, red) opencv accepts BGR values not RGB

BLACK = (0,0,0)

WHITE = (255,255,255)

BLUE = (255,0,0)

RED = (0,0,255)

CYAN = (255,255,0)

YELLOW =(0,255,255)

MAGENTA = (255,0,255)

GRAY = (128,128,128)

GREEN = (0,255,0)

PURPLE = (128,0,128)

ORANGE = (0,165,255)

PINK = (147,20,255)

points_list =[(200, 300), (150, 150), (400, 200)]

switch1 = False

switch2 = False

switch_one_voice_count = 0

switch_two_voice_count = 0

wheelchair_voice_count = 0

 # api-endpoint

input1on = "http://192.168.165.124/4/on"

input1off = "http://192.168.165.124/4/off"

input2on = "http://192.168.165.124/14/on"
```

```
input2off = "http://192.168.165.124/14/off"

#GPIO

pin_1=20

pin_2=21

#GPIO.setup(pin_1,GPIO.OUT)

#GPIO.setup(pin_2,GPIO.OUT)

engine = pyttsx3.init()


def drawColor(img, colors):

    x, y = 0,10

    w, h = 20, 30

    for color in colors:

        x += w+5


def colorBackgroundText(img, text, font, fontScale, textPos,
textThickness=1,textColor=(0,255,0), bgColor=(0,0,0), pad_x=3, pad_y=3):

    (t_w, t_h), _= cv.getTextSize(text, font, fontScale, textThickness) # getting the text
size

    x, y = textPos

    cv.rectangle(img, (x-pad_x, y+ pad_y), (x+t_w+pad_x, y-t_h-pad_y), bgColor,-1) #
draw rectangle

    cv.putText(img,text, textPos,font, fontScale, textColor,textThickness ) # draw in
text

    return img
```

```
def textWithBackground(img, text, font, fontScale, textPos,
textThickness=1,textColor=(0,255,0), bgColor=(0,0,0), pad_x=3, pad_y=3,
bgOpacity=0.5):

    (t_w, t_h), _= cv.getTextSize(text, font, fontScale, textThickness) # getting the text
size

    x, y = textPos

    overlay = img.copy() # coping the image

    cv.rectangle(overlay, (x-pad_x, y+ pad_y), (x+t_w+pad_x, y-t_h-pad_y), bgColor,-
1) # draw rectangle

    new_img = cv.addWeighted(overlay, bgOpacity, img, 1 - bgOpacity, 0) #
overlaying the rectangle on the image.

    cv.putText(new_img,text, textPos,font, fontScale, textColor,textThickness ) # draw
in text

    img = new_img

    return img


def textBlurBackground(img, text, font, fontScale, textPos,
textThickness=1,textColor=(0,255,0),kneral=(33,33) , pad_x=3, pad_y=3):

    (t_w, t_h), _= cv.getTextSize(text, font, fontScale, textThickness) # getting the text
size

    x, y = textPos

    blur_roi = img[y-pad_y-t_h: y+pad_y, x-pad_x:x+t_w+pad_x] # croping Text
Background

    img[y-pad_y-t_h: y+pad_y, x-pad_x:x+t_w+pad_x]=cv.blur(blur_roi, kneral)  #
merging the blured background to img

    cv.putText(img,text, textPos,font, fontScale, textColor,textThickness )

