VISION BASED WHEEL CHAIR CONTROL FOR QUADRIPLEGICS

A PROJECT REPORT

Submitted by

S VIDYA CHANDRAN (1306046)
SUBHAM PRASAD (1306073)
SIDDHARTHA BHASKAR ARJA (1306078)

Under the guidance of
Ms. PUNITHAVATHY MOHAN, M.E

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN
ELECTRONICS AND COMMUNICATION ENGINEERING

HINDUSTAN UNIVERSITY, CHENNAI - 603103

MAY 2017

VISION BASED WHEEL CHAIR CONTROL FOR QUADRIPLEGICS

A PROJECT REPORT

Submitted by

S VIDYA CHANDRAN (1306046)
SUBHAM PRASAD (1306073)
SIDDHARTHA BHASKAR ARJA (1306078)

Under the guidance of
Ms. PUNITHAVATHY MOHAN, M.E

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN
ELECTRONICS AND COMMUNICATION ENGINEERING

HINDUSTAN UNIVERSITY, CHENNAI - 603103

MAY 2017

BONAFIDE CERTIFICATE

Certified that this project report **"VISION BASED WHEEL CHAIR CONTROL FOR QUADRIPLEGICS"** is the bonafide work of **S VIDYA CHANDRAN (1306046), SUBHAM PRASAD (1306073)** and **SIDDHARTHA BHASKAR ARJA (1306078)** who carried out the project work under my supervision during the academic year **2016-2017.**

SIGNATURE
Dr. ABY K THOMAS, Ph.D
HEAD OF THE DEPARTMENT
Professor
Electronics & Communication Engineering
HINDUSTAN UNIVERSITY
Padur, Chennai – 603103

SIGNATURE
Ms. PUNITHAVATHY MOHAN, M.E
SUPERVISOR
Assistant Professor
Electronics & Communication Engineering
HINDUSTAN UNIVERSITY
Padur, Chennai – 603103

INTERNAL EXAMINER

Name: _____

Designation:_____

EXTERNAL EXAMINER

Name: _____

Designation:_____

Institution Name:_____

Project viva - voce conducted on _____

ACKNOWLEDGEMENT

It is our extreme pleasure to thank our Chancellor **Dr. (Mrs.) Elizabeth Verghese,** Vice-Chancellor **Dr. S. Ramachandran,** Hindustan University for providing a conducive environment which helped us to pursue our project in a diligent and an efficient manner.

We wish to express our sincere gratitude to **Dr. N. Vasudevan,** Dean (Academics) for his encouragement and support to complete this project.

We are thankful to **Dr. Aby K Thomas,** Head of the Department, Electronics and Communication Engineering for having evinced keen interest in our project and for his continued support.

We also thank **Mr. M. Abraham,** our project coordinator for his support throughout the completion of our project. We are indebted to our project supervisor Ms. Punithavathy Mohan, Assistant Professor, Department of Electronics and Communication Engineering for her simulating suggestions and encouragement which helped us to carry-out our project.

Last but not the least, we would also like to thank our family and friends who were keen in helping us out to complete the project and to give their blessings.

DEDICATION

To everyone who has aided us in any way to become what we are today.

Their sacrifices seeded our success, especially our parents.

ABSTRACT

Quadriplegia is paralysis caused by illness or injury that results in the partial or total loss of use of all four limbs and torso. People who were affected with this problem have lost the ability to control upper and lower limbs due to quadriplegia or ageing side effects. They require special control systems to use an electrical wheelchair instead of using traditional control by joystick. In this project a novel auto calibrated head orientation controller for wheelchair application is proposed. A device has been developed for the patients in which it uses a webcam attached PC and a hardware kit which consists of a TX/RX, MAX232, Arduino, motors and its driver circuit, power supply and an LCD display. The idea is to create a Vision Monitored System which allows movement of the patient's wheelchair depending on the eye movements. This device helps patients sitting on the Wheel Chair assembly to move in a direction based on their eye movement. Image capturing and image processing detects the movement of the eyes which passes instructions to the micro-controller board (Arduino) via a TX/RX. The Arduino board processes the instructions and operates the DC motors through a driver circuit. Viola Jones algorithm is used to detect the eyes and theirs motion. Eye movement is processed to determine the direction of movement of the wheelchair. Threshold values are used to avoid unwanted jerks of the wheelchair assembly. The system is cost effective and thus can be used by patients over a large economy range.

TABLE OF CONTENT

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	**v**
	LIST OF FIGURES	**ix**
	LIST OF TABLES	**xi**
	LIST OF ABBREVIATIONS	**x**
1	**INTRODUCTION**	**1**
	1.1 Background	1
	1.2 Problem Statement	1
	1.3 Range of Solutions	1
	1.4 Summary	2
2	**LITERATURE SURVEY**	**3**
3	**PROJECT DESCRIPTION**	**5**
	3.1 Design Overview	5
	3.1.1 Eye-Detection and Motion Tracking	5
	3.1.2 ATmega328P Controlled Wheel Chair Assembly	5
	3.1.2.1 Initialization	6
	3.1.2.2 Image and Video Processing	6
	3.1.2.3 Estimation	7
	3.1.2.4 Detection	7
	3.1.2.5 Error Handling	7
	3.1.2.6 Motion	7
	3.1.2.7 Serial Communication	8
	3.2 Software Design	8
	3.2.1 Initialization of Variables and Setting up Serial Communication	8
	3.2.2 Image Capture and Eye Detection	8
	3.2.3 Image Processing	9
	3.2.4 Movement Detection	9

	3.2.5 Motor Signals	10
	3.3 Firmware Design	10
	3.4 Hardware Design	10
	3.4.1 Motor Control	11
	3.4.2 Mechanical Design	11
	3.4.3 Driver Circuit	12
	3.4.3.1 Working	13
4	**METHODOLOGY ADOPTED**	**14**
	4.1 Arduino UNO	14
	4.1.1 Programming	15
	4.1.2 Warnings	16
	4.1.3 Difference with Boards	16
	4.1.4 Power	16
	4.1.5 Memory	17
	4.1.6 Input and Output	17
	4.1.7 Communication	18
	4.2 LCD	19
	4.3 MAX 232	21
	4.4 MATLAB	23
	4.5 Image Processing using MATLAB	24
	4.5.1 Image Enhancement	24
	4.5.2 Image Analysis	25
	4.5.3 Image Transforms	25
	4.5.4 Image Segmentation	26
	4.5.5 Cascade Object Detector	26
5	**TESTING AND ANALYSIS**	**27**
	5.1 Testing Strategy	27
	5.2 Speed of Execution	27
	5.3 Accuracy	27
	5.3 Safety Features	27
6	**CONCLUSION**	**30**
7	**FURTHER ENHANCEMENTS**	**31**

REFERENCES **32**

APPENDIX **33**

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	Block Diagram	6
3.2	Circuit Diagram	11
3.3	Mechanical Design	12
3.4	Pin Diagram of L293D	13
4.1	Arduino UNO Board	14
4.2	Pin Diagram of ATmega328	17
4.3	Pin Diagram of LCD	19
4.4	MAX232 Setup	21
4.5	Pin Diagram of MAX232	22
5.1	Left Movement Detection	28
5.2	Right Movement Detection	28
5.3	Straight Movement Detection	29
5.4	No Movement Detection	29

LIST OF ABBREVIATIONS

ATmega	-	Atmel Mega
DC	-	Direct Current
IC	-	Integrated Circuit
ICSP	-	In-Circuit Serial Programming
LCD	-	Liquid Crystal Display
LED	-	Light Emitting Diode
MATLAB	-	Matrix Laboratory
MISO	-	Master Input Slave Output
MOSI	-	Master Output Slave Input
PWM	-	Pulse Width Modulation
RX	-	Receiver
SPI	-	Serial Peripheral Interface
TTL	-	Transistor-Transistor Logic
TX	-	Transmitter

LIST OF TABLES

CHAPTER NO.	TITLE	PAGE NO.
4	4.1 Technical Specification	15
4	4.2 Pin Description of LCD	20
4	4.3 Pin Description of Max232	22

CHAPTER 1
INTRODUCTION

1.1 Background

The ability to move freely is highly valued by all people. However, it is sometimes difficult for a person with a physical disability. Nowadays, an electric wheelchair is commercially available for disabled people. It generally requires considerable skill to operate. Moreover, some disabled people cannot drive an electric wheelchair manually, even with a joystick, because they lack the physical ability to control the movement. To enable a disabled person to drive a wheelchair safely and easily so that they can enjoy a higher quality of life, researchers have proposed several electric wheelchair systems. The use of voice commands to control an electric wheelchair is one research result. Scientist Stephen W. Hawking is perhaps the most well-known victim of major paralysis. Hawking was diagnosed with incurable Amyotrophic Lateral Sclerosis (ALS) in 1962, thereafter using a wheelchair to move. Many of those suffering close to or complete paralysis usually however still can control their eye movement which inspired us to develop a vision based electric wheelchair. The idea of a vision based wheelchair was inspired by so many past projects which work on the finger movements of the handicapped person to guide the wheel chair in the desired direction.

