

A VOICE CONTROLLED AND VISION BASED SMART WHEEL CHAIR FOR PARALYZED PEOPLE

A PROJECT REPORT

Submitted by

MUTHU SELVAM.S (310617205044)

AJAI KUMAR.M (310617205004)

KAUSIC NARAYANAN M.N (310617205025)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY



EASWARI ENGINEERING COLLEGE, CHENNAI

(Autonomous Institution)

affiliated to

ANNA UNIVERSITY: CHENNAI 600 025

JULY 2021

EASWARI ENGINEERING COLLEGE, CHENNAI

(AUTONOMOUS INSTITUTION)

AFFILIATED TO ANNA UNIVERSITY, CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report “**A voice controlled and vision based smart wheel chair for paralyzed people**” is the bonafide work of “**Muthu Selvam. S (310617205044), Ajai Kumar. M (310617205004), Kausic Narayanan M.N(310617205025)**” who carried out the project work under my supervision.

SIGNATURE

Dr.N.Ananthi,M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

Department of Information Technology

Easwari Engineering College

Ramapuram, Chennai-89

SIGNATURE

Dr.R.Priyatharshini, M.E.,Ph.D.,

SUPERVISOR

Department of Information Technology

Easwari Engineering College

Ramapuram, Chennai-89

Submitted for Semester Examination held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

VIVA VOCE EXAMINATION

The viva voce examination of the project work, submitted by

MUTHU SELVAM. S **Register Number: 310617205044**

AJAI KUMAR. M **Register Number: 310617205004**

KAUSIC NARAYANAN M.N **Register Number: 310617205025**

is held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We express our sincere thanks and gratitude to our Founder Chairman **Dr.T.R.PARIVENDHAR** and our beloved Chairman **Dr.R.SHIVAKUMAR**, for their support and inspiration. We would like to convey our due respect and regards to our Principal **Dr.R.S.KUMAR** and Head of the Department **Dr.N.ANANTHI**, for their constant encouragement and guidance.

With a deep sense of gratitude we would like to thank our professors **Dr.A.K.MARIAPPAN** and **Dr.D.SIVAKUMAR** for their motivation, timely help and valuable suggestions.

We owe our profound gratitude to our project supervisor **Dr.R.PRIYATHARSHINI**, Associate Professor, who took keen interest in our project work and guided us all along in bringing out this project in complete shape by providing all the necessary information for developing a good system.

We would like to sincerely thank our project coordinator **Dr.R.PRIYATHARSHINI**, Associate Professor for their valuable assistance and support.

Finally we would like to extend our token of appreciation to all the teaching and non-teaching staff of Department of Information Technology for their kind co-operation throughout the project.

ABSTRACT

There are 144 million disabled people who use a wheelchair in 32 developed and 155 developing countries, accounting for 1.82 percent of the global population. Furthermore, lots of people suffer from diseases that cause them to be unable to produce coordinated movement in any of their limbs or even their heads. An eyeball and voice-controlled system is developed to control the movement of wheel chair for paralyzed people. The movement of the eye ball is tracked using the camera module, and the motor driver controls the direction of the wheel chair based on pupil location. The wheel chair movement also controlled via voice control. The proposed system also senses an obstacle with an ultrasonic sensor to halt the wheel chair if any obstacle is detected. The notification module also developed to communicate with their guardian through voice command via the message in the event of an emergency.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGENO.
	ABSTRACT	v
	LIST OF TABLES	ix
	LIST OF FIGURES	x
	LIST OF ABBREVIATION	xi
1.	INTRODUCTION	1
	1.1 GENERAL	1
	1.2 OBJECTIVE	2
	1.3 ORGANISATION	3
	OF PROJECT REPORT	
	1.4 SUMMARY	3
2.	LITERATURE SURVEY	4
	2.1 INTRODUCTION	4
	2.2 RELATED WORKS	4
	2.3 LIMITATIONS OF	6
	EXISTING SYSTEM	
	2.4 SUMMARY	6

3.	PROPOSED SYSTEM	7
	3.1 INTRODUCTION	7
	3.2 PROPOSED SYSTEM	7
	3.3 SYSTEM ARCHITECTURE	8
	3.3.1 Eye movement based wheel Chair control	9
	3.3.2 Voice based wheel chair control	10
	3.4 SUMMARY	10
4.	SYSTEM IMPLEMENTATION	11
	4.1 INTRODUCTION	11
	4.2 HARDWARE AND SOFTWARE SPECIFICATION	11
	4.2.1 Software Requirements	11
	4.2.2 Hardware Requirements	11
	4.3 COMPONENTS USED	12
	4.3.1 Raspbian OS	12
	4.3.2 VNC Viewer	12
	4.3.3 Python	13
	4.3.4 Open CV	13
	4.3.5 Raspberry PI	14
	4.3.6 Web camera	15
	4.3.7 Ultrasonic Sensor	16
	4.3.8 DC Motor	16

4.4	MODULE IMPLEMENTATION	17
4.4.1	Eye ball movement tracker	17
4.4.1.1	Flow chart for eye tracking	20
4.4.1.2	Output for eye ball movement tracker	21
4.4.2	Speech Synthesis and Recognition	22
4.4.2.1	Flow chart for Speech Recognition	23
4.4.2.2	Output of speech synthesis and recognition	24
4.4.3	Wheel chair motor driver	24
4.5	SUMMARY	25
5.	PERFORMANCE ANALYSIS	26
5.1	INTRODUCTION	26
5.2	PERFORMANCE TESTING	26
5.3	PERFORMANCE EVALUATION	27
5.4	SUMMARY	30
6.	CONCLUSION AND FUTURE WORK	31
6.1	CONCLUSION	31
6.2	FUTURE WORK	31
	APPENDIX	32
	REFERENCES	44

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
5.1	Results of left eye ball movement detections	27
5.2	Results of Forward movement detections	28
5.3	Results of right movement detections	28
5.4	Performance analysis of voice controlled wheel chair	29
5.5	Performance analysis of Eye controlled wheel chair	29

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.3	System Architecture	9
3.3.2	Block diagram of voice controlled wheel chair	10
4.3.5	Raspberry PI diagram	15
4.3.7	Ultrasonic sensor	16
4.4.1.1	Flow chart for eye tracking	20
4.4.1.2	Output of eye detection in normal position	21
4.4.1.3	Output of eye motion detected while looking right	21
4.4.1.4	Output of eye motion detected while looking left	22
4.4.2.1	Flow chart for user voice command	23
4.4.2.2	Output of voice control module	24
6.1	User interface of the application	42
6.2	Output of alert message module	42
6.3	Output of obstacle detection	43

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
HMI	Human Machine Interface
EEG	Electroencephalography
EMG	Electromyography
EOG	Electrooculography
MEMS	Micro-electro-mechanical-systems
GUI	Graphical User Interface
IR	InfraRed
IDE	Integrated Development Environment
DC	Direct Current
CPU	Central Processing Unit
VNC	Virtual Network Computing
LXDE	Lightweight X11 Desktop Environment
USB	Universal Serial Bus
HD	High Definition
AC	Alternating Current

CHAPTER-1

INTRODUCTION

1.1 GENERAL

The number of persons who are paralyzed and therefore dependent on others due to loss of self mobility is growing with the population. Significant strides made in the fields of rehabilitation, artificial intelligence (AI) (especially around the implementation of complex algorithms for analysis and interpretation of human cognition), and human machine interfaces (HMIs), have opened a new evolutionary pathway for the development of smart mobility aids . People who accidentally lose their lower limbs or suffer from conditions such as quadriplegia or stroke, resulting in paralysis, and muscle stiffness are unable to make use of conventional wheelchairs.

Researchers across the world are engaged in developing medical devices/rehabilitation aids for physically challenged populations, such as quadriplegics, to enable them to carry out their daily work without or with minimal assistance from caregivers and nurses, etc. . They are thus increasing the self-esteem and functional capabilities of such patients with the ultimate goal of improving the patients' quality of life. Biosignals recorded through electroencephalography (EEG), electromyography (EMG), and electrooculography (EOG), etc. have been exploited by researchers in developing smart, responsive and real-time rehabilitative control systems.

The development of the wheelchair for paralyzed users is surprisingly recent starting with the conventional manually powered wheelchairs and advancing to electrical wheelchairs. Conventional wheelchair use tends to focus exclusively on manual use which assumes users still able to use their hands which excludes those unable to do so. Diseases or accidents injuring the nervous system also frequently because people lose their ability to move their

voluntary muscle. Because voluntary muscle is the main actuator enabling people to move their body, paralysis may cause a person not move their locomotor organ such as arm, leg and others. Paralysis may be local, global, or follow specific patterns. Most paralysis are constant, however there are other forms such as periodic paralysis (caused by genetic diseases), caused by various other factors.

