AI-ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS

A PROJECT REPORT

Submitted by

ARSHATHUL MOHAMED HAQ B (811519104010)

BOGAR S (811519104018)

FARHAAN N (811519104029)

in partial fulfilment of the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

K.RAMAKRISHNAN COLLEGE OF ENGINEERING (AUTONOMOUS),
SAMAYAPURAM,TRICHY – 621 112.

APRIL 2023

K. RAMAKRISHNAN COLLEGE OF ENGINEERING (AUTONOMOUS)

BONAFIDE CERTIFICATE

Certified that this project report **"AI-ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS"** is the Bonafide work of **"FARHAAN N, ARSHATHUL MOHAMED HAQ B and BOGAR S"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. T.M. NITHYA, M.E., Ph.D.,	Mrs. N. NITHYA, M.E,
Associate Professor	Assistant Professor
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Computer Science & Engineering	Computer Science & Engineering
K.Ramakrishnan College of	K.Ramakrishnan College of
Engineering(Autonoumous).	Engineering(Autonoumous).
Samayapuram,	Samayapuram,
Trichy – 621 112.	Trichy – 621 112.

Submitted for the Project Viva-Voce Examination held on ………………….

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We thank the almighty GOD, without whom it would not have been possible for us to complete our project.

We wish to address our profound gratitude to **Dr.K.RAMAKRISHNAN, Chairman, K.Ramakrishnan College of Engineering (Autonomous),** who encouraged and gave us all help throughout the course.

We express our hearty gratitude and thanks to our honourable and grateful Executive Director **Dr.S.KUPPUSAMY, B.Sc., MBA., Ph.D., K.Ramakrishnan College of Engineering (Autonomous).**

We are glad to thank our principal **Dr.D.SRINIVASAN,M.E., Ph.D., FIE.,MIIW.,MISTE.,MISAE.,C.Engg,** for giving us permission to carry out this project.

We wish to convey our sincere thanks to **Dr.T.M.NITHYA, M.E., Ph.D., Head of the Department, Computer Science and Engineering** for giving us constants encouragement and advice throughout the course.

We are grateful **to Mrs.N.NITHYA, M.E, Assistant Professor in the Department of Computer Science and Engineering, K.Ramakrishnan College of Engineering (Autonomous)**, for his guidance and valuable suggestions during the course of study.

Finally, we sincerely acknowledged in no less term for all our staff members, colleagues, our parents and friends for their co-operation and help at various stages of this project work.

DECLARATION

I hereby declare that the work entitled **"AI-ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS"** is submitted in partial fulfilment of the requirement for the reward of the degree in B.E., Anna University, Chennai, is a record of our own work carried out by me during the academic year 2022-2023 under the supervision and guidance of **Mrs. N. NITHYA, M.E, Assistant professor, Department of Computer Science and Engineering, K.Ramakrishnan College of Engineering (Autonomous).** The extent and source of information are derived from the existing literature and have been indicated through the dissertation at the appropriate places. The matter embodied in this work is original and has not been submitted for the award of any degree or diploma, either in this or any other University.

FARHAAN N
(811519104029)

I certify that the declaration made by above candidate is true.

Mrs. N. NITHYA , M.E.,

Assistant Professor/CSE

DECLARATION

I hereby declare that the work entitled **"AI-ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS"** is submitted in partial fulfilment of the requirement for the reward of the degree in B.E., Anna University, Chennai, is a record of our own work carried out by me during the academic year 2022-2023 under the supervision and guidance of **Mrs. N. NITHYA, M.E, Assistant professor, Department of Computer Science and Engineering, K.Ramakrishnan College of Engineering (Autonomous).** The extent and source of information are derived from the existing literature and have been indicated through the dissertation at the appropriate places. The matter embodied in this work is original and has not been submitted for the award of any degree or diploma, either in this or any other University.

ARSHATHUL MOHAMED HAQ B
(811519104010)

I certify that the declaration made by above candidate is true.

Mrs. N. NITHYA , M.E.,

Assistant Professor/CSE

DECLARATION

I hereby declare that the work entitled **“AI-ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS”** is submitted in partial fulfilment of the requirement for the reward of the degree in B.E., Anna University, Chennai, is a record of our own work carried out by me during the academic year 2022-2023 under the supervision and guidance of **Mrs. N. NITHYA, M.E, Assistant professor, Department of Computer Science and Engineering, K.Ramakrishnan College of Engineering (Autonomous).** The extent and source of information are derived from the existing literature and have been indicated through the dissertation at the appropriate places. The matter embodied in this work is original and has not been submitted for the award of any degree or diploma, either in this or any other University.

BOGAR S
(811519104029)

I certify that the declaration made by above candidate is true.

Mrs. N. NITHYA , M.E.,

Assistant Professor/CSE

ABSTRACT

People who are incapable of managing all of their muscles other than their head and eyes. There are some medical conditions, such as Locked in Syndrome, that can induce paralysis or motor speech disorders in persons, which can result in voice or speech impairments.Traditionally, many people can communicate by tracking their movements and blinking their eyes. Communication is important for allowing people of this type to express how they feel and what they need. Using an IOT module and an eye tracking system based on CNN (Convolutional Neural Network), designed a technique of interpersonal communication for a tetraplegic patient in this system. The use of eye blink detection and movement tracking for communication is also possible for quadriplegic patients. With a voice board speaker and this project concept, talks can be delivered while imparting all necessary information. The goal of this research was to develop an IOT module that would enable tetraplegic people to send emergency information to the care team using deep learning and digital image processing. Following the end of the interaction, the technology will be able to flawlessly display the audio signals from the gaze movement.

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	**EXPANSION**
AI	Artificial Intelligence
ConvNet/CNN	Convolutional Neural Network
ANN	Artificial Neural Network
IOT	Internet of Things
CV	Computer Vision
NumPy	Numeric Python
ReLU	REctified Linear activation Unit
SMS	Short Message Service
CUDA	Compute Unified Device Architecture
UML	Unified Modeling Language
VR	Virtual Reality
HMD	Head-Mounted-Displays
MOOC	Massive Open Online Course
MQTT	Message Queuing Telemetry Transport
PIL	Python Image Library

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Eye tracking technology has considerably developed over the past few years and has become a crucial tool in a number of fields, human-computer interaction, and market research. By detecting the movements of a person's eyes, eye tracking devices can reveal crucial information about that person's visual attention and decision-making processes. The addition of artificial intelligence (AI) to the systems has improved the accuracy, dependability, and adaptability of eye tracking technology. Eye tracking systems augmented by artificial intelligence do, however, have significant challenges. It is challenging to locate high-quality training data for AI systems. Finding the enormous amounts of accurately labeled data required for this can be difficult. AI may be challenging for real-time applications since the methods can be computationally expensive. The objective of this conference paper is to provide a summary of the most current advancements in AI-enhanced eye tracking systems. The study begins by describing the background, context, and applications of eye tracking technology. The aim of the study is to assess the present status of AI enhanced eye tracking systems and their potential for future development. Recent advances in artificial intelligence (AI) have led to the development of AI-enhanced eye-tracking technology. AI algorithms can analyse eye-tracking data and automatically recognize patterns and behaviours that may be difficult or impossible for humans to detect. This technology can help individuals with tetraplegia improve their accuracy and speed of eye movements, leading to more efficient communication and control of electronic devices.

