Automatic Wheelchair Control by Tracking Eye Movement and Using IR Sensors

A Project Report

Submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

by

Devansh Mittal

14BEC0139

Under the guidance of

Prof. Rajalakshmi S.

School of Electronics Engineering

Vellore Institute of Technology, Vellore-632014

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

April 2018

DECLARATION

I hereby declare that the project work entitled "**Automatic Wheelchair Control by Tracking Eye Movement and Using IR Sensors**" submitted by me, for the award of the degree of *Bachelor of Technology in Electronics and Communication Engineering* to Vellore Institute of Technology is a record of bonafide work carried out by me under the supervision of Prof. **Rajalakshmi S**.

I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore **Signature of the Candidate**

Date : 20.04.2018

CERTIFICATE

This is to certify that the project work entitled "Automatic Wheelchair Control by Tracking Eye Movement and Using IR sensors" submitted by **Devansh Mittal**, School of Electronics Engineering, Vellore Institute of Technology, for the award of the degree of *Bachelor of Technology in Electronics and Communication Engineering*, is a record of bonafide work carried out by him/her under my supervision, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The report fulfills the requirements and regulations of the institute and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 20.04.2018

Signature of the Guide

The project work is satisfactory / unsatisfactory

Internal Examiner **External Examiner**

Approved by

Head of the Department
Department of Communication Engineering

School of Electronics Engineering

ACKNOWLEDGEMENT

I would like to take this opportunity to thank Vellore Institute of Technology and our Honourable Chancellor G**.** Viswanathan for providing us with an excellent platform to utilize the technical skills we have acquired over the course of the last four years.
I would like to thank Dean SENSE **Dr. Elizabeth Rufus** and HOD **Dr. Renuga Devi** for providing me with this wonderful opportunity. I wish to express my sincere thanks and regards to SENSE School and my guide, Prof. S. Rajalakshmi for providing me with all the necessary facilities and supporting for the project. I am extremely thankful and indebted to her for sharing expertise, sincere and valuable guidance and encouragement extended to me. I thank my parents for the unceasing encouragement, support and attention. I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this project.

Devansh Mittal

Executive Summary

Quadriplegia is paralysis caused by illness or injury to humans that results in partial or complete loss of limbs and torso. It’s a phenomenon which confines the ability of a person to move by himself, and he must rely on someone to carry him around. Technology can intervene and help reinstate the self-reliance of people suffering from Quadriplegia, by creating a medium, through which they can move at will. Such people may be able to move only their eyes and partially their head. This project precisely aims at targeting the movements of the eye and the head.

The idea is to create an eye monitored wheelchair system where a camera constantly stares at the person’s eyes and based on the combined movement of eye and head, decides to move the wheelchair in the direction the person desires to move in. The web camera of the laptop is used which has a MATLAB script running which constantly captures snapshots and processes them. The script based on the processing determines whether the person wants to move in a direction, and communicates this information using USB communication to a microcontroller which drives motors of the wheelchair in the desired direction. In addition to this, we have created a safety mechanism wherein an IR sensor constantly detects if there is any obstacle in front of the wheelchair and stops the wheelchair if it detects an obstacle.

CONTENTS

List of Figures

List of Tables

List of Terms and Abbreviations

DC	Direct Current
ECG	Electrocardiography
EEG	Electroencephalography
EOG	Electrooculogram
IC	Integrated Circuit
IR	Infrared
PWM	Pulse Width Modulation
RGB	Red, Green and Blue
ROI	Region of Interest
USB	Universal Serial Bus

List of Symbols and Notations

bboxes	Bounding box of Viola Jones Algorithm
I	Original image
I2	Eye Detection Image
I4	Final processed image
SE	Structuring Element

1.1 INTRODUCTION

1.1. MOTIVATION

The number of persons who are paralyzed and therefore dependent on others due to loss of self-mobility is growing with the population. 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 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 to not move their locomotors 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. Many of those suffering close to or complete paralysis usually however still can control their eye movement which was the inspiration to develop an eye-controlled electric wheelchair.

1.2 BACKGROUND

With the rapid pace of technology, people are looking into more modern and foolproof methods for the development of a wheelchair which will help people suffering from quadriplegia. Several other interfaces have been used to control a wheelchair like voice recognition or EEG/ECG signals. However, eye tracking has been easier to use and proved to give more accurate results. In addition, using eye tracking as alternative interface, control or communication methods is beneficial for a wide range of severely disabled people who are left with minimal ability to perform voluntary motion. Eye movements are the least affected by disabilities because, for example, spinal cord injuries do not affect the ability to control them, as they are directly controlled by the brain. Eye tracking can provide a larger number for possible control commands to be used with assistive technologies such as a wheelchair.

There are many different methods which have been used to track eye movements since the use of eye-tracking technology was first pioneered by Rayner & Pollatsek, 1989. Electro-oculographic techniques, for example, relied on electrodes mounted on the skin around the eye that could measure differences in electric potential so as to detect eye movements. These methods proved quite invasive and inconvenient. There was a need to develop a solution where there was no contact with the skin and therefore, most modern eye-tracking systems now use video images of the eye to determine where a person is looking.

1.2.1 LITERATURE SURVEY

Some eye tracking systems detect and analyze eye movements based on electric potentials measured with electrodes placed in the region around the eyes. This electric signal detected using two pairs of electrodes placed around one eye is known as electrooculogram (EOG). When the eyes are in their origin state, the electrodes measure a steady electric potential field. If the eyes move towards the periphery, the retina approaches one electrode and the cornea approaches the other. This changes the orientation of the dipole and results in a change in the measured EOG signal. Eye movement can be tracked by analyzing the changes in the EOG signal, which is presented in Paper [14].

Different pattern recognition techniques, such as template matching and classification, have proved effective in the field of eye tracking. Raudonis et al. [2] used principal component analysis (PCA) to find the first six principal components of the eye image to reduce dimensionality problems, which arise when using all image pixels to compare images. Then, Artificial Neural Network (ANN) is used to classify the pupil position. The training data for ANN is gathered during calibration where the user is required to observe five points indicating five different pupil positions.

The system requires special hardware which consists of glasses and a single head-mounted camera and thus might be disturbing to the patient as it is in their field of view. The use of classification slows the system and hence it requires some enhancements to be applicable. In addition, the system is not considered a real-time eye tracking system. The proposed algorithm was not tested on a known database which means the quality of the system might be affected by changes in lighting conditions, shadows, distance of the camera, the exact position in which the camera is mounted, etc. The algorithm requires processing which cannot be performed by low computational hardware such as a microcontroller.

Tang and Zhang [3] suggested a method that uses the detection algorithm combined with gray prediction to serve eye tracking purposes. The GM(1,1) model is used in the prediction of the location of an eye in the next video frame. The predicted location is used as the reference for the region of eye to be searched. The method uses low-level data in the image in order to be fast but there are no experimental results evaluating the performance of the method.

Kuo et al. [4] proposed an eye tracking system that uses the particle filter which estimates a sequence of hidden parameters depending on the data observed. After detecting possible eyes positions, the process of eye tracking starts. For effective and reliable eye tracking, the gray level histogram is selected as the characteristics of the particle filter. Using low-level features in the image makes it a fast algorithm. High accuracy is obtained from the system; however, the real-time performance was not evaluated, the algorithm was tested on images not videos and the images were not taken from a known database and, thus, the accuracy and performance of the algorithm may decrease when utilized in a real-world application.

Lui et al. [6] suggested a fast and robust eye detection and tracking method which can be used with rotated facial images. The camera used by the system is not head mounted. A Viola-Jones face detector, which is based on Haar features, is used to locate the face in the whole image. Then, Template Matching (TM) is applied to detect eyes. Zernike Moments (ZM) is used to extract rotation invariant eye characteristics. Support Vector Machine (SVM) is used to classify the images to eye/non-eye patterns. The exact positions of the left and right eyes are determined by selecting the two positions having the highest values among the found local maximums in the eye probability map. Detecting the eye region is helpful as a pre-processing stage before iris/pupil tracking. Especially, it allows for eye detection in rotated facial images. This work presented a simple eye tracking algorithm but the results of the method evaluation were not reported and thus the proposed eye tracking method is weak and not usable.