More recently, eye gesture control-based systems have gained significant attention due to the fact that even in the most seriously physically challenged population, such as quadriplegics, eye movements are still intact; the main operating mechanism of the eye-controlled based systems . Therefore, keeping in view this fact and the need to advance the adoption of independent mobile rehabilitation technology, such as smart wheelchairs and walkers, a distinctive eyeball movement-based technique for controlling a wheelchair is presented, leading to increased patient comfort.

1.2 OBJECTIVE

The main objectives of the work are expressed as follows:-

- To develop an Eye ball movement and voice command based wheel chair controller for paralyzed people.
- The device ought to be ready to detect obstacle during the wheel chair movement.
- The device ought to send message to the guardian in case of any emergency.

1.3 ORGANISATION OF THE PROJECT REPORT

The work done in various phases has been organized into chapters. Chapter 2 gives details about the existing system and its drawbacks. Chapter 3 gives details about proposed system, design and architecture. Chapter 4 gives details about different modules and their operations. Chapter 5 provides detail description of various tests performed and the performance and result analysis. Chapter 6 discuss about conclusion and the possible future enhancements.

1.4 SUMMARY

This chapter gives the general introduction of the domain. The required concepts which are made use of in the project are analyzed and the objective of our project is also discussed.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

A literature survey comprises of a comprehensive review of the published and unpublished work and findings from various sources related in the areas of specific interest to the researcher.

2.2 RELATED WORKS

A motion control method of intelligent wheelchair based on hand gesture recognition(2018)

In this paper Sarangi.P et al designed a device to regulate a motion-based approach based on the movement of the organs. Hand gestures are detected and measured by a Micro-electro-mechanical-systems (MEMS) instrument and then transmitted to a computer system via Bluetooth network, where a Markov algorithm is used for hand gesture movement training and another segmentation algorithm is used to classify gestures. However, the main concern occurs when a patient is unable to move all of his or her body parts.

A low cost Human Computer Interface for Disabled People based on Eye Blink detection using Brain Signal(2017)

Rajesh kannan introduced a device to detect and capture eye movements, the system described in this paper uses a Brain Controlled Interface based approach. Every minute, the patient's brain waves are recorded, and the data is used to determine whether or not eye motion is present, and the wheelchair is then pushed based on that information. However, the biggest disadvantage of this approach is that using Brain Wave Sensors all of the time makes paralyzed people

nervous and makes them reliant on external devices.

Autonomous camera based eye controlled wheelchair system using Eye ball cursor movement(2019)

Dr. zhang and pheng built a system for the disabled people who are unable to perform activities with their hands or legs are the subjects of this paper. A graphical user interface (GUI) is being created for such handicapped people. The GUI is shown on a screen with options for a variety of tasks. Up to this point, four separate motions have been performed and regulated using the GUI: forward, backward, left, and right.

Integrating Human Input with autonomous behaviors on an Intelligent Wheelchair Platform(2020)

An IR based eye pupil detection device was suggested by R. Barea, L. Boquete which consist of an IR transmitter and IR receiver used to drive the wheelchair. The light is transmitted over the iris by the IR transmitter, and the reflected light is absorbed by the IR receiver. The controller can detect the user's intention of wheelchair movement based on the strength of reflected light absorbed by the receiver. The importance of this procedure is that it can have reliable results; however, this device has an effect on the eye and can cause vision loss.

Emotion Recognition based smart wheelchair for disabled person(2018)

The prime objective of the device developed by S. Ravi Kumar, S. Shenbagavadivu, and et-al is to control wheelchair using image analysis method where it utilizes camera in-order to capture the facial images and software is developed and uploaded into Raspbian model where it processes and classifies the facial expression into happy, sad, angry and transmitted to the system via

ZigBee to move the wheelchair It is not helpful to all users, especially those who suffer from restrictions in facial expressions due to diseases like facial paralysis. Moreover, classification of facial expressions is more challenging than the eye-controlled system, where only the eye is targeted.

2.3 LIMITATIONS OF EXISTING SYSTEM

- GUI and cursor based methodology was used in the existing system which requires a eye blink every time as it makes the system a uncomfortable one for the user.
- Infrared rays are also used to control the wheel chair by using eye ball movement which is very inefficient as it requires a external device every time and it is also harmful.
- Using Brain Wave Sensors all the time makes paralyzed people uncomfortable.
- Handicapped people will always be dependent on third persons or external device such as remote controller if they use remote control type of wheel chair controlling system.
- Gesture based wheel chair cannot be used by Paralyzed people as free movement of hands are not possible by them.

2.4 SUMMARY

This chapter covers the survey of the existing systems and their limitations. A smart wheel chair for paralyzed people which can be controlled by voice commands and eye ball movement has been proposed in order to overcome the limitations present in the current scenario.

CHAPTER 3

PROPOSED SYSTEM DESIGN

3.1 INTRODUCTION

System design is that the method of process the design, components, modules, interfaces and information for a system to satisfy such requirements. System design may well be seen because the application of systems theory to development. System design chiefly deals with planning and making process of the proposed system. It explains the general flow of operations within the planned system.

3.2 PROPOSED SYSTEM

In this proposed system, the camera focuses on the eye, and live footage is recorded. Then, using OpenCV, we must calculate the eyeball's centroid. Then, depending on the pupil location, a different command set for the wheelchair is created. We can adjust the wheel chair by tracking the eyeball movement. The wheelchair would be able to travel forward, left, right, and stop which is controlled by the motor driver's regulation of both speed and direction. It is particularly helpful to the elderly and disabled who need assistance in their everyday lives. By blinking our eyes, we can start and stop the wheelchair's operation. A mobile can also be used to control the wheelchair by using voice commands. To detect the obstacle, ultrasonic sensors are used, which are mounted on the wheelchair. When a wheelchair meets an hindrance, instructions are sent to the driving circuit to stop the motor immediately. In our proposed system there are two ways to steer the wheelchair in the desired direction by using Eye ball movement or by Voice command.

3.3 SYSTEM ARCHITECTURE

The system design as shown in Fig.3.1 below consists of functionalities and every one the info relating the system are going to be displayed during this process. The info includes that the user has to sits in front of the display screen of private computer or pc, a specialized video camera established above the screen to study the consumer's eyes. The laptop constantly analysis the video photo of the attention and determines wherein the consumer is calling at the display screen. not anything is attached to the consumer's head or body. To "pick out" any key, the user seems at the key for a exact period of time and to "press" any key, the consumer just blink the eye. On this device, calibration procedure is not required. For this system enter is simplest eye. No outside hardware is connected or required. Camera gets the input from the eye. After receiving these streaming movies from the cameras, it'll spoil into frames. After receiving frames, it will check for lights conditions because cameras require enough lighting fixtures from external sources in any other case blunders message will show at the screen. The captured frames which can be already in RGB mode are transformed into Black 'n' White. Five. Pics (frames) from the enter supply focusing the eye are analysed for Iris detection (middle of eye).After this, a mid point is calculated through taking the suggest of left and right eye centre point. Eventually the mouse will pass from one position to any other at the display and consumer will perform clicking with the aid of blinking their eyes for 5 seconds.