1.1 TETRAPLEGIA PATIENTS

Tetraplegia, also known as quadriplegia, is a medical condition that affects a person's ability to move or feel in their arms, legs, and torso. It is caused by damage to the spinal cord, which can result from a variety of factors such as traumatic injury, disease, or congenital conditions. The severity of tetraplegia varies from person to person, depending on the level and extent of the spinal cord injury. In general, the higher the level of injury on the spinal cord, the more severe the tetraplegia will be. People with tetraplegia may experience paralysis, loss of sensation, and other complications that can affect their daily life. There are several types of tetraplegia, depending on the level of injury on the spinal cord. For example, a person with C1-C4 tetraplegia may have paralysis in their arms, legs, and torso, and may require a ventilator to breathe. A person with C5-C8 tetraplegia may have partial or complete paralysis in their arms, but may be able to use their hands and fingers to some extent. One of the primary challenges of living with tetraplegia is the loss of independence and mobility. Many people with tetraplegia require assistance with daily activities such as eating, dressing, and bathing. They may also need specialized equipment and devices to help them move around and perform tasks. In recent years, there have been many advances in technology and assistive devices that can help people with tetraplegia live more independently. For example, powered wheelchairs can help people with mobility impairments move around more easily, while assistive technology such as voice recognition software and eye tracking systems can help them control electronic devices. In addition to physical challenges, people with tetraplegia may also face psychological and emotional challenges. The loss of independence and mobility can be difficult to come to terms with, and many people with tetraplegia may experience depression, anxiety, and other mental health issues. It is important for people with tetraplegia to have access to support and resources to help them manage these challenges. There is ongoing research into new treatments and therapies for tetraplegia, such as stem cell therapy and spinal cord stimulation. While there is no cure for tetraplegia at present, these and other emerging treatments hold promise for improving the lives of people with this condition in the future. In conclusion, tetraplegia is a medical condition that can have a significant impact on a person's life, including their mobility, independence, and emotional well-being. While there are many challenges associated with tetraplegia, there are also many resources and technologies available to help people with this condition live more fulfilling and independent lives. Ongoing research and innovation in this field hold promise for continued progress in improving the lives of people with tetraplegia.

1.2 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI), sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans. Leading AI textbooks define the field as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. Colloquially, the term "artificial intelligence" is often used to describe machines (or computers) that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving". Artificial intelligence (AI) is an area of computer science that emphasizes the creation of intelligent machines that work and react like humans. Some of the activities computers with artificial intelligence are designed for include:

- Speech recognition
- Learning
- Planning

- Problem solving

1.3 DEFINITIONS

Computer science defines AI research as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. A more elaborate definition characterizes AI as a system’s ability to correctly interpret external data, to learn from such data, and to use those learnings to achieve specific goals and tasks through flexible adaptation. Artificial intelligence is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry. Research associated with artificial intelligence is highly technical and specialized. The core problems of artificial intelligence include programming computers for certain traits such as:

- Knowledge
- Reasoning
- Problem solving
- Perception
- Learning
- Planning

The integration of AI into eye tracking systems has produced a number of benefits, including increased accuracy and dependability, innovative insights and capabilities, and improved human computer interaction. It’s crucial to consider some of the challenges that AI-enhanced eye tracking systems provide. One of the main benefits of AI-enhanced eye tracking systems is the increase in dependability and accuracy. The accuracy of the data gathered by eye trackers can be considerably improved by employing AI algorithms to detect and repair errors in gaze data, such as blinks or gaze changes. This may have important implications for a range of applications, from market research to psychology. To identify and study patterns in eye movements, such as reading patterns, attention patterns, and decision-making patterns, AI algorithms can be used. Insightful information on reading speed, visual attention, and other cognitive processes can be found in these patterns, which can be useful in a variety of situations. Some of the key findings from previous studies and research in this area include:

• Data accuracy and dependability can be considerably increased by integrating AI into eye tracking systems, according to a number of studies. This is because AI systems can identify and fix mistakes in gaze data, including blink, or gaze changes.

• Several studies have shown that AI-enhanced eye tracking systems can offer new insights and capabilities, such as the capacity to recognise and analyse patterns in eye movements, such as reading patterns, attention patterns, and decision-making patterns.

The core of deep learning is represented by artificial neural networks (ANN's), A subset of machine learning. Multi-layer perceptrons, or neural networks, are what they seem to be. From the first stage to multi-layered neural networks, the perceptron models the brain's capacity for learning. To create the final result, input data are processed against one hidden layer and adjusted against a second hidden layer. A Convolutional Neural Network (ConvNet/CNN) is a deep learning algorithm that can take in an input image, assign significance (i.e. weights and biases) to distinct aspects/objects in the image, and be able to distinguish one from the other. The creation of algorithms that power and will continue to power AI as a whole in the near future relied heavily on CNN's of different architectures, which are readily available. Some of them have been listed below:

- LeNet
- AlexNet
- VGGNet
- GoogLeNet
- ResNet
- ZFNet

Focusing on the eye to track eye movement allows for real-time data recording. The key to this paradigm is eye movement, which is interpreted to reveal the process's outcome. Using the eye dataset, which has four categories, the model is trained. Eyes closed, focused on the left, focused on the right, focused on the up, focused on the down.

