Camera Based Eye Movement and EOG Detection to Control Mobility Assistive Device Using Graphical User Interface.

AUTHOR
ANUM RASHID

REGISTRATION NUMBER
00000276791

SUPERVISOR
Dr. ASIM WARIS

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY,

ISLAMABAD

JULY 2022

Camera Based Eye Movement and EOG Detection to Control Mobility Assistive Device Using Graphical User Interface.

Author
ANUM RASHID

Registration
Number
00000276791

A thesis submitted in partial fulfillment of the requirements for
the degree of

MS Biomedical Engineering

Thesis Supervisor

Dr. ASIM WARIS

Thesis Supervisor's Signature: _____

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY,

ISLAMABAD

JULY 2022

Thesis Acceptance Certificate

It is certified that final copy of MS thesis written by Ms. Anum Rashid, Registration No. 00000276791 of MS Biomedical Sciences and Engineering (SMME) has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfilment for award of MS degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have been incorporated in the said thesis.

Supervisor: _____
Dr. Asim Waris

HOD: _____
Dr. Syed Omer Gilani

Principal: _____
Dr. Javed Iqbal

Declaration

I certify that this research work titled “Camera based eye movement and EOG detection for mobility assistive device using Graphical User Interface” is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

Signature of Student

Anum Rashid

2022-NUST-Ms-BME-00000274977

Proposed Certificate for Plagiarism

It is certified that MS Thesis Titled Detection of **Camera Based Eye Movement and EOG Detection to Control Mobility Assistive Device Using Graphical User Interface** by **Anum Rashid** has been examined by us. We undertake the follows:

a. Thesis has significant new work/knowledge as compared already published or are under consideration to be published elsewhere. No sentence, equation, diagram, table, paragraph or section has been copied verbatim from previous work unless it is placed under quotation marks and duly referenced.
b. The work presented is original and own work of the author (i.e. there is no plagiarism). No ideas, processes, results, or words of others have been presented as Author own work.
c. There is no fabrication of data or results which have been compiled/analyzed.
d. There is no falsification by manipulating research materials, equipment, or processes, or changing or omitting data or results such that the research is not accurately represented in the research record.
e. The thesis has been checked using TURNITIN (copy of originality report attached) and found within limits as per HEC plagiarism Policy and instructions issued from time to time.

Name & Signature of Supervisor

Name: Dr. Asim Waris

Signature: _____

Copyright Statement

Acknowledgements

I am grateful to my Creator Allah Subhana-Watala for guiding me through every step of my job and for every new notion He instilled in my mind to help me enhance it. Without His invaluable assistance and instruction, I would have been unable to accomplish anything. Nobody is more deserving of recognition than Him, whether it was my parents or any other individual who assisted me throughout the length of my thesis.

I am eternally grateful to my loving parents, who reared me from the time I was unable to walk and have continued to support me in every aspect of my life.

I would like to thank my supervisor, Dr. Asim Waris, as well as GEC colleagues Dr. Omer Gillani, and Dr. Adeeb Shahzad, for their assistance with my thesis.

Finally, I would like to extend my gratitude to Dr Zia Mohy Ud Din, HOD Biomedical Department, Air University and my colleagues who have helped me with my research.

Dedicated to my parents and my husband Ali Raza Asif and for their unconditional support throughout my degree.

Abstract

Researchers from all over the world have recently become increasingly interested in bio-based human machine interfaces (HMI) for the assistance of paralyzed people enabling them to live an assistance free life. Among various approaches of designing a Human machine interface, eye signals are considered the most appropriate type of input. In general, eye-tracking systems assess a person's eyeball position and gaze direction and are classified into two approaches: electrooculography-based and computer vision based. This research uses EOG, and computer vision technique to predict which input method is more appropriate and user friendly for the mobility of an electric wheelchair. EOG data is acquired for four different eye movements i.e., right, left, upward, downward using BIOPAC. Video based data set is acquired using a webcam mounted at a fixed distance from the subject. EOG dataset is then processed and classified using eleven different classifiers among which the Decision tree shows the highest accuracy and F1 score i.e., 88.94 ± 13.82, 89.12 ± 13.58 respectively. The other data set of videos is processed using computer vision. Deep learning algorithms are used to classify the results. Both systems mentioned in this study have their own limitations. For EOG based system, the attachment of electrodes is a must requirement. This causes irritation to the user and sometimes generates motion artifacts which can be a source of hinderance for the motion of any HMI. For computer vision-based system, camera is a must requirement. However, it can't be used in dark rooms, outdoor; during night times, wearing sunglasses and in similar other situations. For such situations, another alternative is an infrared camera, but prolonged usage of such camera can damage the eye. Therefore, a hybrid system should be developed which involves both techniques i.e, EOG and a camera which can effectively drive any mobility assistive device.

Key Words: ***Computer Vision, Electrooculogram, EOG, Gaze tracking, faster RCNN, Deep learning***

Table of Contents

List of Figures

List of Tables

CHAPTER 1: INTRODUCTION

1.1 Background

People with accidental loss of limbs; upper, lower or both due to stroke, hemorrhage and other medical condition leads to complete or partial loss of their limb control resulting in paralysis, loss of sensation and even an amputee [1]. For paralyzed patients, external help is required to achieve daily life activities [2] which is usually provided by medical care personal like caretakers, nurses, and other paramedics. This may be time consuming, require manpower and for some people, unaffordable [3].

Figure 1.1: A real time Human Machine Interface [4]

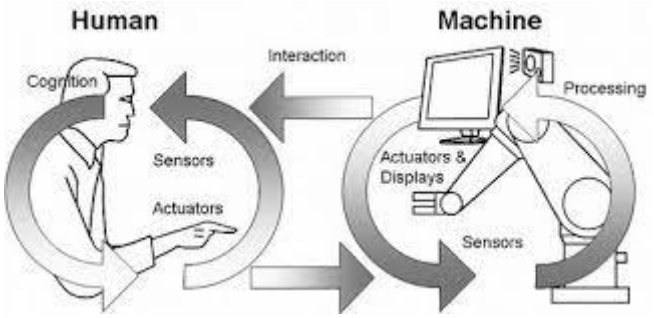


Figure 1.2: Description of HMI [5]

Considerable effort has been made in the fields of rehabilitation and artificial intelligence prioritizing the application of intricate algorithms for scrutinizing and interpretation of HMI: human machine interfaces. Human Machine Interface is an interfacing technology between the operator and a machine which is very supportive for a disabled person as it tends to enhance their life by facilitating their activities and improving their life. This has paved way for the development of smart mobility assistive devices to help handicapped individuals live independently **[6]**.

1.1.1 Types of Paralysis

Paralysis is the loss of control over a muscle or a group of muscles. This loss of control is caused by the destruction of tissues in the central nervous system that cannot be cured. The reason behind is that the tissues in brain are highly specialized that it loses the ability to regenerate. Any nerve cells or nerve fibers in the brain or spinal cord which have been destroyed are lost permanently. Thus, resulting in the loss of pathway for signal transmission from brain to the muscle. Different body parts are affected in different conditions leading to various types of paralysis. The figure below shows which type affects which area of the body.

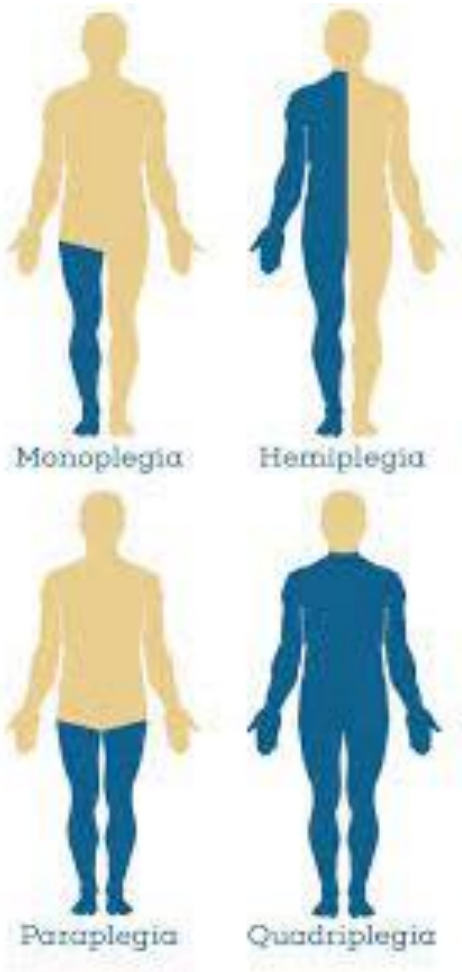


Figure 2.3: Types of Paralysis [7]

1) Monoplegia

Monoplegia is the condition which involves the paralysis of any one limb, either arm or leg.

2) Hemiplegia

It involves the paralysis of one side of the body and is usually caused by stroke, which damages the one side of the brain.

3) Paraplegia

Paraplegia is used to describe the inability to voluntarily move the lower parts of the body. It usually effects the areas of toes, feet, legs and sometimes include abdominal region as well.

4) Quadriplegia

It's a condition which involves the paralysis of all four limbs and sometimes, along with them, certain organs are effected too.

1.1.2 Mobility Aids

To help handicapped patients take care of themselves self-reliantly, mobility assistive devices are widely used to realize two basic and essential limb functions: (i) Walking function of the lower limb and (ii) Grasping function of the upper limbs [4].

Figure 1.4: Types of Conventional mobility aids [8]

Mobility aids designed and manufactured over the years have gone through different stages. Researchers across the world are putting efforts to develop assistive devices particularly wheelchairs for rehabilitative purpose of physically challenged population [9]. From manual to automatic and intelligent devices, researchers have developed user friendly mobility assistive aids to improve their daily life activities without the help of personal assistance or caretakers hence improving their self-esteem, practical abilities, and quality of life. Among these devices, the most common is wheelchair for the lower limbs and robotic hands for the upper limbs [10].

Figure 1.5: Modern standing wheelchair [11]

Figure 1.6: Electric powered Wheelchair [12]

Figure 1.7: Prosthetic hand [13]

1.2 Scope, and Motivation

According to WHO (World Health Organization) around 10% of the world population suffer from physical disabilities. From this percentage of the population who live with disabilities, 10% require the use of a wheelchair, but only a small number have access to them, and very few have access to the appropriate type of wheelchair [10]. Such handicapped patients are unable to make use of conventional wheelchairs. Moreover, among many of this handicapped population, very few are those who have access to the right type of wheelchair [10].

Previous researchers have focused on different techniques to upgrade the mechanical system of conventional wheelchair to modern, smart system. Some of the most worked techniques involves control system based on sensors which are attached on the user in a non-invasive manner which drives the wheelchair, hence forming an HMI [14]. Other examples of HMIs designed to augment lifestyle of paralyzed people are artificial prosthesis like prosthetic limbs and virtual

keyboards [15]. Valbuena et al. introduced the Rhombus layout SSVEP-based virtual keyboard using five flickering stimuli [16]. These and many such devices use magnetoencephalograms and electroencephalograms, called as Brain computer Interfaces (BCIs) to design HMIs [16].

Within the interfaces, specific systems are connected, which take body signals and turn them into control signals as electrical impulses. These body signals also known as biopotentials which are often used for this purpose are Electroencephalography (EEG), Electrooculography (EOG) and Electromyography (EMG) [14]. These bio signals however differ in their application.

1.2.1 Electromyography (EMG) controlled mobility assistive devices

Electromyography is a combination of electrical signals which are produced due to the muscle activation [17]. The motor neurons of the motor units of the muscle transmits electrical signals which results in muscle contraction [18]. The EMG signal represents the functional and structural properties of the muscle [18].

Recently, EMG has been applied and implemented in control systems such as prosthetics and robotics. This technology allows users to control any mechatronics device with small muscle movements without physically straining the body. Electromyography is used where partial paralysis has occurred; in other words, for EMG to be used, the body must have some muscles which can perform normal functioning of contraction and relaxation. Prosthetic devices such as artificial limbs, prosthetic hands, EMG based exoskeletons, wheelchairs, robotic hands are a few examples of EMG based HMIs [19].

The two types of EMG recording techniques include the Surface EMG and the intramuscular EMG; both of which use devices called electrodes. Electrodes are attached on the surface of the body part where the respective muscle group of interest is present. It is ideal to wipe down the body where the sensors will be placed before placing the sensors with alcohol to increase the quality of contact and better conductivity. It is also advised not to apply the sensors to oily skin, skin with make-up, dry skin, scar tissues, or hairy skin.

Yavendra Singh et al used BIOKIT Datascope signal acquisition device to acquire surface EMG signals from the arm and interpreted the grip/lift operation using time domain features for the development of a prosthetic arm [20]. In another study, Ananda et al designed a hand gesture based controlled wheelchair using muscles of forearm to operate the direction and movement of

the wheelchair. The sensors were placed on the forearm and the EMG signals were extracted form Extensor Carpi Radialis and Flexor Carpi Radialis muscles [21].

Chaiko Mokri et al used quadriceps, vastus medialis, vastis lateralis and rectus femoris muscles to design a knee rehabilitation device and to estimate muscle force. They applied different machine learning classifiers like Support vector machine, Support Vector regression and random forest to check the accuracy of the device. Moreover, they devised a genetic algorithm for parameter optimization which resulted in 98.67% accuracy for lower limb muscles [22].

P Geethanjali et al in their research designed a robotic hand using EMG signals collected from the forearm of the subject to study the different pattern recognition techniques and real time implementation [20].

In [21] A B Barreto et al studied the design and development of input device which could be used even by those individuals which suffer with severe motor disabilities. Their system utilized EMG signals from cranial muscles which were translated into controls for 2D movement of the cursor [21].

Pedro Nuno Fernandes et al in their research designed EMG based control using a Powered knee orthosis to provide assistive commands according to the motion intention of the user. In another research Minsang Seo et al designed a control system of a prosthetic hand for trans radial amputees using surface EMG signals [22].

1.2.2 Electroencephalography (EEG) controlled mobility assistive device

EEG and EOG on the other hand are used where complete body paralysis has occurred. Conditions like quadriplegia, paraplegia and tetraplegia affects the normal functioning of arms, legs and in cases like tetraplegia, both limbs are entirely affected [23]. In such cases where body's voluntary muscles are inactive, but brain and eye signals are working properly, EEG and EOG based assistive devices are beneficial [24].

Neurorehabilitation, often known as neuroscience-based rehabilitation, has gained more and more attention in recent years. Due to its noninvasive and straightforward assessment of human brain activity, electroencephalography (EEG) has been widely employed in clinical practice as a technique for the evaluation and treatment of rehabilitation. Due to its suitability for the measurement requirements of the field, which include straightforward, secure, and portable equipment, EEG is generally implemented in rehabilitation. The main purpose of EEG analysis in

the past has been to record brain activity as an electric field in conjunction with a phenomenon or while performing a task, and then to infer the source of that action from the distribution of that activity on the scalp. On the other hand, recent developments have resulted in the creation of a technique that can record variations in the strength of beats in a certain frequency range [25].