1.2 Problem Statement

To develop a vision based wheel chair control system for physically challenged persons to move in any desirable direction.

1.3 Range of Solutions

The motive of this project is to develop an inexpensive system and thus can be afforded by all. The main task in the design was to accurately detect the eye movements. Since, the system is for human use, an extra care is taken about the safety of the system.

For the eye detection, so many strategies are implemented in past projects. One was to detect the eye movements using the infrared light. The idea behind this method was to concentrate the infrared light onto the eyes from the forehead. The voltage rating of the infrared light is under the permissible levels that can be used to concentrate in the eyes. The infrared light concentrated through the forehead of the user will be collimated through the pupil of the eye and

this collimated light was used to obtain a voltage drop across the photodiode which will drive the motors.

The photo detector circuit has a backdrop that it is not able to obtain accurate results and the voltage drop across the photo diode was not constant and kept on varying. Thus, the issue with this idea was that the voltage drop obtained across the diode was just 0.7 volts and it was not sufficient to detect accurately.

Another strategy was to connect the web camera to the Raspberry Pi directly which would then process the snapshots, eliminating the need to have the camera attached to the laptop. But the time taken to understand the new controller board is a lot. Also, normally these days such users have a laptop somehow attached to their wheel chair and thus it is simple to go ahead with a microcontroller board which is easy to understand.

The attempted design is to capture the images using a webcam that will be attached to the laptop placed on the wheelchair of the user. These captured images will be used to detect the eyes and hence detect the movements using a MATLAB script running on the laptop, which then sends serial commands to the microcontroller circuitry driving the motors attached to the wheel chair.

1.4 Summary

It is decided to use the web camera to detect the eye movements which will be further processed to drive the motors. For the simplicity and to make a prototype, an attempt is made to design a small motorized, wooden platform and the laptop webcam. Serial port communication is used between the web camera and the microcontroller. The microcontroller will be placed on the wheel chair which will be connected to the motors, driving the wheel chair in the direction the person sitting on the chair desires to move in.

CHAPTER 2
LITERATURE SURVEY

2.1 S. Tameemsultana and N. Kali Saranya, "Implementation of Head and Finger Movement Based Automatic Wheel Chair", Bonfring International Journal of Power Systems and Integrated Circuits, vol. 1, Special Issue, pp 48-51, December 2011.

An automated system was developed to control the motor rotation of wheel chair based on head and finger movement of physically challenged person. In order to facilitate these people for their independent movement, an accelerometer device was fitted on person's head and a flex sensor was fixed in a glove which was worn by the person. Based on the head and finger movements the accelerometer and the flex sensor drives the motor fitted to the wheel chair. The wheel chair was driven in any of the four directions. The automated wheelchair was based on simple electronic control system and the mechanical arrangement that was controlled by a Programmable Interface Controller. This automatic wheel chair also helped people who had various other disabilities to sit on the chair and just hold the accelerometer and move it over to control the vehicle movements. In this paper the wheel chair was mimicked using a robot.

2.2 S. Vijayprasath, DR. R Sukanesh and S. Palanival Rajan "Innovative Techniques using Wireless Sensors in Designing Smart System for Disabled Patients" International Journal of Power Control Signals and Computation (IJPCSC) VOL. 1 NO. 4.

Physically challenged persons find their movements very tough with the existing assistive devices (Joysticks) in cases of higher disability. Though there were many methods available in recent times to enable their motility they require fine and precise control which wasn't possible most of the time. There had been various control systems developing specialized for people with various disorders and disabilities. This paper reported the preliminary work in developing a robotic wheelchair system that involved the movement of eyeball and head kinematics in directing the wheel chair. The system enabled the patient to have command over the chair, its

direction of movement and also sense and alarm the user about the obstacles in the path to avoid collision. This wheelchair helped the patients to move in environments with ramps and doorways of little space. Generally an automated wheelchair must be highly interactive to enable the system to work most efficiently.

2.3 Ankur Thakkar, Darshan Shah, "Eye monitored Wheel Chair Control"

The idea was to create an Eye Monitored System which allowed movement of the patient's wheelchair depending on the eye movements. It was known that a person suffering from quadriplegia can partially move his eyes and tilt his head, thus presenting an opportunity for detecting those movements. A device was developed where a patient sitting on the Wheel Chair with attached camera, was able to move in a direction just by looking in that way. The camera signals were monitored by a MATLAB script, which then guided the motors wired to the ATmega1284P Microcontroller over the Serial Interface to move in a particular direction.

2.4 Viola Jones Algorithm - https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

The Viola–Jones object detection framework was the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones. Although it could be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection.

CHAPTER 3

PROJECT DESCRIPTION

3.1 DESIGN OVERVIEW

There are two major components from the system design standpoint –

1) Eye-Detection and motion tracking.
2) Arduino UNO controlled Wheel Chair Assembly.

3.1.1 Eye-Detection and Motion Tracking

An inbuilt webcam of a laptop or a PC is used in this project which continuously stares at the user's eyes. The webcam, wired to the patient's laptop runs a MATLAB application designed to monitor and react to eye movements. Based on a series of snapshots taken and thereafter processed, the motion of the user's eyes are detected, decision to move the Wheel Chair in a particular direction is taken and communicated serially to ATmega328P microcontroller. MATLAB 2015b has an image processing toolbox which has been utilized for the eye detection and a webcam support plug-in installer. A command 'CascadeObjectDetector', capable of detecting eye-shaped objects based on their shape and size. It uses the Viola Jones Algorithm for the same.

Continuous snapshots are taken and feature points extracted are saved. Based on the position of the feature points in previous snapshot and current snapshot, a movement is detected and this is communicated to the wheelchair assembly via the serial port (MAX232).

3.1.2 ATmega328P controlled Wheel Chair Assembly

A decision based on the processing done by the MATLAB application is communicated and received by the ATmega328P. The controller on reception forces the port pin high on which the motors have been connected for desired motion of the Wheel Chair.

The circuit assembly block diagram is shown in figure 3.1.

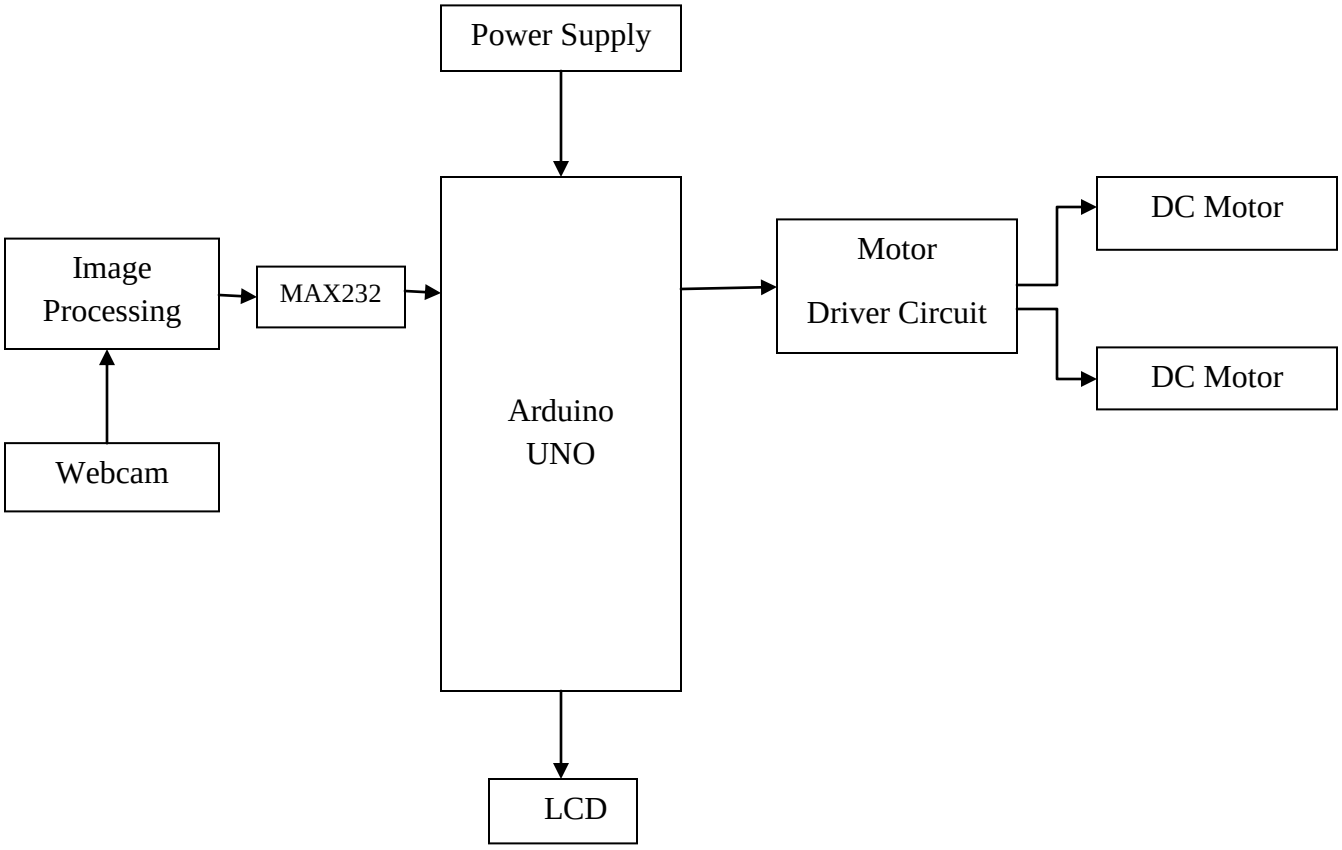

Figure 3.1 Block Diagram

There are two parts in the code structure. The first part is to detect the eye movements and the other part is to drive the motors.