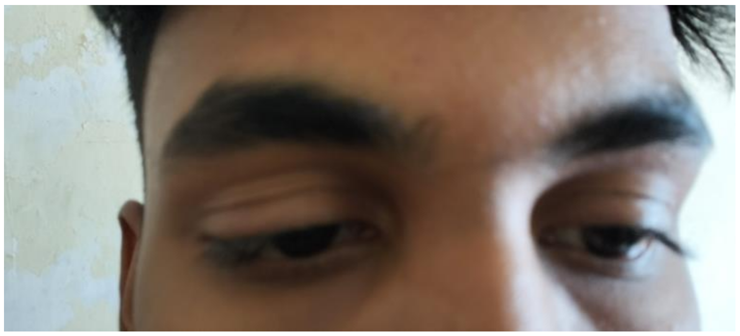


Fig 1.2.1: Eye Down

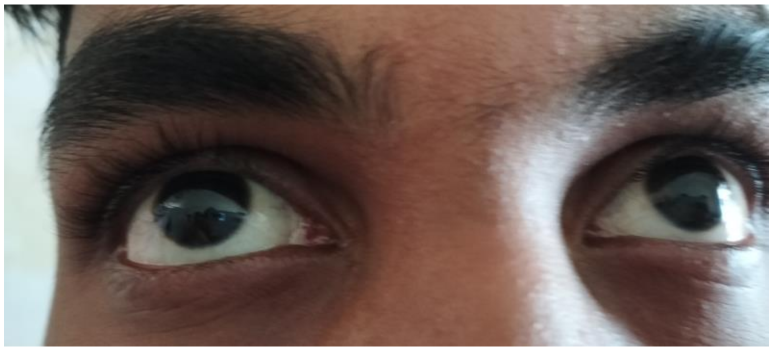


Fig 1.2.2: Eye Up

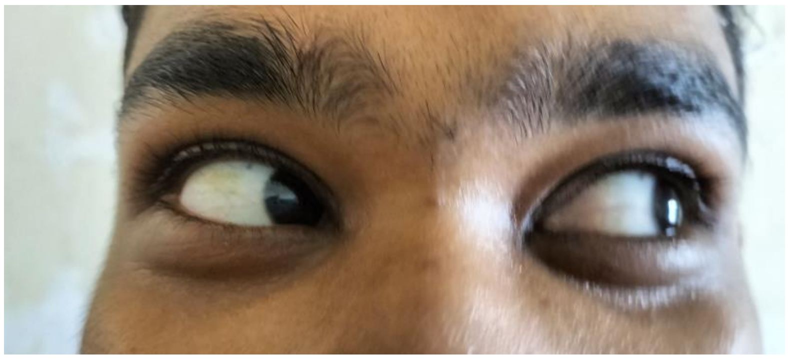


Fig 1.2.3: Eye Left

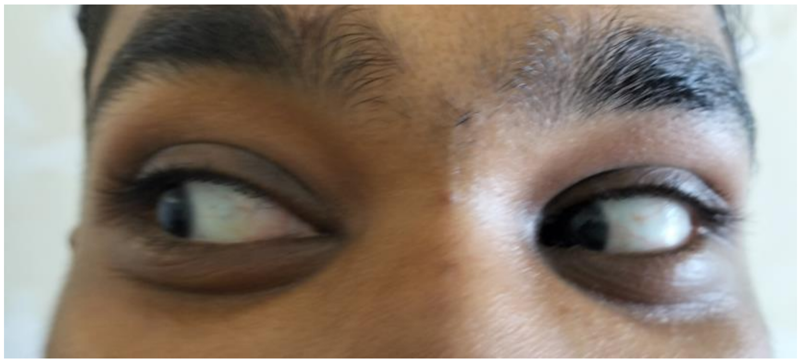


Fig 1.2.4: Eye Right

This model will prompt the user to indicate whether they require assistance by asking three times in a row whether they are blinking. When the eyes are closed and opened three times in a short period of time, the model will automatically understand that the eyes are closed and will blink three times continuously.

1.4 OBJECTIVE

People can communicate by tracking their movements and blinking their eyes. Communication is important for allowing people of this type to express how they feel and what they need.

Using an IOT module and an eye tracking system based on CNN (Convolutional Neural Network), designed a technique of interpersonal communication for a tetraplegic patient in this system.The use of eye blink detection and movement tracking for communication is also possible for quadriplegic patients.

With a voice board speaker and this project concept, talks can be delivered while imparting all necessary information. The goal of this research was to develop an IOT module that would enable tetraplegic people to send emergency information to the care team using deep learning and digital image processing.

CHAPTER 2

LITERATURE SURVEY

1.Zhixuan Mu et al proposed "Application of eye tracking analyzes In the Micro-teaching Environment" IEEE-2020

In most situations, eye movement were controlled by tasks or goals, in which information from different sources is obtained and processed. There is a strong relationship between eye-movement and selective mechanism in visual information processing, so eye- movement becoming a valuable paradigm in education and psychological research. this paper look back the eye-movement history, the article analyzes the 3D camera tracking of eye tracking system. At last, several point are made about apply eye tracking analyzes In the Micro-teaching environment.

2.Bonita Sharif; Cole Peterson; Drew Guarnera; Corey Bryant; Zachary Buchanan; Vlas Zyrianov; Jonathan Maletic et al proposed "Practical Eye Tracking with iTrace" IEEE-2020

The evolution and effort in designing and implementing iTrace, an infrastructure for integrating eye tracking into developer environments, is presented. The goal is to make eye tracking practical for various stakeholders in software engineering namely researchers, practitioners, and educators. An overview of iTrace and the general process involved in conducting an eye tracking study with human subjects using iTrace is presented in this tool demo paper. Upcoming features and ongoing plans for community involvement are also presented.

3.<u>Wolfgang Mehringer</u>; <u>Markus Wirth</u>; <u>FrankaRisch</u>; <u>Daniel Roth</u>; <u>Georg Michelson</u>; <u>BjoernEskofier</u> et al proposed "Hess Screen Revised: How Eye Tracking and Virtual Reality change Strabismus Assessment" IEEE-2021

Strabismus is a visual disorder characterized by eye misalignment. The extent of ocular misalignment is denoted as the deviation angle. With the advent of Virtual Reality (VR) Head-Mounted-Displays (HMD) and eye tracking technology, new possibilities measuring strabismus arise. Major research addresses the novel field of VR strabismus assessment by replicating prism cover tests while there is a paucity of research on screen tests. In this work the Hess Screen Test was implemented in VR using a HMD with eye tracking for an objective measurement of the deviation angle. In a study, the functionality was tested and compared with a 2D monitor-based test. The results showed significant differences in the measured deviation angle between the methods. This can be attributed to the type of dissociation of the eyes.

4. <u>Amer Al-Rahayfeh</u>; <u>MiadFaezipour</u> et al proposed "Enhanced frame rate for real-time eye tracking using circular hough transform" IEEE-2020

Eye-gaze detection and tracking has been widely investigated and presented as a way of unconventional human computer interaction. This area has provided convenience for many fields of practical applications, such as assistive systems and technology for people with severe disabilities, virtual reality, driver assistance and monitoring systems. Many methods for eye tracking have been introduced in literature. In this paper, a real-time eye tracking system is presented. To locate the iris of the eye in the captured video frames, the system uses the Circular Hough Transform which aims to recognize circular patterns in an image. Generally, the speed of eye motion is not as high as the used video frame rate of 30 Frames per Second (FPS) which is the frame rate used in general live video. In other words, the eye cannot move as fast as 30 motions per second. This led to proposing an enhancement to the eye tracking system being presented. This enhancement improved the CPU processing time requirements. The enhancement presented in this paper suggests that not all captured live video frames need to be processed for eye detection because the same eye movement will be captured on multiple subsequent video frames. Processing only a subset of frames will be enough to detect all eye movements in the video. The required CPU processing time is improved by selecting the minimum accepted video frame rate sufficient for accurately detecting all eye motions in a video. This was investigated for both the low and high speed eye movements. For low speed eye movements, the improvement in required CPU time was 1500%. For high speed eye movements, it was 750%. The improvement in CPU time is general and applies to different eye tracking algorithms when using the proposed enhancement. These improvements are a result of the elimination of the redundant video frames which are no longer processed in the procedure of eye detection.