    # cv.imshow('blur roi', blur_roi)
```

```
    # cv.imshow('blured', img)
    return img


def fillPolyTrans(img, points, color, opacity):
    list_to_np_array = np.array(points, dtype=np.int32)
    overlay = img.copy()  # coping the image
    cv.fillPoly(overlay,[list_to_np_array], color )
    new_img = cv.addWeighted(overlay, opacity, img, 1 - opacity, 0)
    # print(points_list)
    img = new_img
    cv.polylines(img, [list_to_np_array], True, color,1, cv.LINE_AA)
    return img


def rectTrans(img, pt1, pt2, color, thickness, opacity):
    overlay = img.copy()
    cv.rectangle(overlay, pt1, pt2, color, thickness)
    new_img = cv.addWeighted(overlay, opacity, img, 1 - opacity, 0) # overlaying the
rectangle on the image.
    img = new_img


    return img


# constants
CLOSED_EYES_FRAME =3
FONTS =cv.FONT_HERSHEY_COMPLEX
```

```
# Left eyes indices

LEFT_EYE =[ 362, 382, 381, 380, 374, 373, 390, 249, 263, 466, 388, 387, 386, 385,384, 398 ]

LEFT_EYEBROW =[ 336, 296, 334, 293, 300, 276, 283, 282, 295, 285 ]

# right eyes indices

RIGHT_EYE=[ 33, 7, 163, 144, 145, 153, 154, 155, 133, 173, 157, 158, 159, 160, 161 , 246 ]

RIGHT_EYEBROW=[ 70, 63, 105, 66, 107, 55, 65, 52, 53, 46 ]

map_face_mesh = mp.solutions.face_mesh

# camera object

camera = cv.VideoCapture(1)


# landmark detection function

def landmarksDetection(img, results, draw=False):

    img_height, img_width= img.shape[:2]

    mesh_coord = [(int(point.x * img_width), int(point.y * img_height)) for point in results.multi_face_landmarks[0].landmark]

    if draw :

        [cv.circle(img, p, 2, (0,255,0), -1) for p in mesh_coord]

    # returning the list of tuples for each landmarks

    return mesh_coord


# Euclaidean distance

def euclaideanDistance(point, point1):

    x, y = point
```

```
    x1, y1 = point1
    distance = math.sqrt((x1 - x)**2 + (y1 - y)**2)
    return distance


# Blinking Ratio
def blinkRatio(img, landmarks, right_indices, left_indices):
    # Right eyes
    # horizontal line
    rh_right = landmarks[right_indices[0]]
    rh_left = landmarks[right_indices[8]]
    # vertical line
    rv_top = landmarks[right_indices[12]]
    rv_bottom = landmarks[right_indices[4]]
    # draw lines on right eyes
    cv.line(img, rh_right, rh_left, GREEN, 2)
    cv.line(img, rv_top, rv_bottom, WHITE, 2)
    # LEFT_EYE
    # horizontal line
    lh_right = landmarks[left_indices[0]]
    lh_left = landmarks[left_indices[8]]
    # vertical line
    lv_top = landmarks[left_indices[12]]
    lv_bottom = landmarks[left_indices[4]]
```

```
    # draw lines on left eyes
    cv.line(img, lh_right, lh_left, GREEN, 2)
    cv.line(img, lv_top, lv_bottom, WHITE, 2)
    rhDistance = euclaideanDistance(rh_right, rh_left)
    rvDistance = euclaideanDistance(rv_top, rv_bottom)
    lvDistance = euclaideanDistance(lv_top, lv_bottom)
    lhDistance = euclaideanDistance(lh_right, lh_left)
    reRatio = rhDistance/rvDistance
    leRatio = lhDistance/lvDistance
    ratio = (reRatio+leRatio)/2
    return ratio


    # Eyes Extrctor function,
def eyesExtractor(img, right_eye_coords, left_eye_coords):
    # converting color image to  scale image
    gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
    # getting the dimension of image
    dim = gray.shape
    # creating mask from gray scale dim
    mask = np.zeros(dim, dtype=np.uint8)
    # drawing Eyes Shape on mask with white color
    cv.fillPoly(mask, [np.array(right_eye_coords, dtype=np.int32)], 255)
    cv.fillPoly(mask, [np.array(left_eye_coords, dtype=np.int32)], 255)
```