Hotrakool et al. [7] introduced an eye tracking method based on gradient orientation pattern matching along with automatic template updates. The method detects the iris based on low level features and motion detection between subsequent video frames. The method can be used in applications that require real-time eye tracking with high robustness against change in lighting conditions during operation. The computational time is reduced by applying down-sampling on the video frames. The method achieves high accuracy. However, the experiments were performed on videos of a single eye, which eliminates all surrounding noise, and a known database was not used. The method detects the iris but does not classify its position. The method requires minimal CPU processing time among other real-time eye tracking methods investigated in this survey. The motion detection approach discussed in this paper is worth being considered in new algorithms to obtain the minimal required CPU processing time in eye tracking applications.

Yuan and Kebin [10] presented Local and Scale Integrated Feature (LoSIF) as a new descriptor for extracting the features of eye movement based on a non-intrusive system, which gives some tolerance to head movements. The feature descriptor uses two-level Haar wavelet transform, multi-resolution characteristics and effective dimension reduction algorithm, to find the local and scale eye movement features. Support Vector Regression is used in mapping between the appearance of the eyes and the gaze direction, which correspond to each eye's appearance. The focus of this method is to locate the eye without classifying its position or gaze direction. The method was found to achieve high accuracy in iris detection, which makes this method useful in iris detection and segmentation which is important for eye tracking. However, the real-time performance was not evaluated.

Fu and Yang [11] proposed a high-performance eye tracking algorithm in which two eye templates, one for each eye, are manually extracted from the first video frame for system calibration. The face region in a captured frame is detected and a normalized 2-D cross-correlation is performed for matching the template with the image. Eye gaze direction is estimated by iris detection using edge detection and Hough circle detection. They used their algorithm to implement a display control application. However, it has an inflexible calibration process. The algorithm was not tested on a variety of test subjects and the results were not clearly reported which requires the algorithm to be investigated carefully before choosing to implement it.

Mehrubeoglu et al. [12] introduced an eye detection and tracking system that detects the eyes using template matching. The system uses a special customized smart camera which is programmed to continuously track the user's eye movements until the user stops it. Once the eye is detected, the region of interest (ROI) containing only the eye is extracted with the aim of reducing the processed region. From their work, it can be concluded that it is a fast eye tracking algorithm with acceptable performance. The algorithm could be a nice feature to be added to modern cameras. A drawback is that the experiments were not performed using a database containing different test subjects and conditions, which reduces the reliability of the results. In addition, the algorithm locates the coordinates but does not classify the eye gaze direction to be left, right, up or down.

Yang et al. [8] proposed a scheme which employs gray difference between the face, pupils and corneal reflection points for eye detection. The proposed scheme was tested under a cross-ratio-invariant-based eye tracking system. The test included users wearing glasses and other accessories and the results showed the ability of the system to eliminate the optical reflective effect of accessories and glasses. The scheme first prepares for gaze tracking by a preprocessing stage applied on cropped faces.

This is particularly useful in applications which use a very close camera. The results are not detailed and not performed on a database containing various test subjects under different conditions which makes the algorithm weak when considered for use in real-world applications. In addition, the required CPU time was not addressed and thus the algorithm needs optimization to determine whether it works in real-time applications.

Chen and Kubo [1] proposed a technique where a sequence of face detection and Gabor filters is used. The potential face regions in the image are detected based on skin color. Then, the eye candidate region is determined automatically using the geometric structure of the face. Four Gabor filters with different directions ($0, \pi/4, \pi/2, 3\pi/4$) are applied to the eye candidate region. The pupil of the eye does not have directions and thus, it can be easily detected by combining the four responses of the four Gabor filters with a logical product. The system uses a camera which is not head-mounted. The accuracy of the algorithm is not investigated and the required CPU time is not mentioned which does not make the algorithm preferable for real-world applications compared to other algorithms.

Khairosfaizal and Nor'aini [5] presented a straightforward eye tracking system based on mathematical Circular Hough transform for eye detection applied to facial images. The first step is detecting the face region which is performed by an existing face detection method. Then the search for the eye is based on the circular shape of the eye in a two-dimensional image. Their work added value to academic research but not to real-world applications.

Pranith and Srikanth [9] presented a method which detects the inner pupil boundary by using Circular Hough transformation whereas the outer iris boundary is detected by circular summation of intensity from the detected pupil center and radius. This algorithm can be used in an iris recognition system for iris localization because it is applied to cropped eye images. However, it needs further analysis to obtain its accuracy and real-time performance by applying it on a database containing variant images.

Alioua et al. [13] presented an algorithm that handles an important part of eye tracking, which is analyzing eye state (open/closed) using iris detection. Gradient image is used to mark the edge of the eye. Horizontal projection is then computed for the purpose of detecting upper and lower boundaries of the eye region and the Circular Hough transform is used for iris detection. This algorithm can be useful in eye blinking detection, especially since the possible output classes are limited: open, closed or not an eye. When used in a control application, it can increase the number of possible commands. Due to its importance, the algorithm is required to be fast but no required CPU time was mentioned, although indicating the required CPU time could be useful

1.3 OBJECTIVE

- The main objective of this project is to design a vision based wheelchair system, using the camera to acquire user images and analyzing user intent using head gestures. The main idea is to come up with the system that is not expensive 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 we have to take an extra care about the safety of the system. This safety mechanism is carried out by installing a system to detect any obstacles in front of it and stopping the wheelchair if it detects an obstacle.
- The project captures the images using the webcam of 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 commands to the Arduino circuitry driving the motors attached to the wheel chair.

1.4 ORGANIZATION OF THE REPORT

Section 1 of this report discusses the motivation, background and objective of the project. In section 2 each element in the block is explained. Along with the description of the project, the goals too are mentioned in section 2. Section 3 talks about the technical specifications of this project. In section 4, the design flow and exact approach is elaborated upon, step by step. Section 5 includes the schedule, milestones and tasks in this project while section 6 is a demonstration of the project and displays the results. Section 7 gives a brief idea of the cost analysis. Section 8 is a conclusion wherein we draw observations from this project. Section 9 contains all the references that were used during the course of this project.

2. PROJECT DESCRIPTION AND GOALS

There are three major parts from the system design standpoint –

- Software Design
- Hardware Design
- Mechanical Design

2.1 SOFTWARE DESIGN

There are multiple aspects to the software design of this project. Since majority of computational work is done in software, a lot of time went in software design and testing.

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 USB transmission of the decision to move.

The MATLAB design can be structured into many small sub-parts each of which is described below –

2.1.1 INITIALIZATION OF VARIABLES AND SETTING UP USB COMMUNICATION

MATLAB 2018a contains the 'MATLAB Support Package for Arduino Hardware' which is used in this project to set up communication with the Arduino. Once this package is downloaded, the Arduino is automatically configured to communicate with the laptop and we must just initialize a variable for the Arduino.

After setting up the USB communication to enable the data link between MATLAB and the Arduino, we reset the variables needed during the program to their initial valuables.

2.1.2 IMAGE CAPTURE AND EYE DETECTION

MATLAB 2018a equips an Image Processing Toolbox, which we have used majorly in this section of the Software Design. We utilize the webcam of the laptop which streams continuous video signals on MATLAB using the video processing toolbox available. The function 'imaqhwinfo' is used to recognize all video capture adaptors. Identifying the laptop web camera and then using it to stream the video signal is the next step.

The requirement of the design was to continuously look at different frames, based on which determine motion. It is practically impossible to do a lot of processing on a per frame basis.

That is why we try to sample every 25th frame. So, a snapshot of every 25th frame is captured and processed. We used the ‘getsnapshot’ command to capture these snapshots.

The image is then converted to grayscale image, as we do not need color information 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. After the initial pre-processing, we move towards the eye detection. 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 the large rectangle is made up of many such smaller rectangles, which are fundamentally feature points on a human face. It is based on the principle that all human faces share some similar properties. These regularities may be matched using Haar Features. Figure 2.1 shows the Haar Feature that looks similar to the eye region and is darker than the upper cheeks which is applied onto a face.

Figure 2.1 – Haar Feature for Eye

The ‘cascadeobjectdetector’ on MATLAB, utilized this algorithm to extract and detect the eyes of the person. Based on the x and y co-ordinates of the bounding box of one eye and the height and width of that same eye, we get a cropped image of any one eye on which further image processing will be carried out in order to detect the eye movements.

2.1.3 IMAGE PROCESSING

This step involves various morphological operations carried out on the cropped image of the eye. The field of mathematical morphology contributes a wide range of operators to image processing, all based around a few simple mathematical concepts from set theory. The operators are particularly useful for the analysis of binary images and common usages include edge detection, noise removal, image enhancement and image segmentation. The various processes carried out are shown in the form of a block diagram in Figure 2.2.