3.1.2.1 Initialization

Primarily the serial communication is initialized which will be used later for the interface between MATLAB and the controller, the video capture and the program variables.

3.1.2.2 Image and Video Processing

Continuous video frames are taken and the input is sampled to save it as the screen shots. Each frame is then converted into the black and white frames. For the accurate results, contrast

stretching on each frame is needed to make the dark region darker and bright region brighter. This enables the detection of the eyes better.

3.1.2.3 Estimation

After working on the each snapshot, detection of the eyes is aimed. This is done by estimating the position of left as well as the right eye. Threshold value limits the vague detection of the eyes which can be used for the further processing.

3.1.2.4. Detection

The idea is to compare the current position of the eye with the previous position. Thus, the difference in the coordinates helps to predict the motion in the particular eye. But sometimes, it may be possible that only one of the either eye gets detected. In that case, the preference is given to the eye that is detected currently.

3.1.2.5 Error Handling

If the detection results give a height and width value lesser or greater than the threshold, the value is voided and not considered for the decision making.

3.1.2.6 Motion

After detecting the eye movements, a decision algorithm is incorporated that helps the controller to drive the motors accordingly: Valid Left: The decision to turn left will be considered as valid if the eye turns left and stays there for a cycle. This action will be detected as a left turn request. A cycle is regarded as couple of seconds. If the patient then turns right for forward motion, it should be considered as void. Valid Right: Similarly, the decision to turn right will be considered as valid if the eye turns right and stays there for a cycle. This action will be detected as a right turn request. If the patient then turns left for forward motion, it should be considered as void. Valid Straight: The signal to go straight is when a person looks left and right or right and then left. This will be detected as to go straight.

3.1.2.7 Serial Communication

According to the detected command, the MATLAB application transmits 3, 4 or 1 for left, right and straight respectively to the controller to drive the motors.

3.2 Software Design

Image Processing techniques like capturing the eye regions and detection of eye movements are done using MATLAB. The firmware component deals with the reception of serial signal to drive the motor, connected to the port pins, forcing the wheelchair in the direction it is supposed to move in. The MATLAB design can be structured into many small sub-parts each of which are discussed in 3.2.1.

3.2.1 Initialization of Variables and Setting up Serial Communication

MATLAB R2015b can easily be configured to serially transmitting data on the port mentioned in the code. Initially, already set up serial ports are disabled. The current port is mentioned, by checking the 'Device Manager' which indicates the port in use. The baud rate of communication is set to 9600. The communication is set to have no flow control and parity check is disabled.

After setting up the serial communication to enable the data link between MATLAB and ATmega328P, the variables needed in the course of the program are redefined to their initial values.

3.2.2 Image Capture and Eye Detection

MATLAB R2015b equips an Image Processing Toolbox, used majorly in this section of the Software Design. An 'Integrated Web Camera' which is connected via a USB cable to the laptop on which the MATLAB script is running. Streaming continuous video signals on MATLAB coming from the camera using the Video Processing Toolbox is available. Identifying the correct device and then using it to stream the video signal is the next step.

The requirement of the design is to continuously look at different frames, based on which the motion is determined. Command 'snapshot(cam)' is used to capture the snapshots. The image is then converted to grayscale image as no color information is needed to detect eye feature points. The conversion in fact makes the detection easier. The 'imadjust' command is used then

to contrast stretch the image to make darker sections even darker, enhancing the eye feature points useful for the application.

This pre-processing of the image makes the image easier to process and extract the eyes from the acquired snapshots. After this initial pre-processing, the Eye Detection is done using the Viola-Jones Object Detection Algorithm. Primarily, this algorithm was designated for face detection though it is used for all sorts of object detections. The algorithm is designed to work on sum of pixels in a rectangular area. Viola-Jones algorithm says that face can be detected by looking for rectangle. And then a large rectangle is made up of many such smaller rectangles, which are fundamentally feature points on a human face.

The 'CascadeObjectDetector' on MATLAB, utilizes this algorithm to extract and detect the eyes of the person. Eyes are detected by plotting the rectangle at the appropriate position of the eyes.

3.2.3 Image Processing

Initially, monitoring of the eyes is done to detect the eye feature points. If not detected, set a flag and display it on the debug screen. To increase the detection accuracy, neglect all other points on the screen except the actual eye of the person. The reason being, if anyone except quadriplegic person comes in front of the camera, the person should not affect the system. Also certain things seemingly looking like eyes should be rejected as well. To avoid false detection of the eyes, a threshold value is used to compare to the respective positions of the eyes and reject everything which is outside of the threshold value.

A flag is set each time no valid eyes are detected. It is assumed that the position of the camera is fixed relative to which the left and the right eye approximate positions can be estimated. Using this, it distinguishes and store the left and the right eye in different matrices. This helps getting a clear discrimination between both the eyes, helping in easy movement detection.

3.2.4 Movement Detection

The movement detection is done with a very basic principle. The feature points for both left and right eyes are recorded. The difference in pixels of the left eye position and right eye position in the current snapshot is compared to the previous snapshot. It defines the threshold for

the minimum movement of the eye required to be qualified as a valid attempt. In each snapshot the difference is evaluated, and if this difference above the threshold in any direction left or right, the flags indicating left movement or right movement are set. If the difference is not above the threshold, the flag which says that no movement has occurred is set.

Sometimes due to non-linearities, both the eyes are not detected. At such instances while evaluating the difference for detecting movement, we would give a bias to the eye which was detected in the previous snapshot. After detecting the eye movements, serial signals are sent to the micro-controller.

3.2.5 Motor signals

To qualify a valid right, left and straight attempts to move, there is a need to incorporate many factors. A valid movement is recognized by the motion of eye's pupil towards right/left/straight. The eye needs to stay at the same position until the snapshots are captured for the image processing for determining the movement of the wheelchair.

Flags are set for left and right movements each time the wheel chair moves, avoiding precisely this unwanted behavior of the system. The decision is transmitted to the micro-controller via the serial port thereby determining the direction by which the wheel chair moves accordingly.

3.3 Firmware Design

The firmware design is fairly straight forward as all the computation has been done on MATLAB and the only thing which the micro-controller has to do is to control the motors to move in a particular direction. The firmware constantly monitors the serial input. The firmware turns ON the port pin based on this received signal.

3.4 Hardware Design

The hardware part serially transfers the signals to the microcontroller board which controls the DC motors through a driver circuit.

3.4.1 Motor Control

The signal obtained from the MATLAB script is used to drive the motors. The MATLAB program does all the decision making for the motor to run in a particular direction. So, the script sends the bit 3, 4 and 1 for left, right and straight direction. The circuit diagram is shown in Figure 3.2.