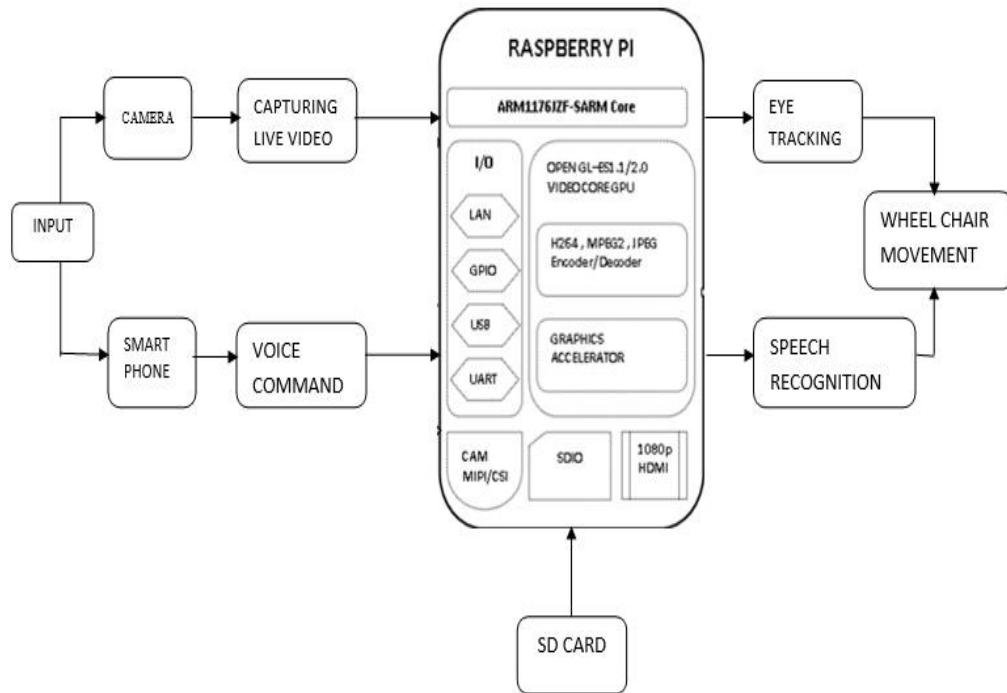


Fig 3.3 System Architecture

The system architecture consists of two modules. They are

1. EYE MOVEMENT BASED WHEEL CHAIR CONTROL
2. VOICE COMMAND-BASED WHEEL CHAIR CONTROL

3.3.1 Eye Movement based wheel chair control

A vision based approach is implemented in this system. To begin with, the device detects multiple or single faces and uses eye movement to move the wheelchair. The algorithm used is the HAAR cascade algorithm. The HAAR classifiers are qualified for facial features, and a user's face is detected using that information to locate the eye-region. Feature detection, eye pupil centre localization, and eye-tracking are all performed using the Hough circle transform algorithm.

3.3.2 Voice based wheel chair control

A Voice mode selection is made available for indoor use for the easy movement of the chair and the operation of the module requires command from the user. The Raspberry Pi program includes an integrated development environment (IDE) that allows you to upload and write built-in commands as well as configure messages. To send commands, a Voice Bot App is installed on the patient's phone. The voice module is fully reliant on a Bluetooth connection. In the event of an emergency, the user may use an SMS-based message feature.

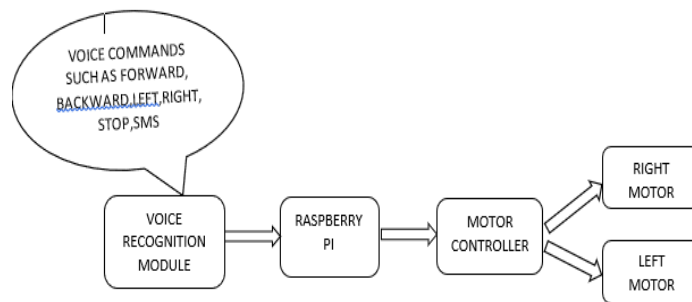


Fig:3.3.2. Block diagram of voice controlled wheelchair.

3.4 SUMMARY

This chapter summarizes details about the proposed system's design. System architecture of the proposed system is drawn and the modules in the architecture are explained.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 INTRODUCTION

The system implementation is a technical specification of requirements for the hardware products. It is the first step in the requirements analysis process. It lists the requirements of a particular hardware system including functional, performance and security requirements. The whole system is hardware and software oriented and built for specific purpose and goals.

4.2 HARDWARE AND SOFTWARE SPECIFICATION

4.2.1 Software Requirements

- Raspberry pi OS: Raspbian stretch
- VNC viewer
- Programming language: python 3
- Library: OpenCV

4.2.2 Hardware Requirements

- Raspberry pi
- SD card
- Camera
- Ultrasonic sensor
- DC Motor

4.3 COMPONENTS USED

4.3.1 Raspbian OS

Raspberry Pi OS[3] (formerly Raspbian) is a Debian-based operating system for Raspberry Pi. Since 2015, it has been officially provided by the Raspberry Pi Foundation as the primary operating system for the Raspberry Pi family of compact single-board computers. The first version of Raspbian was created by Mike Thompson and Peter Green as an independent project. The initial build was completed in June 2012.

Raspberry Pi OS is highly optimized for the Raspberry Pi line of compact single-board computers with ARM CPUs. It runs on every Raspberry Pi except the Pico microcontroller. Raspberry Pi OS uses a modified LXDE as its desktop environment with the Openbox stacking window manager, along with a unique theme. The distribution is shipped with a copy of the algebra program Wolfram Mathematical and a version of Minecraft called Minecraft: Pi Edition, as well as a lightweight version of the Chromium web browser

4.3.2 VNC VIEWER

VNC stands for virtual network computing. This is a desktop sharing system that allows you to remotely control another computer. VNC works by transmitting all of your keyboard and mouse movements from your thin client computer to the other, large client computer.

VNC is platform-independent – there are clients and servers for many GUI-based operating systems and for Java. Multiple clients may connect to a VNC server at the same time. Popular uses for this technology include remote technical support and accessing files on one's work computer from one's home computer, or vice versa.

4.3.3 PYTHON

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant white space. It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative functional and procedural, it also has a comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software]and has a community based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation. Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than C or Pascal.

4.3.4 OPEN CV

OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then It sees (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online

documentation. Wrappers in other languages such as C#, Perl, Ch, Haskell, and Ruby have been developed to encourage adoption by a wider audience. All of the new developments and algorithms in OpenCV are now developed in the C++ interface. OpenCV runs on the following desktop operating systems : Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD. OpenCV runs on the following mobile operating systems: Android, iOS, Maemo, BlackBerry 10. The user can get official releases from Source Forge or take the latest sources from GitHub. OpenCV uses CMake.

4.3.5 RASPBERRY PI

Raspberry Pi (/paɪ/) is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The original model became more popular than anticipated, selling outside its target market for uses such as robotics. It is widely used in many areas, such as for weather monitoring, because of its low cost, modularity, and open design. It is typically used by computer and electronic hobbyists, due to its adoption of HDMI and USB devices. Raspberry pi board is brain of the system. Raspberry pi board have its own operating system is known as raspbian which is Linux based operating system and compatible with raspberry pi board. A real time data receive and determine the digital data by raspberry pi B+ model board, which is very efficiently work with the multiple images. Raspberry pi sends the command to motor driver which is enabling the GPIO pin to raspberry pi.

There are 3 Processors

The processor at the heart of the Raspberry Pi is a Broadcom BCM28XX.

This is the Broadcom System on Chip (SOC) chip use in the Raspberry Pi. The processor from first to third generations include:

- Raspberry Pi 1: Broadcom BCM2835 SOC with 700MHz CPU speed, L2 cache of 128kb with ARM compatibility AR1176JZF-S (ARMv6) 32-bit RISC ARM.
- Raspberry Pi 2: Broadcom BCM 2836 SOC with 900MHz CPU speed, L2 cache of 256kb with 32-bit quad-core ARM cortex-A7 (ARMv7).
- Raspberry Pi 3: Broadcom BCM2837 SOC with 1.2GHz 64-bit quad-core –A53 with 512 kb shared L2 cache (64-bit instruction set ARMv8).

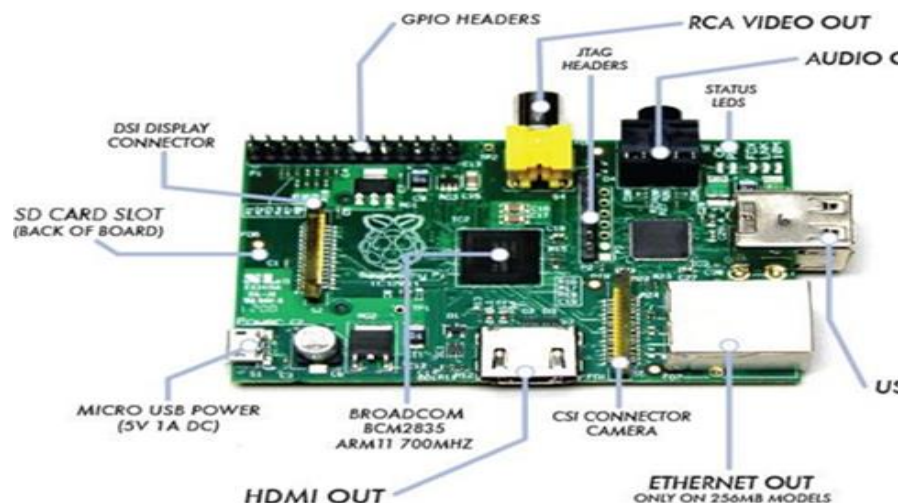


Fig 4.3.5 : Raspberry pi diagram

4.3.6 WEB CAMERA

Web camera is used for capturing the image. We can also use HD (high definition) camera but it increase the memory size, system can't able to read the image and it will increase the processing time. UV4L driver is needed for interfacing a camera with raspberry pi board.