Electroencephalography is a noninvasive method of collecting brain signals using surface electrodes. Commonly 10-20 lead system is used to acquire EEG signals. The scalp is cleaned, and tiny electrodes are attached on the specific positions marked before the process is initiated. The electrodes measure the neuronal activity of brain i.e., the potential changes due to the ionic current [26]. EEG signal is classified into different bands like alpha, beta, delta, thelta and mu suppression. Each band represent different state of being like relaxed, active, focused etc.

EEG has proven to be a powerful component in the stroke diagnosis process. It is now routinely utilized to monitor the stroke rehabilitation process. Following a stroke, patients are advised to engage in physical activity to help in the recovery of impaired motor function.

Satakshi Singh et al in their study used EEG signals from the patients in the process of rehabilitation to track the improvement in their motor ability. They used Fugl-Meyer Assessment (FMA) for this study [27].

Imran Ali Mirza et al devised a mind-controlled wheelchair using an EEG headset. They used the Mind Wave Headset to record EEG signals which were then forwarded to the Brain computer interface of the wheelchair. The wheelchair's electronic assembly was coded to perform the basic movements of moving forward and left, right rotational movement [28]. The wheelchair's movement was solely dependent on the signals generated by the brain [28].

In [29], Madiha Tariq et al identified the potentials of EEG based BCI applications for locomotion and mobility rehabilitation. The EEG signals such as sensorimotor rhythms (SMR), event- related potentials (ERP) and Visual evoked potentials (VEP) are reviewed for user's mental tasks related to lower limb for BCI reliability.

In [30], Maged S. AL- Qureshi et al reviewed EEG-based control for upper and lower limb exoskeletons and prostheses. In their study they emphasized on rehabilitative devices which uses brain signals to develop human robot interfaces to estimate the motion intent.

In another study, Dennis J. McFarland et al used EEG signals to develop brain computer interface to provide an alternative communication channel for either completely paralyzed patients

or those having extreme motor damage. They analyzed EEG signals in real time and its conversion into device control [31].

However, low signal-to-noise ratio and short-term recording capabilities are typical disadvantages of EEG signals. As a result, inaccurate samples and irrelevant data could impair the functioning of the HMI. Therefore, Electrooculography (EOG) was then used by researchers since it has high signal to noise ratio.

1.2.3 Electrooculography (EOG) controlled mobility assistive devices

Many people affected by muscular and neurological disorders such as cerebral palsy lose their ability to control most of their muscles, however they retain the ability to control the movement of their eyes. Electrooculography (EOG) signals are generated between two parts of the eye i.e., retina and cornea present between the front and back of the eye. The difference in potential arises due to the electrically active nerves present in the retina as compared to the cornea. Electrodes placed near to the eye can detect these potential changes thereby leading to the development of EOG controlled rehabilitation aids. For such patients, HMIs are designed by chasing and distinguishing eye movements to make things more convenient for them.

The EOG signal has a specific pulse shape for both forward and backward eyeball movement. Depending on the angle at which the eyeball was moved, the EOG signal's amplitude varies. The voltage stays positive (or negative) when the eyeball is tilted to one side and goes back to zero when the eyeball is pointed straight ahead. Both the amplitude and duration of the pulse produced by leftward movement are almost identical to those produced by rightward movement. Even when the eyes are closed, the signal potential is unaffected [32].

Jason J et al created a robotic prosthetic eye system based on EOG. The true eye location was ascertained using the EOG approach, and a fake eye movement was controlled using the signal that was received. A reference electrode was placed on the midline of the forehead, and two tiny electrodes were utilised to capture contact points around the eyes. The AD card's 30Hz sample frequency was used to send the amplified potential difference between the two electrodes to the computer [33]. To determine the correlation between the sensor's output and the eye, a calibration curve was created. The artificial eyeball was controlled by the controller using the eye position signal after the calibration curve had been obtained, allowing the artificial eyeball to be oriented

similarly to the natural eye. A servomotor with prosthetic eyeballs placed on it was driven by a microprocessor utilising a pulse-modulated signal [33].

M. Trikha et al unveiled a novel method for classifying multiple channel EOG. The classifier used Deterministic Finite Automata as its foundation (DFA). The programme classified sixteen different EOG signals. To collect EOG signals, five silicon rubber electrodes with impedances under 10K were positioned around the eye. In contrast to Ag/AgCl electrodes, silicon rubber conducting electrodes can detect signals with extremely low amplitudes. The most noticeable frequency components, according to FFT analysis, are up to 40 Hz, whereas the highest frequency components are at 4 Hz. They employed four threshold values, which correspond to the High and Low threshold levels in the Horizontal and Vertical channel EOG signals, respectively: THH, THL, TVH, and TVL. The greatest positive/negative amplitude of the measured EOG signal was +/- 37.5% of these 23 Threshold values. The positive/negative region above/below the threshold values in the vertical/horizontal channel is represented by the symbols [CV+, CV-, CH+, and CH-]. Rest zone areas in vertical and horizontal channels are represented by the symbols [CV0, CH0]. These Signals served as the main input for the Peak Detection Deterministic Finite Automata (PDDFA), which recognizes positive and negative peaks as well as the rest zone in EOG [34].

A.P. Vinod, et al created an EOG-based typing system. It used eight different sorts of distinct EOG patterns on a virtual keyboard to form letters on the monitor. The amplitude and timing of the signal's negative and positive components were used to identify the EOG pattern. EOG signals were measured using five electrodes. On the right mastoid, the reference electrode was positioned. EOG signals were filtered, amplified, and converted from analogue to digital using a gUSB amplifier [35].

1.2.4 Computer Vision

Artificial intelligence of computer vision enables computers and systems to extract useful information from digital photos, videos, and other visual inputs and to conduct actions or offer recommendations in response to that information. If AI provides computers the ability to think, computer vision gives them the ability to see, observe, and comprehend. In contrast to human vision, which requires retinas, optic nerves, and a visual cortex, computer vision teaches robots to execute these tasks in much less time with the use of cameras, data, and algorithms.

1.2.5 How does computer vision work?

A large amount of data is required for computer vision. It repeatedly analyses the data until it can distinguish between things and recognize photos. For instance, a computer needs to be fed a huge amount of tyre photos and tyre-related things to be trained to detect automotive tyres. This is especially true of tyres without any flaws. To achieve this, deep learning, a subset of machine learning, and convolutional neural networks are used as two key technologies (CNN).

With the use of algorithmic models, a computer can learn how to understand the context of visual input using machine learning. The computer will "look" at the data and educate itself to distinguish between different images if enough data is sent through the model. Instead of needing to have an image recognition programme written for the computer, algorithms allow the machine to learn on its own.

A CNN assists a machine learning or deep learning model in "seeing" by breaking images down into pixels that are tagged or labelled. It utilizes the labels to perform convolutions (a mathematical procedure on two functions to produce a third function) and make predictions about what it is "seeing." The neural network performs convolutions and evaluates the precision of its predictions repeatedly until the predictions begin to come true. Once this happens, it will be able to recognize or perceive images equivalent to humans.

1.2.6 Gaze Tracking

In the scientific literature, gaze tracking is defined as the automatic detection of eye locations, their temporal consistency, and their mapping into a line of sight in the real world. With a remarkable and expanding range of application domains, this has recently become a very popular topic in the world of computer vision. Research organizations are working laboriously each day to identify novel techniques in making independent and smart assistive devices for the benefit of handicapped patients [36].

The technology that allows wide-ranging use of eye-tracking laws includes areas in the automotive industry, medical science, exhaustion simulation, cognitive tests, computer vision, behavior recognition, etc. Over a period, the importance of eye recognition and monitoring in industrial applications has increased significantly thus leading to robust and efficient designs which is a requirement of modern healthcare industry. Computer vision and image processing techniques are used by eye gaze tracking systems to determine where the user is looking. There

are several different types of gaze tracking systems, both invasive and non-invasive, utilising one or more cameras [11].

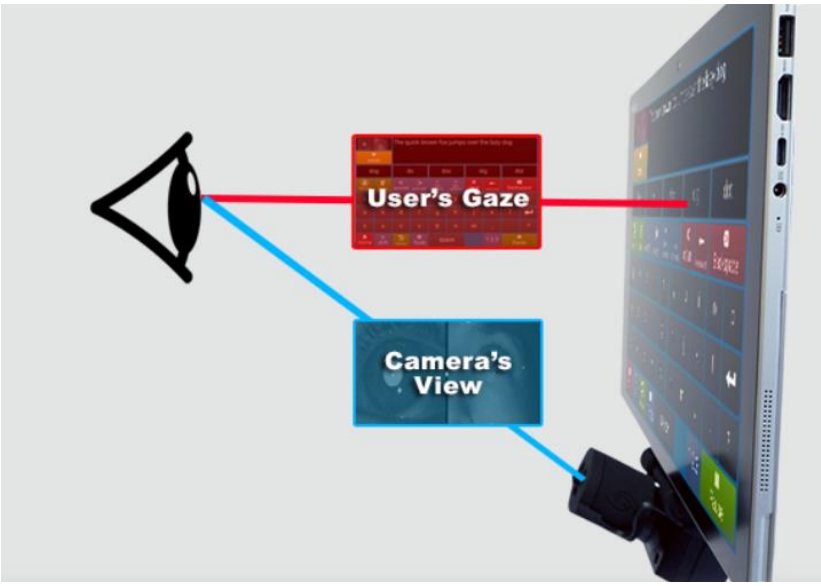


Figure 1.8: Eye gaze tracking system

1.2.7 Gaze Tracking and mobility assistive devices

Eye-tracking technology is a powerful tool for people with movement disabilities. An eye-tracking device combined with an assistive robot can improve user interaction speed and comfort. As a result, researchers are interested in developing an eye-gaze system interface for people who are paralyzed or physically disabled [37].

In 2007, the first system for controlling an electric wheelchair with eye movement was introduced, which used a standard web camera mounted on a head-mounted display (HMD) to capture the user's face and send the video stream to a computer [pp]. The computer analyses the captured video, estimates the line-of-sight to determine where the user is looking, and commands the electric wheelchair to move to a desired location.

In [38], the researcher created a wheelchair that is controlled by eye-tracking. A PID controller is used to control the wheelchair's DC motor after the camera captures the user's image and detects the eye position using a Laplacian edge detection technique and segmentation

algorithm. The detection accuracy was 90 percent. However, calibration time is not calculated, and the Laplacian technique is too sensitive to noise. Because the Haar cascade algorithm is based on edge or line detection, it is appropriate for face and eye detection but not for detecting eye movement.

The authors of [39] postulate a system that allows a person to control a wheelchair using their gaze. The system includes a wheelchair, eye-tracking glasses, and a depth camera, as well as Ambient Space Engineering, Portable Computer Sit Flexible Stand for Maximum Comfort, and a safety off switch to turn the system off when necessary. The CNN algorithm is used by the author to capture the eye and calculate its direction. Their study's accuracy was 92 percent, and the calibration time was 36 seconds. Because it is complex, the CNN algorithm is not recommended for face and eye detection. Furthermore, the calibration time is 36 seconds according to the results

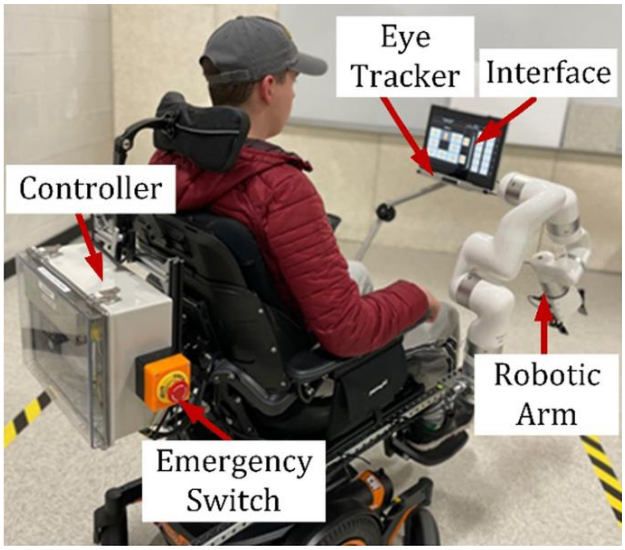


Figure 1.9: Gaze tracking for wheelchair users

1.2.8 Gaze tracking and computer vision

Hadish Habte Tesfamikael et al in their research paper discussed about an electric wheelchair based on eye tracking. They focused on cost effectiveness and enhancemnet in the system’s performance by performing edge detection to identify the eye pupil position in the face. Additionally, they improved the controller's validation parameters, which produced an accuracy of 90% [40].

S.R. Rupanagudi et al in their research [41] designed an eye gazed recognition system for motor neuron disease patients. They used oculographic movements of the subjects to train a model using Otsu algorithm which achieved the accuracy of 95%. They also compared it with Viola algorithm which resulted in increased speed as well.

1.2.9 Computer Vision Vs EOG

EOG was best chosen method for rehabilitative research area until a new field of computer vision was identified. Computer vision uses eye gaze tracking to develop smart assistive devices. For EOG data to be recorded, it is necessary for the user to wear the electrodes all the time which is inconvenient and irritating as well. For instance, for a person to drive the wheelchair using EOG method he would have to always wear the electrodes. This causes agitation especially when the leads can get tangled and might result in another damage.

On the contrary, for the computer vision technique, only a webcam is required which can be mounted anywhere on the wheelchair. It is a convenient, easy, and portable technique which nowadays researches more interested in instead of EOG system.

A Picot et al in their research of testing drowsiness also focused on using a camera instead of EOG electrode system reporting the same issue of inconvenience of attaching electrodes all the time [42].

1.2.10 Object detection, region proposal and Convolutional Neural Network

In recent years, the interdisciplinary field of computer vision has experienced tremendous growth (since CNN). Object detection is a crucial component of computer vision. Pose estimation, vehicle detection, surveillance, etc. are all made easier by object detection. In contrast to classification algorithms, object detection algorithms attempt to locate the object of interest within the image by attempting to construct a bounding box around it. An immature method in solving problems with different detectable objects of interest is to use CNN to classify the objects within that region. This is because the objects of interest might have different spatial locations within the image and different aspect ratios. Consequently, numerous regions must be selected which could computationally blow up the system. Therefore, special algorithms like R-CNN, YOLO, faster RCNN have been designed to speed up the computational process [50].

The success of region proposal approaches and region-based convolutional neural

networks (RCNNs) has propelled recent developments in object detection. The cost of region based CNNs has been significantly lowered even though they were computationally expensive when they were first developed in due to the sharing of convolutions between proposals. The most recent iteration, Fast R-CNN, uses very deep networks to attain near real-time speeds while ignoring the time required for region suggestions. Modern detection systems' test-time computational bottleneck is now proposals [51].