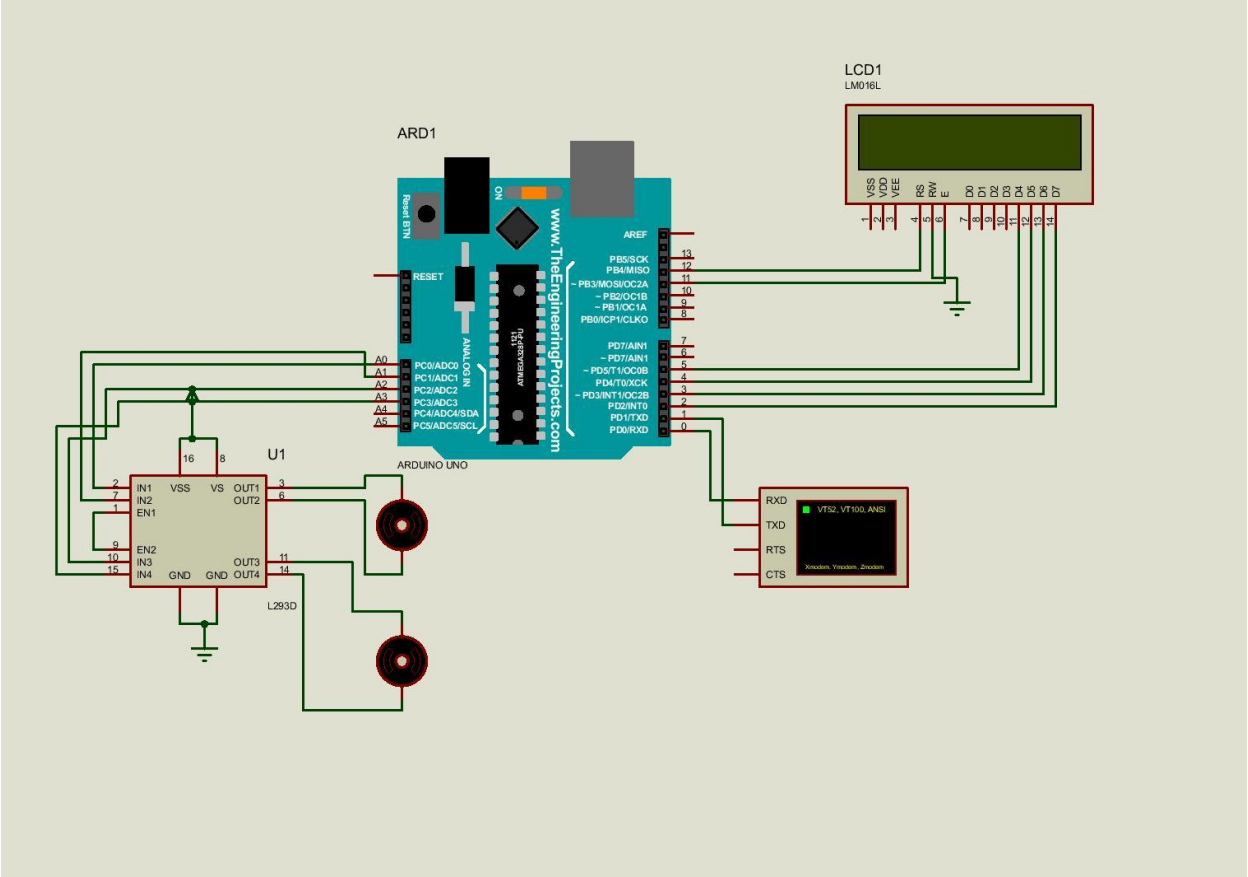

Figure 3.2 Circuit Diagram

3.4.2 Mechanical Design

The wheel chair prototype is made on a sunmica laminated circuit board with three wheels where only two of them are connected to the motors and the other is used for stability of the board. Mechanical Design is shown in the Figure 3.3. In order to detect the eye movements, the web camera should be placed right in front of the user. The connections are soldered to avoid displacement of the wiring.

3.4.3 Driver Circuit

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that one can control two DC motors with a single L293D IC. Dual H-bridge Motor Driver integrated circuit (IC).The L293D can drive small and quite big motors as well.

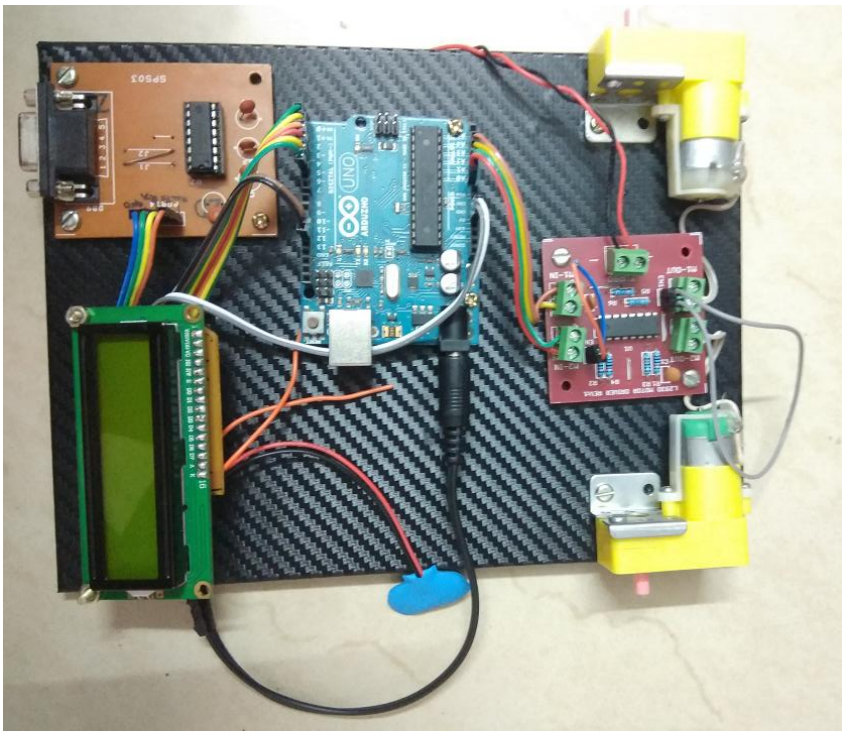

Figure 3.3 Mechanical Design

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. Voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction. Hence, H-bridge IC is ideal for driving a DC motor.

In a single L293D chip there are two H-Bridge circuit inside the IC which can rotate two dc motor independently. Due to its size it is very much used in robotic application for controlling DC motors. Pin diagram of a L293D motor controller is given in the Figure 3.4. There are two Enable pins on L293D pin 1 and pin 9. To drive the motors, the pin 1 and 9 needs to be high. For driving the motors with left H-bridge, enable pin 1 to high, for right H-Bridge, make the pin 9

high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working.

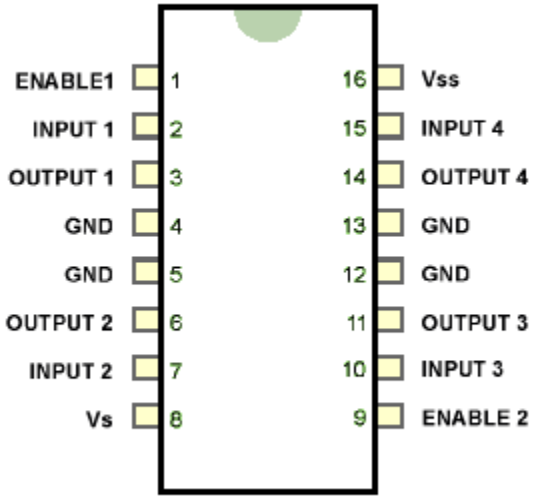

Figure 3.4 Pin Diagram of L293D

3.4.3.1 Working

There are 4 input pins for L293D, pin 2, 7 on the left and pin 15, 10 on the right as shown in the Fig. 3.4. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side. The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1. In simple, one needs to provide Logic 0 or 1 across the input pins for rotating the motor.

Considering a motor connected on left side output pins (pin 3, 6), for rotating the motor in clockwise direction the input pins has to be provided with Logic 1 and Logic 0.

- **Pin 2 = Logic 1** and **Pin 7 = Logic 0** | Clockwise Direction
- **Pin 2 = Logic 0** and **Pin 7 = Logic 1** | Anticlockwise Direction
- **Pin 2 = Logic 0** and **Pin 7 = Logic 0** | Idle [No rotation] [Hi-Impedance state]
- **Pin 2 = Logic 1** and **Pin 7 = Logic 1** | Idle [No rotation]

In a very similar way the motor can also operate across input pin 15, 10 for motor.

CHAPTER 4

METHODOLOGY ADOPTED

4.1 Arduino UNO:

Arduino/Genuino UNO is a microcontroller board based on theATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power in with a AC-to- DC adapter or battery to get started. The Arduino UNO board is shown in Figure 4.1.