Webcams can be used as security cameras. Software is available to allow PC-connected cameras to watch for movement and sound,[7] recording both when they are detected. These recordings can then be saved to the computer, e-mailed, or

uploaded to the Internet.

4.3.7 ULTRASONIC SENSOR

Ultrasonic sensor is used to detect obstacle in the path of wheelchair. Sensor is directly connected to the raspberry pi board. It receives the data and measuring the distance between wheelchair and obstacle. If any obstacle is detected every close to wheelchair, motors will stop to run the wheels. Ultrasonic sensor is a very affordable proximity / distance sensor that has been used mainly for object avoidance in various robotics projects. Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception. An optical sensor has a transmitter and receiver, whereas an ultrasonic sensor uses a single ultrasonic element for both emission and reception. In a reflective model ultrasonic sensor, a single oscillator emits and receives ultrasonic waves alternately. This enables miniaturization of the sensor head.



Fig 4.3.7 : Ultrasonic sensor

4.3.8 DC Motor

An electric motor is an electrical machine that converts electrical energy into mechanical energy. Most electric motors operate through the interaction between the motor's magnetic field and electric current in a wire winding to generate force

in the form of torque applied on the motor's shaft. Electric motors can be powered by direct current (DC) sources, such as from batteries, motor vehicles or rectifiers, or by alternating current (AC) sources, such as a power grid, inverters or electrical generators. An electric generator is mechanically identical to an electric motor, but operates with a reversed flow of power, converting mechanical energy into electrical energy.

4.4 MODULE IMPLEMENTATION

The proposed system consists of three stages. They are

1. Eye ball movement tracker
2. Speech synthesis and recognition
3. Wheel chair motor driver module

4.4.1 Eye ball movement tracker

The camera captures the live feed of the user's face, the user's eye, and the pupil of the user's eye. The machine must recognize the pupil of the eye and characterize its internal focuses. This is achieved by the use of various image processing techniques such as blurring, edge detection, thresholding, colour transfer, Hough transform, and filtering. When the eye area is marked, two rectangular boxes are drawn around it. To detect and clear blur images, a filtering technique is used. Following the identification of the eye area, the eye edge is labelled in order to locate the eye pupil using coordinate points. The coordinate system is used to locate the pupil, which assists in the identification of eye-pupil movement, allowing the wheelchair to be directed in the desired direction. The captured images are processed using the OpenCV library, and the classifier is run using the OpenCV Haar Detect Objects functions.

Face and Eye-detection using HAAR cascade algorithm:

The HAAR cascade algorithm is a Machine Learning (ML) based technique that is trained for various images and is used to detect both face and eye. There are four stages in this algorithm

1. Feature selection using HAAR
2. Constructing integral images
3. Adaboost training
4. Cascading HAAR classifiers

Haar-Cascade is an artificial intelligence object location calculator that identifies features in images. To arrange the classifier, the calculation first requires a large number of positive and negative images (pictures without faces). In the meantime, we need to untangle it from points of interest. Haar does this by highlighting the picture region that has been used. As far as our convolutional-section is concerned, they're plenty constant. To obtain each part of solitary value, the whole numbers of pixels under white-square shape are subtracted from a group of pixels under dark-square shape in this process.

ALGORITHM FOR EYE MOVEMENT TRACKER

Haar-Cascade is an AI based object location calculator used to identify features in an image. At first, the calculation wants plenty of positive-images and negative to arrange the classifier. In the mean-time, have got to disencumber places of interest from it. For this, Haar marks the utilized image area that is beneath. They're ample constant as of our convolutional- portion.

Input : Pick(maximum acceptable false positive rate per layer) and d (minimum acceptable detection rate per layer)

Output : Set of facial features in a captured image

Initialization:

- Let F_{target} is target overall positive rate
- Let P is a set of positive examples
- Let n is a set of negative examples
- Let $F_0=1$, $D_0=1$, and $i=0$ (F_0 : overall false positive rate at layer 0)
- D_0 : acceptable detection rate at layer 0, and i : is the current layer)

Step 1 : While F_i is greater than F_{target} (F_i : overall false positive rate at layer i)

Step 2: Increase the count of I by 1.

Step 3 : Initialize n_i to 0 and $F_i=F_{i-1}$ (n_i : negative example i)

Step 4 : While F_i is greater than $f \cdot F_{i-1}$:

Step 5 : Increase the count of n_i by 1.(check a next negative example)

Step 6 : Use P and N to train with Adaboost to make a xml (classifier)

Step 7 : check the result of new classifier for F_i

Step 8 : Decrease threshold for new classifier to adjust detection rate

Step 9 : Check if r is greater than $d \cdot F_{i-1}$

Step 10 : $N=\text{empty}$

Step 11 : If F_i is greater than F_{target} then

Step 12 : use the current classifier and false detection set to N .

4.4.1.1 FLOW CHART FOR EYE TRACKING

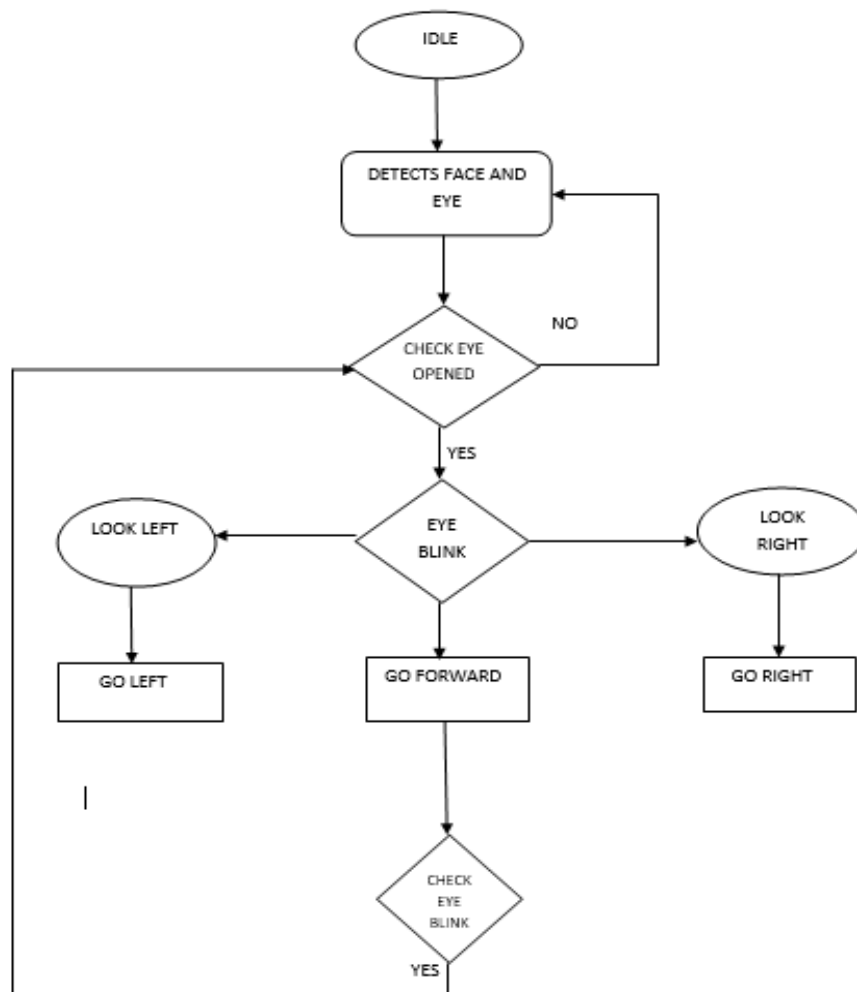


Fig 4.4.1.1 : Flow chart for Eye tracking

4.4.1.2 OUTPUT FOR EYE BALL MOVEMENT TRACKER



Fig 4.4.1.2 : Output of eye motion detected in normal position

Fig 4.4.1.2 shows a person who looks straight into the camera, the pupil is detected from the live video stream which is processed by the system as normal position and therefore makes the wheel chair halt



Fig 4.4.1.3 : Output of eye motion detection while looking right

Figure 4.5 shows a person with movement of eye in right direction, the pupil is detected based on eyeball movement that leads to the right turn of wheel chair.