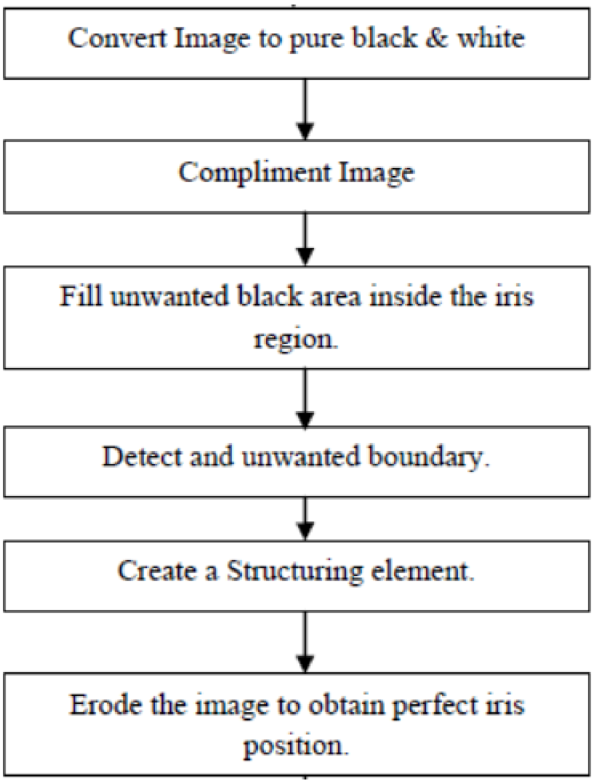

Figure 2.2 – Processes Involved in Image Processing

Morphological operations on an image require the image to be in binary format. Since the acquired images are in the grayscale format, we first convert the grayscale image into a pure black and white image. This is by done by using the 'imbinarize' function available in MATLAB 2018a. This quantizes the grayscale image into 2 levels, 1 and 0. After multiple tests, we fix a threshold value of 0.1 for this function. What this means is that all pixel values above 0.1 are considered to be white pixels and pixel values below 0.1 are considered as black pixels. This threshold will vary depending on the lighting condition and the size of the eye of different persons.

This image is then complemented to reverse the order of white and black pixels. This is carried out using the ‘imcomplement’ function available in MATLAB 2018a. This results in an iris area which is white and other area to be black.

Due to limitations of video device we get some unwanted noise in the captured image which is mainly caused due to reflection of light from eyeball. This is seen as black spots in the iris area in the cropped image. So, in order to remove this noise from image “imfill” operation is performed which removes unwanted pixels from our area of interest.

Also, image boundaries need to be cleared so as to remove the region of the eyebrow captured by the camera. This is done by using a MATLAB function “imclearborder”.

Erosion of the image is performed next. For this we first create a structuring element. The shape is chosen to be disk so that the structuring element closely resembles the iris. The erosion process moves the disk-shaped structuring element from left to right and top to bottom. At the centre point (indicated by the centre of the structuring element), the process will look for a complete overlap of the image region with the structuring element. If there is no overlap then the centre pixel will be set to be white. Thus, the result of erosion process is a well-defined iris area.

2.1.4 MOVEMENT DETECTION

The movement detection is done with a very basic principle. This final processed image is then split into 9 cells. The main concept of such a division is to increase the accuracy when alignment with the camera is disturbed. We then calculate the sum of the values of each pixel in each cell. A black pixel is represented by 0. Thus, the cell with the maximum white pixels will have the highest count. This gives the position of the iris. If cell 1, 2 or 3 return this highest count iris is in right position. Similarly, if cells 4, 5 or 6 return this highest count iris is in centre position. And if cells 7, 8 or 9 return this highest count iris is in left position.

In order to stop the wheelchair, the person has to close his eyes for one iteration. To detect if the eye is open or closed, we use the grayscale image, prior to it being converted to a binary image, as an input and find the number of maximum peaks in the image. The MATLAB function ‘findpeaks’ returns a vector with the local maxima (peaks) of the input signal vector (data sample that is larger than its two neighboring samples). Based on the number of peaks in an image of a closed eye (Figure 2.3) and open eye (Figure 2.4), a threshold is set to differentiate the two as the pattern maximas are located far when eye is closed. On the other hand, the pattern maximas are located close to each other when the eye is open and therefore the number of peaks will be greater than the number of peaks when the eye is closed.

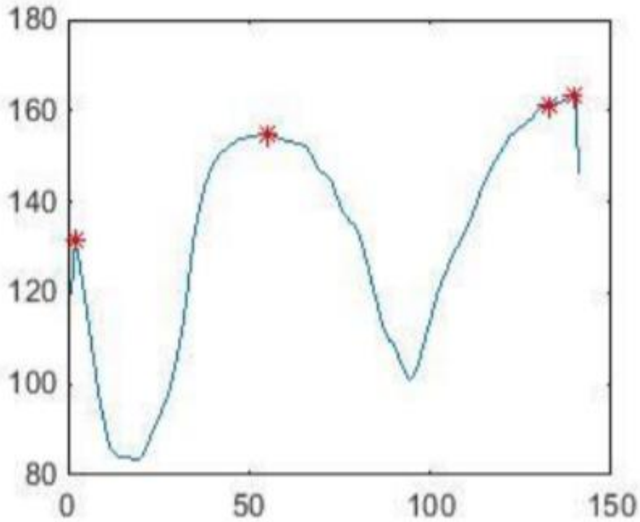

Figure 2.3 – Peaks of Image When Eye is Closed

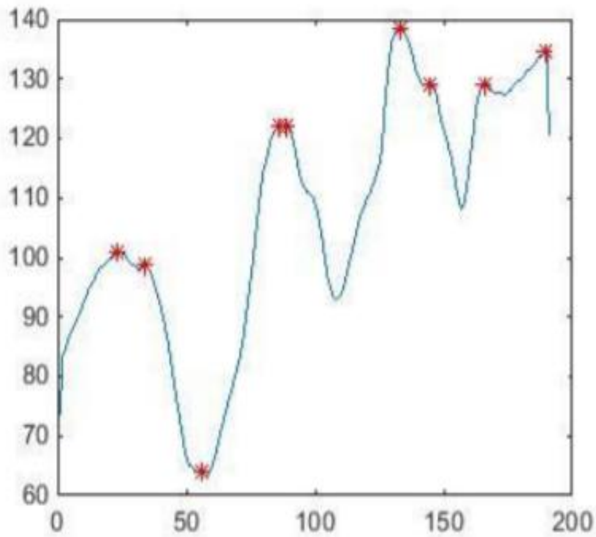

Figure 2.4 – Peaks of Image When Eye is Open

After detecting the eye movements, the corresponding signal is sent to the Arduino which ultimately is used to control the motors. This is done by using the ‘writeDigitalPin’ command which is available under the ‘MATLAB Support Package for Arduino Hardware’ package. Either 1 or 0 is sent to the Arduino which denotes motion or no motion respectively.

2.2 HARDWARE DESIGN

The hardware part consists only to control the motor. Figure 2.5 shows the motor controller circuit.

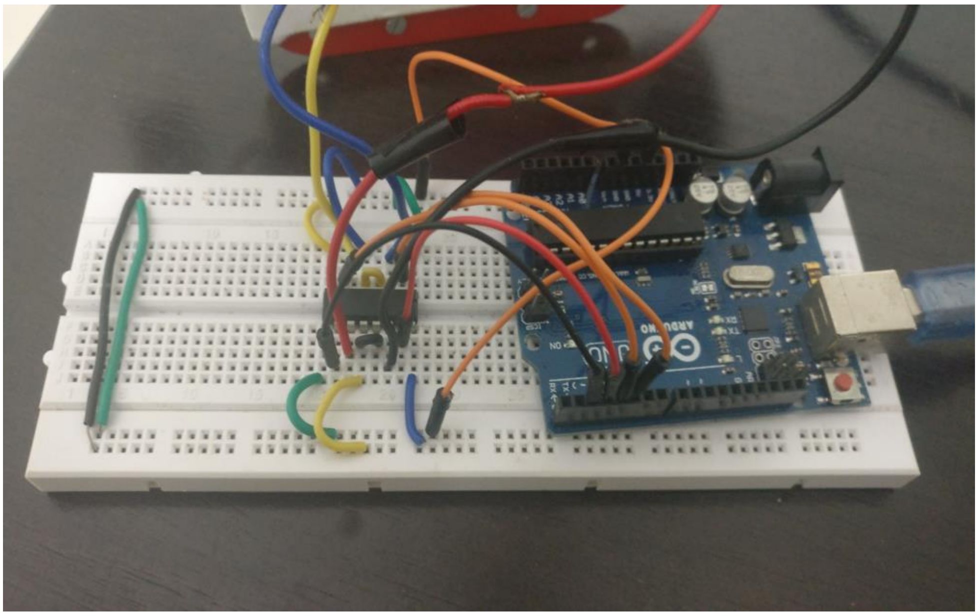

Figure 2.5 – Motor Control Circuit

2.2.1 MOTOR CONTROL

The signals that we obtain from the MATLAB script, we then use that to drive the motors. The MATLAB program makes the decision of which wheel will be motion and which wheel will not be in motion, thereby deciding the direction in which the wheelchair will move. So, the script sends the bit 0 and 1 for motion or no motion. The circuit shown in Figure 2.6 is fairly safe and used to drive the motors. The signal is sent to the Arduino UNO board via the USB connection to the laptop. This signal is then sent to the motor driver ‘L293D’ IC which is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.
L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction.