1.3 Objective

The study in this research focuses on

1. Application of industry standard EOG classification machine learning algorithms compared with computer vision deep learning method.
2. Effectiveness of new deep learning methods for image classification compared to traditional deep learning and machine learning method.
3. Comparison of EOG based method and Computer vision method to detect which of the two methods shows results with greater accuracy for driving any mobility assistive device.

1.4 Areas of application

Current study targets the following areas of application:

1. Human machine interface
2. Biomedical industry
3. Rehabilitation industry
4. Artificial Intelligence

1.5 Thesis Overview

In this thesis, Chapter 1 is introduction about human machine interface and different data acquisition techniques using biomedical signals to operate any mobility assistive device. Chapter 2 covers the literature review regarding the problem statement, and its provided solutions in the past. Chapter 3 provides with the methodology proposed in this study and how it is performed step by step. Chapter 4 is about results from the proposed methodology. Chapter 5 is about conclusions that has been drawn from the whole study and imparts future work.

CHAPTER 2: LITERATURE REVIEW

2.1 Principle of Electrooculography

The continuous electric potential field that exists in the eye has very little to do with light stimulation. Even with the eyes closed or in utter darkness, this field is detectable. A fixed dipole with a positive pole at the cornea and a negative pole at the retina can be used to represent this continuous electric potential. This corneoretinal potential has a value between 0.4 and 1.0 mV. This potential is brought about by increased metabolic activity in the retina, not by excitable tissue. The polarity of this potential difference is opposite in invertebrates than it is invertebrates, such as humans [43].

The direction of gaze, which may be determined by surface electrodes attached on the skin surrounding the eyes, rotates with the corneoretinal potential because it is nearly aligned with the optic axis. A signal is detected at two periorbital surface electrodes based on the rotation of the eye and the corneoretinal potential. The signal is called an electrooculogram (EOG).

2.2 Anatomy of an Eye

The eye is one of the most intricate sense organs in the body and functions somewhat as an extension of the brain. It is the supreme sense organ among all the others due to its large number of sensory neurons, precise optics, and magnificent architecture. The distance between the two eyes in animals (which is around 6 cm in humans) aids in the formation of a separate image by each eye, which is then processed by the brain through superimposition of the two images, giving the surroundings a sense of depth and three-dimensionality. The shape and structure of the eye also aid in determining the light source's distance [44].

The cornea, which forms a protective covering for the iris and the pupil and permits light passage into the eye, is the most transparent portion of the eye in the direction that light enters the visual system. The conjunctiva, a mucous-filled membrane that also covers the inner surface of the eyelids, conceals the visible portion of the eye. The eye receives moisture from it. The sclera is the name for the eyeball's outside layer, which is white in color. It creates a shield over the eye and conceals the optic nerves that are located behind the organ. The anterior chamber, a hollow area within the cornea, is filled with a transparent liquid known as aqueous humor [44].

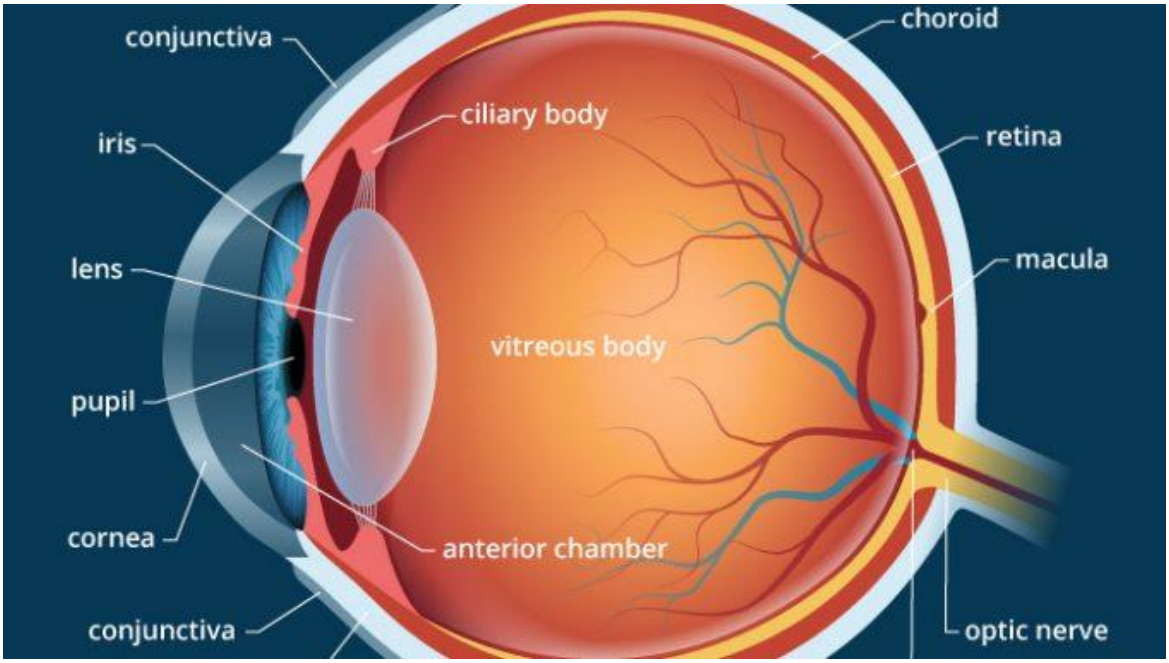


Figure 3.1: Anatomy of an Eye. [45]

The nutrients in this fluid aid in the survival of the cornea and lens. Fresh nutrition media are continuously injected into the system, replenishing the aqueous humour. When this fluid cannot drain properly due to glaucoma, intraocular pressure builds up, causing visual loss. The pupil, which creates the dark circle in the center of the eye, is located underneath the anterior chamber. Its job is to control how much light the retina receives. The colored area of the eye that surrounds the pupil, known as the iris, helps with this function. The iris spreads out to create the choroid, which is composed of several blood capillaries that provide the retina's cells nutrition and is located between the retina and the vitreous body. The eye's lens, located below the iris, directs incoming light toward the retina. The lens is a flexible, crystalline component of the eye that can change in size depending on how much focusing is needed, which in turn relies on how far away the object is (a characteristic known as "accommodation"). The lens needs to be thicker as the object gets farther away. The ciliary body, which is responsible for accommodating, producing aqueous humour, and positioning the lens, as well as a group of fibers known as the zonules of Zinn, which keep the lens securely in place, control the function and location of the lens. The posterior chamber is a further tiny hollow region located between the lens and the iris. The vitreous body, which is filled with a jelly-like substance called vitreous humour, is located behind the lens

and the retina. Prior to reaching the retina, this jelly aids in the refraction of light. After passing through the cornea, pupil, lens, and vitreous humour, light ultimately reaches the retina, which is made up of millions of photosensitive cells (sometimes referred to as rods and cones) and photoreceptive nerve cells that transmit visual signals to the optic center of the brain. The macula, an extremely dense area at the center of the retina is a pigmented yellow area that has the greatest number of cone cells and is hence in charge for vision with great resolution. The optic nerve head, also referred to as the optic disc, which constitutes the blind spot of the eye, is where blood capillaries and optic nerves congregate. The brain receives the nerves and capillaries together, where the photoreceptor signals are processed, and an image is conceived. The location and movement of the eye, governed by the numerous muscle systems connected to the eye, are necessary for proper ocular function [46].

2.3 Types of Muscles in the eye

Two pairs of rectus muscles and one pair of oblique muscles work together to govern the eye's movement. These muscles give the eyes orientation, giving the eye a total of six degrees of freedom to move in any direction. The lateral rectus and the medial rectus, whose roles are to abduct the eyes toward and away from the nose respectively, are responsible for controlling the eye's single movement in the horizontal plane. The other four muscles work in the vertical plane. The superior and inferior rectus lift and lower the eye, respectively, while the superior and inferior oblique muscles govern the eye's intorsion and extortion. The optic nerves in the back of the eye that lead to the brain are protected by these muscles as well [47].

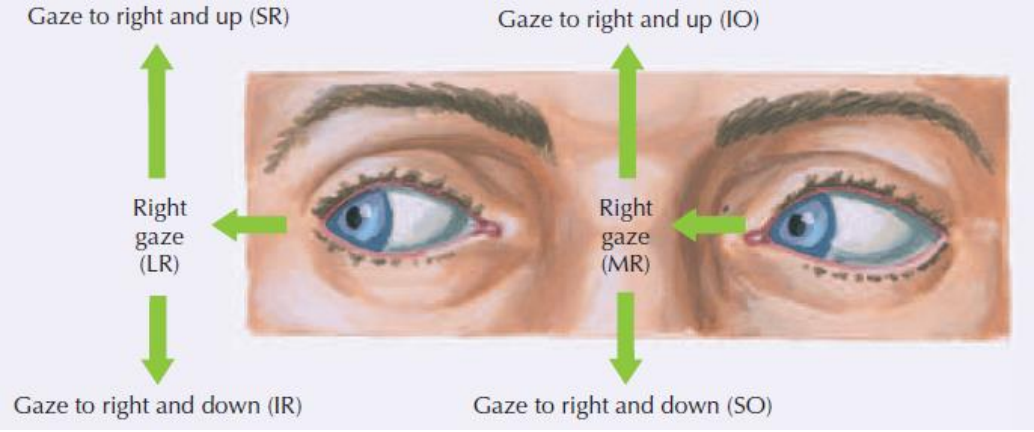


Figure 2.2: Muscles involved in eye movement [48]

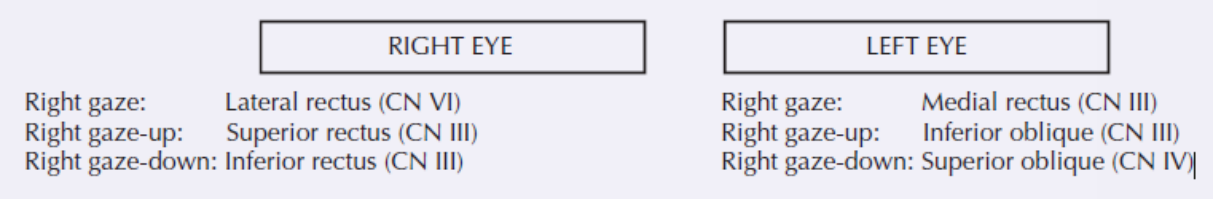


Figure 2.3: Cranial nerves involved in the movement of eye muscles [49]

CHAPTER 3: METHODOLOGY

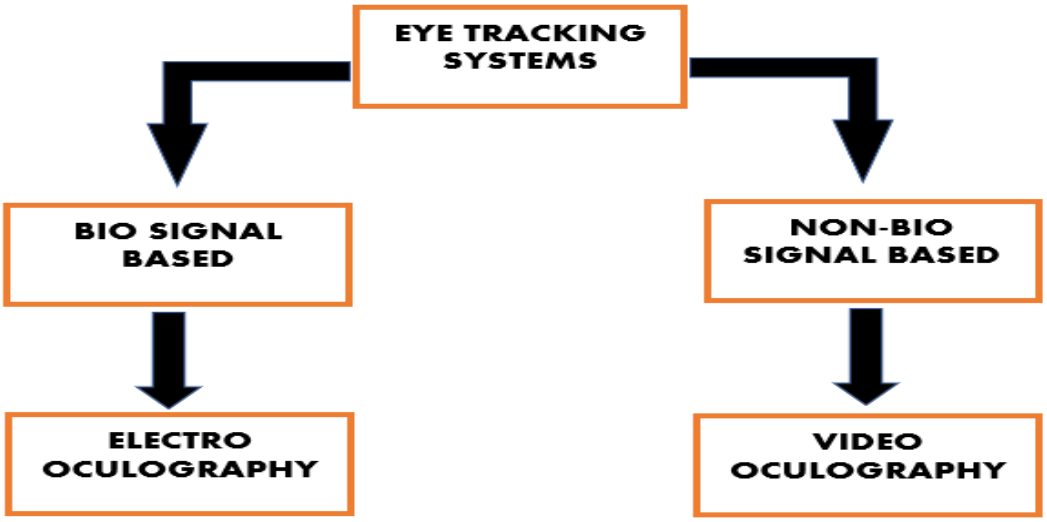
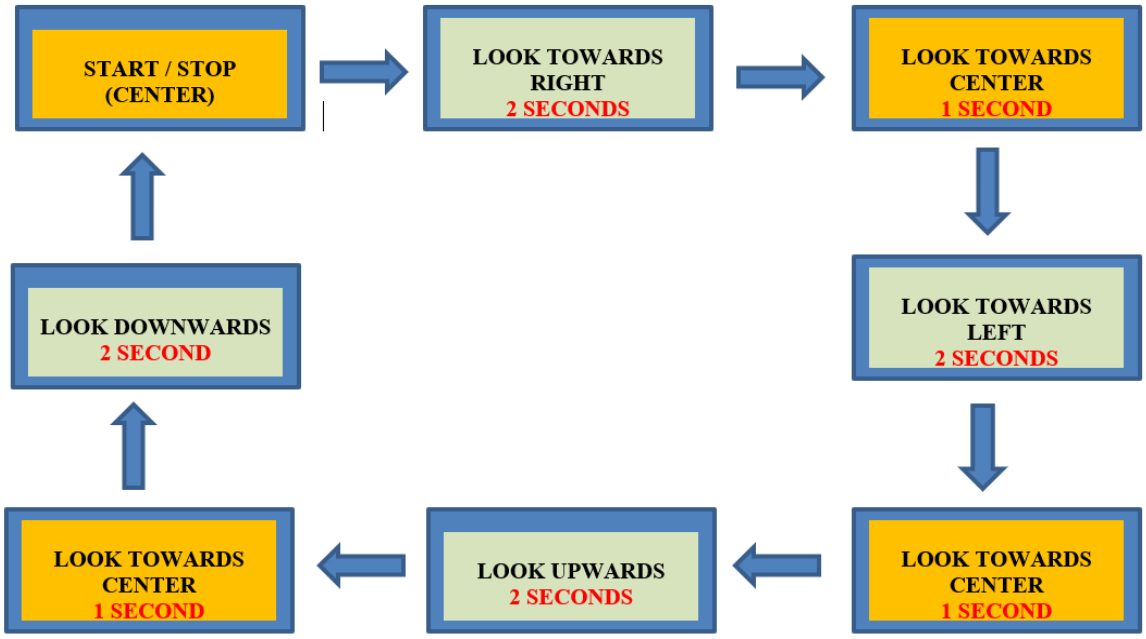


Figure 3.1: Eye tracking techniques

Dataset acquired for this study was in the form of: (I) EOG signals and (II) videos of eye gesture. Subjects were clearly explained about the study and a consent form was signed by each of them. The willing candidates were selected as subjects and the study was performed on 20 subjects. The data acquisition protocol is shown in the Figure 3.2.