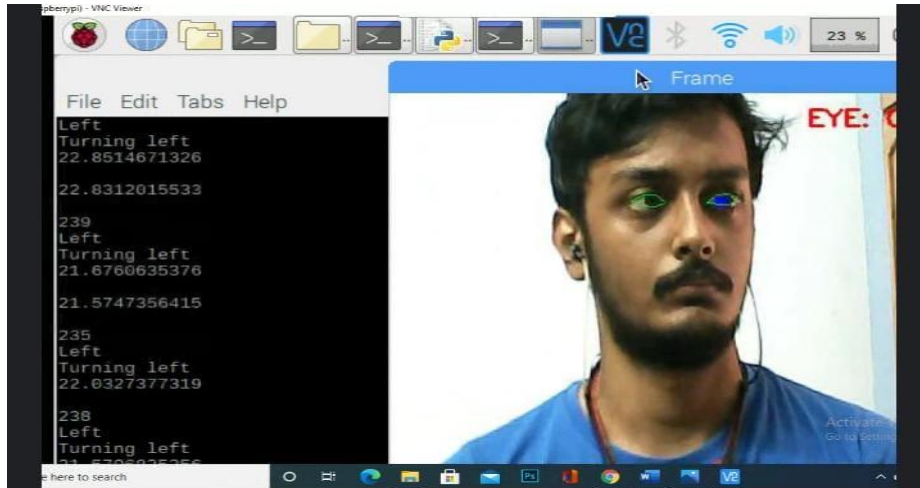


Fig 4.4.1.4: Output of eye motion detection while looking left

Figure 4.6 depicts a person with left eye movement, the pupil is detected based on motion of the eyeball that corresponds to the left turn of wheel chair.

4.4.2 SPEECH SYNTHESIS AND RECOGNITION

This module's main feature is the voice recognition module, which is used to configure the desired voice command and performance. Voice customization, voice capture, and voice recognition are the three steps of the process. The process of matching the desired voice recorded to the desired output signal is known as voice customization. The voice capture process records and stores the desired person's voice command depending on the customization setup. The voice recognition process is the final phase, in which this module recognizes a voice command and sends a particular signal to the microcontroller to perform the required task. The recorded voice command is checked by repeating the five

commands that were previously recorded. Forward, Backward, Turn Left, Turn Right, and Stop are the five voice commands used. In order to monitor the wheelchair movement, the Bluetooth module is used as a wireless communication medium between the Raspberry pi and the cell phone. This eliminates the need for long, tangled wires.