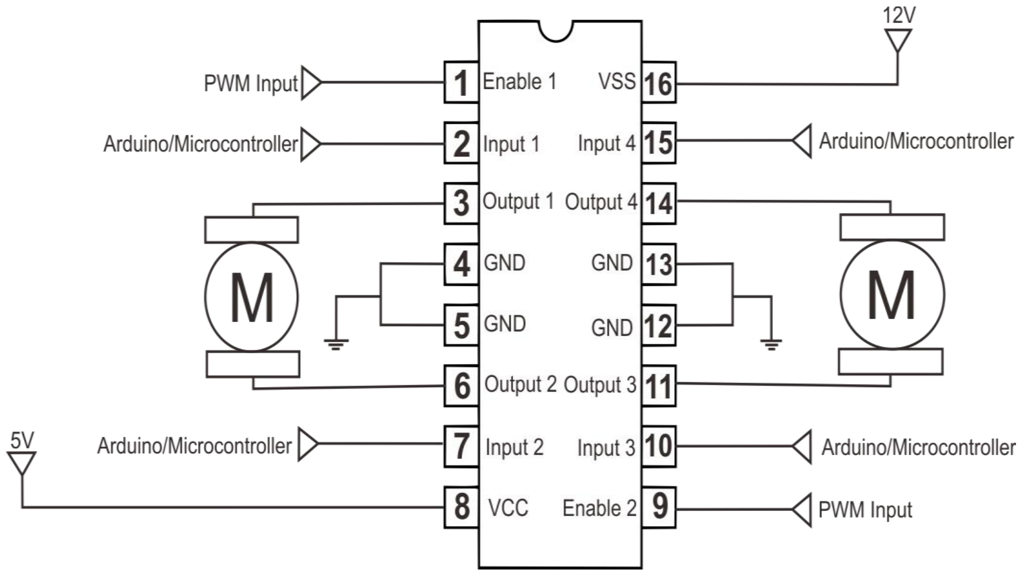

Figure 2.6 – Circuit Diagram

Therefore, the amplified current from the motor driver is then sent to the DC motors. These motors are connected to the wheel of the wheelchair. Based on the initial value computed by the MATLAB program, the wheel will move in that particular direction.

2.2.2 SAFETY MECHANISM

Safety is of paramount importance, especially for a patient suffering from quadriplegia as they might be unable to see an obstacle in front of them. Even if they see an obstacle in front of them, they might find it difficult to react quickly to stop the wheelchair in time. Therefore, this solution prioritizes the safety of the patients and aims to stop the wheelchair immediately in case any obstacle is detected in front of the wheelchair. To develop this solution, we have made use of a couple of IR sensors to detect the obstacles. The IR sensors are placed on the front left and front right side of the wheelchair. The IR sensor consists of a transmitter and a receiver. Whenever there is no obstacle in front of the wheelchair, the IR signal transmitted from the transmitter of the IR sensors is not reflected to the IR sensor and hence the receiver does not receive the IR signal. Therefore, the IR module gives a low signal to the Arduino. The MATLAB reads this value from the Arduino at every iteration and continues with normal operation.

However, whenever there is an obstacle in front of the wheelchair, the IR signal transmitted from the transmitter of the IR sensor is reflected due to this obstacle and the receiver of the IR sensor receives this IR signal. Therefore, the IR module gives a high signal to the Arduino. The MATLAB program scans for this high value at every iteration. If it receives a high value from the Arduino, it stops the wheelchair. The wheelchair operation is resumed only when the obstacle is removed. Figure 2.7 shows the entire hardware circuit along with the IR sensor.

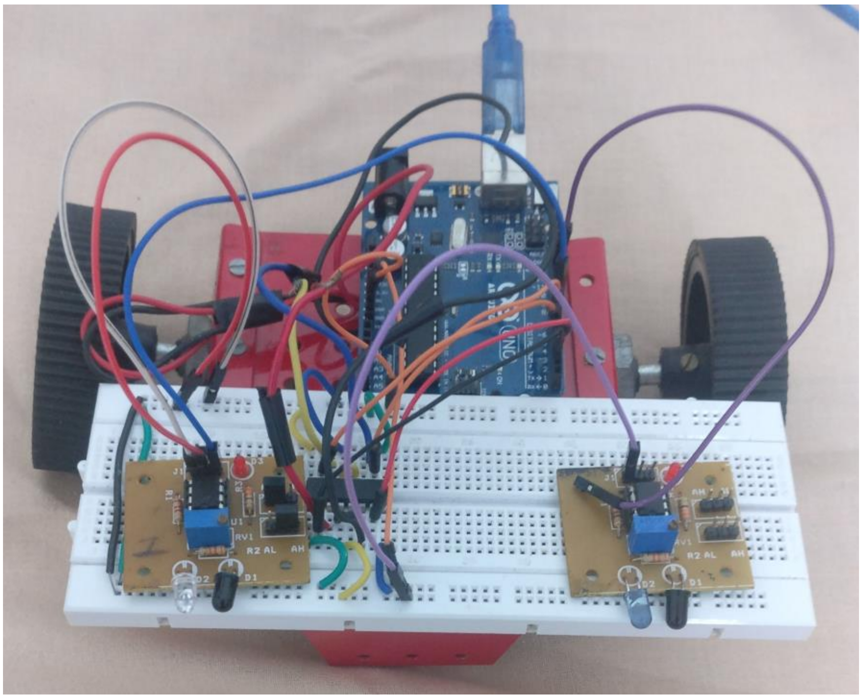

Figure 2.7 - Entire Hardware Circuit

2.3 MECHANICAL DESIGN

For the mechanical part, we have worked on trying to create a wheelchair prototype. We make use of a metallic chassis to which 2 rubber wheels and one caster wheel are attached. Figure 2.8 shows the wheelchair prototype.

In this project, we are using the laptop webcam to detect the eye movements. In an actual wheelchair, we can have a tray attached to the wheelchair on which a light weight laptop can be accommodated.

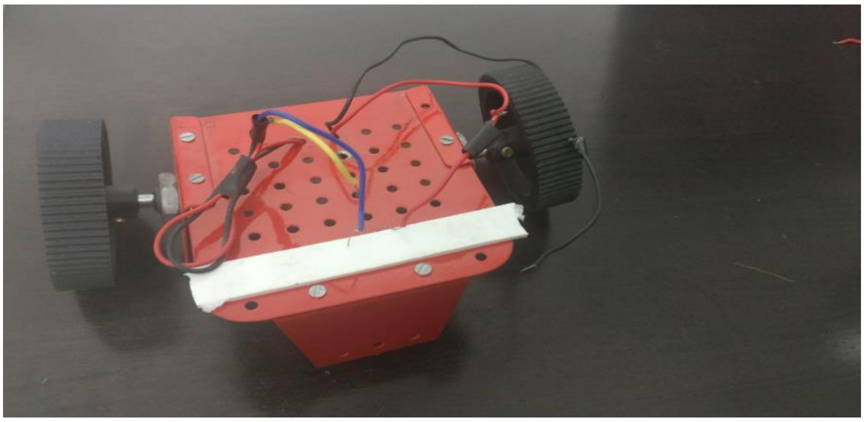

Figure 2.8 –Wheelchair Assembly

3. TECHNICAL SPECIFICATIONS

3.1 ARDUINO UNO

Arduino Uno is a microcontroller board based on the ATmega328P. 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; it simply needs to be connected to a computer with a USB cable or powered with an AC-to-DC adapter or battery to get started. The Arduino IDE is used to program the Arduino. Figure 3.1 shows the Arduino UNO. The following are its technical specifications:

- 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

Figure 3.1 – Arduino UNO

3.2 L293D Motor Driver

The L293D device is a quadruple high-current half-H driver. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. It is designed to drive inductive loads such as relays, solenoids, DC and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications. Figure 3.2 shows the L293D Motor Driver IC. The following are its technical specifications:

- Wide Supply-Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- Internal ESD Protection
- High-Noise-Immunity Inputs
- Output Current: 600 mA per channel
- Peak Output Current: 1.2 A per channel
- Output Clamp Diodes for Inductive Transient Suppression

Figure 3.2 – L293D Motor Driver IC

3.3 Image

- Dimensions of cropped image (grayscale): 516 x 516 pixels
- Dimensions of image (RGB): 516 x 516 x 3 pixels
- Number of bits per pixel (RGB): 8

3.4 IR Sensor PX Module

An object can be detected with an infrared system consisting of an infrared transmitter and a receiver. More in detail an IR transmitter, also known as IR LED, sends an infrared signal with a certain frequency compatible with an IR receiver which has the task to detect it. There are different kinds of IR sensors for different type of application. IR technology is used, for example, in proximity sensors to detect a near object, in contrast sensors to find a path or in counting sensors to count objects.