This data was collected at times of COVID 19 pandemic which was why subjects were forced to wear masks. Due to this limitation, dataset of videos was only collected keeping the eyes as the focal point instead of full face thereby using eye detection algorithms instead of face detecting algorithms like HOUGH transform, HAAR Cascade, Viola jones technique etc.

3.1 Data Acquisition Protocol



. **Figure 3.2:** Data Acquisition protocol

3.1.1 Eye gaze recording using camera

A webcam with 1080P, 30fps was used to capture videos of eyeball moving in different directions. Each subject was asked to sit in front of the web cam mounted at a fixed distance. Instructions were given to move the eye keeping their head in straight direction. The movements were in the order as shown in the figures below. Subjects were told to look at the extremities for 2 seconds while only 1 second was defined for the rest position i.e., center. Each repetition lasted for 12 seconds which began from looking at the right position and ended at the double blink of the eye. The protocol of eye directions is shown in the figure 3.1. Subjects were allowed to have rest if they intend by pausing the video after every repetition so that they do not tire themselves. However, few candidates who wore glasses had some difficulty in keeping their eye fixed at

extreme positions as they faced trouble focusing, headaches, and eye strains, however, this ratio was very small.

3.1.2 EOG Recording using BIOPAC

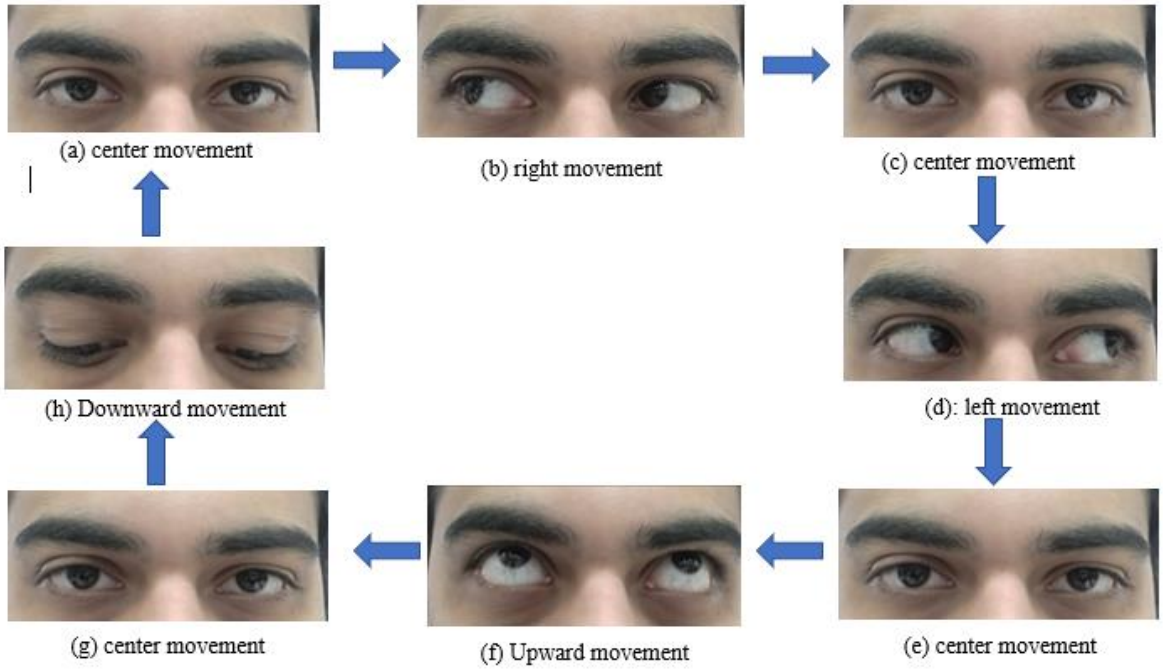


Figure 3.3 (a-h): Eye gaze movement in different directions

EOG signals were acquired by BIOPAC device. Surface gel electrodes were positioned at a specific location around the eye in a manner that both vertical and horizontal movements could be recorded. The placement of electrodes is shown in figure 6. Electrodes were placed after carefully wiping the regions with alcohol swabs. For recording horizontal movements, they were placed between the corner of the eye and the hairline of the temporal region while for the vertical recording they were placed above and below the eye keeping the center of the eyeball as a vertical axis. The ground electrodes were attached on the center of the forehead. Dual channels were used for the recording, each for vertical and horizontal signal. As shown in the figure 3.3 below, 2 red, black and white leads are attached on the subject.

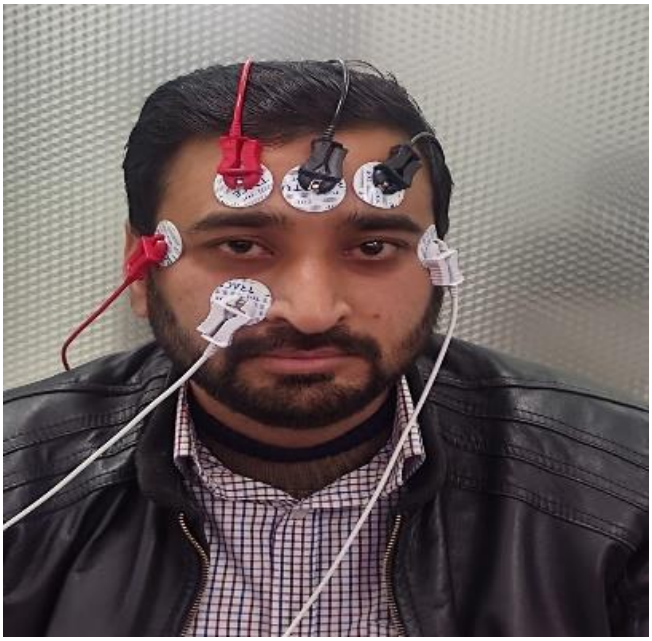


Figure 3.4: Attachment of EOG electrodes

Figure 3.5: BIOPAC System connection

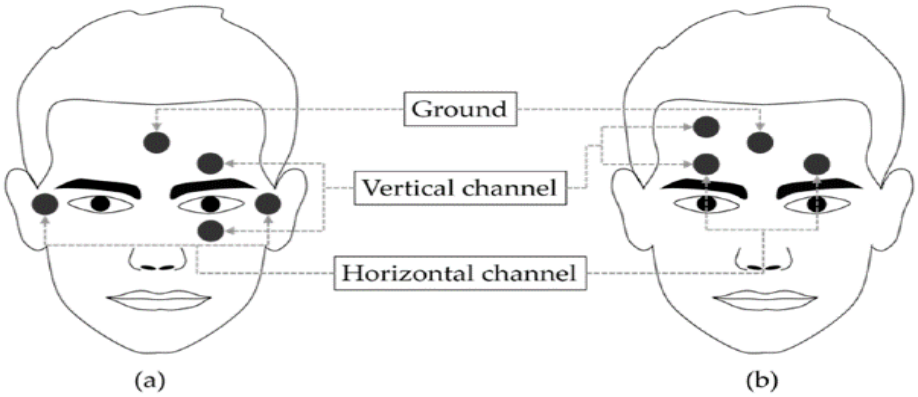


Figure 3.6 (a-b): Facial positions for attachment of EOG electrodes [52]

Red leads represent positive, white represent negative and black represent ground. Red and white leads attached on the side of the eyes are for horizontal recording while those attached above and below the right eye are for vertical recording. Subjects were asked to perform similar movement of eyes as in the previous experiment of eye gesture. They were instructed to move their eyes in the similar order as in figure 3.1 i.e., *Right – center, Left – center, Up – center, Down – center*

Table 3.1: Parameters of EOG dataset

Total sessions	20
Approximate time of each session	1 min
Repetitions in each session	5
Time of each repetition	12 seconds
Total subjects	20
Cut off frequency of low pass filter	35 Hz
Cut off frequency of high pass filter	0.5 Hz
Channel 1	ECG
Channel 2	Horizontal movement EOG
Channel 3	Vertical movement EOG

Time duration for contraction and rest was same as that of video dataset i.e., 2 second of contraction and 1 second of rest. Sampling frequency was set to 1000 Hz and bandpass filter was set with 0.05 Hz lowpass cut off frequency and 35 Hz high pass cut off frequency. The blue signal as shown in figure 3.6 represents data of up and down movement of eyeball while green signal represents righ t and left movement of the eyeball.

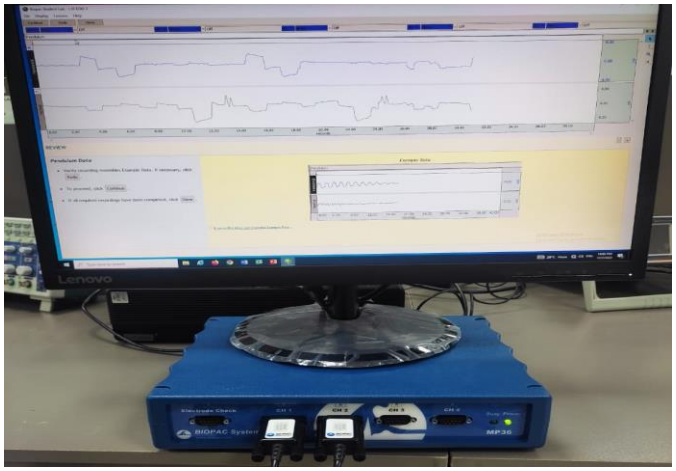


Figure 4.7: BIOPAC for EOG data acquisition

3.1.3 EOG Signal interpretation

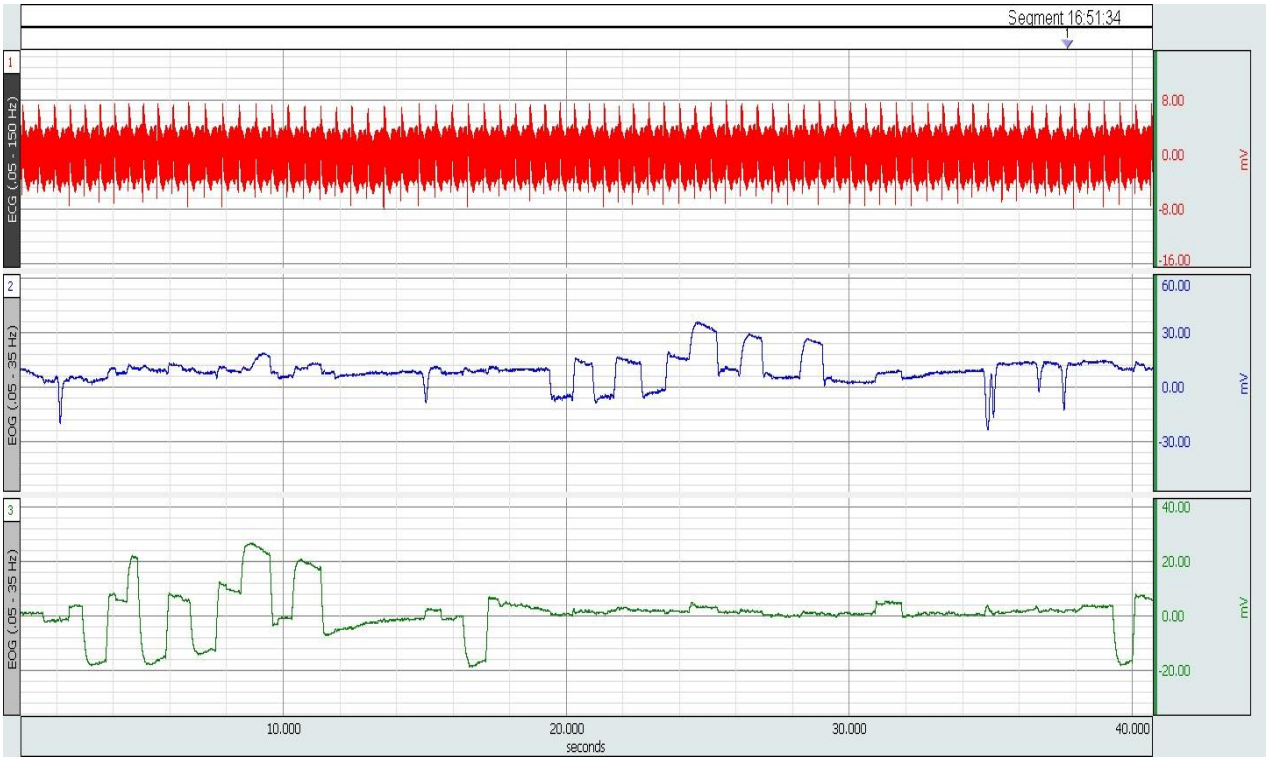


Figure 3.8: Raw EOG signal of vertical and horizontal movement acquired using BIOPAC

The figure shown above (3.6) is a graphical representation of an EOG signal. (Channel 1 shows ECG waveform which was acquired along with the EOG dataset for a different study which has not been incorporated in this study). Channel 2 shows movement of eye in upward and downward direction and Channel 3 shows movement of eye in right and left direction.

The signal was acquired against time axis which is displayed in x axis and the y axis displays the amplitude of the signal. EOG signal shows peaks when eye is moved to extreme position and straight line when looking at the center. The latter represents rest phase while the former represents contraction phase. The subject was told to look at the directions announced to

them. Each session was of 5-minute duration which include 5 repetitions of twelve seconds. Each repetition began with looking towards right and ended on double blink.