4.4.2.1 FLOW CHART FOR SPEECH RECOGNITION

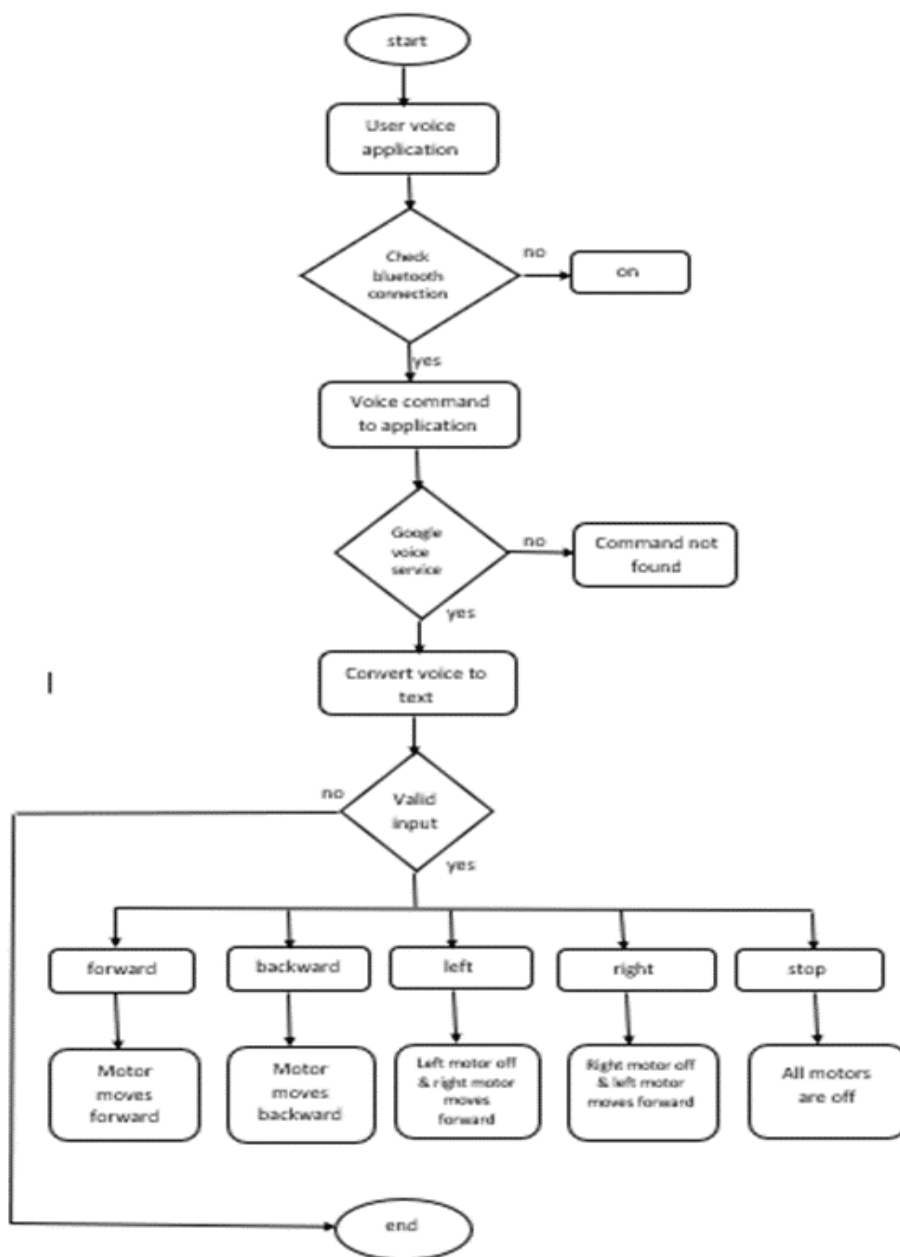


Fig 4.4.2.1 : Flowchart for User Voice Command

4.4.2.2 OUTPUT OF SPEECH SYNTHESIS AND RECOGNITION

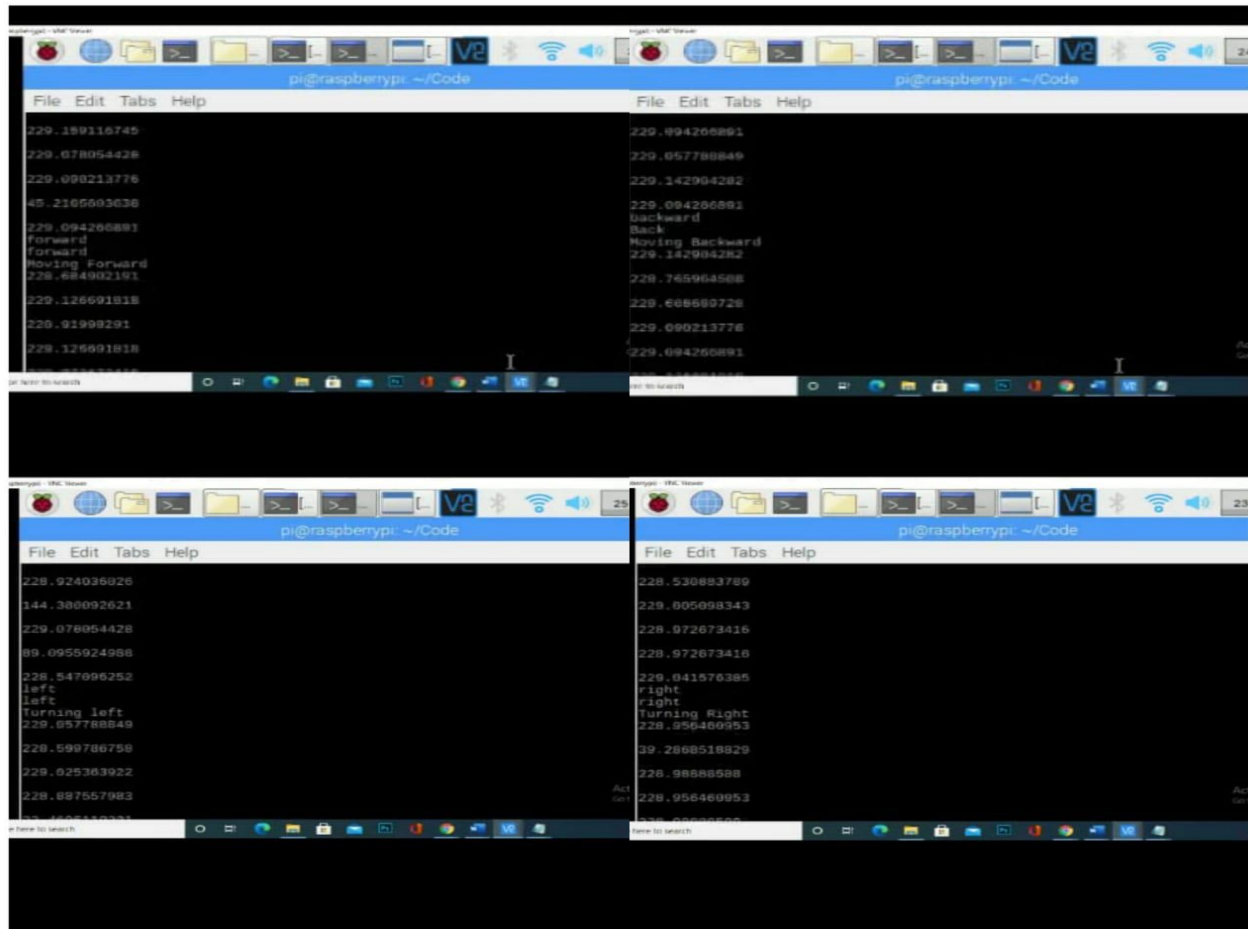


Fig 4.4.2.2: Output of voice control module

4.4.3 Wheel chair motor driver

The movement of a small wheel chair prototype is powered by a dc motor. The Motor driving IC is controlled by a two-channel relay board, and power is supplied by a battery. The Raspberry Pi is equipped with a motor driving circuit that aids in the operation of the device. This module is set up to monitor the user's eye movements and record voice feedback in order to guide the wheelchair in the desired direction. Each command will have a unique motor control path that can be modified on the fly. For example, if the voice command is to turn left, the left wheel will be set to reverse and the right wheel to forward, while if the voice

command is to turn right, the left wheel will be set to forward and the right wheel will be set to backward. The forward role not only regulates the speed of the wheelchair's motor, but it also senses obstacles on both sides. This role is heavily reliant on ultrasonic sensors to sense the distance between the obstacle and the wheelchair in order to complete a feedback loop that adjusts the wheel speed and allows the wheelchair to change direction.

4.2.4 SUMMARY

This chapter provides details about the proposed system's implementation. It gives a detailed view on the technologies used, the various phases involved and their implementation.

CHAPTER 5

PERFORMANCE ANALYSIS

5.1 INTRODUCTION

Performance analysis is the process of evaluating how a particular software program is functioning. This process normally begins with how the program loads and what happens when each step in using the program is executed. The objective of performance analysis is to ensure the software program is working at optimum efficiency and to identify and correct any issues that may negatively impact that efficiency. Performance analysis is a measure of the success or failure of a project using various parameters. It helps in developing a positive culture of project management that yields excellent results.

5.2 PERFORMANCE TESTING

Performance Testing is an activity to verify that a correct system is being built and is performed with the intent of finding faults in the system. Testing is a process of executing a program or application with the intent of finding software bugs. It can also be stated as the process of validating and verifying that a software program or application or product to meet the business and technical requirements that guided its design and development, and to Work as expected and to verify whether they are implemented with the same characteristic. Testing process is done after the development phase of the system. A system is bound to be imperfect without proper testing.

5.3 PERFORMANCE EVALUATION

The model is ready for performance analysis. The proposed model is compared to two other existing models. The traditional model is implemented and drives the input application. Since the duration of requests varies for each and every model, we have calculated the performance by running the algorithm thrice a week. The proposed system is compared to the existing systems on the basis of three criteria:

- i. The number of datasets given as input
- ii. The runtime in seconds averaged over solved problems
- iii. The average cost of the solution obtained

TABLE 5.1 Results of left eye ball movement detections

S.NO	Total no. of detections	Total no. of accurate left detections	Accuracy
1	60	19/20	95%
2	60	19/20	95%
3	60	19/20	95%
4	60	18/20	90%
5	60	20/20	100%
6	60	20/20	100%
7	60	20/20	100%
8	60	19/20	95%
9	60	18/20	90%
10	60	19/20	100%

Table 5.1 shows the performance measure of eye movement tracker module while moving left.

TABLE 5.2 Results of Forward movement detections

S.NO	Total no. of detections	Total no. of accurate forward detections	Accuracy
1	60	20/20	100%
2	60	19/20	95%
3	60	20/20	100%
4	60	19/20	95%
5	60	20/20	100%
6	60	19/20	95%
7	60	20/20	100%
8	60	20/20	100%
9	60	20/20	100%
10	60	20/20	100%

Table 5.2 shows the performance measure of eye movement tracker module while moving forward.

TABLE 5.3 Results of right eye ball movement detections

S.NO	Total no. of detections	Total no. of accurate right detections	Accuracy
1	60	19/20	95%
2	60	20/20	100%
3	60	18/20	90%
4	60	19/20	95%
5	60	18/20	90%
6	60	19/20	95%
7	60	19/20	95%
8	60	20/20	100%
9	60	19/20	95%
10	60	19/20	95%

Table 5.3 shows the performance measure of eye movement tracker module while moving right,.

TABLE 5.4 Performance Analysis for voice controlled wheel chair

S.NO	TITLE	Total no of tests	Total no of successful detection	Accuracy
1	Voice Controlled wheel Chair system	100	90	90%
2	Voice controlled Wheelchair using Arduino and voice recognition module	100	93	93%
3	A voice controlled and vision based wheel chair for paralyzed people(proposed)	100	98	98%

Table 5.4 shows the comparison of performance analysis with the existing systems for voice controlled wheel chair

TABLE 5.5 Performance Analysis for eye controlled wheel chair

S.NO	TITLE	Total no of tests	Total no of successful detection	Accuracy
1	Information Fusion based wheel chair control for paralyzed patient	100	92	92%
2	Iris movement based wheel chair control using arduino	100	91	91%
3	A voice controlled and vision based wheel chair for paralyzed people(proposed)	100	96	96%

Table 5.5 shows the comparison of performance analysis with the existing systems for eye controlled wheel chair

5.4 SUMMARY

This chapter mainly focuses on the performance analysis. The various testing methods are discussed. The performance of the proposed system is evaluated. It is seen that the accuracy is significantly increased.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

This initiative would be very beneficial to blind people as well as disabled people who are unable to see or walk properly. The benefit of this project is that the camera module is combined with the voice module, allowing the individual to offer the command as well. In the case that the camera fails to identify the eye properly, the voice module may be used as a substitute.

6.2 FUTURE WORK

The scope of the project includes the Information fusion based wheelchair control for the paralyzed patient. To make our system more effective, more number of sensors can be added that are highly capable of detecting obstacles accurately and readily. And also GPS can be provided to track the person if he is in the outdoor area. The system can also include a more sophisticated system which includes an LCD or monitor which includes basic user needs like food, water and toilet so that, when the eye-blink counts stop, it can ring an alarm to signal the guardians quickly.

APPENDIX

CODE

```
# USAGE
# python eyeballcursor.py --shape-predictor
# shape_predictor_68_face_landmarks.dat

# import the necessary packages
import RPi.GPIO as GPIO
from scipy.spatial import distance as dist
from imutils.video import VideoStream
from imutils import face_utils
from threading import Thread
import numpy as np
import serial
##import playsound
import argparse
import imutils
import time
import dlib
import cv2
import math
import pyautogui
import subprocess
#import urllib
import robot
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(19,GPIO.OUT)
GPIO.setup(26,GPIO.IN)
#url="http://192.168.1.44:8080/shot.jpg"
port=serial.Serial("/dev/serial0",9600,timeout=0.1)
def sound_alarm(path):
    # play an alarm sound
    playsound.playsound(path)

def eye_aspect_ratio(eye):
    # compute the euclidean distances between the two sets of
    # vertical eye landmarks (x, y)-coordinates
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])
    kk1=eye[1];
    kk2=eye[5];
```

```

x1=kk1[1];
y1=kk2[0];
h1=kk1[1]-kk2[1];
w1=kk2[0]-kk1[0];

# compute the euclidean distance between the horizontal
# eye landmark (x, y)-coordinates
C = dist.euclidean(eye[0], eye[3])
## print eye[0],eye[3]
vert1=eye[0];
vert2=eye[3];
x=vert1[1];
y=vert1[0];
h=vert1[1]-vert2[1];
w=vert2[0]-vert1[0];
if w==0:
    w=-1;
if h==0:
    h=1;
## print w

# compute the euclidean distance between the horizontal
# eye landmark (x, y)-coordinates
C = dist.euclidean(eye[0], eye[3])
## print eye[0],eye[3]

# compute the eye aspect ratio
EYE = (A + B) / (2.0 * C)

# return the eye aspect ratio
cnter=(eye[0]+eye[3])/2;
return EYE,h,w,x,y,cnter

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()

ap.add_argument("-p", "--shape-predictor", required=True,
    help="path to facial landmark predictor")

##ap.add_argument("-a", "--alarm", type=str, default="",
## help="path alarm .WAV file")
ap.add_argument("-w", "--webcam", type=int, default=0,
    help="index of webcam on system")
args = vars(ap.parse_args())