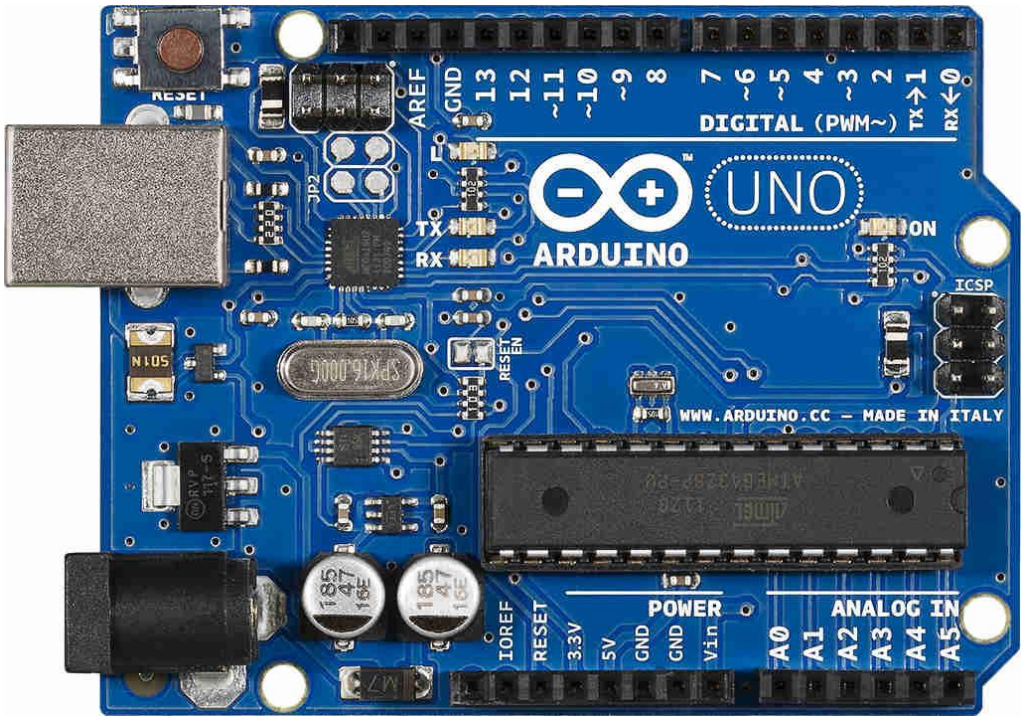

Figure 4.1 Arduino UNO Board

"UNO" means 'One' in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The UNO board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases.

4.1 Technical Specification:

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

4.1.1 Programming

The Arduino/Genuino UNO can be programmed with the (Arduino Software (IDE)). Select "Arduino/Genuino UNO from the Tools > Board menu (according to the microcontroller on your board).

The ATmega328 on the Arduino/Genuino UNO comes preprogrammed with a boot loader that allows uploading new code to it without the use of an external hardware programmer.

It communicates using the original STK500 protocol. One can also bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP.

4.1.2 Warnings

The Arduino/Genuino UNO has a resettable polyfuse that protects your computer's USB ports from shorts and over current. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

4.1.3 Differences with Other Boards

The UNO differs from all preceding boards because it does not use the FTDI USB-to-serial driver chip. Instead, it features the ATmega16U2 (ATmega8U2 up to version R2) programmed as a USB-to-serial converter.

4.1.4 Power

The Arduino/Genuino UNO board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If the input voltage crosses 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

- Vin: The input voltage to the Arduino/Genuino board when it's using an external power source.
- 5V: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board.
- A 3.3 volt supply generated by the on-board regulator. Maximum current drawn is 50 mA.

- GND: Ground pins.
- IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

4.1.5 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

4.1.6 Input and Output

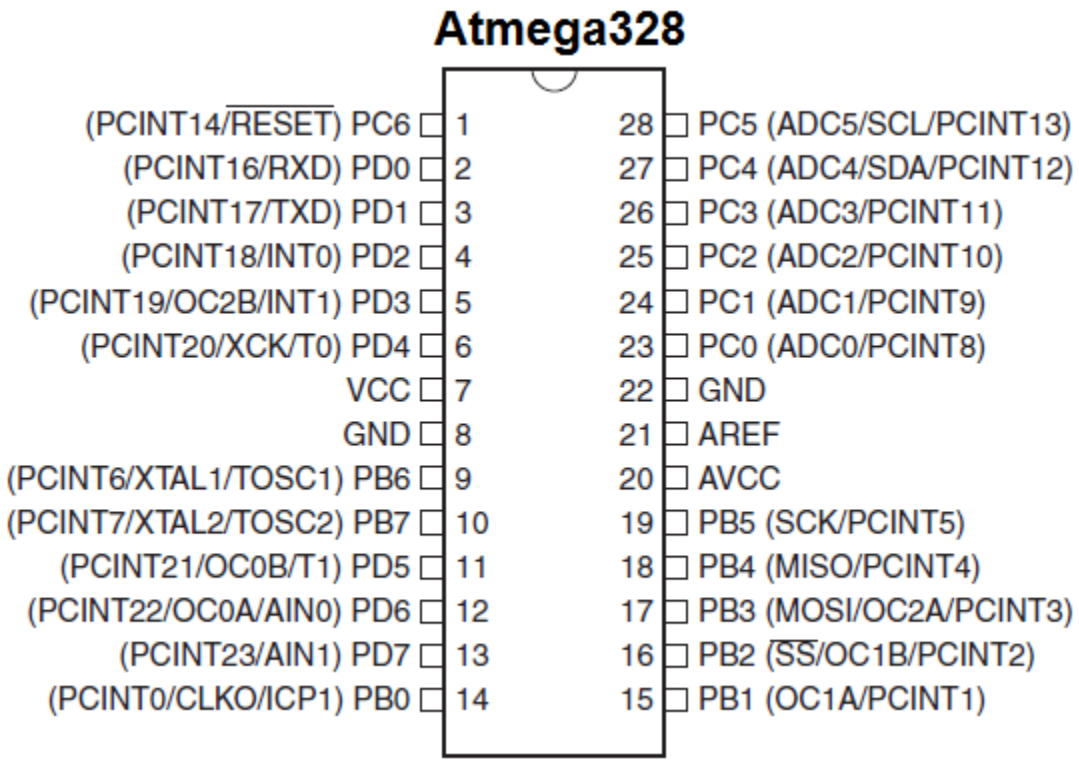

Figure 4.2 Pin Diagram of ATmega328

Each of the 14 digital pins on the UNO can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up

resistor (disconnected by default) of 20-50kΩ. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The pin diagram of ATmega328 is shown in Figure 4.2. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The UNO has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and the analogReference() function.

4.1.7 Communication

Arduino/Genuino UNO has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows serial communication on any of the UNO's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus. For SPI communication, SPI library is used.

4.2 LCD

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.

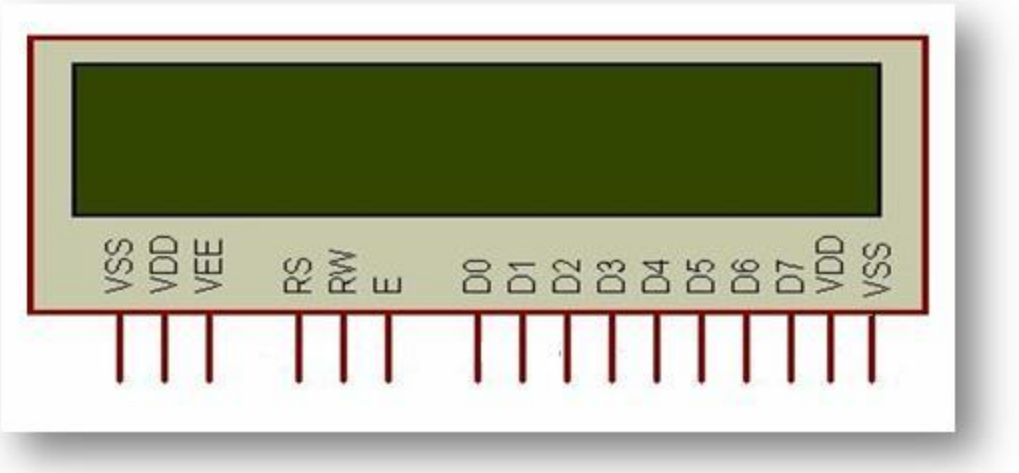

Figure 4.3 Pin Diagram of LCD

A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The pin diagram of LCD is mentioned in Figure 4.3.

4.2 Pin Description of LCD

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on

the LCD. The data is the ASCII value of the character to be displayed on the LCD. The pin description of LCD is shown in table 4.2.

4.3 MAX232:

The **MAX232** is an integrated circuit first created in 1987 by Maxim Integrated Products that converts signals from a TIA-**232** (RS-**232**) serial port to signals suitable for use in TTL-compatible digital logic circuits. The MAX232 is a dual transmitter / dual receiver that typically are used to convert the RX, TX, CTS and RTS signals.