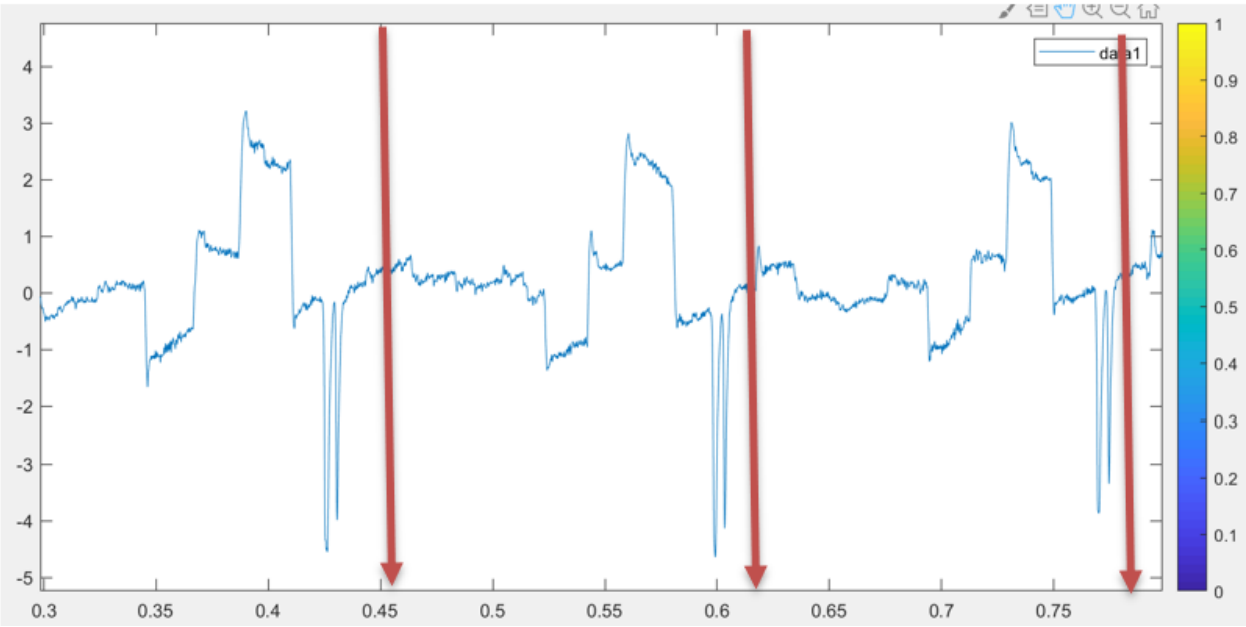


Figure 3.9: Vertical movement of eye from channel 2 showing 3 repetitions

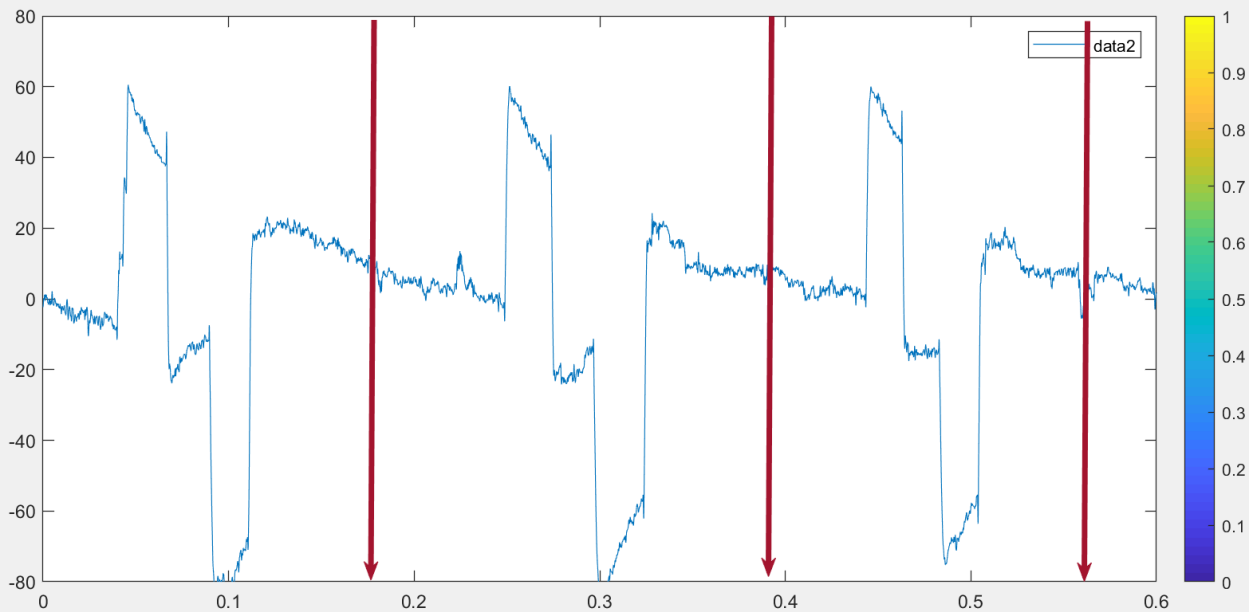


Figure 3.10: Horizontal movement of eye from channel 3 showing 3 repetitions

3.2 Data Processing of EOG signal

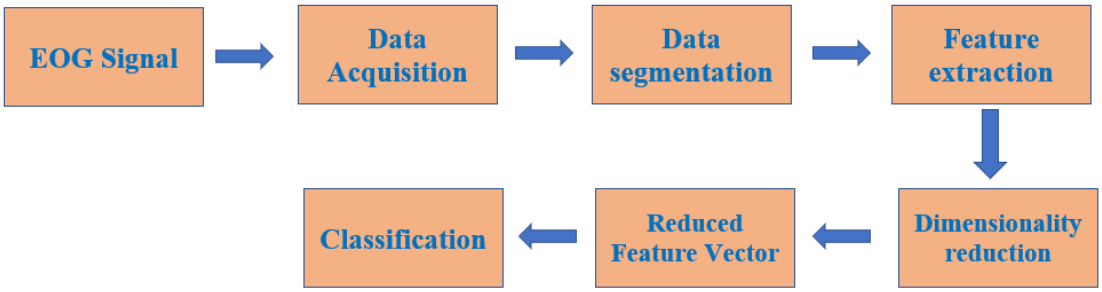


Figure 3-11: Flow chart of Data Processing

Data was collected in "acq" form and was converted into "txt, csv" format to be processed further. Data classification was performed in PYTHON software on JUPYTER notebook. Eleven different machine learning classifiers were tested to check which of it attains maximum average accuracy using five-fold cross validation. Code for data classification is written in APENDIX A.

4.2.1 Data Segmentation

The entire signal length of EOG data was about 20,000 samples. It was split into segments of 200 seconds for each movement. The segmentation was decided by taking the product of sampling frequency and the length of one repetition. Sampling frequency of the signal was 1000 Hz.

4.2.2 Feature selection

For the training of data, it was necessary to extract some features on which the classification will be decided. By skimming the literature regarding feature selection of EOG dataset, it was decided to use mean velocity of the eye movement in horizontal and vertical positions, peak velocity of eye in horizontal and vertical positions, PSD ratio between low and high frequency on horizontal channel, PSD ratio between low and high frequency on vertical channel, Number of threshold crossing value, maximum peak and valley amplitude values, area under the curve value and EOG variance derived from both vertical and horizontal signals.

3.2.3 Dimensionality reduction

Principal component analysis (PCA) was applied on the feature set to select the best features. Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.

3.2.4 Reduced Feature vector

After dimensionality reduction, it was decided that the following features will be considered for the data training to classify the motions more precisely**.**

Table 3.2: Reduced feature set for data classification

Feature	**Abbreviation**
Mean velocity of eye	MVE
Area under the curve	Area
EOG variance for vertical channel	V.Var
EOG variance for horizontal channel	H.Var
Peak to peak	ptp
SEM proportion	SEM

3.2.5 Classification

A classifier is trained on instances with labels it can see in the training set and then used to forecast labels for examples with labels it cannot see in the test set. The accuracy of the classifier, or the percentage of instances for which the prediction was accurate, can then be calculated by comparing the predicted labels to the actual labels.

Table 3.3: List of labels and associated gesture

Class Label	Gesture associated
r	Right
l	Left
u	Upward
d	Downward
c	Center

1. K Nearest Neighbor
2. C- Support vector Machine
3. NU- Support Vector Machine
4. Decision Tree
5. Random Forest
6. AdaBoost
7. Gradient Boosting
8. Gaussian Naïve Bayes
9. Linear Discriminant Analysis
10. Quadratic Discriminant Analysis
11. Logistic Regression

a) K Nearest Neighbor

It is a simple and easy to implement algorithm which is used to solve classification and regression problems for supervised dataset. As the name defines, KNN considers its neighbors to predict the class. KNN makes high accurate predictions but its speed decreases as the size of the dataset increases. It is most useful when the labelled data is impossible to obtain or is too expensive [54]. KNN uses feature similarity to predict the values of new data points [53] i.e., it stores all the given cases and classifies the new case based on similarity measure. The parameter K in KNN denotes the number of proximal neighbors to involve in the majority voting process.

It is calculated using the distance formula: $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

b) C- Support Vector Machine and Nu Support Vector Machine

SVMs methods are used for classification, regression, and outliers detection. They fall under the branch of supervised learning methods. The two topologies of SVM as mentioned in the headings have similar methods but vary in mathematical formulation and accept slightly different sets of parameters. [53] SVM techniques use super-finite degrees of polarity to categorize data and train models, resulting in a 3-dimensional classification model that extends beyond X/Y predictive axes. The SVM selects the hyperplane that perfectly separates (distinguishes) the two objects. This is merely a straight line in two dimensions. The best hyperplane for maximizing machine learning model training is the one with the greatest distance between each object.

c) Decision Tree

These classifiers capture descriptive decision-making knowledge from the given data. They can be produced from training sets [53]. Decision Tree classifier falls under the branch of supervised learning algorithms and can be implemented for queries of regression and classification. The aim of this classifier is to give output based on decision rules deduced from training data [53]. A decision tree uses a tree-like structure to develop classification or regression models. It employs a collection of if-then rules that are exhaustive and mutually exclusive for classification. Using one training set of data at a time, the rules are successively learned. The tuples covered by the rules are eliminated after each time a rule is learned. On the practice set, this procedure is continued until a termination requirement is satisfied. decision tree uses a tree-like structure to develop classification or regression models.

d) Random Forest

Random forest classifiers are included in the broad category of ensemble-based learning techniques [55]. They are easy to set up, operate quickly, and have had great success across a wide range of domains. Each individual tree shows a class prediction and the class with the most polls becomes the prediction value of the model [56]. The greater the number of trees is the higher the accuracy is. Random forest is a versatile, user-friendly machine learning technique that provides excellent results most of the time even without hyperparameter adjustment. Because of its

simplicity and diversity, it is also one of the most popular algorithms which can be used for both classification and regression tasks.

e) Adaptive boosting

Boosting is an ensemble technique that aims to combine several weak classifiers into one strong classifier. To achieve this, a model is first constructed using the training data, and a second model is then developed to fix the flaws in the first model. The training set is predicted exactly, or a predetermined number of models are added, depending on which comes first.

The first truly successful boosting technique for binary classification was called AdaBoost. AdaBoost is the foundation for several current boosting techniques, most notably stochastic gradient boosting machines.

Short decision trees are utilised with AdaBoost. Following the initial tree's creation, the effectiveness of the tree on each training instance is utilised to determine how much emphasis the following tree should place on each training instance. Training data which is difficult to determine is assigned a greater weight value, while the simple to predict are assigned less weight. Each model updates the weights on the training instances that have an impact on how well the subsequent tree in the sequence learns. Models are produced progressively, one after the other. Following the construction of all the trees, predictions are made using fresh data, and the performance of each tree is weighted based on how well it performed using training data. However, clean data without any outlier is the key requirement of the algorithm.

f) Gradient Boosting

One of the most potent algorithms used in machine learning is the gradient boosting technique. Bias error and variance error are the two basic categories into which errors in machine learning systems can be grouped. As one of the boosting strategies, gradient boosting is used to reduce the model's bias error. One kind of machine learning boosting is gradient boosting. It is predicated on the hunch that when prior models are coupled with the best possible upcoming model, the overall prediction error is minimised. To reduce mistake, the crucial concept is to set the desired results for this subsequent model.

g) Gaussian Naïve Bayes

Naive Bayes is a straightforward yet unexpectedly effective method for predictive modelling. The model is made up of two different kinds of probabilities, both of which can be calculated from your training data: 1) the probability of each class, and 2) the conditional probability for each class given each x value. Once determined, the probability model can be applied to Bayes Theorem to produce predictions for new data. It is typical to assume that your data will have a Gaussian distribution (bell curve), which will make it simple for you to estimate these probabilities, when your data are real-valued. Assuming that each input variable is independent is what gives Bayes its reputation of being naive. Although the technique is particularly effective on a wide variety of complex situations, this is a significant assumption and unrealistic for real data.

h) Linear Discriminant Analysis

The Linear Discriminant Analysis procedure is the preferred linear classification method for more than two classes. LDA is portrayed in a simple manner. It is made up of statistical characteristics of the data that were computed for each class which includes mean for each class and variance computed across all classes. For each class, a discriminate value is calculated, and the class with the highest value is used as the subject of the forecast. It is advisable to remove outliers from your data beforehand because the technique relies on the assumption that the data has a Gaussian distribution (bell curve). For problems involving categorization predictive modelling, it is a straightforward and effective approach.

i) Quadratic Discriminant Analysis

Quadratic Discriminant Analysis can learn quadratic boundaries which makes it more efficient and flexible to counter problems as compared to linear discriminant Analysis which only distinguishes clusters using linear boundaries. This classifier has closed form solutions which are easy to compute and have no hyperparameters to tune. Both LDA and QDA can be derived from simple probabilistic models where each class will have its own covariance matrix i.e., the covariance matrix is estimated separately for each class.

$$\delta_k(x) = -\tfrac{1}{2}\log|\Sigma_k| - \tfrac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log\pi_k$$

j) Logistic Regression

Another statistical method that machine learning has adopted is logistic regression. It is the preferred method for issues involving binary categorization (problems with two class values). Similar to linear regression, the aim of logistic regression is to determine the coefficient values that weight each input variable. In contrast to linear regression, a non-linear function known as the logistic function is used to change the estimate for the output. The logistic function, which resembles a large S, will change any value into the range between 0 and 1. This is helpful since we can use a strategy to predict a class value and apply it to the logistic function's output to snap values to 0 and 1 (for example, IF less than 0.5 then output 1). The predictions provided by logistic regression can also be used as the likelihood that a specific data instance belongs to class 0 or class 1 because of how the model is learned. This can be helpful in situations where you must provide more support for a prediction.

3.3 Data processing of Video set using Computer Vision.

Data classification code was developed using Google Colab as it has an inbuilt GPU to compute a huge dataset. Therefore, it allows the user to train the data in a very less time thereby increasing the processing speed. The libraries and software packages for iris detection were imported. The most important library for gaze detection is computer vision, CV2.

3.3.1 Classes and labels

Video data set was first converted into images by taking screenshot of every motion. Each motion was then appended in a separate column vector known as classes. This new data was then assigned with a label and class name.

Table 3.4: Class names and labels assigned for Computer Vision

3.3.2 RCNN, Faster RCNN, YOLO

Conventionally, object detection techniques have 3 major steps to follow. The initial step is to generate several region proposals which are dedicated as the candidates having objects within them. These regions can be 2000 or greater in number and can be selected using selective search or EdgeBoxes algorithms.