```
    # draw eyes image on mask, where white shape is
    eyes = cv.bitwise_and(gray, gray, mask=mask)
    # change black color to gray other than eys
    eyes[mask==0]=155
    # getting minium and maximum x and y  for right and left eyes
    # For Right Eye
    r_max_x = (max(right_eye_coords, key=lambda item: item[0]))[0]
    r_min_x = (min(right_eye_coords, key=lambda item: item[0]))[0]
    r_max_y = (max(right_eye_coords, key=lambda item : item[1]))[1]
    r_min_y = (min(right_eye_coords, key=lambda item: item[1]))[1]
    # For LEFT Eye
    l_max_x = (max(left_eye_coords, key=lambda item: item[0]))[0]
    l_min_x = (min(left_eye_coords, key=lambda item: item[0]))[0]
    l_max_y = (max(left_eye_coords, key=lambda item : item[1]))[1]
    l_min_y = (min(left_eye_coords, key=lambda item: item[1]))[1]
    # croping the eyes from mask
    cropped_right = eyes[r_min_y: r_max_y, r_min_x: r_max_x]
    cropped_left = eyes[l_min_y: l_max_y, l_min_x: l_max_x]
    # returning the cropped eyes
    return cropped_right, cropped_left


# creating pixel counter function
def pixelCounter(first_piece, second_piece, third_piece,fourth_piece,fifth_piece):
    # counting black pixel in each part
```

```
right_part = np.sum(first_piece==0)
center_part = np.sum(second_piece==0)
left_part = np.sum(third_piece==0)
up_part = np.sum(fourth_piece==0)
down_part = np.sum(fifth_piece==0)
# creating list of these values
eye_parts = [right_part, center_part, left_part, up_part, down_part]

# getting the index of max values in the list
max_index = eye_parts.index(max(eye_parts))
pos_eye =''
if max_index==0:
    pos_eye="RIGHT"
    color=[BLACK, GREEN]
elif max_index==1:
    pos_eye = 'CENTER'
    color = [YELLOW, PINK]
elif max_index ==2:
    pos_eye = 'LEFT'
    color = [GRAY, YELLOW]
elif max_index ==3:
    pos_eye = 'UP'
    color = [GRAY, YELLOW]
elif max_index ==4:
```

```
        pos_eye = 'DOWN'
        color = [BLACK, YELLOW]
    else:
        pos_eye="CLOSED"
        color = [WHITE, YELLOW]
    return pos_eye, color
# Eyes Postion Estimator
def positionEstimator(cropped_eye):
    # getting height and width of eye
    h, w =cropped_eye.shape
    # remove the noise from images
    gaussain_blur = cv.GaussianBlur(cropped_eye, (9,9),0)
    median_blur = cv.medianBlur(gaussain_blur, 3)
    # applying thrsholding to convert binary_image
    ret, threshed_eye = cv.threshold(median_blur, 130, 255, cv.THRESH_BINARY)
    # create fixd part for eye with
    piece = int(w/3)
    place = int(h/3)
    down = int(h/2)
    # slicing the eyes into three parts
    right_piece = threshed_eye[0:h, 0:piece]
    center_piece = threshed_eye[0:h, piece: piece+piece]
    left_piece = threshed_eye[0:h, piece +piece:w]
    up_piece=threshed_eye[0:down, 0:w]
```

```
    down_piece=threshed_eye[down:h, 0:w]
    # calling pixel counter function
    eye_position, color = pixelCounter(right_piece, center_piece, left_piece, up_piece, down_piece)
    return eye_position, color


with map_face_mesh.FaceMesh(min_detection_confidence =0.5, min_tracking_confidence=0.5) as face_mesh:
    # starting time here
    start_time = time.time()
    # starting Video loop here.
    while True:
        frame_counter +=1 # frame counter
        ret, frame = camera.read() # getting frame from camera
        #  resizing frame
        frame = cv.resize(frame, None, fx=1.5, fy=1.5, interpolation=cv.INTER_CUBIC)
        frame_height, frame_width= frame.shape[:2]
        rgb_frame = cv.cvtColor(frame, cv.COLOR_RGB2BGR)
        results  = face_mesh.process(rgb_frame)
        if results.multi_face_landmarks:
            mesh_coords = landmarksDetection(frame, results, False)
            ratio = blinkRatio(frame, mesh_coords, RIGHT_EYE, LEFT_EYE)
            colorBackgroundText(frame,  f'Ratio : {round(ratio,2)}', FONTS, 0.7, (30,100),2, PINK, YELLOW)
```