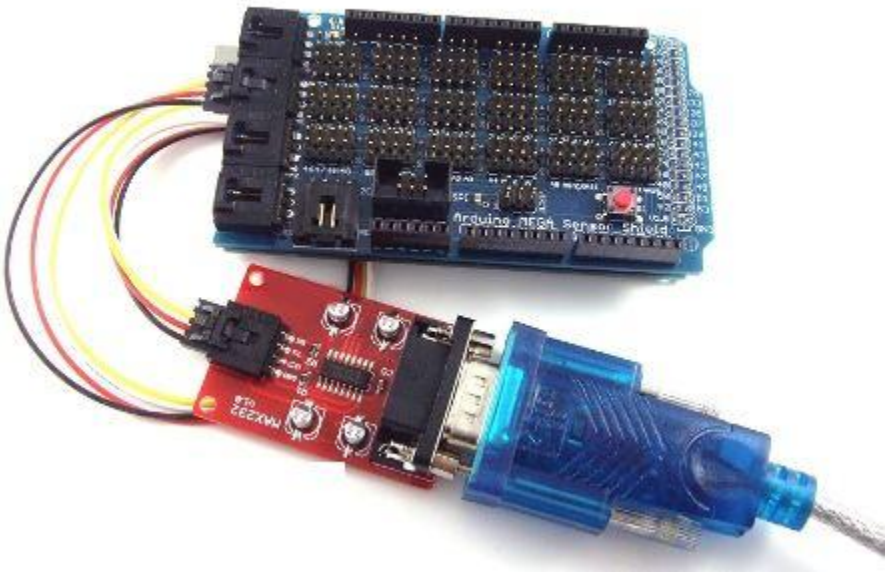

Figure 4.4 MAX232 Setup

It is used only to communicate between PC/Laptop and microcontroller. It converts the TTL voltage levels to RS232 level and vice versa. MAX232 setup and pin diagram of MAX232 are shown in Figure 4.4 and 4.5 respectively.

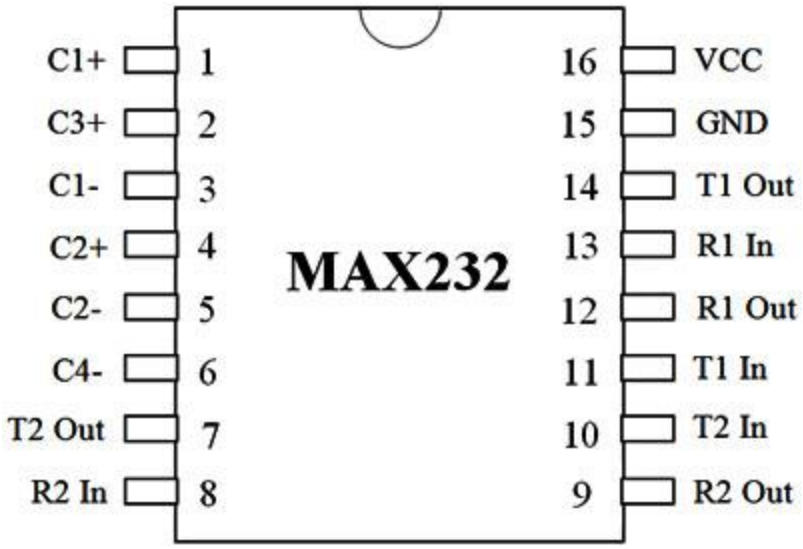

Figure 4.5 Pin Diagram of Max232

4.3 Pin Description of Max232

Pin No	Function	Name
1	Capacitor connection pins	Capacitor 1 +
2		Capacitor 3 +
3		Capacitor 1 -
4		Capacitor 2 +
5		Capacitor 2 -
6		Capacitor 4 -
7	Output pin; outputs the serially transmitted data at RS232 logic level; connected to receiver pin of PC serial port	T_2 Out
8	Input pin; receives serially transmitted data at RS 232 logic level; connected to transmitter pin of PC serial port	R_2 In
9	Output pin; outputs the serially transmitted data at TTL logic level; connected to receiver pin of controller.	R_2 Out
10	Input pins; receive the serial data at TTL logic level; connected to serial transmitter pin of controller.	T_2 In
11		T_1 In
12	Output pin; outputs the serially transmitted data at TTL logic level; connected to receiver pin of controller.	R_1 Out
13	Input pin; receives serially transmitted data at RS 232 logic level; connected to transmitter pin of PC serial port	R_1 In
14	Output pin; outputs the serially transmitted data at RS232 logic level; connected to receiver pin of PC serial port	T_1 Out
15	Ground (0V)	Ground
16	Supply voltage; 5V (4.5V – 5.5V)	Vcc

4.4 MATLAB

There are multiple aspects to the software design of this project. The MATLAB component is responsible for capture of regular snapshots, processing of those snapshots, determining the movement of eyes, algorithm for movement of wheelchair and serial transmission of the decision to move. The firmware component deals with receiving the serial signal, based on which drive the motor connected to the port pins, forcing the wheelchair in the direction it is supposed to move in.

The name MATLAB stands for Matrix Laboratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide. It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering. The major tools within or accessible from the main window of MATLAB are:

• The Command Window

• The Command History

•The Workspace

• The Current folder

• The Help Browser

4.5 Image Processing Tool Using MATLAB

Image Processing Toolbox provides a comprehensive set of reference standard algorithms, functions, and apps for image processing, analysis, visualization, and algorithm development. It offers image analysis, image segmentation, image enhancement, noise reduction, geometric transformations, and image registration. Many toolbox functions support multicore processors, GPUs, and C-code generation. Image Processing Toolbox supports a diverse set of image types, including high dynamic range, gigapixel resolution and embedded ICC profile. Visualization functions and apps provide images and videos, examine a region of pixels, adjust color and contrast, create contours or histograms, and manipulate regions of interest (ROIs). The toolbox supports workflows for processing, displaying, and navigating large images. The key features are

- Image analysis, including segmentation, morphology, statistics, and measurement.
- Image enhancement, filtering, and deblurring.
- Geometric transformations and intensity-based image registration methods.
- Image transforms, including FFT, DCT, Radon, and fan-beam projection.
- Large image workflows, including block processing, tiling, and multi resolution display.
- Visualization apps, including Image Viewer and Video Viewer.
- Multicore- and GPU-enabled functions and C-code generation support.

4.5.1 Image Enhancement

Image enhancement techniques in Image Processing Toolbox increase the signal-to-noise ratio and accentuate image features by modifying the colors or intensities of an image. The

toolbox includes specialized filtering routines and a generalized multidimensional filtering function that handles integer image types, offers multiple boundary-padding options, and performs convolution and correlation. Predefined functions of this provide:

- Filter with morphological operators
- Deblur and sharpen
- Remove noise with linear, median, or adaptive filtering
- Perform histogram equalization
- Remap the dynamic range
- Adjust the gamma value
- Adjust contrast

4.5.2 Image Analysis

Image analysis is the process of extracting meaningful information from images such as finding shapes, counting objects, identifying colors, or measuring object properties. Image Processing Toolbox provides a comprehensive suite of reference standard algorithms and visualization functions for image analysis tasks such as statistical analysis, feature extraction, and property measurement.

4.5.3 Image Transforms

Image transforms play a critical role in many image processing tasks, including image enhancement, analysis, restoration, and compression. Image Processing Toolbox provides several image transforms, including Hough, Radon, FFT, DCT, and fan beam projections. Reconstruction of images from parallel-beam and fan-beam projection data is made lot easier.

4.5.4 Image Segmentation

Image segmentation algorithms determine region boundaries in an image. It is simple to explore many different approaches to image segmentation, including progressive methods, automatic thresholding, edge-based methods, and morphology-based methods such as the watershed transform that is often used to segment connected objects.

4.5.5 Cascade Object Detector

Cascade Object Detector detects object using the Viola-Jones algorithm. detector = vision.CascadeObjectDetector creates a System object that detects objects using the Viola-Jones algorithm. The detector is capable of detecting a variety of objects, including faces and a person's upper body. The type of object to detect is controlled by the Classification Model property. By default, the detector is configured to detect faces. This creates a System object, detector, configured to detect objects defined by MODEL. MODEL is a string describing the type of object to detect. There are several valid MODEL strings. Examples include 'FrontalFaceCART', 'UpperBody', and 'ProfileFace'.

CHAPTER 5
TESTING AND ANALYSIS

5.1 Testing Strategy

The hardware of the project is first checked for any error. A connection is made to the MAX232 with a USB-to-Serial cable. The COM port is checked on the PC to determine the port number that is been used. Mentioning the determined port in the MATLAB script file, the program is executed for the Vision Detection.

5.2 Speed of Execution

Speed of execution is an important aspect of this system. But there were couple of issues because of which led to limit this. This limitation helps the user never making a sudden move, precisely a jerk. Hence, the motor speed should be at slow pace. Also, the system can be secure by taking a while to evaluate the movement and thereafter making a decision, given the criticality of the application.

5.3 Accuracy

The project performs well with an accuracy of around 80%. Better lighting can improve the accuracy by providing brighter snapshots to process. However, a flashlight can be used in low light conditions. The initial pre-processing contrast stretches the image around the mean, which helps in improving the accuracy by making the detection more accurate.