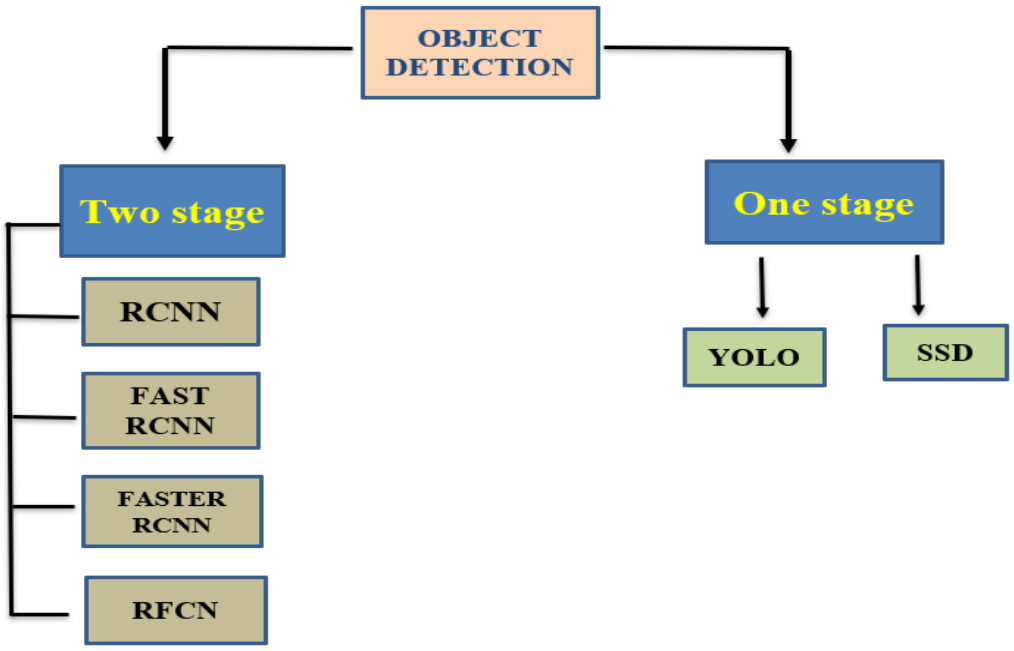


Figure 3.12: Flowchart of object detection algorithm

a. CNN

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning method that can take in an input image, give importance to distinct aspects and objects in the image (learnable weights and biases), and be able to distinguish between them. Comparing a ConvNet to other classification algorithms, the amount of pre-processing needed is significantly less. Contrary to hand-engineered filters used in more rudimentary techniques, ConvNets can learn these filters and properties with enough training. With pertinent filters, a ConvNet may effectively capture the spatial and temporal dependencies in a picture. Because there are fewer factors to consider and the

weights can be reused, the architecture provides a better fitting to the picture dataset. In other words, the network may be trained to better comprehend the level of complexity in the image.

The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction. This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets [57].

For feature extraction and classification, a CNN's functionality can be divided into two categories. The neural network extracts important features from the input image for use in classification during feature extraction. Convolution layers are used to convolve the input data, and pooling layers are used to down sample the input and avoid overfitting. The classification component consists of a fully connected layer that is then connected to the output layer, which has its own activation function and is responsible for making the final inference.

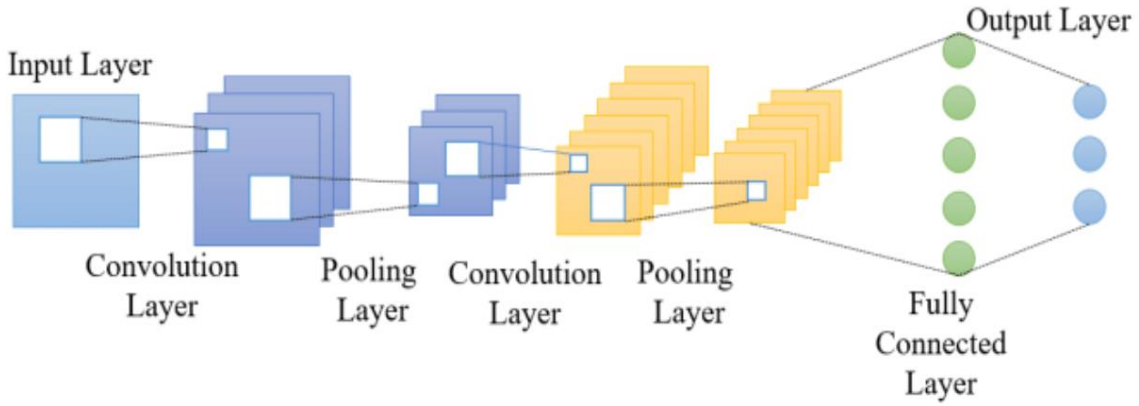


Figure 3.13: Architecture of Convolutional Neural Network [58]

b. RCNN

Region based CNN is the first approach towards designing a model to detect objects which extracts features using a pre trained CNN. It can detect 80 various types of objects in the images. The main advantage of RCNN is to extract the features based on a Convolutional Neural Network. However, some drawbacks of this model lead to the development of another algorithms. Since it is a multi-stage independent model, therefore it cannot be trained end to end. It also requires a large storage space on the disks to compute efficiently and thirdly, each region proposal is fed independently to the CNN for feature extraction which makes it impossible to run in real time [59].

c. Faster RCNN

Faster R-CNN is a deep convolutional network for object detection that presents to the user as a single, end-to-end, unified network. The network can predict the positions of various items rapidly and correctly. Object detection architecture known as Faster R-CNN was created by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. It was developed as an upgrade from the original R-CNN and Fast R-CNN and as a response to the complexity needed for precise localization in object identification challenges.

Most object identification techniques rely on a hunch or educated estimate as to where the target object might be. The accuracy of most object identification systems now heavily depends on how well their region proposal performs thus leading to the creation of region proposal algorithms. Despite this, it was eventually discovered that the region proposal technique, which applies a CNN to each object, might potentially make object recognition incredibly slow [60].

d. YOLO

A region proposal network is commonly used by well-known two-step algorithms like Fast-RCNN and Faster-RCNN to suggest regions of interest that may contain objects. A classifier divides the areas into classes using the output from the RPN. Even while approach produces precise object detection with a high mean Average Precision (mAP), it necessitates numerous iterations on the same image, which slows down the algorithm's detection speed and prevents real-time detection. Before YOLO, object detection algorithms would employ classifiers to conduct detection, but YOLO suggests using an end-to-end neural network that predicts bounding boxes and class probabilities simultaneously.

YOLO achieves cutting-edge results, outperforming previous real-time object detection algorithms by a significant margin by taking a fundamentally different approach to object recognition. The YOLO algorithm divides the image into N grids, each of which has an equal-sized SxS region. These N grids are each in charge of finding and locating the thing they contain. Accordingly, these grids forecast the object label, the likelihood that the object will be present in the cell, and B bounding box coordinates relative to their cell coordinates. As cells from the image handle both detection and recognition, this technique significantly reduces computation, but—Due to numerous cells predicting the same object with varying bounding box predictions, it generates a lot of duplicate predictions. Non-Maximal Suppression is used by YOLO to address

this problem.

3.3.3: Architecture of faster R-CNN

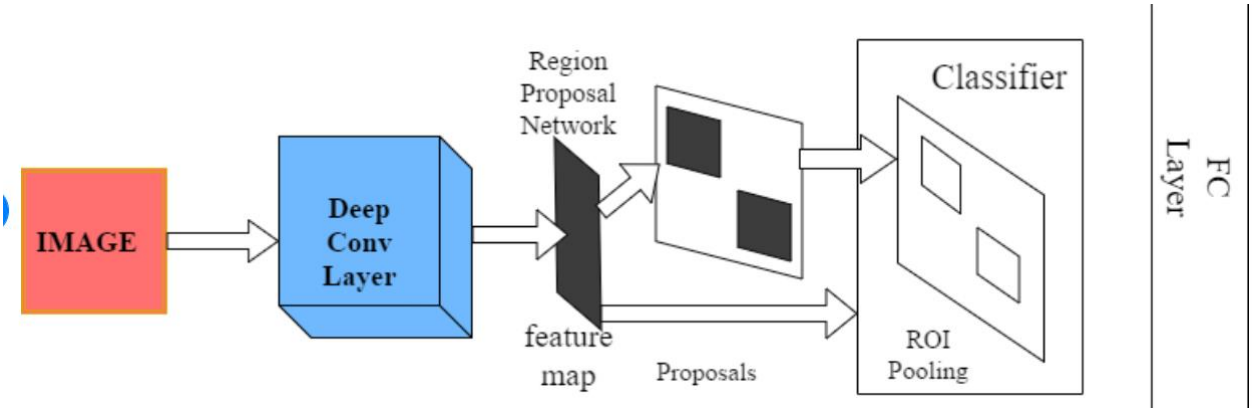


Figure 3.14: Architecture of faster R-CNN []

The faster R-CNN is composed of two parts:

a. A CNN that provides possible locations for the component to be recognised. This is the RPN. It provides instructions to the second Faster R-CNN component on where it assumes the object should be searched for. The RPN takes an image as input and returns rectangular objects that are its suggestions for the object's location. It also returns a value that represents its possibility of belonging to a specific class in comparison to the rest of the image. Search selective is used here to select sections with similar textures and colours and place them in separate boxes. The Softmax activation function is used to classify these choices, which are then output.

b. The Faster R-CNN object detector is the second component of the Faster R-CNN. It takes the feature maps produced by the initial CNN and applies ROI pooling to them.
ROI pooling extracts the regions of interest from the set of regions produced by the initial CNN. These ROI are then passed to a fully connected layer, where they are flattened before being passed to the output layer, where they are classified and given bounding boxes.

3.4 Graphical User Interface

The Graphical User Interface for testing the images for eye gaze detection was designed on Google collab using “PYTORCH”. The interface consists of an OPEN button which is connected at back end to the windows folder from which the files of images can be accessed. Another button is the “SAVE” button which saves the detected files. Figure 3.9 shows the screenshot of the interface.

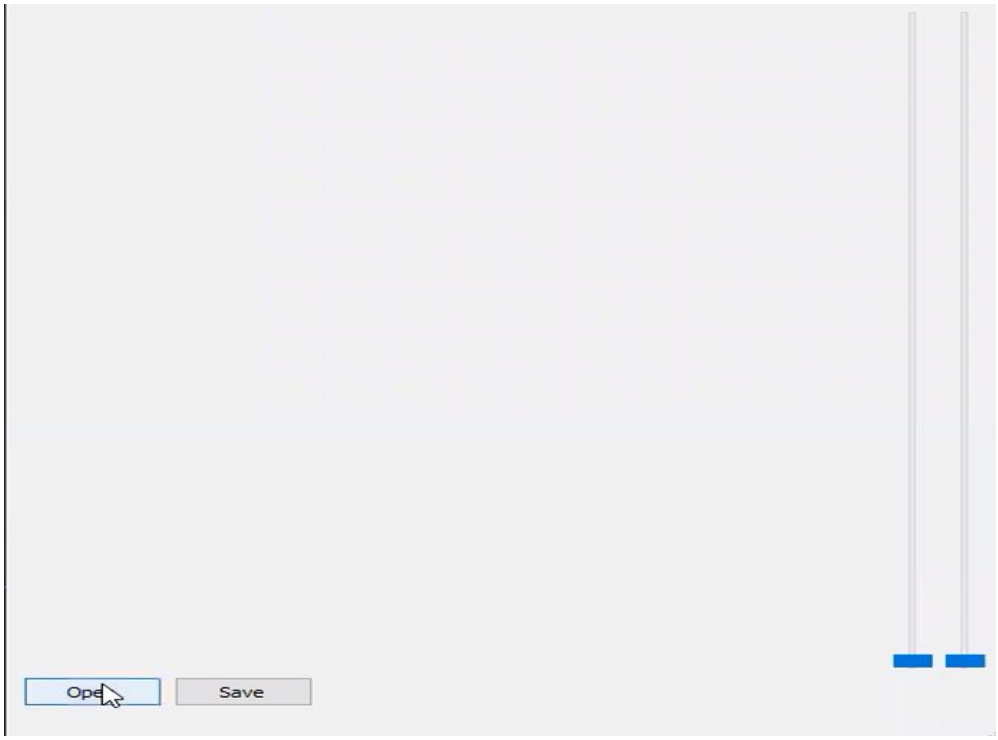


Figure 3.15: Graphical User Interface for eye detection

CHAPTER 4: RESULTS AND DISCUSSION

1.1.Results of EOG classification

The model was trained offline using 11 different classifiers. The results are shown in the tables below. The results predicted via machine learning are based on the confusion matrix which helps in displaying performance of the model. Table 1 shows the average accuracies achieved by the classifiers for EOG dataset as well as their F1 score while table 2 shows the precision and recall averaged over 5 folds for each classifier. 5-fold cross validation was used to evaluate the results for this study. Precision and recall are performance metrics used for pattern recognition and classification in machine learning.

1.1.1. Precision, Recall, F1 Score, Average Accuracy

a. **Precision: Precision is defined as** the ratio of True Positive with that of Total Number of data points predicted as Positive by the model. It helps in visualizing the reliability of the model to classify it as positive.

Formula: TP/ (TP+FP)

b. **Recall:** The recall is calculated as the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples. The *recall measures the model's ability to detect positive samples*. The higher the recall, the more positive samples detected.

Formula: TP/TP+FN

c. **F1 score:** F1 score combines precision and recall relative to a specific positive class. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

Formula: F1 = 2 * (precision * recall) / (precision + recall)

d. Cross validation: Cross-validation is a statistical method used to estimate the performance (or accuracy) of machine learning models. It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited.

Table 4-1: Classification accuracy and F1 score for EOG dataset

Classifier name and abbreviation		Average Accuracy	F1 Score
Nu-Support Vector Classifier	*Nu-SVM*	52.21 $\pm$ 12.76	51.15 $\pm$ 13.24
Decision Tree	*DT*	88.94 $\pm$ 13.82	89.12 $\pm$ 13.58
Random Forest	*RF*	86.81 $\pm$ 17.77	86.87 $\pm$ 17.69
Gradient Boosting	*GB*	87.66 $\pm$ 17.46	87.73 $\pm$ 17. 38

Table 4-2: Precision and recall averaged over 5 folds for each classifier

CLASSIFIER	AVERAGE PRECISION	AVERAGE RECALL
Nu-Support Vector Classifier	2.64	2.58
Decision Tree	4.47	4.45
Random Forest	4.38	4.36
Gradient Boosting	4.52	4.45

Table 4-1 shows that among all the classifiers tested, Decision Tree has the highest accuracy and F1 score i.e., 88.94 and 89.12 respectively. However, highest precision and recall is of gradient boosting i.e., 4.52 and 4.45 respectively as displayed in table 4-2. Therefore, for EOG signal having respective characteristics, Decision tree is the best-chosen classifier.

4.2 Results of Computer Vision

The Faster R-CNN model is an innovative method of using neural networks for object detection that has successfully preserved and, in some cases, improved the high level of accuracy of the standard Convolutional Neural Networks while also shortening the training period. The Faster R-CNN is simply an improvement on the Fast R-CNN, which is an advancement over the CNN. It was able to identify the movement of eyes in all four-direction using 50 epochs. Increasing number of epochs increases the accuracy of data classification.