```
        if ratio >4.5:
            CEF_COUNTER +=1
            CLOSED_TIME +=1
            TOTAL_BLINKS +=1
            colorBackgroundText(frame,  f'Blink', FONTS, 1.7, (int(frame_height/2), 100), 2, YELLOW, pad_x=6, pad_y=6, )
        else:
            if CEF_COUNTER>CLOSED_EYES_FRAME:
                CEF_COUNTER =0
        colorBackgroundText(frame,  f'Total Blinks: {TOTAL_BLINKS} Total CEF={CEF_COUNTER} Total CLOSED TIME={CLOSED_TIME}', FONTS, 0.7, (30,150),2)
        if TOTAL_BLINKS>9:
            TOTAL_BLINKS=0
            CLOSED_TIME=0
        cv.polylines(frame,  [np.array([mesh_coords[p] for p in LEFT_EYE ], dtype=np.int32)], True, GREEN, 1, cv.LINE_AA)


        # Blink Detector Counter Completed
        right_coords = [mesh_coords[p] for p in RIGHT_EYE]
        left_coords = [mesh_coords[p] for p in LEFT_EYE]
        crop_right, crop_left = eyesExtractor(frame, right_coords, left_coords)
        eye_position_right, color = positionEstimator(crop_right)
        eye_position_left, color = positionEstimator(crop_left)
        colorBackgroundText(frame, f'R: {eye_position_right}', FONTS, 1.0, (40, 220), 2, color[0], color[1], 8, 8)
```

```
        if(TOTAL_BLINKS==6):
            if(wheelchair_voice_count == 0):
                engine.say('WheelChair selected')
                engine.runAndWait()
                wheelchair_voice_count=1
                switch_one_voice_count=0
                switch_two_voice_count=0
            if(eye_position_right=="DOWN" ):
                print("forward")
                GPIO.output(pin_1,GPIO.HIGH)
                GPIO.output(pin_2,GPIO.HIGH)
            elif(eye_position_right=="LEFT" ):
                print("left")
                GPIO.output(pin_1,GPIO.HIGH)
                GPIO.output(pin_2,GPIO.LOW)
            elif(eye_position_right=="RIGHT"):
                print("right")
                GPIO.output(pin_1,GPIO.LOW)
                GPIO.output(pin_2,GPIO.HIGH)
            else:
                GPIO.output(pin_1,GPIO.LOW)
                GPIO.output(pin_2,GPIO.LOW)
                print("stop")
```

```
        if(TOTAL_BLINKS==4):
            if(switch_two_voice_count == 0):
                engine.say('Switch Two Selected')
                engine.runAndWait()
                switch_two_voice_count=1
                switch_one_voice_count=0
                wheelchair_voice_count=0
            if(eye_position_right=="RIGHT"):
                requests.get(url = input1off)
                print("switch 1 off")
            if(eye_position_right=="LEFT"):
                requests.get(url = input1on)
                print("switch 1 on")
        if(TOTAL_BLINKS==2):
            if(switch_one_voice_count == 0):
                engine.say('Switch One Selected')
                engine.runAndWait()
                switch_one_voice_count=1
                switch_two_voice_count=0
                wheelchair_voice_count=0
            if(eye_position_right=="RIGHT"):
                requests.get(url = input2off)
                print("switch 2 off")
            if(eye_position_right=="LEFT"):
```