5. <u>Audi I. Al-Btoush</u>; <u>Mohammad A. Abbadi</u>; <u>Ahmad B. Hassanat</u>; <u>Ahmad S. Tarawneh</u>; <u>Asad Hasanat</u>; <u>V. B. Surya Prasath</u> et al proposed "New Features for Eye-Tracking Systems: Preliminary Results" IEEE-2020

Due to their large number of applications, eye-tracking systems have gain attention recently. In this work, They proposed 4 new features to support the most used feature by these systems, which is the location (x, y). These features are based on the white areas in the four corners of the sclera; the ratio of the whites area (after segmentation) to the corners area is used as a feature coming from each corner. In order to evaluate the new features, They designed a simple eye-tracking system using a simple webcam, where the users faces and eyes are detected, which allows for extracting the traditional and the new features. The system was evaluated using 10 subjects, who looked at 5 objects on the screen. The experimental results using some machine learning algorithms show that the new features are user dependent, and therefore, they cannot be used (in their current format) for a multiuser eye-tracking system. However, the new features might be used to support the traditional features for a better single-user eye-tracking system, where the accuracy results were in the range of 0.90 to 0.98.

CHAPTER - 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The interaction with the various learners in a Massive Open Online Course (MOOC) is often complex. Contemporary MOOC learning analytics relate with click-streams, keystrokes and other user-input variables. Such variables however, do not always capture users' learning and behavior (e.g., passive video watching). In this paper, They presented a study with 40 students who watched a MOOC lecture while their eye movements were being recorded and then proposed a method to define stimuli based gaze variables that can be used for any kind of stimulus. The proposed stimuli-based gaze variables indicate students' content-coverage (in space and time) and reading processes (area of interest based variables) and attention (i.e., with-meness), at the perceptual (following teacher's deictic acts) and conceptual levels (following teacher discourse). In our experiment, They identified a significant mediation effect of the content coverage, reading patterns and the two levels of with-me-ness on the relation between students' motivation and their learning performance. Such variables enable common measurements for the different kind of stimuli present in distinct MOOCs. Our long-term goal is to create student profiles based on their performance and learning strategy using stimuli-based gaze variables and to provide students gaze-aware feedback to improve overall learning process. One key ingredient in the process of achieving a high level of adaptation in providing gazeaware feedback to the students is to use artificial intelligence algorithms for prediction of student performance from their behavior. In this contribution, They presented a method combining state-of-the-art AI technique with the eye-tracking data to predict student performance. The results show that the student performance can be predicted with an error of less than 5%. Finally, from the prediction results, They were able to show that the heat-maps cannot be only used as a popular visualization tools, but also as a source of features to predict performance and other traits, such as motivation. The best prediction results for the performance was with a 5.04% normalized error. In terms of a quiz-based evaluation of learning, which in our case are 10 questions, this error translates to less than one question. For example, if a student answers 9 questions correctly, our method will predict the score within the range of [8.5–9.5]. Similarly, on the motivation scale, which is a 5-point Likert scale making it in the range of [0 -- 50], the error of 9.04% would translate to one incorrect prediction out of ten on the scale proposed by Biggs et al. (2001). Additionally, in this contribution, the eye-tracking variables they defined had different pre-processing requirement. These variables also have capacities in terms of being used within an adaptive and real-time system. The computation of content coverage is realtime and requires no pre-processing of the data or the stimulus. The Scan-path variables can also be computed in real-time and there is small amount of pre-processing required in term of defining the area of interest (AOI) to be able to compute them. The pre-processing for computing the perceptual with-me-ness could be automatized since there are computer-vision methods to detect pointing/other deictic gestures of the teacher. Once this detection is done, the real-time computation of Perceptual with-meness if fairly straightforward. Finally, the conceptual with-me-ness, requires a few manual interventions in transcribing the teachers' dialogues and mapping them to the content. This acts as a hindrance in the real-time computation of the conceptual with-me-ness, and therefore, this is the only gaze-based measure used in this study that requires further work to be used as within a personalised adaptive gaze-based feedback system. To gain further insight into the design of MOOC videos and the affordances of the respective systems, they need to consider eye-gaze measurements (or can call them gaze analytics) that they found to not only strongly associated with learning, but also mediate the influence of other variables (i.e., motivation). Discussing these features from a technical standpoint can give rise to practical implications for the design of MOOC videos (e.g., designed in a way to draw students' attention (Kizilcec et al., 2014) and the respective video-based learning systems (e.g., offer an indication of students' attention based on the web-camera).

3.2 PROPOSED SYSTEM

The system would include a user-friendly interface, allowing tetraplegia patients to control external devices or communicate using their eye movements. The interface should be optimized for ease of use and minimal effort required. The system would be customizable to the specific needs and abilities of each tetraplegia patient, taking into account their individual eye movements and preferences. The system would be able to integrate with external devices, such as computers or communication devices, to allow tetraplegia patients to control these devices using their eye movements. The system would include feedback mechanisms, such as visual or auditory cues, to provide tetraplegia patients with feedback on their eye movements and actions. The system would prioritize data privacy and security, ensuring that the personal information and eye movement data of tetraplegia patients are protected and not misused.