The IR transmitter sends an infrared signal that, in case of a reflecting surface (e.g. white colour), bounces off in some directions including that of the IR receiver that captures the signal detecting the object. When the surface is absorbent (e.g. black colour) the IR signal isn't reflected and the object cannot be detected by the sensor. This result would occur even if the object is absent.

The IR transmitter is a particular LED that emits radiation in the frequency range of infrared, invisible to the naked eye. An infrared LED just works as a simple LED with a voltage of 3V DC and a current consumption of about 20mA. The IR receiver, such as a photodiode or a phototransistor, is capable of detect infrared radiation emitted from the IR transmitter. Aesthetically it is similar to a LED but the external capsule can be wrapped by a dark colour film.

This sensor detects objects at a distance in range between 2~30cm. With the potentiometer you can calibrate the sensitivity according to the application and environmental conditions (e.g. brightness). The IC LM393 is an open-collector voltage comparator which provides an output if there is a pull-up R between the output of the IC (DO) and the power supply Vcc (R=10KΩ). The output DO is:

- high if the object is not detected;
- low if the object is detected.

Figure 3.3 shows the IR sensor module. Following are some technical specifications of this sensor:

- Operating Voltage: 5V
- Mode Selection: Configurable HIGH / LOW Output State (Using AH and AL pins)
- Adjustable Range using preset (potentiometer on board)

Figure 3.3 – IR Sensor

3.5 Programming Environment

The programming environment used for writing and executing the MATLAB code is MATLAB and the version of programming used is MATLAB 2018a. MATLAB is a multi-paradigm numerical computing environment. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.

4. DESIGN APPROACH AND DETAILS

4.1 DESIGN APPROACH

The basic design of this project is as shown in Figure 4.1. We will have a look at the overall code structure of our algorithm and the logic behind the decision making.

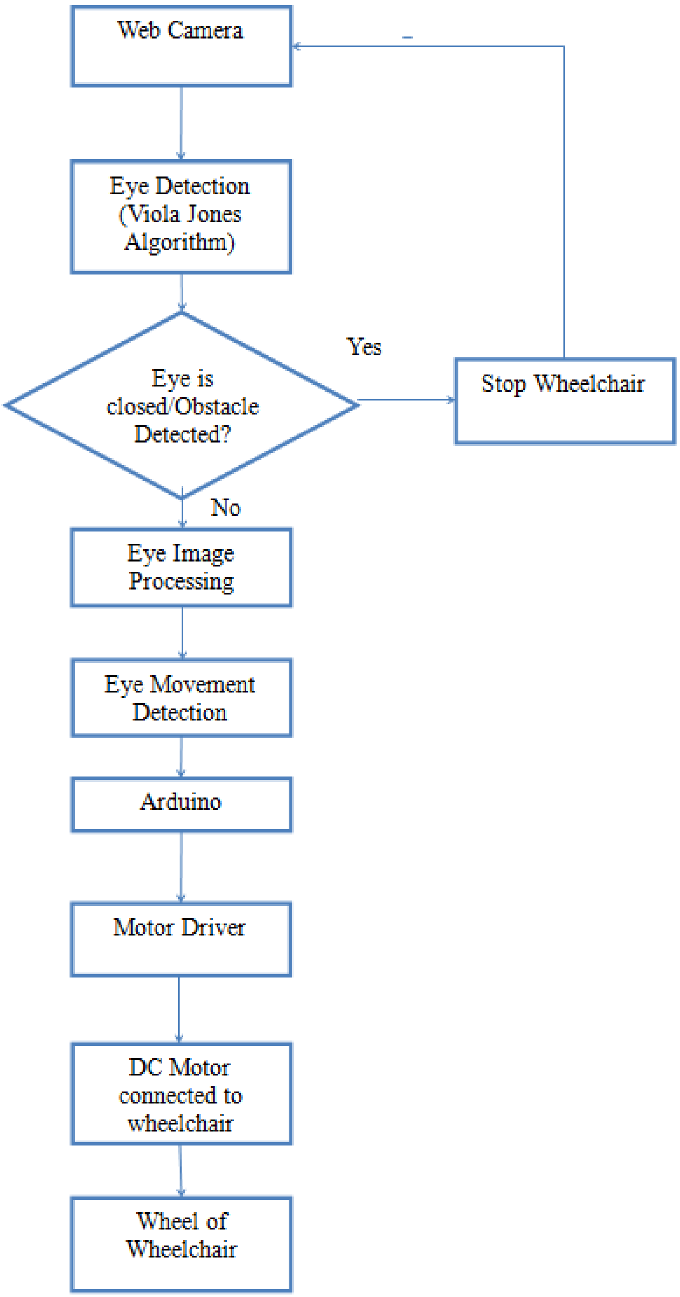

Figure 4.1 - Block Diagram of Project

Figure 9 shows us that 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. The code structure can be explained in the following steps:

4.1.1 EYE MOVEMENT DETECTION:

Step 1: Initially we set up the USB communication that will be used later for the interface between MATLAB and the Arduino, the video capture and the program variables. The web camera is also initialized to an object to enable video capture of the eyes.

Step 2: We then take continuous video frames and sample the input every 25^{th} frame and save it as the screen shots.

Step 3: Each frame is then converted into grayscale frames. For the accurate results, we perform contrast stretching on each frame to make the dark region darker and bright region brighter. This will enable the detection of the eyes better.

Step 4: Now, after working on each frame, we try to detect the eyes. This we do by estimating the position of left as well as the right eye using the Viola Jones Algorithm.

Step 5: To avoid detection errors, we incorporated an error handling mechanism, which specifies a threshold for the height and width of a valid eye, by calibrating it for the user. 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. Then it displays an error saying, 'invalid eye' and face detection procedure starts again from the beginning.

Step 6: Based on the height and width of the eyes and the x and y co-ordinates of the eyes, we select a single eye, which can be either the left or right eye, on which further processing will take place and therefore movement can be detected.

Step 7: We check if the eye is closed or open. This is done by finding the number of maximum peaks in the image. A closed eye will have their peaks spread out due to the increased distance between the eyebrow and the eyelids and therefore will have lesser number of peaks. On the other hand, an open eye will have their peaks close together and therefore a higher number of peaks. By setting a threshold, we can estimate if the eye is open or closed.

Step 8: Simultaneously, IR sensor detects if there is an obstacle in front of the wheelchair. We make use of 2 IR sensors on either side of the front of the wheelchair to carry out this detection.

Step 9: If the eye is closed, it indicates that the wheelchair should stop. Otherwise if either of the IR sensors return a high value to the Arduino and therefore to the MATLAB program, it indicates that there is an obstacle in front of the wheelchair. In either case, a command is sent to both the left and right wheel of the wheelchair to come to a halt. The camera will continue to take snapshots of the eye and in order to resume the movement of the wheelchair, the person has to remove the obstacle in front of the wheelchair, if there was an obstacle in front of the wheelchair in the first place, and close his eyes for one iteration again.

Step 10: If the eye is open or if the wheelchair is to be resumed after it is stopped, further image processing is carried out on the eye.

Step 11: The grayscale image is converted to a binary image which contains just 1 or 0 pixel values.

Step 12: The image is then complemented.

Step 13: Due to refection of light, the image contains unwanted black spots in the iris area. These black spots are detected and removed.

Step 14: Unwanted boundaries are detected and removed.

Step 15: A structuring element is created. Using this structuring element, the image is eroded to obtain perfect iris position.