```

```

# define two constants, one for the eye aspect ratio to indicate
# blink and then a second constant for the number of consecutive
# frames the eye must be below the threshold for to set off the
# alarm
EYE_AR_THRESH = 0.25
EYE_AR_CONSEC_FRAMES = 1

# initialize the frame counter as well as a boolean used to
# indicate if the alarm is going off
COUNTER = 0
ALARM_ON = False

# initialize dlib's face detector (HOG-based) and then create
# the facial landmark predictor
##
##print("[INFO] loading facial landmark predictor...")
##detector = dlib.get_frontal_face_detector()
##predictor = dlib.shape_predictor(args["shape_predictor"])

cam = cv2.VideoCapture(0)
detector = dlib.get_frontal_face_detector()
predictor =
dlib.shape_predictor('/home/pi/Code/shape_predictor_68_face_landmarks.dat')

# grab the indexes of the facial landmarks for the left and
# right eye, respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

# start the video stream thread
print("[INFO] starting video stream thread...")
#vs = VideoStream(src=args["webcam"]).start()
time.sleep(1.0)
data=['q','w','e','r','t','y','u','i','o','p'];
# loop over frames from the video stream
cursdata=[[1,192],[193,384],[385,576]];
muldiffcentX=0;
muldiffcentY=0;
CentInitX=1920/2;
CentInitY=1080/2;
FrameCentX=960/2;
FrameCentY=540/2;
lp=0;

##light=19
##fan=26

```

```

##
##GPIO.setup(light, GPIO.OUT)
##GPIO.setup(fan, GPIO.OUT)
##
##GPIO.output(light, False)
##GPIO.output(fan, False)

flag1=0
flag2=0
flag3=0
def ultra():
    end=0
    start=0
    GPIO.output(19,True)
    time.sleep(0.0001)
    GPIO.output(19,False)
    w = time.time()
    while(GPIO.input(26)==0):
        start = time.time()
        if(time.time()-start>3):
            break
    while(GPIO.input(26)==1):
        end = time.time()

    dur = end-start
    distance = dur*34000/2
    print(distance)
    return(distance)
while True:
    # grab the frame from the threaded video file stream, resize
    # it, and convert it to grayscale
    # channels)
    distance = ultra()
    rcv = port.readline()
    print(rcv)
    __,frame = cam.read()
    if(rcv == "forward"):
        print("forward")
        robot.Up()
    elif(rcv == "backward"):
        print("Back")
        robot.Down()
    elif(rcv=="left"):
        print("left")
        robot.Left()
    elif(rcv == "right"):

```

```

        print("right")
        robot.Right()
    elif(rcv == "stop" ):
        print("stop")

        robot.Stop()
    elif(distance<20):
        print("obstacle detected")
        print("stop")
        robot.Stop()

elif(rcv == "SMS"):
    subprocess.Popen("sudo python sms.py",shell = True)
    #imgPath=urllib.urlopen(url)
    #imgNp=np.array(bytearray(imgPath.read()),dtype=np.uint8)
    #frame=cv2.imdecode(imgNp,-1)

    frame = cv2.resize(frame,(450, 450), interpolation = cv2.INTER_CUBIC)
## if lp==0:
##     dat=input('initialize the keyboard');
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # detect faces in the grayscale frame
    rects = detector(gray, 0)

    # loop over the face detections
    for rect in rects:
        # determine the facial landmarks for the face region, then
        # convert the facial landmark (x, y)-coordinates to a NumPy
        # array
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)

        # extract the left and right eye coordinates, then use the
        # coordinates to compute the eye aspect ratio for both eyes
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        EYE,h,w,x,y,cnter = eye_aspect_ratio(leftEye)
        if lp>0:
            c11=x;
            c22=y;
            x=int(math.ceil(cnter[0]));
            y=int(math.ceil(cnter[1]));
            print(x)

```

```

##### HOME
AUTOMATION #####
    if (x<220):

        print("Right")
        robot.Right()
    ##        GPIO.output(light, True)
    ##        while(x>220):
    ##            print("waiting....")
    ##            flag1=1
    elif (x>230):

        print("Left")
        robot.Left()

    cv2.circle(frame,(x,y),5,255,-1);
    if lp>0:
        diffcentX=FrameCentX-x;
        diffcentY=FrameCentY-y;
        muldiffcentX=diffcentX*6;
        muldiffcentY=diffcentY*6;
    ##        pyautogui.moveTo(CentInitX+muldiffcentX,CentInitY-
muldiffcentY);
        lp=1;
        rightEAR,hrr,wr,xr,yr,cnterr = eye_aspect_ratio(rightEye)
    ##        rightEAR = eye_aspect_ratio(rightEye)

    # average the eye aspect ratio together for both eyes
    ##        ear = (leftEAR + rightEAR) / 2.0

    # compute the convex hull for the left and right eye, then
    # visualize each of the eyes
    leftEyeHull = cv2.convexHull(leftEye)
    rightEyeHull = cv2.convexHull(rightEye)
    cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
    cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)

    # check to see if the eye aspect ratio is below the blink
    # threshold, and if so, increment the blink frame counter
    if EYE < EYE_AR_THRESH:
    ##        pyautogui.doubleClick()
    ##        pyautogui.click(CentInitX+muldiffcentX,CentInitY-
muldiffcentY, button='left')
        COUNTER += 1

    # if the eyes were closed for a sufficient number of

```



```

# then sound the alarm
if COUNTER >= EYE_AR_CONSEC_FRAMES:
    # if the alarm is not on, turn it on
    if not ALARM_ON:
        ALARM_ON = True

    # check to see if an alarm file was supplied,
    # and if so, start a thread to have the alarm
    # sound played in the background
    if args["alarm"] != "":
        t = Thread(target=sound_alarm,
                    args=(args["alarm"],))
        t.daemon = True
        t.start()

    # draw an alarm on the frame
    cv2.putText(frame, "BUTTON PRESSED!", (10, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    if(flag3==0):
        print("Forward")
        robot.Up()
        GPIO.output(fan, True)
        flag3=1
    elif(flag3==1):
        print("stop")
        robot.Stop()
        GPIO.output(fan, False)
        flag3=0

# otherwise, the eye aspect ratio is not below the blink
# threshold, so reset the counter and alarm
else:
    COUNTER = 0
    ALARM_ON = False
    if rightEAR < EYE_AR_THRESH:
        pyautogui.click(CentInitX+muldiffcentX,CentInitY-
muldiffcentY, button='left')
        COUNTER += 1
        pyautogui.click(button='right')
        # if the eyes were closed for a sufficient number of
        # then sound the alarm
        if COUNTER >= EYE_AR_CONSEC_FRAMES:
            # if the alarm is not on, turn it on
            if not ALARM_ON:
                ALARM_ON = True

```

```

## # check to see if an alarm file was supplied,
##           # and if so, start a thread to have the alarm
##           # sound played in the background
####           if args["alarm"] != "":
####               t = Thread(target=sound_alarm,
####                   args=(args["alarm"],))
####               t.daemon = True
####               t.start()
##
##           # draw an alarm on the frame
##           cv2.putText(frame, "RIGHT CLICK PRESSED!", (10, 30),
##               cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# otherwise, the eye aspect ratio is not below the blink
# threshold, so reset the counter and alarm
##     else:
##         COUNTER = 0
##         ALARM_ON = False
# draw the computed eye aspect ratio on the frame to help
# with debugging and setting the correct eye aspect ratio
# thresholds and frame counters
cv2.putText(frame, "EYE: {:.2f}".format(EYE), (300, 30),
    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# show the frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF

# if the `q` key was pressed, break from the loop
if key == ord("q"):
    break

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()

```

Code for robot

```

import RPi.GPIO as GPIO
import time
##import serial
GPIO.setmode(GPIO.BCM)