3.3 BLOCK DIAGRAM

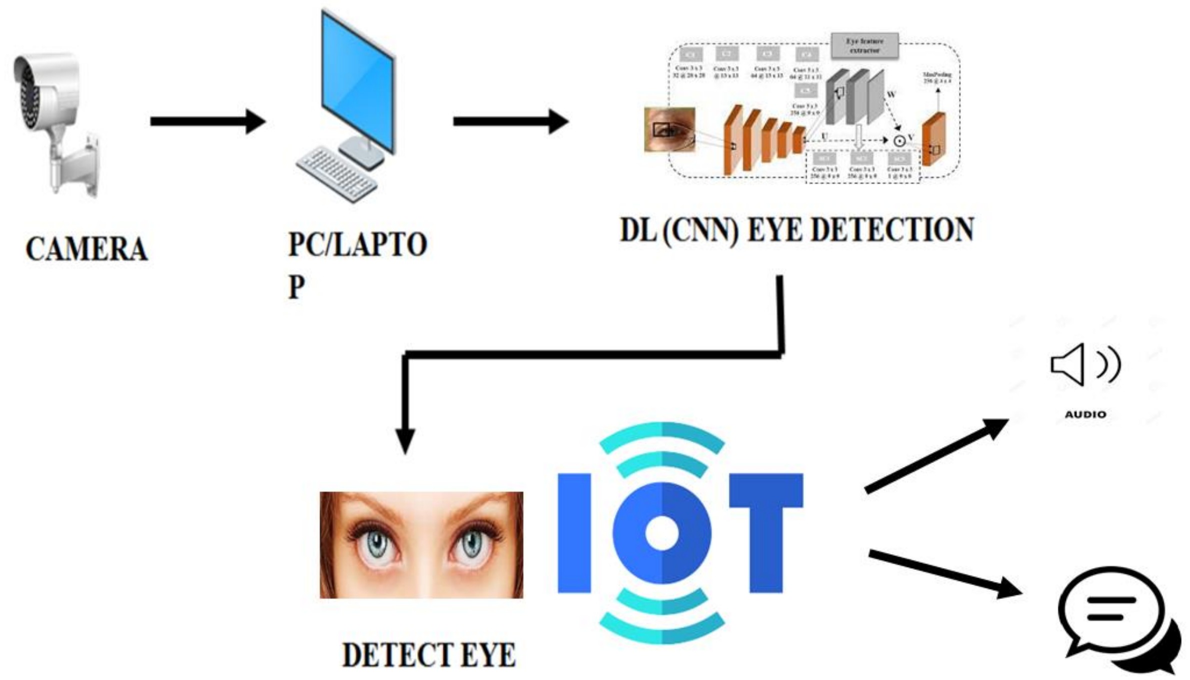


Fig 3.3 Block Diagram

CHAPTER 4

MODULE DESCRIPTION

4.1. IMAGE DATASET COLLECTION

The eye dataset collection module is an essential component of the project, as it provides the foundation for training the machine learning algorithm to recognize and classify different features of the eyes. This module involves collecting images or videos of the eyes under various conditions, capturing images of the eyes with different medical conditions or diseases, and labeling or annotating the images with information about the specific features or characteristics of the eyes. Once the dataset has been collected and annotated, it can be used to train our machine learning model to recognize and classify the features of the eyes. The eye dataset collection module is critical for the success of the project, as it ensures that our machine learning model is trained on a diverse and comprehensive dataset that accurately represents the characteristics of the eyes that needs to be analyzed. The image dataset is classified into four categories: i) Eye watching upwards, ii) Eye watching downwards, iii) Eye watching left side iv) Eye watching rightside.

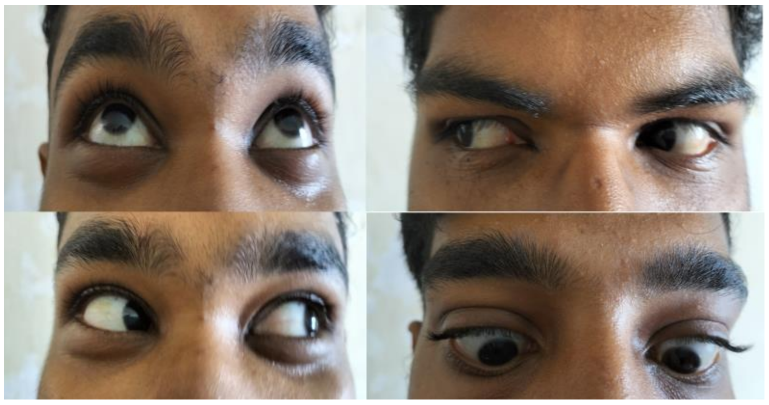


Fig 4.1 Sample Dataset images

4.2. IMAGE PREPROCESSING

The eye dataset preprocessing module is a critical component of the project, as it enables us to clean, transform, and prepare the eye images or videos for use in training our machine learning algorithm. This module involves several steps, including removing any noise or artifacts from the images or videos, resizing and cropping the images to a standard size, normalizing the pixel values to a common scale, and augmenting the dataset with additional images or videos to increase its size and diversity. In addition, this module also involves splitting the dataset into training, validation, and testing sets, to ensure that our machine learning model is trained on a diverse and representative dataset and can generalize well to new data. The eye dataset preprocessing module is essential for the success of the project, as it ensures that our machine learning model is trained on a high-quality and well-curated dataset that accurately represents the features and characteristics of the eyes that needs to be analyzed.

4.3. TRAINING THE MODEL

This module involves feeding the preprocessed dataset into the machine learning model, adjusting the model's parameters, and optimizing its performance using various techniques such as regularization and hyper parameter tuning. During the training process, it monitors the performance of the model on the validation set, adjusting the model's architecture or hyper parameters as needed to improve its performance. Once the model has been trained and evaluated its performance on the testing set, to ensure that it can generalize well to new and unseen data. The training the model module is essential for the success of our project, as it ensures that our machine learning model is accurately trained and optimized to recognize and classify the features of the eyes.

4.4. FINAL COMBINATION OF IOT AND DEEP LEARNING MODEL

This module is connected to IOT server and deep learning model. In which the computer vision will get turned on and each frame will get resized and passed as the input for the model. The predicted output will be categorized based on the accuracy and model will predict five times continuously and the final result will be based on the frequency. MQTT stands for Message Queuing Telemetry Transport. It is a lightweight, publish-subscribe network protocol designed to efficiently transport messages between devices, especially in low-bandwidth, high-latency or unreliable networks. MQTT was developed in the late 1990s by Dr. Andy Stanford-Clark of IBM and Arlen Nipper of Arcom. MQTT works on a client-server architecture, where clients are the devices that produce or consume data, and servers are the brokers that receive and route the messages between the clients. The MQTT protocol uses a publish-subscribe model, where publishers (clients) send messages to a broker, which then distributes the messages to all the subscribed clients. The broker acts as an intermediary and does not store messages unless a client is unavailable at the time of message delivery. The output will be delivered as sound using pyttsx3 and a sms message from the Twilio as per the basis of MQTT protocol.

CHAPTER 5

SOFTWARE DESCRIPTION

5.1 PYTHON

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

• Web development (server-side),

• Software development,

• Mathematics,

• System scripting.

There are two attributes that make development time in Python faster than in other programming languages:

1. Python is an interpreted language, which precludes the need to compile code before executing a program because Python does the compilation in the background. Because Python is a high-level programming language, it abstracts many sophisticated details from the programming code. Python focuses so much on this abstraction that its code can be understood by most novice programmers.