Step 16: The resultant image is divided into 9 cells. This is done by creating 9 separate images from the single image.

Step 17: The pixels in each sub image are summed. We then check which of the combination of cells, 1, 2, 3 or 4, 5, 6 or 7, 8, 9 has the highest sum.

4.1.2 DRIVING THE MOTOR SIGNAL:

Step 1: Now after detecting the eye movements, we have to come up with a decision algorithm that will help the controller to drive the motors accordingly:

- Valid Left: The decision to turn left will be considered as valid if the sum of pixels of cells 7, 8, 9 is higher than the sum of pixels in the other cells. Then, a high signal is sent to the Arduino pin which is connected to the right wheel and a low signal is sent to the Arduino pin which is connected to the left wheel in order to facilitate this left movement.

- Valid Right: The decision to turn right will be considered as valid if the sum of pixels of cells 1,2,3 is higher than the sum of pixels in the other cells. Then, a high signal is sent to the Arduino pin which is connected to the left wheel and a low signal is sent to the Arduino pin which is connected to the right wheel in order to facilitate this right movement.
- Valid Straight: The decision to move straight will be considered as valid if the sum of pixels of cells 4,5,6 is higher than the sum of pixels in the other cells. Then, a high signal is sent to the Arduino pins which is connected to the left and right wheels in order to facilitate this straight movement.

Step 2: The above signals are then sent to the from the Arduino to the motor driver IC 'L293D'. This IC amplifies the input current to a level which can drive the DC Motors.

Step 3: The appropriate signals are then sent to the corresponding DC motor. This DC motor is attached to the wheel of the wheelchair and therefore moves the wheelchair in the direction determined by the eye.

4.2 CODES AND STANDARDS

The software used in this project is MATLAB 2018a. Following are the codes and standards that significantly affect this project:

- ASCII – American Standard Code for Information Interchange
- USB – Universal Serial Bus
- 610.4-1990 – IEEE Standard for Image Processing

4.3 CONTSTRAINTS, ALTERNATIVES and TRADE-OFFS

CONSTRAINTS:

- Sampling every 25^{th} frame as it will be impossible to do a lot of processing on a per frame basis.
- The threshold has to be changed from person to person and therefore only a single person can use it at a time.
- Slow speed of execution in order to maintain accuracy of the system.
- Low light conditions and quality of web camera may have an effect on the accuracy of results.

TRADE-OFFS:

- We have chosen convenience over speed of execution. A faster speed of execution will result in jerks during the movement of the wheelchair. The speed of execution is limited in order to provide for comfortable movement for the person.
- We have selected accuracy over speed of execution. It will be impossible to do a lot of processing on a per frame basis. Therefore, we increase the frame interval per snapshot in order to provide enough time for the system to process the image and make an accurate decision on the movement of the wheelchair.
- We have chosen the web camera of the laptop as it is a very cost effective option. High quality web camera with excellent video capturing regardless of lighting conditions will provide very accurate results, however the cost of the system will drastically increase.

ALTERNATIVES:

- Using a better quality web camera if there are no budget constraints.
- Other eye detection techniques instead of the Viola Jones Algorithm.
- Other iris movement detection techniques.
- Using other microcontrollers instead of Arduino
- Making use of a small computer like Raspberry Pi etc. instead of a laptop to carry out all the detection and processing.
- Using an ultrasonic sensor instead of IR sensor.

5. SCHEDULES, TASKS AND MILESTONES

- **November – December 2017**

  Literature Survey was carried out and research on various iris movement detection methodologies was done.
- **December 2017**

  Review 1 – Research Methodology was finalized and Abstract was submitted.
- **January 2018**

  MATLAB coding was done for detecting the eye.
- **February 2018**

  Algorithm development and MATLAB coding was done for detecting the iris movement.
- **24th February 2018**

  Review 2 – Implementation of existing system was presented successfully to review panel and further work on hardware execution was discussed.
- **March 2018**
  - Research was carried out on various methods to detect if eye is closed or open and algorithm development and MATLAB coding was done to implement it.
  - Arduino was interfaced with MATLAB.
  - The hardware needed to implement the movement of wheelchair was assembled and connected.
  - Further research into making this system safer was considered and IR sensors were used for obstacle detection.
- **April 2018**
  - Code was optimized by removing unnecessary lines of code. Benefits of using functions was explored and therefore functions were incorporated into the code.
  - Final thesis including all results and poster was prepared for final review.

6. PROJECT DEMONSTRATION

The image has undergone various processing stages in this project. This section discusses the results of these processes in detail. The debug screen of MATLAB was the most useful aspect of our testing strategy. Figure 6.1 shows complete setup of the system including the hardware and the laptop with the debug screen which constantly displays the direction the wheelchair will move in and any errors, if present.

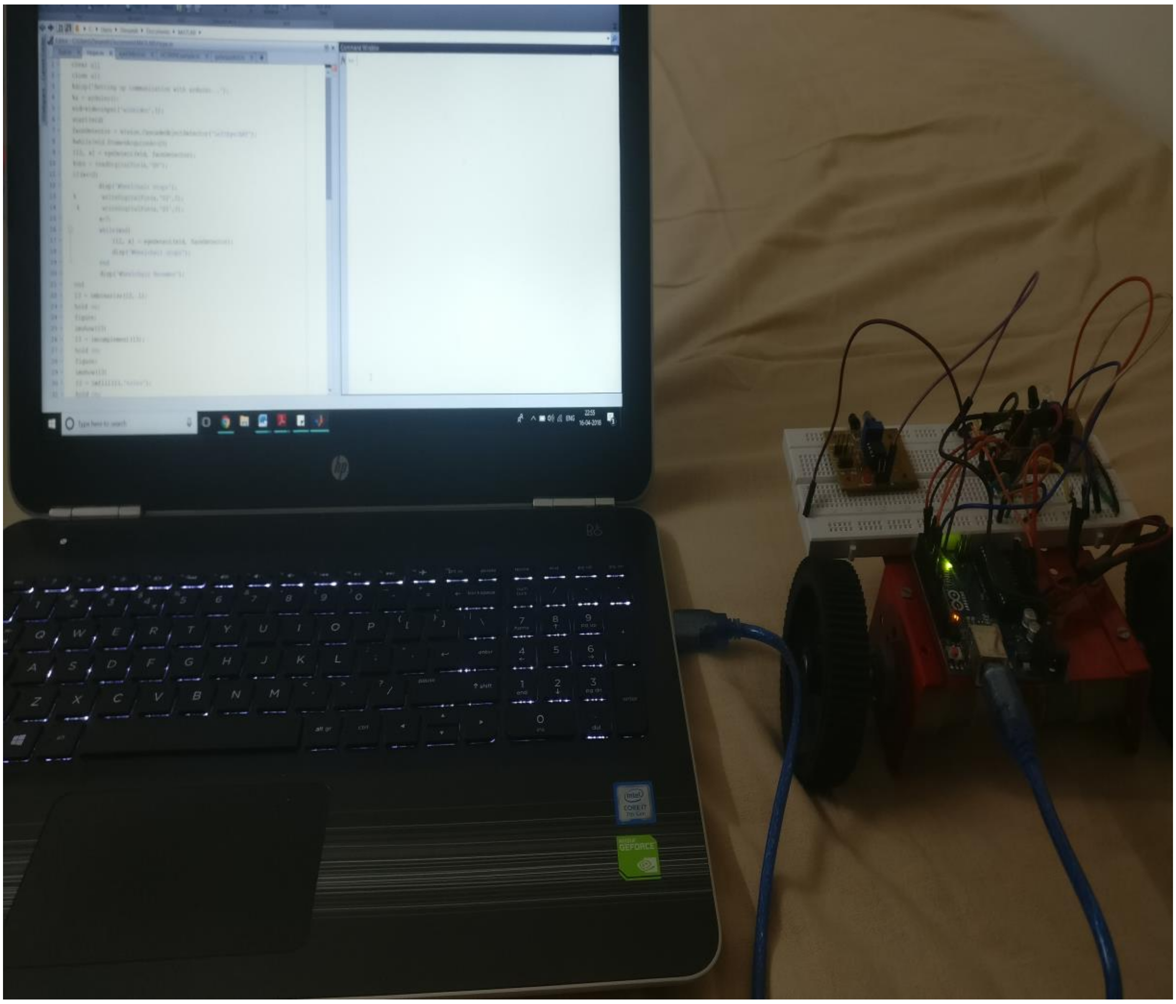

Figure 6.1 - Complete Setup of the System

6.1 EYE DETECTION

Figure 6.2 shows the detected eyes by using the Viola Jones algorithm. It can be seen that the image is a grayscale image and the image has been contrasted to make the dark regions darker and light regions lighter which aids in the implementation of the above algorithm.