```
                requests.get(url = input2on)
                print("switch 2 on")
            #time.sleep(0.15)


        # calculating  frame per seconds FPS
        end_time = time.time()-start_time
        fps = frame_counter/end_time
        frame =textWithBackground(frame,f'FPS: {round(fps,1)}',FONTS, 1.0, (30, 50), bgOpacity=0.9, textThickness=2)
        cv.imshow('frame', frame)
        key = cv.waitKey(2)
        if key==ord('q') or key ==ord('Q'):
            GPIO.cleanup()
            break
    cv.destroyAllWindows()
    camera.release()
```

A2: Arduino program in Node MCU

```
#include <ESP8266WiFi.h>
// Replace with your network credentials
const char* ssid     = "SHRAVAN";   //wifi name
const char* password = "12345678";      //wifi password
 // Set your Static IP address
IPAddress local_IP(192, 168, 74, 124);
```

```
// Set your Gateway IP address
IPAddress gateway(192, 168, 1, 1);


IPAddress subnet(255, 255, 0, 0);
IPAddress primaryDNS(8, 8, 8, 8); // optional
IPAddress secondaryDNS(8, 8, 4, 4); // optional
// Set web server port number to 80
WiFiServer server(80);


// Variable to store the HTTP request
String header;


// Auxiliar variables to store the current output state
String output0State = "off";
String output1State = "off";
String output2State = "off";
String output3State = "off";


// Assign output variables to GPIO pins
const int output0 = D2;     //GPIO4
const int output1 = D5;     //GPIO14
const int output2 = D6;     //GPIO12
const int output3 = D1;     //GPIO5
```

```
void setup() {
  Serial.begin(115200);
  // Initialize the output variables as outputs
  pinMode(output0, OUTPUT);
  pinMode(output1, OUTPUT);
  pinMode(output2, OUTPUT);
  pinMode(output3, OUTPUT);

  // Set outputs to LOW
  digitalWrite(output0, LOW);
  digitalWrite(output1, LOW);
  digitalWrite(output2, LOW);
  digitalWrite(output3, LOW);

  // Connect to Wi-Fi network with SSID and password
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  // Print local IP address and start web server
  Serial.println("STARTING.....");
```

```
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  server.begin();
}
void loop()
{
  WiFiClient client = server.available();   // Listen for incoming clients
  if (client)
  {                             // If a new client connects,
    Serial.println("New Client.");          // print a message out in the serial port
    String currentLine = "";                // make a String to hold incoming data from the
client
    while (client.connected()) {            // loop while the client's connected
      if (client.available()) {             // if there's bytes to read from the client,
        char c = client.read();             // read a byte, then
        Serial.write(c);                    // print it out the serial monitor
        header += c;
        if (c == '\n') {                    // if the byte is a newline character
          // if the current line is blank, you got two newline characters in a row.
          // that's the end of the client HTTP request, so send a response:
          if (currentLine.length() == 0) {
            // HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)
            // and a content-type so the client knows what's coming, then a blank line:
            client.println("HTTP/1.1 200 OK");
```

```
client.println("Content-type:text/html");
client.println("Connection: close");
client.println();

// turns the GPIOs on and off
if (header.indexOf("GET /4/on") >= 0)
{
  Serial.println("GPIO 4 on");
  output0State = "on";
  digitalWrite(output0, HIGH);
}
else if (header.indexOf("GET /4/off") >= 0)
{
  Serial.println("GPIO 4 off");
  output0State = "off";
  digitalWrite(output0, LOW);
}
// turns the GPIOs on and off
else if (header.indexOf("GET /14/on") >= 0)
{
  Serial.println("GPIO 14 on");
  output1State = "on";
  digitalWrite(output1, HIGH);
}
```