5.4 Safety Features

The safety features incorporated in the system were

a) Controlled speed of detection and wheel chair drive.

b) Eye width threshold.

c) Controlled movement in either direction for limited time period.

d) Less jerk by incorporating delay in the wheel chair drive.

The movements detected are mentioned in Figure 5.1, 5.2, 5.3 and 5.4 below:

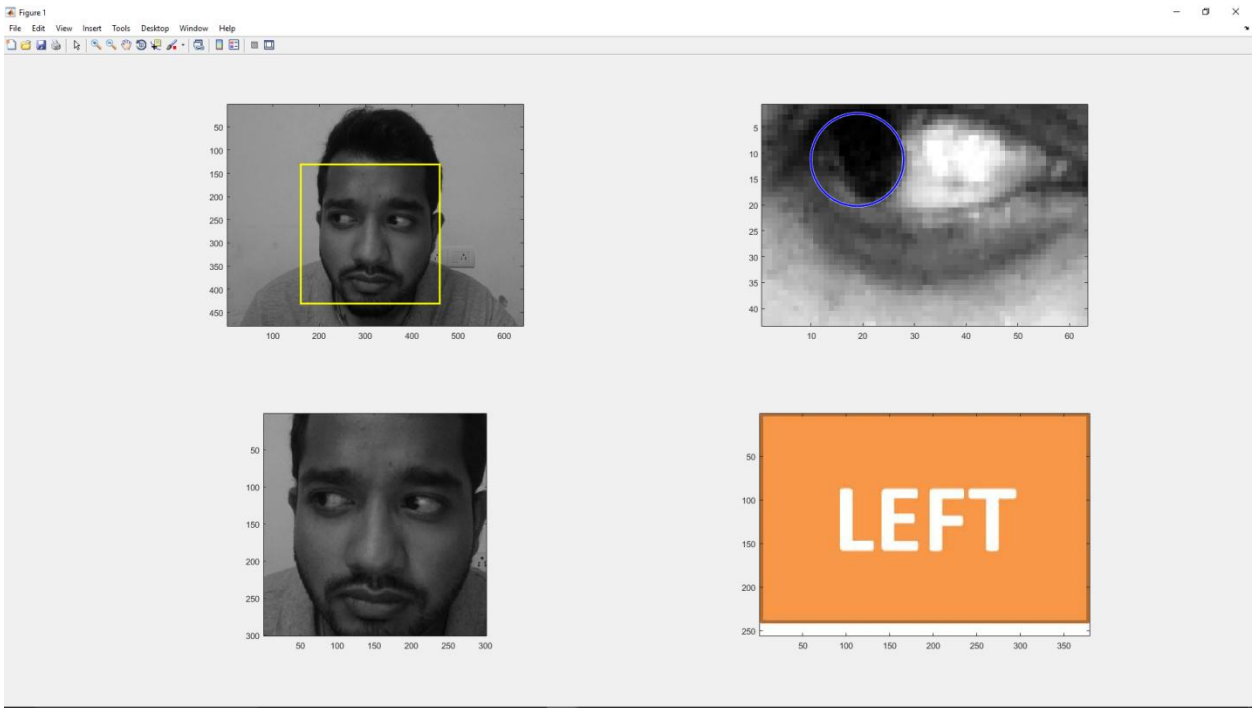

Figure 5.1 Left Movement Detection

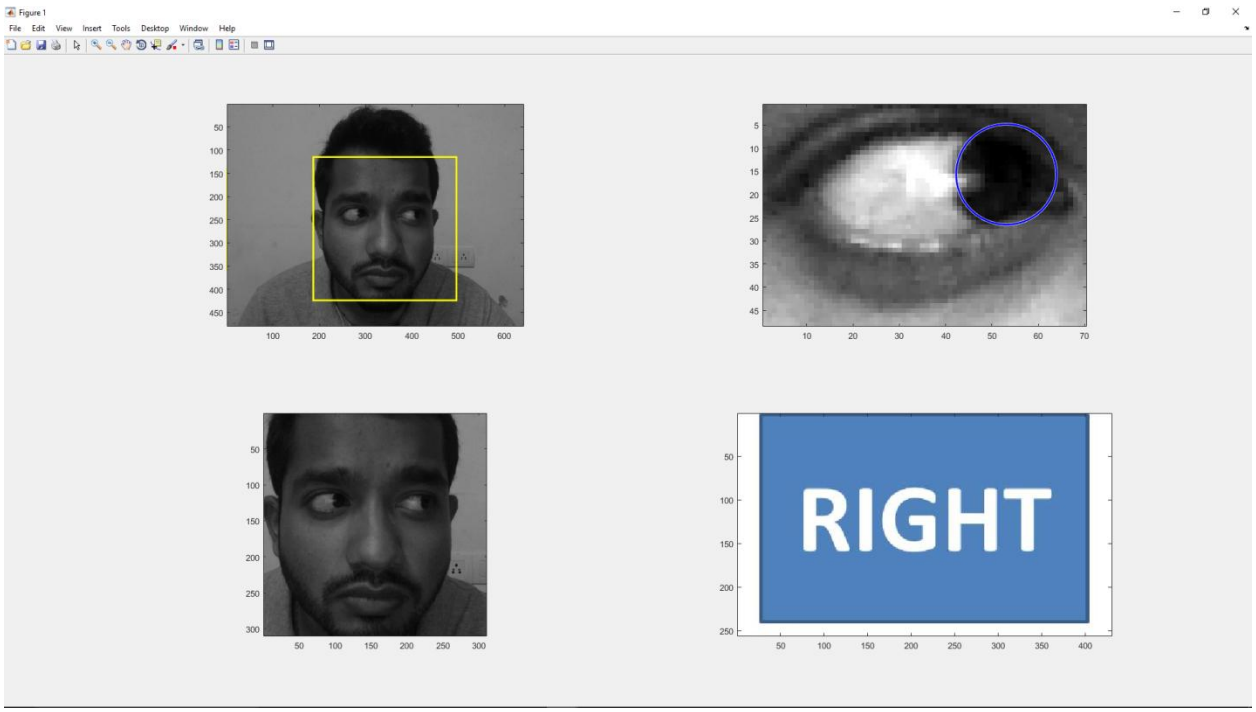

Figure 5.2 Right Movement Detection

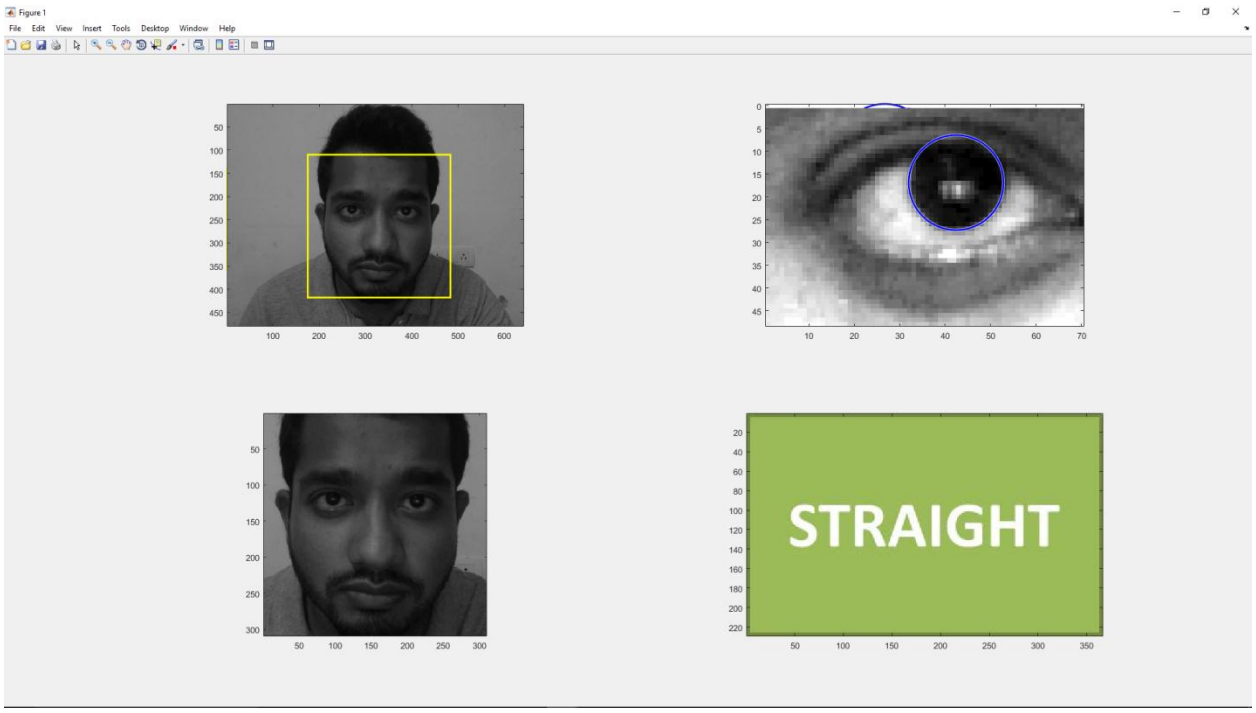

Figure 5.3 Straight Movement Detection

Figure 5.4 No Movement Detection

CHAPTER 6
CONCLUSION

The project on ‘Vision Based Wheelchair Control for Quadriplegics’ was successfully carried out with positive results and deeper insights/awareness into the field.