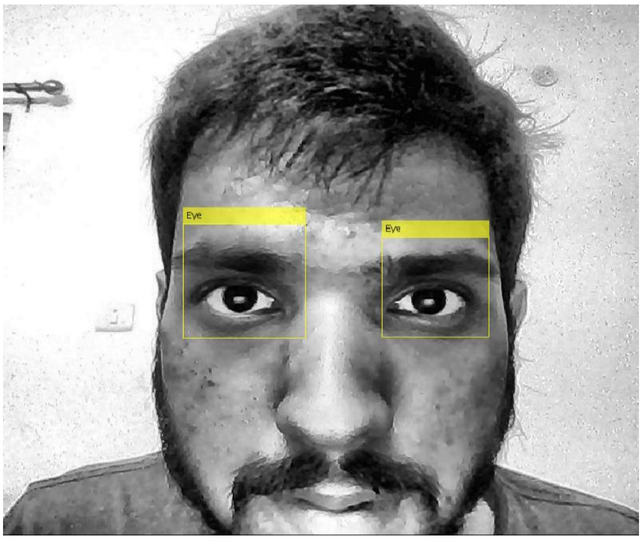

Figure 6.2 - Eye detection using Viola Jones Algorithm

From Figure 6.3, we see that the left eye and the right eye are detected separately. Each of the bounding box consists of a 1x4 matrix where the 1st element and 2nd element are the x and y co-ordinates of the top left corner of the box respectively. The 3rd and 4th element indicate the width and height of the eye respectively. Figure 15 shows the debugging screen of MATLAB. Here we can see the co-ordinates of both the left and right eye contained in the variable ‘bboxes’.

```
bboxes =

   708   316   136   136
   456   298   156   156


bb =

   708   316   136   136
```

Figure 6.3 - Values of the Bounding Box

In order to ensure that only a valid eye is detected by the algorithm, we reject any detected box which has the width value outside the range of 100-200. If the width values are within the 100-200 range, we proceed by selecting the any one of the 2 detected eyes. The values of the selected eye are stored in the variable ‘bb’ as seen in Figure 15. In order to only work on the eye and eliminate an unnecessary processing of the face, we crop the original image according to the values mentioned in the variable ‘bb’. Figure 6.4 shows the cropped image.

Figure 6.4 - Cropped Image f Eye

Before we move to the processing this image, we have to first detect if the eye is closed or open. This is done by finding the number of maximum peaks in the image. Figure 6.5 shows the plot of the peaks when the eye is open and Figure 6.6 shows the plot of the peaks when the eye is closed.

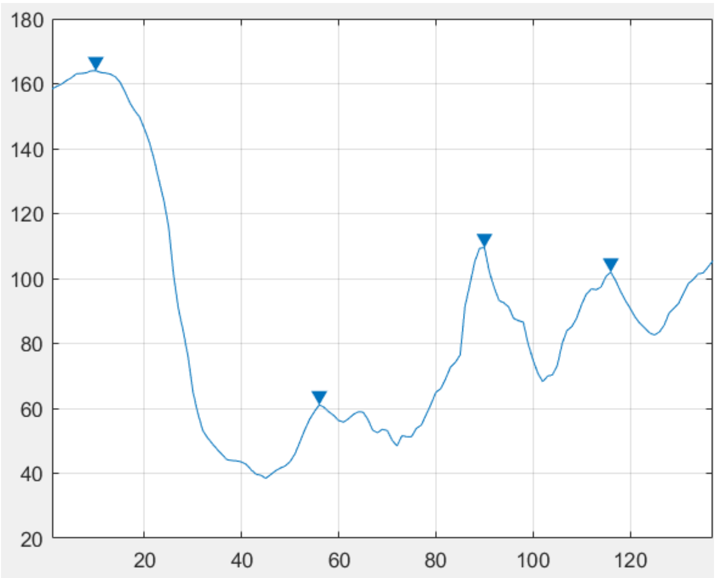

Figure 6.5 - Maximum Peaks of Image When Eye is Open

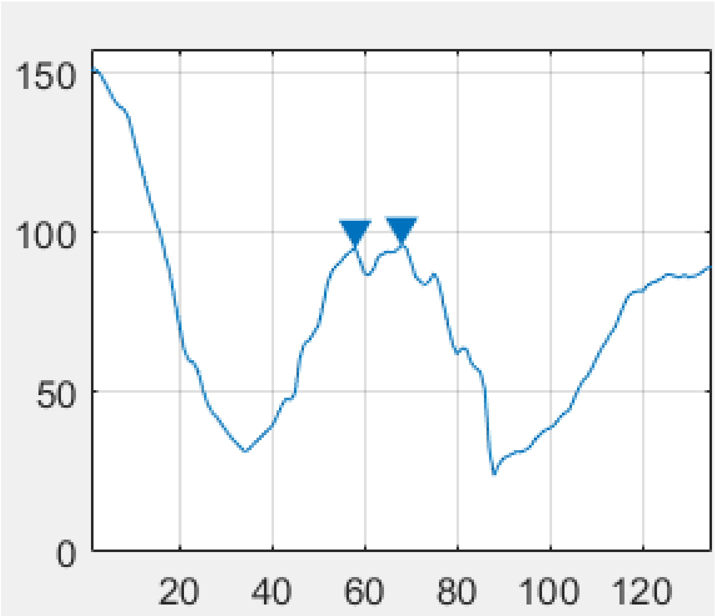

Figure 6.6 - Maximum Peaks of Image When Eye is Closed

6.2 IMAGE PROCESSING OF EYE

The next step is to binarize the image by converting the pixel values from between 0 and 1 to 0 or 1. Figure 6.7 shows the binary image.

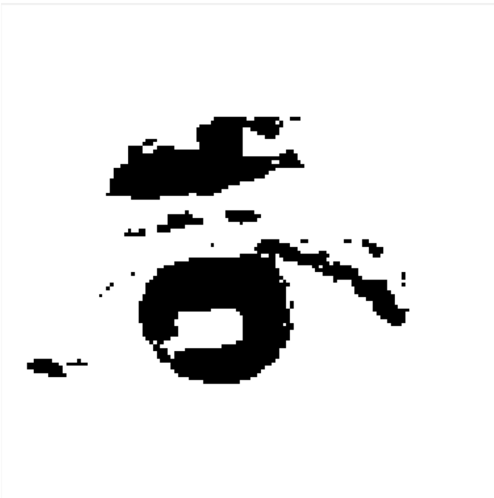

Figure 6.7 - Converted Binary Image

Next, the image is complemented where the 0 pixels values are changed to 1 and 1 pixel values are changed to 0. Figure 6.8 shows the complemented image.

Figure 6.8 - Complemented Image

In the above image, black spots can be seen in the middle of the iris. This black spot is due to the reflected light on the eye which is bound to happen because of the low-quality web camera of the laptop. However, we have a solution for precisely such a problem. The ‘imfill’ operation searches for such black pixels surrounded by white pixels and fills it with white pixels. Figure 6.9 shows such the result of such an operation.

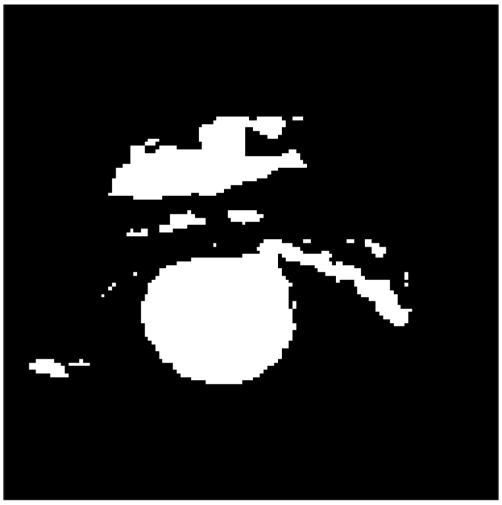

Figure 6.9 - Black Spots Within Iris Area are Removed

In the above image, we therefore see that we get an image which does have any noise clouding the iris area.

Once we create a structuring element resembling a disk, the erosion function is performed on this image. The result of such an operation is just the white pixels of the iris completely surrounded by black pixels. This is seen in Figure 6.10.