Table 4-3: Table of Average precision and Intersection over Union

Area	IoU	Average Precision
ALL	**0.5**	**0.868**
SMALL	**0.7**	**0.526**
MEDIUM	**0.5:0.95**	**0.60**
LARGE	**0.5 : 0.95**	**0.69**

Mean Average Precision (mAP) is a metric used to evaluate object detection models such as Faster R-CNN.

Intersection over Union indicates the overlap of the predicted bounding box coordinates to the ground truth box. Higher IoU indicates the predicted bounding box coordinates closely resembles the ground truth box coordinates.

IoU threshold was kept 0.5.

The mean of average precision (AP) values is calculated over recall values from 0 to 1.

The figures below show the classification results of the subject. Figure 4.1 shows that the classifier has detected the center position of the pupil while figure 4.2 shows the left movement of the eye detected by the faster R-CNN.

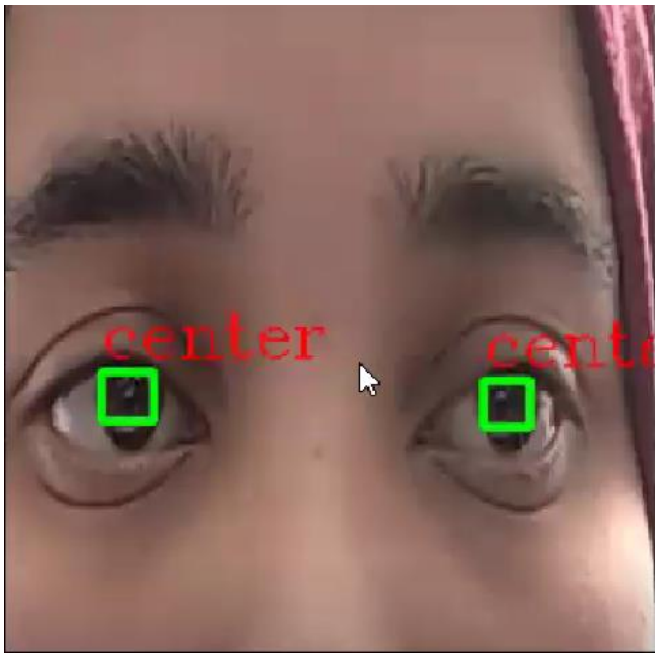


Figure 4.1: Algorithm detecting center eye movement

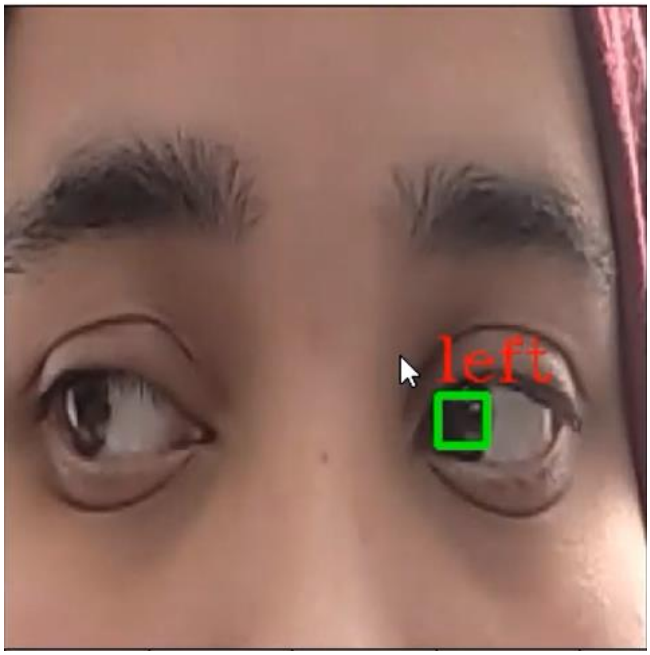


Figure 4.2: Algorithm detecting left eye movement

CHAPTER 5: CONCLUSION AND FUTURE WORKS

This research was completed to identify the application of industry standard EOG classification machine learning algorithms versus computer vision deep learning method. Furthermore, the effectiveness of new deep learning methods for image classification was checked against traditional deep learning and machine learning methods. It is devised from the study that faster RCNN algorithm achieves greater results with an accuracy of 77 %. Faster RCNN combined is the best approach for classifying dataset containing videos by frame-by-frame division.

For the EOG dataset, conventional machine learning classifiers were tested to identify the best classifier which could provide maximum accuracy for the movement of pupil. Among the tested classifiers, Decision tree showed the highest average accuracy and F1 score i.e., 88.94 and 89.12 respectively. The main industry applicable classifiers used in modern era are K Nearest Neighbor, Gradient boosting, Gaussian Naïve Byes, and Random Forest for the signals which are acquired against time series. Since EOG signals are plotted against time series, therefore among all the classifiers, Decision tree gave the best results.

Both systems mentioned in this study have their own limitations.

For EOG based system, the attachment of electrodes is a must requirement. This causes irritation to the user and sometimes generates motion artifacts which can be a source of hinderance for the motion of any HMI.

For computer vision-based system, camera is a must requirement. However, it can't be used in dark rooms, outdoor; during night times, wearing sunglasses and in similar other situations. For such situations, another alternative is an infrared camera, but prolonged usage of such camera can damage the eye.

Therefore, a hybrid system should be developed which involves both techniques i.e, EOG and a camera which can effectively drive any mobility assistive device.

In future. a device incorporating both EOG signals and Camera based eye gesture-controlled system for effectively driving any mobility assistive device should be designed which uses state of the art object detection technique known as "YOLO" (YOU ONLY LOOK ONCE) for iris detection and compare with the results of Faster RCNN, thereby choosing the best algorithm for the device.

REFERENCES

1. Jiboye, A. and Weir, R. (2005). A heuristic fuzzy logic approach to emg pattern recognition for multifunctional AnEMG-basedAssistiveOrthosisforUpperLimbRehabilitation 327 prosthesis control. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 13(3):280 –291.

2. Albert, S. and Kesselring, J. (2012). Neurorehabilitation of stroke. Journal of Neurology, 259:817 – 832. Andreasen, D. S., Allen, S., and Backus, D. (2005). Exoskeleton with EMG Based Active Assistance for Rehabilitation. In IEEE 9th International Conference on Rehabilitation Robotics, pages 333–336. blind review Ownpaper (2013). b.

3. Bohn, C. and Atherton, D. (1995). An Analysis Package Comparing PID Anti-Windup Strategies. IEEE Control Systems, 15:34–40. Brewer, B. R., McDowell, S. K., and Worthen-Chaudhari, L. C. (2007). Poststroke Upper Extremity Rehabilitation: A Review of Robotic Systems and Clinical Results. Topics in Stroke Rehabilitation, 14:22 – 44.

4. Burden, A. and Bartlett, R. (1999). Normalisation of EMG amplitude: an evaluation and comparison of old and new methods. Medical Engineering and Physics, 21(4):247 – 257. Deaton, C., Froelicher, E. S., Wu, L. H., Ho, C., Shishani, K., and Jaarsma, T. (2011). The Global Burden of Cardiovascular Disease. Journal of Cardiovascular Nursing, 26:5 – 14. DiCicco, M., Lucas, L., and Matsuoka, Y. (2004). Co

5. . P. Geethanjali and K. K. Ray, "A Low-Cost Real-Time Research Platform for EMG Pattern Recognition-Based Prosthetic Hand," in IEEE/ASME Transactions on Mechatronics, vol. 20, no. 4, pp. 1948-1955, Aug. 2015, doi: 10.1109/TMECH.2014.2360119.

6. E. Martin, “Breakthrough research on real-world driver behavior released,” National Highway Traffic Safety Administration, 2006.

7. D. F. Dinges and R. Grace, “Perclos: A valid psychophysiological measure of alertness as assessed by psychomotor vigilance,” Federal Highway Administration Office of motor carriers, Tech. Rep. MCRT-98- 006, 1998.

8. F. Friedrichs and B. Yang, “Drowsiness monitoring by steering and lane data based features under real driving conditions,” Signal Processing, pp. 209–213, 2010.

9. J. Berglund, “In-vehicle prediction of truck driver sleepiness: Steering related variables,” Ph.D. dissertation, Linkoping, 2007. ¨

10. A. Rechtschaffen and A. Kales, “A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects,” 1968.

11. L.-C. Shi and B.-L. Lu, “Dynamic clustering for vigilance analysis based on EEG,” in Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vancouver, BC Canada, pp. 54-57, 2008.

12. L.-C. Shi and B.-L. Lu, “EEG-based vigilance estimation using extreme learning machines,” Neurocomputing, vol. 102, pp. 135–143, 2013.

13. L.-C. Shi, H. Yu, and B.-L. Lu, “Semi-supervised clustering for vigilance analysis based on EEG,” in Proceedinds of International Joint Conference on Neural Networks, pp. 1518-1523, 2007.

14. L.-C. Shi and B.-L. Lu, “Off-line and on-line vigilance estimation based on linear dynamical system and manifold learning,” in Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Buenos Aires, Argentina, pp. 6587-6590, 2010.

15. J.-X. Ma, L.-C. Shi, and B.-L. Lu, “Vigilance estimation by using electrooculographic features,” in Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Buenos Aires, Argentina, pp. 6591-6594, 2010.

16. A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems, pp. 1106–1114, 2012.

17. H. P. Martinez, Y. Bengio, and G. N. Yannakakis, “Learning deep physiological models of affect,” IEEE Computational Intelligence Magazine, , vol. 8, no. 2, pp. 20–33, 2013.

18. H. Cecotti and A. Graser, “Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 33, no. 3, pp. 433-445, Mar. 2011.

19. J. Masci, U. Meier, D. Cires¸an, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in Proceedings of International Conference of Artificial Neural Networks and Machine Learning 2011. pp. 52-59, 2011.

20. J. Weston, F. Ratle, H. Mobahi, and R. Collobert, “Deep learning via semi-supervised embedding,” in Proceedings of International Conference of Machine Learning, pp. 1168-1175, 2008.

21. N. Jaitly and G. Hinton, “Learning a better representation of speech soundwaves using restricted boltzmann machines,” in Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing , Prague, Czech Republic, pp. 5884–5887, 2011.

22. Z.-P. Wei and B.-L. Lu, “Online vigilance analysis based on electrooculography,” in The 2012 International Joint Conference on Neural Networks (IJCNN). IEEE, pp. 1–7, 2012.

23. R. Schleicher, N. Galley, S. Briest, and L. Galley, “Blinks and saccades as indicators of fatigue in sleepiness warnings: looking tired?” Ergonomics, vol. 51, no. 7, pp. 982–1010, 2008.

24. E. Magosso, F. Provini, P. Montagna, and M. Ursino, “A wavelet based method for automatic detection of slow eye movements: A pilot study,” Medical engineering & physics, vol. 28, no. 9, pp. 860–875, 2006.

25. E. Magosso, M. Ursino, A. Zaniboni, F. Provini, and P. Montagna, “Visual and computer-based detection of slow eye movements in overnight and 24-h eog recordings,” Clinical neurophysiology, vol. 118, no. 5, pp. 1122–1133, 2007.

26. E. Magosso, M. Ursino, A. Zaniboni, and E. Gardella, “A wavelet-based energetic approach for the analysis of biomedical signals: Application to the electroencephalogram and electro-oculogram,” Applied Mathematics and Computation, vol. 207, no. 1, pp. 42–62, 2009.

27. C.-C. Chang and C.-J. Lin, “Libsvm: a library for support vector machines,” ACM Transactions on Intelligent Systems and Technology (TIST), vol. 2, no. 3, p. 27, 2011.

28. G. E. Hinton and R. S. Zemel, “Autoencoders, minimum description length, and helmholtz free energy,” Advances in neural information processing systems, pp. 3–3, 1994.

29. D. Nie, X.-W. Wang, L.-C. Shi, and B.-L. Lu, “EEG-based emotion recognition during watching movies,” in The 5th International IEEE/EMBS Conference on Neural Engineering (NER), pp. 667–670, 2011.

30. R. H. Shumway and D. S. Stoffer, Time series analysis and its applications: with R examples, Springer, 2011

31. Barreto AB, Scargle SD, Adjouadi M. A practical EMG-based human-computer interface for users with motor disabilities. J Rehabil Res Dev. 2000 Jan-Feb;37(1):53-63. PMID: 10847572.

32. Fernandes et al., "EMG-based Motion Intention Recognition for Controlling a Powered Knee Orthosis," 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), 2019, pp. 1-6, doi: 10.1109/ICARSC.2019.8733628.

33. Tariq Madiha, Trivailo Pavel M., Simic Milan, EEG-Based BCI Control Schemes for Lower-Limb Assistive-Robots,"2018 in Frontiers in Human Neuroscience, Volume 12, doi 10.3389/fnhum.2018.00312 ,ISSN=1662-5161.

34. AL-Quraishi, M.S.; Elamvazuthi, I.; Daud, S.A.; Parasuraman, S.; Borboni, A. EEG-Based Control for Upper and Lower Limb Exoskeletons and Prostheses: A Systematic Review. *Sensors* 2018, *18*, 3342.

35. McFarland, D.J., Lefkowicz, A.T. & Wolpaw, J.R. Design and operation of an EEG-based brain-computer interface with digital signal processing technology. *Behavior Research Methods, Instruments, & Computers* 29, 337–345 (1997).

36. A. Picot, A. Caplier and S. Charbonnier, "Comparison between EOG and high frame rate camera for drowsiness detection," *2009 Workshop on Applications of Computer Vision (WACV)*, 2009, pp. 1-6, doi: 10.1109/WACV.2009.5403120.

37. The KNN Algorithm - Explanation, Opportunities, Limitations - neptune.ai

38. K-Nearest Neighbors Algorithm | KNN Regression Python (analyticsvidhya.com)

39. 1.4. Support Vector Machines — scikit-learn 1.1.1 documentation

40. Decision Tree Classifier - an overview | ScienceDirect Topics

41. Decision Tree Algorithm, Explained - KDnuggets

42. Understanding Random Forest. How the Algorithm Works and Why it Is… | by Tony Yiu | Towards Data Science

43. Singh, Yadvendra & Singh, Nirbhow & Agarwal, Ravinder. (2014). ANALYSIS Bio-POTENTIAL TO ASCERTAIN MOVEMENTS FOR PROSTHETIC ARM WITH

DIFFERENT WEIGHTS USING LabVIEW. International Journal of Electronics Communications and Electrical Engineering.