In this project, an innovation in an ordinary wheelchair by adding motor type mechanism and making easier and simple wheelchair to handle by using eye motion tracking for physically disabled and paralyzed is presented. The aim of this system is to contribute to the society in a small way by setting out an idea for a system which could actually better the lives of millions of people across the globe.

CHAPTER 7

FURTHER ENHANCEMENTS

The future scope of this system would be to develop a mobile app to manage the wheelchair control. Also introducing home automation in the system would be an added feature of the wheelchair where a disabled person can turn on/off home appliances without getting up from his position.

Also since the criticality of the application is so high, lot of safety precautions need to be incorporated. It needs to be made sure that the system is not fatal to the health of the person. A lot of testing needs to be done before making this a reality.

REFERENCES

- S. Tameemsultana and N. Kali Saranya, “Implementation of Head and Finger Movement Based Automatic Wheel Chair”, Bonfring International Journal of Power Systems and Integrated Circuits, vol. 1, Special Issue, pp 48-51, December 2011.

- S. Vijayprasath, DR. Rsukanesh and S. palanival Rajan "Innovative Techniques using wireless sensors in designing smart system for disabled patints" International Journal of power control signals and computation (IJPCSC) VOL. 1 NO. 4

- Ankur Thakkar, Darshan Shah, “ Eye monitored Wheel Chair Control”

- Viola Jones Algorithm
  https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

- https://www.arduino.cc/en/main/arduinoBoardUno

- http://www.rakeshmondal.info/L293D-Motor-Drive

Appendix

User Manual

STEP 1: Install the webcam.

STEP 2: Connect the hardware circuit (motors connected to the controller) to the 9V supply and also power the micro controller (Arduino).

STEP 3: Sit upright in front of the Computer's webcam.

STEP 4: Program the microcontroller with the code given and also run the MATLAB script. This will initiate the serial communication between the MATLAB script and the microcontroller.

STEP 5: Tracked Eye movements control the motors to move in the desired direction.

MATLAB Script

```
clc;
clear;
close all;
warning('off','all');
clf('reset');
%%
cam=webcam('Integrated Webcam');   %create webcam object

right=imread('RIGHT.jpg');
left=imread('LEFT.jpg');
noface=imread('no_face.jpg');
straight=imread('STRAIGHT.jpg');

detector = vision.CascadeObjectDetector(); % Create a detector for face using Viola-Jones
detector1 = vision.CascadeObjectDetector('EyePairSmall'); %create detector for eyepair

while true % Infinite loop to continuously detect the face

   vid=snapshot(cam);  %get a snapshot of webcam
   vid = rgb2gray(vid);    %convert to grayscale
   img = flip(vid, 2); % Flips the image horizontally

    bbox = step(detector, img); % Creating bounding box using detector

    if ~ isempty(bbox)  %if face exists
       biggest_box=1;
       for i=1:rank(bbox) %find the biggest face
          if bbox(i,3)>bbox(biggest_box,3)
             biggest_box=i;
```

```
    end
end
faceImage = imcrop(img,bbox(biggest_box,:)); % extract the face from the image
bboxeyes = step(detector1, faceImage); % locations of the eyepair using detector

subplot(2,2,1),subimage(img); hold on; % Displays full image
for i=1:size(bbox,1)    %draw all the regions that contain face
   rectangle('position', bbox(i, :), 'lineWidth', 2, 'edgeColor', 'y');
end

subplot(2,2,3),subimage(faceImage);     %display face image

if ~ isempty(bboxeyes)  %check it eyepair is available

   biggest_box_eyes=1;
   for i=1:rank(bboxeyes) %find the biggest eyepair
      if bboxeyes(i,3)>bboxeyes(biggest_box_eyes,3)
         biggest_box_eyes=i;
      end
   end


bboxeyeshalf=[bboxeyes(biggest_box_eyes,1),bboxeyes(biggest_box_eyes,2),bboxeyes(biggest_
box_eyes,3)/3,bboxeyes(biggest_box_eyes,4)];   %resize the eyepair width in half

   eyesImage = imcrop(faceImage,bboxeyeshalf(1,:));    %extract the half eyepair from the
face image
   eyesImage = imadjust(eyesImage);    %adjust contrast

   r = bboxeyeshalf(1,4)/4;
```

```
        [centers, radii, metric] = imfindcircles(eyesImage, [floor(r-r/4) floor(r+r/2)],
'ObjectPolarity','dark', 'Sensitivity', 0.93); % Hough Transform
        [M,I] = sort(radii, 'descend');

        eyesPositions = centers;

        subplot(2,2,2),subimage(eyesImage); hold on;

        viscircles(centers, radii,'EdgeColor','b');

        if ~isempty(centers)
          pupil_x=centers(1);
          disL=abs(0-pupil_x);    %distance from left edge to center point
          disR=abs(bboxeyes(1,3)/3-pupil_x);%distance from right edge to center point
          subplot(2,2,4);
          if disL>disR+16
             subimage(right);
             ccf = 4;
          else if disR>disL
             subimage(left);
             ccf = 3;
             else
               subimage(straight);
               ccf = 1;
             end
          end
  if ccf == 1
     fwrite(a,'1');
  elseif ccf == 3
     fwrite(a,'3');
  elseif ccf == 4
```

```
      fwrite(a,'4');
   end
   fclose(a);
   pause(2)

          end
       end
    else
      subplot(2,2,4);
      subimage(noface);
    end
    set(gca,'XtickLabel',[],'YtickLabel',[]);
  hold off;
end
```

Arduino Code

```
#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); /// REGISTER SELECT PIN,ENABLE PIN,D4 PIN,D5 PIN, D6 PIN, D7 PIN

int msg;

void forward()

{

digitalWrite(A0,HIGH);

digitalWrite(A1,LOW);

digitalWrite(A2,HIGH);

digitalWrite(A3,LOW);

delay(750);

delay(750);

delay(750);

delay(750);

digitalWrite(A0,LOW);

digitalWrite(A1,LOW);

digitalWrite(A2,LOW);

digitalWrite(A3,LOW);

}
```

```
void left()
{
digitalWrite(A0,LOW);
digitalWrite(A1,LOW);
digitalWrite(A2,HIGH);
digitalWrite(A3,LOW);
delay(750);
delay(750);
delay(750);
delay(750);
digitalWrite(A0,LOW);
digitalWrite(A1,LOW);
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);
}
void right()
{
digitalWrite(A0,HIGH);
digitalWrite(A1,LOW);
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);
```

```
delay(750);

delay(750);

delay(750);

delay(750);

digitalWrite(A0,LOW);

digitalWrite(A1,LOW);

digitalWrite(A2,LOW);

digitalWrite(A3,LOW);

}

void setup() {

  lcd.begin(16, 2);


   pinMode(A0, OUTPUT);

  pinMode(A1, OUTPUT);

  pinMode(A2, OUTPUT);

  pinMode(A3, OUTPUT);

  lcd.print(" EYE CONTROLLED ");//print name

  lcd.setCursor(0, 1); // set the cursor to column 0, line 2

  lcd.print("  WHEEL  CHAIR  ");//print name

  delay(750);//delay of 0.75sec

  // initialize serial communication at 9600 bits per second:
```

```
  lcd.clear();
}
// the loop routine runs over and over again forever:
void loop() {
  Serial.begin(9600);
  delay(750);delay(750);lcd.clear();
if(Serial.available())
  {
   msg=(Serial.read());
  }
   digitalWrite(6,LOW);
   digitalWrite(7,LOW);
   digitalWrite(8,LOW);
   digitalWrite(9,LOW);
if(msg=='1')
  {lcd.clear();
  lcd.setCursor(0, 0); // set the cursor to column 0, line 1
  lcd.print("     FORWARD    ");
  delay(750);
  forward();
  }
```

```
  else if(msg=='3')        //LEFT
  {lcd.clear();
    lcd.setCursor(0, 0); // set the cursor to column 0, line 1
  lcd.print("      LEFT      ");
  delay(750);
left();
  }
  else if(msg=='4')
  {lcd.clear();
    lcd.setCursor(0, 0); // set the cursor to column 0, line 1
  lcd.print("      RIGHT     ");
  delay(750);
right();
  }msg=0;
 Serial.end();
  }
```