```
            else if (header.indexOf("GET /14/off") >= 0)
            {
              Serial.println("GPIO 14 off");
              output1State = "off";
              digitalWrite(output1, LOW);
            }
            // turns the GPIOs on and off
            else if (header.indexOf("GET /12/on") >= 0)
            {
              Serial.println("GPIO 12 on");
              output2State = "on";
              digitalWrite(output2, HIGH);
            }
            else if (header.indexOf("GET /12/off") >= 0)
            {
              Serial.println("GPIO 12 off");
              output2State = "off";
              digitalWrite(output2, LOW);
            }
            // turns the GPIOs on and off
            else if (header.indexOf("GET /5/on") >= 0)
            {
              Serial.println("GPIO 5 on");
              output3State = "on";
```

```
            digitalWrite(output3, HIGH);
          }
          else if (header.indexOf("GET /5/off") >= 0)
          {
            Serial.println("GPIO 5 off");
            output3State = "off";
            digitalWrite(output3, LOW);
          }
          // Display the HTML web page
          client.println("<!DOCTYPE html><html>");
          client.println("<head><meta name=\"viewport\" content=\"width=device-
width, initial-scale=1\">");
          client.println("<link rel=\"icon\" href=\"data:,\">");
          // CSS to style the on/off buttons
          // Feel free to change the background-color and font-size attributes to fit your
preferences
          client.println("<style>html { font-family: Helvetica; display: inline-block;
margin: 0px auto; text-align: center;}");
          client.println(".button { background-color: #195B6A; border: none; color:
white; padding: 16px 40px;");
          client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor:
pointer;}");
          client.println(".button2 {background-color: #77878A;}</style></head>");
          // Web Page Heading
          client.println("<body><h1>Home Automation</h1>");
           client.println("<body><h2>using Local Web Server</h2>");
```

```
            // Display current state, and ON/OFF buttons for GPIO 4
            client.println("<p>GPIO 4 - State " + output0State + "</p>");
            // If the output0State is off, it displays the ON button
            if (output0State == "off") {
              client.println("<p><a href=\"/4/on\"><button
class=\"button\">ON</button></a></p>");
            } else {
              client.println("<p><a href=\"/4/off\"><button class=\"button
button2\">OFF</button></a></p>");
            }
            // Display current state, and ON/OFF buttons for GPIO 14
            client.println("<p>GPIO 14 - State " + output1State + "</p>");
            // If the output1State is off, it displays the ON button
            if (output1State == "off") {
              client.println("<p><a href=\"/14/on\"><button
class=\"button\">ON</button></a></p>");
            } else {
              client.println("<p><a href=\"/14/off\"><button class=\"button
button2\">OFF</button></a></p>");
            }
            // Display current state, and ON/OFF buttons for GPIO 12
            client.println("<p>GPIO 12 - State " + output2State + "</p>");
            // If the output2State is off, it displays the ON button
            if (output2State == "off") {
              client.println("<p><a href=\"/12/on\"><button
class=\"button\">ON</button></a></p>");
```

```
            } else {
              client.println("<p><a href=\"/12/off\"><button class=\"button
button2\">OFF</button></a></p>");
            }
            // Display current state, and ON/OFF buttons for GPIO 5
            client.println("<p>GPIO 5 - State " + output3State + "</p>");
            // If the output3State is off, it displays the ON button
            if (output3State == "off") {
              client.println("<p><a href=\"/5/on\"><button
class=\"button\">ON</button></a></p>");
            } else {
              client.println("<p><a href=\"/5/off\"><button class=\"button
button2\">OFF</button></a></p>");
            }
            client.println("</body></html>");

            // The HTTP response ends with another blank line
            client.println();
            // Break out of the while loop
            break;
          } else { // if you got a newline, then clear currentLine
            currentLine = "";
          }
        } else if (c != '\r') {  // if you got anything else but a carriage return character,
          currentLine += c;      // add it to the end of the currentLine
```

```
        }
      }
    }
    // Clear the header variable
    header = "";
    // Close the connection
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }
}
```