44. Kundu, A.S., Mazumder, O., Lenka, P.K. *et al.* Hand Gesture Recognition Based Omnidirectional Wheelchair Control Using IMU and EMG Sensors. *J Intell Robot Syst* 91, 529–541 (2018). https://doi.org/10.1007/s10846-017-0725-0

45. Mokri C, Bamdad M, Abolghasemi V. Muscle force estimation from lower limb EMG signals using novel optimised machine learning techniques. *Med Biol Eng Comput*. 2022;60(3):683-699. doi:10.1007/s11517-021-02466-z

[46]. Singh, S. *et al.* (2020). Monitoring Post-stroke Motor Rehabilitation Using EEG Analysis. In: Tiwary, U., Chaudhury, S. (eds) Intelligent Human Computer Interaction. IHCI 2019. Lecture Notes in Computer Science(), vol 11886. Springer, Cham. https://doi.org/10.1007/978-3-030-44689-5_2

[47] Banerjee, Anwesha & Datta, Shounak & Konar, Amit & Tibarewala, D.N.. (2012). Development strategy of eye movement controlled rehabilitation aid using Electro-oculogram. http://www.ijser.org/. 3.

[48] Human Eye: Anatomy, parts and structure - Online Biology Notes

[49] Cazzato D, Leo M, Distante C, Voos H. When I Look into Your Eyes: A Survey on Computer Vision Contributions for Human Gaze Estimation and Tracking. Sensors (Basel). 2020 Jul 3;20(13):3739. doi: 10.3390/s20133739. PMID: 32635375; PMCID: PMC7374327.

[50] DOI: https://doi.org/10.36548/jiip.2021.1.003

[51] A. T. Duchowski, Eye Tracking Methodology: Theory and Practice, 2nd ed. London, U.K.: Springer, 2007.

[52] Hadish Habte Tesfamikael, Adam Fray, Israel Mengsteab , Adonay Semere „„„Simulation of Eye Tracking Control based Electric Wheelchair Construction by Image Segmentation Algorithm““, Journal of Innovative Image Processing ,2021.

[53] MOHAMAD A. EID1, NIKOLAS GIAKOUMIDIS, “A Novel EyeGaze-Controlled Wheelchair System for Navigating Unknown Environments: Case Study With a Person With ALS ”,IEEE,2016

[54] *A Comprehensive Guide to Convolutional Neural Networks—the ELI5 way*. [online] Available at: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> [Accessed 11 August 2022].

[55] Gu, Hao & Wang, Yu & Hong, Sheng & Gui, Guan. (2019). Blind Channel Identification Aided Generalized Automatic Modulation Recognition Based on Deep Learning. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2934354.

APPENDIX

```
import os
import collections
import pandas as pd
import numpy as np
import functools
import matplotlib.pyplot as plt
import cv2

from sklearn import preprocessing


import xml.etree.ElementTree as ET

import albumentations as A
from albumentations.pytorch.transforms import ToTensor

import torch
import torchvision

from torchvision.models.detection.faster_rcnn import FastRC
NNPredictor
from torchvision.models.detection import FasterRCNN
from torchvision.models.detection.rpn import AnchorGenerato
r


from torch.utils.data import DataLoader, Dataset
from torch.utils.data import SequentialSampler
from google.colab import drive
drive.mount('/content/drive')
BASE_PATH = "/content/drive/MyDrive/data/"
XML_PATH = os.path.join(BASE_PATH, "annotations")
IMG_PATH = os.path.join(BASE_PATH, "images")
XML_FILES = [os.path.join(XML_PATH, f) for f in os.listdir(
XML_PATH)       class XmlParser(object):

def __init__(self,xml_file):

   self.xml_file = xml_file
   self._root = ET.parse(self.xml_file).getroot()
   self._objects = self._root.findall("object")
   # path to the image file as describe in the xml file
```

```
            self.img_path = os.path.join(IMG_PATH, self._root.f
    ind('filename').text)
        # image id
        self.image_id = self._root.find("filename").text
        # names of the classes contained in the xml file
        self.names = self._get_names()
        # coordinates of the bounding boxes
        self.boxes = self._get_bndbox()

     def parse_xml(self):
        """Parse the xml file returning the root."""

        tree = ET.parse(self.xml_file)
        return tree.getroot()

    def _get_names(self):

        names = []
        for obj in self._objects:
            name = obj.find("name")
            names.append(name.text)

        return np.array(names)

    def _get_bndbox(self):

        boxes = []
        for obj in self._objects:
            coordinates = []
            bndbox = obj.find("bndbox")
            coordinates.append(np.int32(bndbox.find("xmin").text
))
                coordinates.append(np.int32(np.float32(bndbox.f
    ind("ymin").text)))
            coordinates.append(np.int32(bndbox.find("xmax").text
))
            coordinates.append(np.int32(bndbox.find("ymax").text
))
            boxes.append(coordinates)

        return np.array(boxes)
    def xml_files_to_df(xml_files):

"""Return pandas dataframe from list of XML files."""
```

```
    names = []
    boxes = []
     image_id = []
     xml_path = []
    img_path = []
    for file in xml_files:
        xml = XmlParser(file)
        names.extend(xml.names)
        boxes.extend(xml.boxes)
        image_id.extend([xml.image_id] * len(xml.names))
        xml_path.extend([xml.xml_file] * len(xml.names))
        img_path.extend([xml.img_path] * len(xml.names))
    a = {"image_id": image_id,
         "names": names,
         "boxes": boxes,
         "xml_path":xml_path,
         "img_path":img_path}

    df = pd.DataFrame.from_dict(a, orient='index')
    df = df.transpose()

    return df

    df = xml_files_to_df(XML_FILES)
    df.head()
    # check values for per class
    df['names'].value_counts()
    # remove .jpg extension from image_id
    df['img_id'] = df['image_id'].apply(lambda x:x.split('.')).
    map(lambda x:x[0])
    df.drop(columns=['image_id'], inplace=True)
    df.head()
    # classes need to be in int form so we use LabelEncoder for
     this task
    enc = preprocessing.LabelEncoder()
    df['labels'] = enc.fit_transform(df['names'])
    df['labels'] = np.stack(df['labels'][i]+1 for i in range(le
    n(df['labels'])))
    classes = df[['names','labels']].value_counts()
    classes
    # make dictionary for class objects so we can call objects
    by their keys.
    classes= {1:'LeftEye',2:'RightEye'}
    # bounding box coordinates point need to be in separate col
umns
```

```
df['xmin'] = -1
df['ymin'] = -1
df['xmax'] = -1
df['ymax'] = -1

df[['xmin','ymin','xmax','ymax']]=np.stack(df['boxes'][i] f
or i in range(len(df['boxes'])))

df.drop(columns=['boxes'], inplace=True)
df['xmin'] = df['xmin'].astype(np.float)
df['ymin'] = df['ymin'].astype(np.float)
df['xmax'] = df['xmax'].astype(np.float)
df['ymax'] = df['ymax'].astype(np.float)
# drop names column since we dont need it anymore
df.drop(columns=['names'], inplace=True)
df.head()
len(df['img_id'].unique())
image_ids = df['img_id'].unique()
valid_ids = image_ids[-20:]
train_ids = image_ids[:-20]
len(train_ids)
valid_df = df[df['img_id'].isin(valid_ids)]
train_df = df[df['img_id'].isin(train_ids)]
valid_df.shape, train_df.shape
!pip install -q albumentations
import tensorflow as tf
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import os
from albumentations import RandomRotate90
from tensorflow.keras import mixed_precision
import gc
def func(image):
Trgb2lms =np.array( [
     np.array([17.8824, 43.5161, 4.1194]),
     np.array([3.4557,27.1154, 3.8671]),
     np.array([0.0300, 0.1843, 1.4671])
 ])

image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
x,y,z = image.shape
#     print(image.shape)
cvd_due = np.array([
```

```
                    np.array([1 ,0, 0]),
                    np.array([0.494207, 0, 1.24827]),
                    np.array([0, 0, 1]),
          ])
     INV_Trgb2lms = np.linalg.inv(Trgb2lms)

          #     print(image.transpose(2, 0, 1).shape)
    out = np.dot(INV_Trgb2lms, cvd_due)
    out = np.dot(out, Trgb2lms)
    out = np.dot(out, image.transpose(2, 0, 1).reshape(3,-1))
    out = out.reshape(3,x,y).transpose(1, 2, 0)
    out = cv2.cvtColor(np.float32(out), cv2.COLOR_RGB2BGR)

    return out

     class VOCDataset(Dataset):

    def __init__(self, dataframe, image_dir, transforms=None):
        super().__init__()

        self.image_ids = dataframe['img_id'].unique()
        self.df = dataframe
        self.image_dir = image_dir
        self.transforms = transforms

    def __getitem__(self, index: int):
        image_id = self.image_ids[index]
        records = self.df[self.df['img_id'] == image_id]

             image = cv2.imread(f'{self.image_dir}/{image_id}.jp
     g', cv2.IMREAD_COLOR)
        image = func(image)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB).astype(np
.float32)
        image /= 255.0
        rows, cols = image.shape[:2]

        boxes = records[['xmin', 'ymin', 'xmax', 'ymax']].values


        area = (boxes[:, 3] - boxes[:, 1]) * (boxes[:, 2] - boxe
s[:, 0])
        area = torch.as_tensor(area, dtype=torch.float32)

        label = records['labels'].values
```

```
        labels = torch.as_tensor(label, dtype=torch.int64)

        # suppose all instances are not crowd
        iscrowd = torch.zeros((records.shape[0],), dtype=torch.i
nt64)

        target = {}
        target['boxes'] = boxes
        target['labels'] = labels
        # target['masks'] = None
        target['image_id'] = torch.tensor([index])
        target['area'] = area
        target['iscrowd'] = iscrowd

        if self.transforms:
            sample = {
                'image': image,
                'bboxes': target['boxes'],
                'labels': labels
            }
            sample = self.transforms(**sample)
            image = sample['image']

                 target['boxes'] = torch.stack(tuple(map(torch.t
     ensor, zip(*sample['bboxes'])))).permute(1,0)

            return image, target

    def __len__(self) -> int:
        return self.image_ids.shape[0]
     def get_transform_train():
    return A.Compose([
        A.HorizontalFlip(p=0.5),
        A.RandomBrightnessContrast(p=0.2),
        ToTensor()
    ], bbox_params={'format':'pascal_voc', 'label_fields': ['lab
els']})

     def get_transform_valid():
    return A.Compose([
        ToTensor()
    ], bbox_params={'format': 'pascal_voc', 'label_fields':['lab
els']})
     images, targets= next(iter(train_data_loader))
     images = list(image.to(device) for image in images)
```

```
    targets = [{k: v.to(device) for k, v in t.items()} for t in
 targets]

    plt.figure(figsize=(20,20))
    for i, (image, target) in enumerate(zip(images, targets)):
   plt.subplot(2,2, i+1)
   boxes = targets[i]['boxes'].cpu().numpy().astype(np.int32)
   sample = images[i].permute(1,2,0).cpu().numpy()
   names = targets[i]['labels'].cpu().numpy().astype(np.int64)
   for i,box in enumerate(boxes):
       cv2.rectangle(sample,
                     (box[0], box[1]),
                     (box[2], box[3]),
                     (0, 0, 220), 2)
            cv2.putText(sample, classes[names[i]], (box[0],box[
    1]+15),cv2.FONT_HERSHEY_COMPLEX ,0.5,(0,220,0),1,cv2.LINE_A
    A)

   plt.axis('off')
   plt.imshow(sample)



    # load a model; pre-trained on COCO
    model = torchvision.models.detection.fasterrcnn_resnet50_fp
    n(pretrained=True)
    num_classes = 3

    # get number of input features for the classifier
    in_features = model.roi_heads.box_predictor.cls_score.in_fe
atures

    # replace the pre-trained head with a new one
    model.roi_heads.box_predictor = FastRCNNPredictor(in_featur
es, num_classes)

    model.to(device)
    params = [p for p in model.parameters() if p.requires_grad]
    optimizer = torch.optim.SGD(params, lr=0.005, weight_decay=
0.0005)
    lr_scheduler = torch.optim.lr_scheduler.StepLR(optimizer, s
    tep_size=5, gamma=0.1)

    num_epochs = 10
```

```
     for epoch in range(num_epochs):
    # train for one epoch, printing every 10 iterations
        train_one_epoch(model, optimizer, train_data_loader, de
     vice, epoch, print_freq=10)
    # update the learning rate
    lr_scheduler.step()
    # evaluate on the test dataset
    evaluate(model, valid_data_loader, device=device)
     torch.save(model.state_dict(), 'faster_rcnn_state.pth')

     model = torchvision.models.detection.fasterrcnn_resnet50_fp
     n(pretrained=False, pretrained_backbone=False)

     WEIGHTS_FILE = "./faster_rcnn_state.pth"

     num_classes = 3

     # get number of input features for the classifier
     in_features = model.roi_heads.box_predictor.cls_score.in_fe
atures

     # replace the pre-trained head with a new one
     model.roi_heads.box_predictor = FastRCNNPredictor(in_featur
     es, num_classes)

     # Load the traines weights
     model.load_state_dict(torch.load(WEIGHTS_FILE))

     model = model.to(device)
     def obj_detector(img):
    img = cv2.imread(img, cv2.IMREAD_COLOR)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB).astype(np.float32
)


    img /= 255.0
    img = torch.from_numpy(img)
    img = img.unsqueeze(0)
    img = img.permute(0,3,1,2)

    model.eval()

    detection_threshold = 0.70

    img = list(im.to(device) for im in img)
```

```
    output = model(img)

    for i , im in enumerate(img):
        boxes = output[i]['boxes'].data.cpu().numpy()
        scores = output[i]['scores'].data.cpu().numpy()
        labels = output[i]['labels'].data.cpu().numpy()

        labels = labels[scores >= detection_threshold]
        boxes = boxes[scores >= detection_threshold].astype(np.i
nt32)
        scores = scores[scores >= detection_threshold]

        boxes[:, 2] = boxes[:, 2] - boxes[:, 0]
        boxes[:, 3] = boxes[:, 3] - boxes[:, 1]

    sample = img[0].permute(1,2,0).cpu().numpy()
    sample = np.array(sample)
    boxes = output[0]['boxes'].data.cpu().numpy()
    name = output[0]['labels'].data.cpu().numpy()
    scores = output[0]['scores'].data.cpu().numpy()
    boxes = boxes[scores >= detection_threshold].astype(np.int32
)
    names = name.tolist()

    return names, boxes, sample
pred_path = "/content/drive/MyDrive/test/test"
pred_files = [os.path.join(pred_path,f) for f in os.listdir(pred
_path)]

plt.figure(figsize=(20,60))
for i, images in enumerate(pred_files):
    if i > 19:break
    plt.subplot(10,2,i+1)
    names,boxes,sample = obj_detector(images)
    print(boxes)
    for i,box in enumerate(boxes):
        cv2.rectangle(sample,
                      (box[0], box[1]),
                      (box[2], box[3]),
                      (0, 220, 0), 2)
        cv2.putText(sample, classes[names[i]], (box[0],box[1]-
5),cv2.FONT_HERSHEY_COMPLEX ,0.7,(220,0,0),1,cv2.LINE_AA)

    plt.axis('off')
    plt.imshow(sample)
```

```
   plt.savefig('save_image.png', bbox_inches='tight')  # if you
want to save result
```