2. Python code tends to be shorter than comparable codes. Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. Of course, with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations. In

most cases, Python is already included in Linux distributions and Mac OS X machines.

3. Python is a dynamic, high level, free open source and interpreted programming language. It supports object –oriented programming as well as procedural oriented programming. Python is a very easy to code as compared to other language like c , c ++, java etc.. It is also a developer friendly language. Python is also an Integrated language with other language like c, c ++, etc.

5.2 PYTHON NUMPY

NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.

Travis Oliphant created NumPy package in 2005 by injecting the features of the ancestor module Numeric into another module NumPyarray. It is an extension module of Python which is mostly written in C. It provides various functions which are capable of performing the numeric computations with a high speed. NumPy provides various powerful data structures, implementing multi-dimensional arrays and matrices. These data structures are used for the optimal computations regarding arrays and matrices.

NumPy package usage in the project

1. numpy.expand_dims() function implements the multidimensional arrays.
2. numpy.uint8: 8-bit unsigned integer (0 to 255). Most often this is used for arrays representing images.

5.3 OPEN CV

OpenCV is a Python open-source library, which is used for computer vision in artificial intelligence, machine learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos. The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on.

There are a two main task which are used in the project:

- **cv2.VideoCapture()**: Function which helps to read the each frame directly from the camera
- **cv2.resize():** Function which helps to resize the frame size
- **cv2.imshow():** Function which helps to show each frame in new desktop

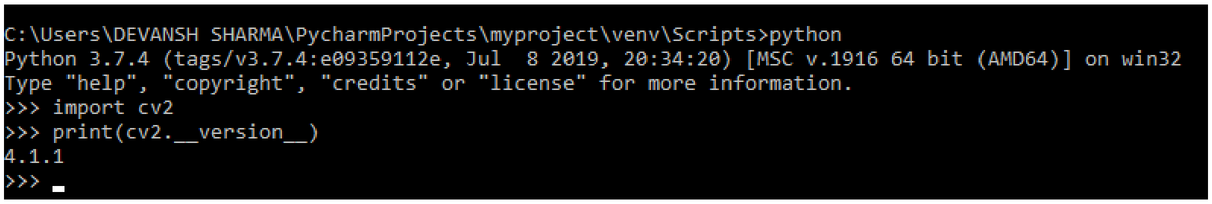
5.3.1 Installation of the OpenCV

Install OpenCV in the Windows via pip

OpenCV is a Python library so it is necessary to install Python in the system and install OpenCV using pip command:

```
pip install opencv-python
```

Open the command prompt and type the following code to check if the OpenCV is installed or not.



```
C:\Users\DEVANSH SHARMA\PycharmProjects\myproject\venv\Scripts>python
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.__version__)
4.1.1
>>> 
```

5.4 SCIPY

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics and many other classes of problems.The main purpose of using the scipy in the project is to calculate the Euclidean distance.The distance module in scipy provides the function named euclidean.

distance.euclidean() – Function that computes the Euclidean distance between two 1-D arrays

5.5 TENSORFLOW

Tensorflow is a software library or framework, developed by the Google team to implement machine learning and deep learning concepts in the easiest manner. It combines the computational algebra of optimization techniques for easy calculation of many mathematical expressions.

TensorFlow — Installation

To install Tensorflow, it is important to have "Python" installed in the system. Python version 3.4+ is considered the best to start with tensorflow installation. Consider the following steps to install tensorflow in Windows operating system.

pip install tensorflow

Tensorflow usage in project:

- **Tensorflow.Graph()**: Returns a serialized GraphDef representation of this graph and can be imported into another Graph (using tf.import_graph_def).
- **Tensorflow.Gfile()**: Tensorflow exports these as tf.io.gfile, so that this function can be used for implementations of saving and loading checkpoints, writing to tensorboard logs, and accessing training data.
- **Tensorflow.squeeze()**: Removes dimensions of size 1 from the shape of a tensor.
- **Tensorflow.cast()**: function that is used to cast a specified Tensor to a another data type.
- **Tensorflow.slice()** : function will slice the tensor to a specified size.
- **Tensorflow.expand_dims() :**function increases the dimensions of the array.
- **Tensorflow.estimator.train_and_evaluate()** : function will train the model based on the estimator.

5.6 PYTHON PILLOW

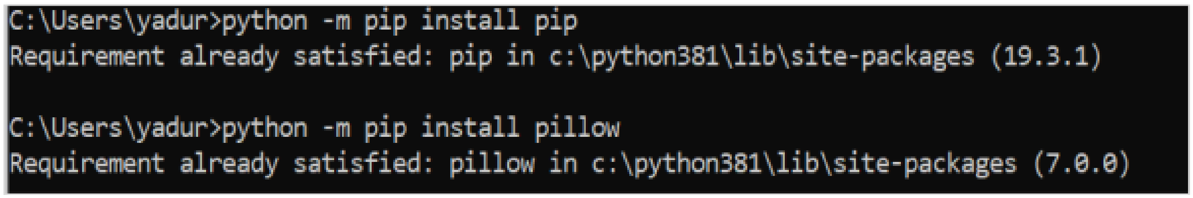
Pillow is built on top of PIL (Python Image Library). PIL is one of the important modules for image processing in Python. However, the PIL module is not supported since 2011 and doesn't support python 3.

Image Processing

The Pillow library contains all the basic image processing functionality which are image resizing, rotation and transformation. Pillow module allows to pull some statistics data out of image using histogram method, which later can be used for statistical analysis and automatic contrast enhancement.

Installing Pillow using pip

To install pillow using pip, run the below command in command prompt:
python -m pip install pip
python -m pip install pillow



```
C:\Users\yadur>python -m pip install pip
Requirement already satisfied: pip in c:\python381\lib\site-packages (19.3.1)

C:\Users\yadur>python -m pip install pillow
Requirement already satisfied: pillow in c:\python381\lib\site-packages (7.0.0)
```

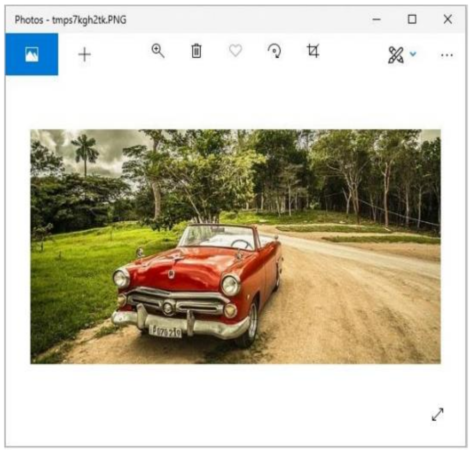
Opening, rotating and displaying an image

To load the image, Import the image module from the pillow and call the Image.open(), passing the image filename. Instead of calling the Pillow module, It will be called as the PIL module as to make it backward compatible with an older module called Python Imaging Library (PIL). That's why our code starts with "from PIL import Image" instead of "from Pillow import Image". To load the image by calling the Image.open() function, which returns a value of the Image object data type. Any modification happened to the image object can be saved to an image file with the save() method. The image object received using Image.open(), later can be used to resize, crop, draw or other image manipulation method calls on this Image object.