```

```

GPIO.setwarnings(False)
print('Wait untill device gets connected....')
leds = [21,20,16,12]
##port=serial.Serial("/dev/serial0",9600,timeout=0.5)
for i in leds:GPIO.setup(i,GPIO.OUT)
def Stop():
    for i in leds:GPIO.output(i,False)
def Left():
    print "Turning left"
    GPIO.output(leds[0],True)
    GPIO.output(leds[1],False)
    GPIO.output(leds[2],False)
    GPIO.output(leds[3],True)
    time.sleep(.2)
    GPIO.output(leds[3],False)
    GPIO.output(leds[0],False)

def Right():
    print "Turning Right"
    GPIO.output(leds[0],False)
    GPIO.output(leds[1],True)
    GPIO.output(leds[2],True)
    GPIO.output(leds[3],False)
    time.sleep(.2)
    GPIO.output(leds[1],False)
    GPIO.output(leds[2],False)

def Up():
    print "Moving Forward"
    GPIO.output(leds[0],False)
    GPIO.output(leds[1],True)
    GPIO.output(leds[2],False)
    GPIO.output(leds[3],True)

def Down():
    print "Moving Backward"
    GPIO.output(leds[0],True)
    GPIO.output(leds[1],False)
    GPIO.output(leds[2],True)
    GPIO.output(leds[3],False)
    time.sleep(1)
    GPIO.output(leds[2],False)
    GPIO.output(leds[0],False)

```

```
## Code for SMS module
```

```
from twilio.rest import Client
```

```
# Your Account SID from twilio.com/console
```

```
account_sid = "ACea0205ede8e46d9b203c9e8db549d603"
```

```
# Your Auth Token from twilio.com/console
```

```
auth_token = "00818030dbe932ac4d73e77e8c4ad5d0"
```

```
client = Client(account_sid, auth_token)
```

```
message = client.messages.create(
```

```
    from_="+15018193640",
```

```
    to="+919237372948",
```

```
    body="emergency")
```

```
print(message.sid)
```

SNAPSHOTS OF OUTPUT:

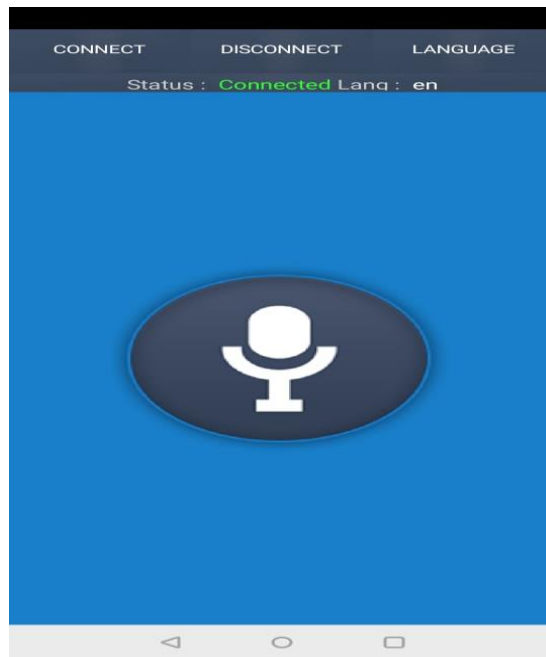


Fig 6.1: User Interface of the application to control Wheelchair.

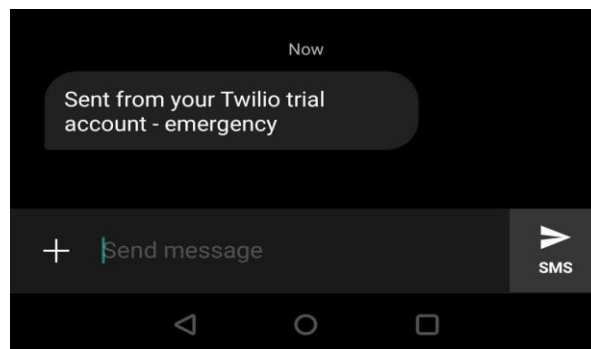
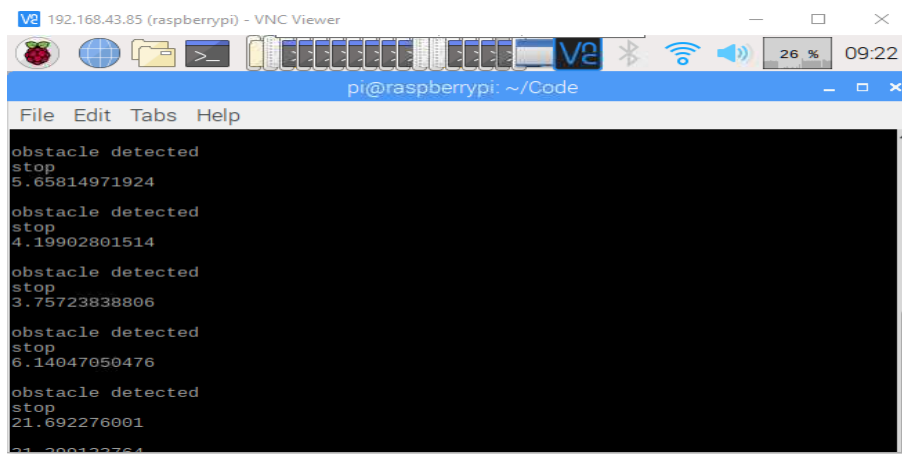


Fig 6.2 : Output of Alert message module

In case of any emergency the user can alert the guardian by using voice module messaging platform.



```
pi@raspberrypi: ~/Code
File Edit Tabs Help

obstacle detected
stop
5.65814971924

obstacle detected
stop
4.19902801514

obstacle detected
stop
3.75723838806

obstacle detected
stop
6.14047050476

obstacle detected
stop
21.692276001

obstacle detected
stop
31.300123764
```

Fig 6.3 : Output of obstacle detection

During the movement of wheel chair, when ultra sonic sensor detects any obstacle found in the path, the wheel chair stops.

REFERENCES

- [1] Sarangi, P., Grassi, V., Kumar, V., Okamoto, "Integrating Human Input with autonomous behaviors on an Intelligent Wheelchair Platform", Journal of IEEE Intelligent System, 22, 2, 33-41, [2018].
- [2]Masato Nishimori, Takeshi Saitohand, Ryosuke Konishi, "Voice Controlled Intelligent Wheelchair," SICE Annual Conference 2007, Takamatsu, 2019.
- [3] T. Lu and et-al "A motion control method of intelligent wheelchair based on hand gesture recognition," 2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, VIC, 2020.
- [4]C. Vijayakumar, M. P. Kumar and et-al, "Sensors based automated wheelchair," 2013 International Conference on Green Computing, Communication and Conservation of Energy (ICGCE), Chennai, 2017.
- [5] S. Ravi Kumar, S. Shenbagavadivu, and et-al "Emotion Recognition based smart wheelchair for disabled person", International Research journal of Engineering and technology Volume No: 06, Issue No: 02, February 2019.
- [6] Udhaya Kumar.S, Vibin Mammen Vinod,"EOG based wheelchair control for quadriplegics" International Conference on Innovation, Embedded and Communication system (ICIIECS), [2015].
- [7] Zhang, Peng & Ito et-al. "Implementation of EOG mouse using Learning Vector Quantization and EOG-feature based methods". IEEE Conference on Systems, Process and Control,2019.
- [8] S. Jaffar Ali et al."Autonomous camera based Eye controlled wheelchair using Raspberry-Pi" (IJITR) International Journal of Innovative Technology and Research Volume No.5, Issue No.2, February – March 2017.
- [9] Kamaraj, et-al "Implementation of Wheelchair Controller using Eyeball Movement for Paralytic" International Journal of Engineering Sciences & Research Technology (IJESRT) (Vol.2, No. 4), April 2014.
- [10] S. N. Patel and V. Prakash, "Autonomous camera based eye controlled wheelchair system using raspberry-pi", Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, 2015
- [11] Preethika Britto, Indumathi, Sudesh srinivasu, Lazar Mathew "Automation of wheel chair using ultrasonic and body kinematics" NCCI 2010 -National Conference

On Computation Instrumentation CSIO chandigarh, INDIA, 19-20 March 2010.

[12] Maja Pantic and Leon J.M. Rothkrantz "Facial action recognition for facial expression analysis from static face images" IEEE Transactions On Systems, Men And Cybernetics-Part B: Cybernetics, VOL. 34, NO. 3, JUNE 2016.

[13] R. Barea, L. Boquete, M. Mazo and E. Lopez, "Wheelchair guidance strategies using eog", Journal of Intelligent and Robotic Systems, , 2013.

[14] G. Pires and U. Nunes, "A wheelchair steered through voice commands and assisted by a reactive fuzzy-logic controller", Journal of Intelligent and Robotic Systems, 2014.

[15] Mr Lokesh Mehta, Mr.Pawan Sharma “Spy night-sight automaton with Moving Wireless Video Camera”. International journal of analysis in engineering technology and management (IJRETM),2017.