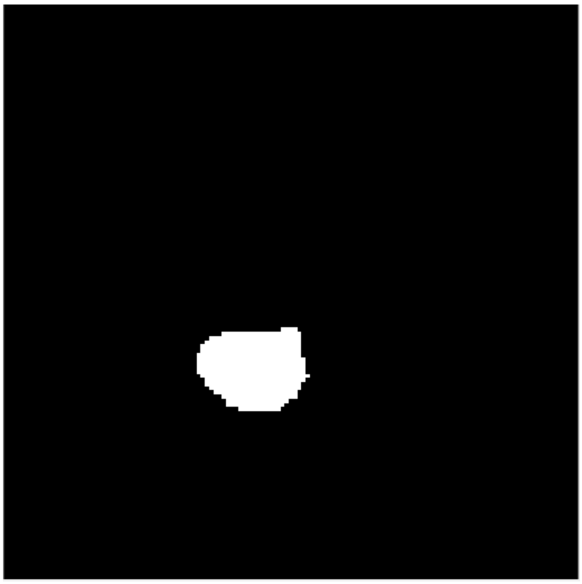

Figure 6.10 - Eroded Image

6.3 EYE MOVEMENT DETECTION

Figure 6.11 is then divided into 9 equal cells. Figure 6.11 shows the divided image

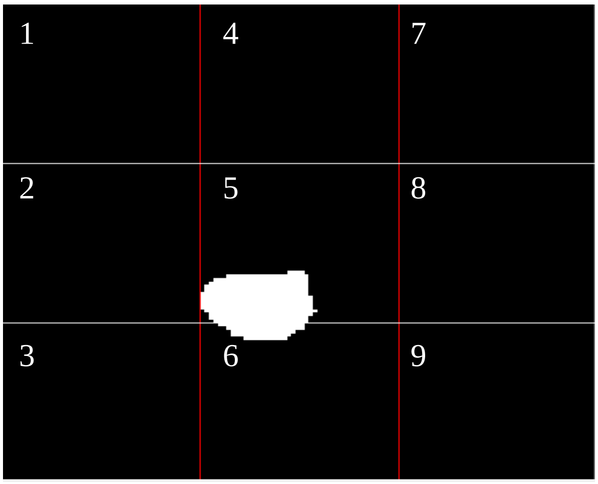

Figure 6.11 - Image divided into 9 cells

The sum of pixels in cells 1, 2, 3/4, 5, 6/7, 8, 9 are separately summed up. In the debug screen, we see that cells 4, 5, 6 have the largest sum and therefore the wheelchair should move straight. Figure 6.12 shows the debug screen indicating this.

```
Wheelchair moves straight
>> sum123

sum123 =

     0

>> sum456

sum456 =

   402

>> sum789

sum789 =

     0
```

Figure 6.12 - Debug Screen Indicating Values of Summation of Pixels and the Direction of Movement of Wheelchair

6.4 ACCURACY

The project performs satisfactory with performance accuracy of around 70-90%. The results after testing it for 100 attempts to move in a random direction were made by me and another person. The results were tabulated as shown in Table 6.1.

Table 6.1 – Accuracy Table

Attempts	**Successful Attempts**	**Accuracy**
100	86	86%
100	74	74%

The first scenario was carried out by testing the movement of the wheelchair using my eye. This shows a good accuracy percentage given how cost effective our solution is. The second scenario was carried out on another person. The dimensions of eye of every person varies and therefore the threshold parameters for every person must be changed. As such, we observe a lesser accuracy for the second scenario as the threshold parameters were fixed for my eye.

Better lighting can improve the accuracy by providing brighter snapshots to process. The initial pre-processing contrast stretches the image around the mean, which helps in improving the accuracy by making the detection more accurate.

Successful attempts were counted as all those attempts which resulted in movement of the wheel chair in the desired direction. For the system to be accurate, each time a system is configured for a person, alter the height and length of the specified eye so that the system recognizes the eye of the person with high precision.

7. COST ANALYSIS

Table 7.1 – Cost Analysis

S.No	COMPONENTS	COST (INR)
1.	Arduino Uno	500
2.	L293D Motor Driver IC	42
3.	DC Motor (2)	314
4.	Rubber Wheel (2)	70
5.	Caster Wheel	21
6.	Chassis	150
7.	Wires	30
8.	IR Sensor	146
	TOTAL COST	**1300**

8. CONCLUSION

The system functions with an accuracy rate of 70-90 % which was above our expectations. The image capture, eye movement detection and the algorithm for validating movement attempts perform very reliably as our results suggest.

The aim of this project is to contribute to the society in our small way by setting out an idea for a system which could actually better the lives of millions of people across the globe. We believe we have done great justice to the idea, and ended up getting more than satisfying results.

Though our prototype performs satisfactorily, but a lot of work needs to be done before making the product commercially viable. Some sort of sequence of events should trigger start of detection, because we do not want the wheel chair to move when the person is just casually glaring in different directions. Similarly, we can incorporate certain sequence for turning ON and OFF electrical devices, or door locks.

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 such a product a reality.

8. References

[1] Y. W. Chen and K. Kubo, "A Robust Eye Detection and Tracking Technique Using Gabor Filters," Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007), Kaohsiung, 2007, pp. 109-112.

[2] V. Raudonis, R. Simutis and G. Narvydas, "Discrete eye tracking for medical applications," 2009 2nd International Symposium on Applied Sciences in Biomedical and Communication Technologies, Bratislava, 2009, pp. 1-6.

[3] J. Tang and J. Zhang, "Eye Tracking Based on Grey Prediction," 2009 First International Workshop on Education Technology and Computer Science, Wuhan, Hubei, 2009, pp. 861-864.

[4] Y. L. Kuo, J. S. Lee and S. T. Kao, "Eye Tracking in Visible Environment," 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kyoto, 2009, pp. 114-117.

[5] W. M. K. Wan Mohd Khairosfaizal and A. J. Nor'aini, "Eyes detection in facial images using Circular Hough Transform," 2009 5th International Colloquium on Signal Processing & Its Applications, Kuala Lumpur, 2009, pp. 238-242.

[6] H. Liu and Q. Liu, "Robust real-time eye detection and tracking for rotated facial images under complex conditions," 2010 Sixth International Conference on Natural Computation, Yantai, Shandong, 2010, pp. 2028-2034.

[7] W. Hotrakool, P. Siritanawan and T. Kondo, "A real-time eye-tracking method using time-varying gradient orientation patterns," ECTI-CON2010: The 2010 ECTI International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, Chaing Mai, 2010, pp. 492-496.

[8] C. Yang, J. Sun, J. Liu, X. Yang, D. Wang and W. Liu, "A gray difference-based pre-processing for gaze tracking," IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS, Beijing, 2010, pp. 1293-1296.

[9] A. Pranith and C. R. Srikanth, "Iris recognition using corner detection," The 2nd International Conference on Information Science and Engineering, Hangzhou, China, 2010, pp. 2151-2154.

[10] Y. Zhang and K. Jia, "A local and scale integrated feature descriptor in eye-gaze tracking," 2011 4th International Congress on Image and Signal Processing, Shanghai, 2011, pp. 465-468.

[11] B. Fu and R. Yang, "Display control based on eye gaze estimation," 2011 4th International Congress on Image and Signal Processing, Shanghai, 2011, pp. 399-403.

[12] M. Mehrubeoglu, Linh Manh Pham, Hung Thieu Le, R. Muddu and Dongseok Ryu, "Real-time eye tracking using a smart camera," 2011 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), Washington, DC, 2011, pp. 1-7.

[13] N. Alioua, A. Amine, M. Rziza and D. Aboutajdine, "Eye state analysis using iris detection based on Circular Hough Transform," 2011 International Conference on Multimedia Computing and Systems, Ouarzazate, 2011, pp. 1-5.

[14] O. V. Acuña, P. Aqueveque and E. J. Pino, "Eye-tracking capabilities of low-cost EOG system," 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, IL, 2014, pp. 610-613.

Curriculum Vitae

Devansh Mittal was born in Bengaluru. He did his schooling from Bishop Cotton Boys' School Bangalore, affiliated to Indian Certificate of Secondary Education. He passed 10th grade with 95%. He completed his 11th and 12th from Delhi Public School Bangalore East, affiliated to Central Board of Secondary Education. He passed 12th grade with 93.6%. He is currently pursuing B.Tech in Electronics and Communication Engineering from VIT University, Vellore. He has a good record of academic performance. His CGPA at the time of submission of this thesis is 8.88. He can be contacted using the following details

Address: 8164, Prestige Shantiniketan, Whitefield Road, Bangalore – 560048

Email id: devansh3@gmail.com

Mobile No: +91-9597728078