Example

Following example demonstrates the rotation of an image using python pillow:

from PIL import Image



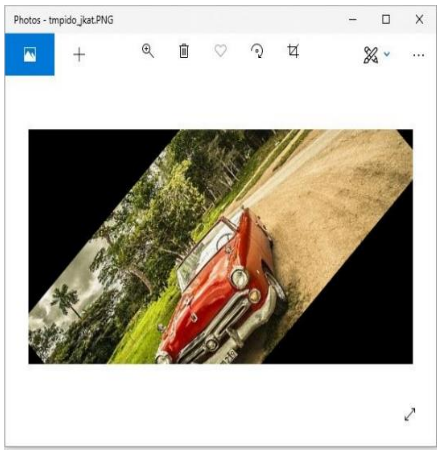


Fig 5.6.1 Normal image **Fig 5.6.2 Rotated image (45 degree)**

CHAPTER-6

HARDWARE DESCRIPTION

6.1 MINIMUM SYSTEM REQUIREMENTES:

- Processor: A 64-bit CPU with AVX support. Most modern CPUs should meet this requirement.
- RAM: At least 4 GB of RAM is recommended, although more may be required for larger models and datasets.
- Operating System: Tensorflow is compatible with Windows, Linux, and macOS operating systems.
- Python: Tensorflow requires Python 3.6 or higher to be installed on the system.
- Disk Space: A minimum of 1 GB of free disk space is required to install TensorFlow and its dependencies. Additional disk space may be required for storing models and datasets.
- GPU (Optional): TensorFlow can also be run on a GPU for improved performance. However, this requires a compatible NVIDIA GPU and the installation of additional software, such as CUDA and cuDNN.

6.2 EXTERNAL CAMERA :

- A webcam is a hardware camera and input device that connects to a computer and the Internet and captures either still pictures or motion video of a driver drowsiness.
- A webcam is a video camera that feeds or streams an image or video in real time to or through a computer to a computer network, such as the Internet. Webcams are typically small cameras that sit on a desk, attach to a user's monitor, or are built into the hardware
- Webcams are used only for the main purpose in this project where tetraplegia patients will lie in the bed always. A stand will be used to hold this camera which is placed above the tetraplegia patients facing their face.

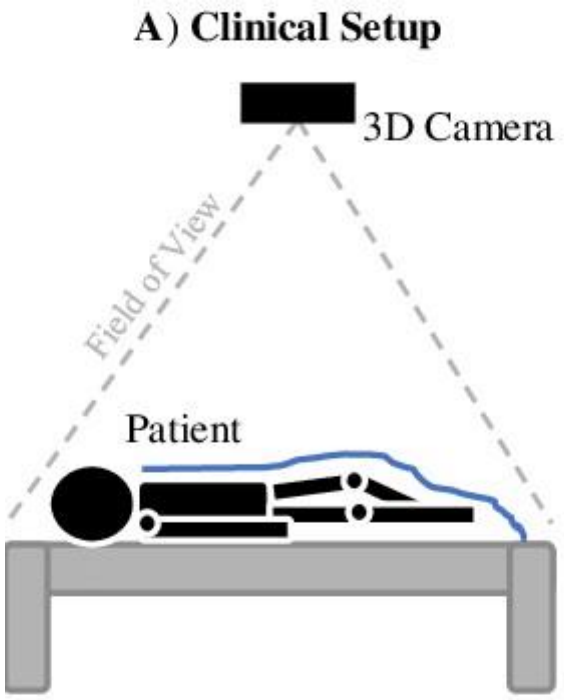


Fig 6.2 Fixation Diagram of the System

CHAPTER – 7

SYSTEM DESIGN

System architecture is the process of defining the architecture, modules, interfaces and data for a system to satisfy specified requirements.

7.1 USECASE DIAGRAM

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

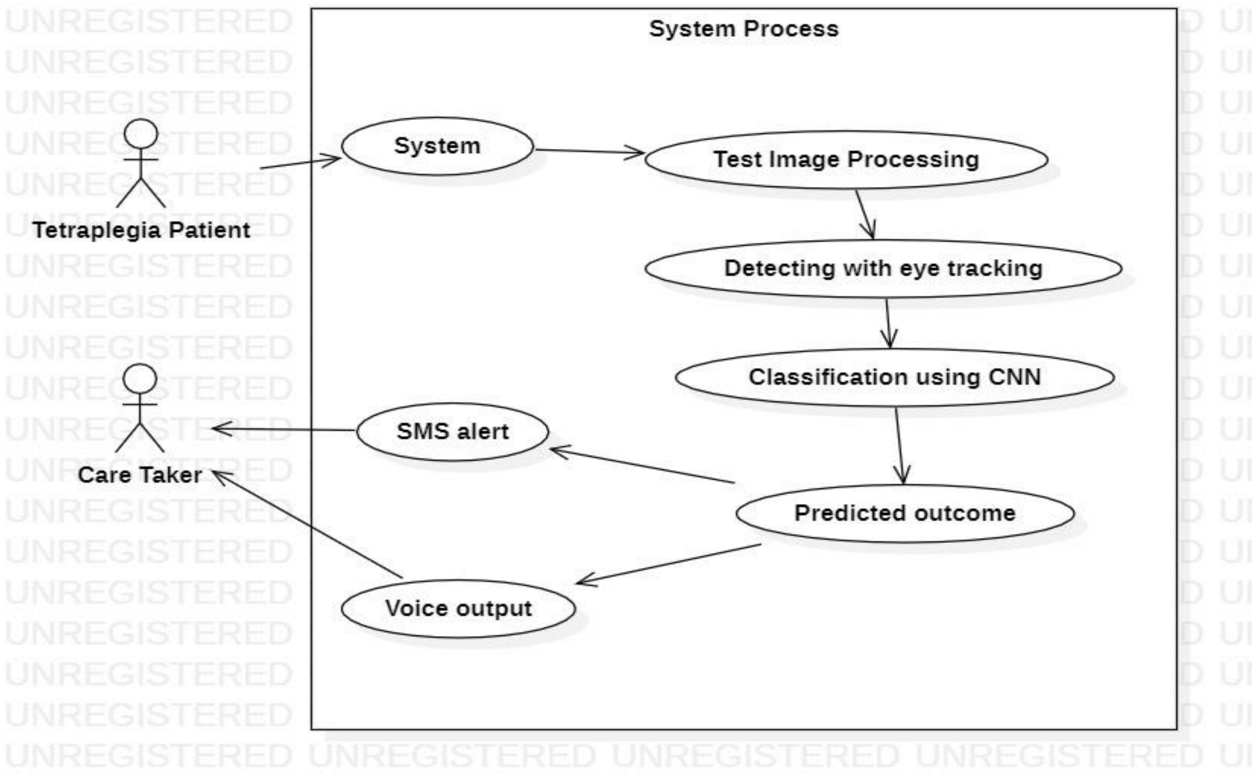


Fig 7.1. Usecase Diagram

7.2 SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

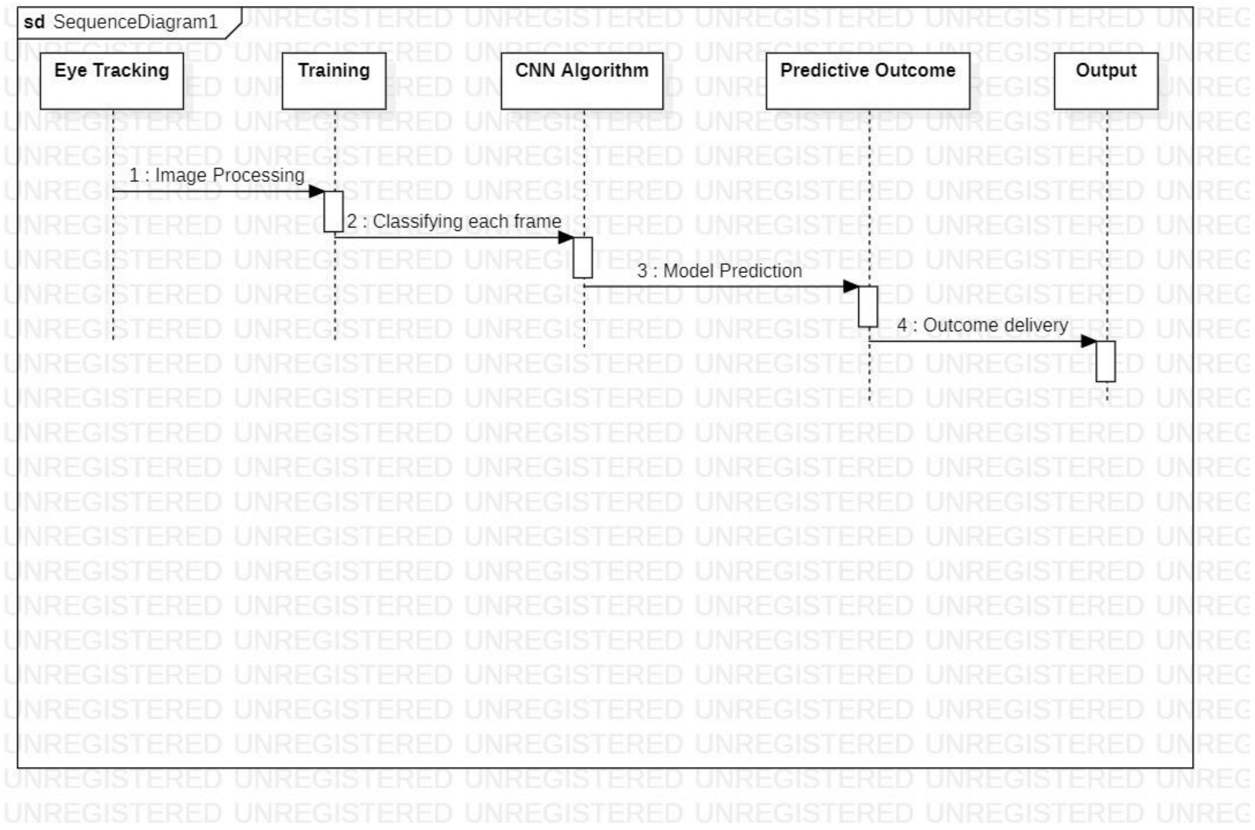


Fig 7.2 Sequence Diagram

7.3 ACTIVITY DIAGRAM

In UML, the activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities. The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

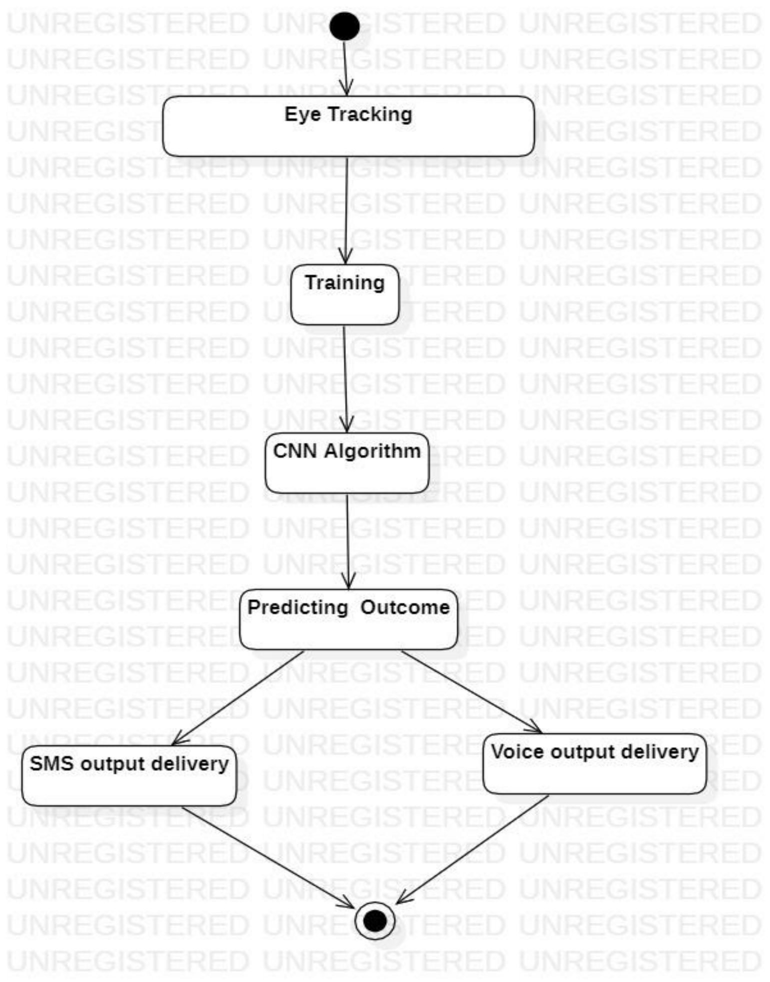


Fig 7.3 Activity Diagram

7.4 CLASS DIAGRAM

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code. It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.

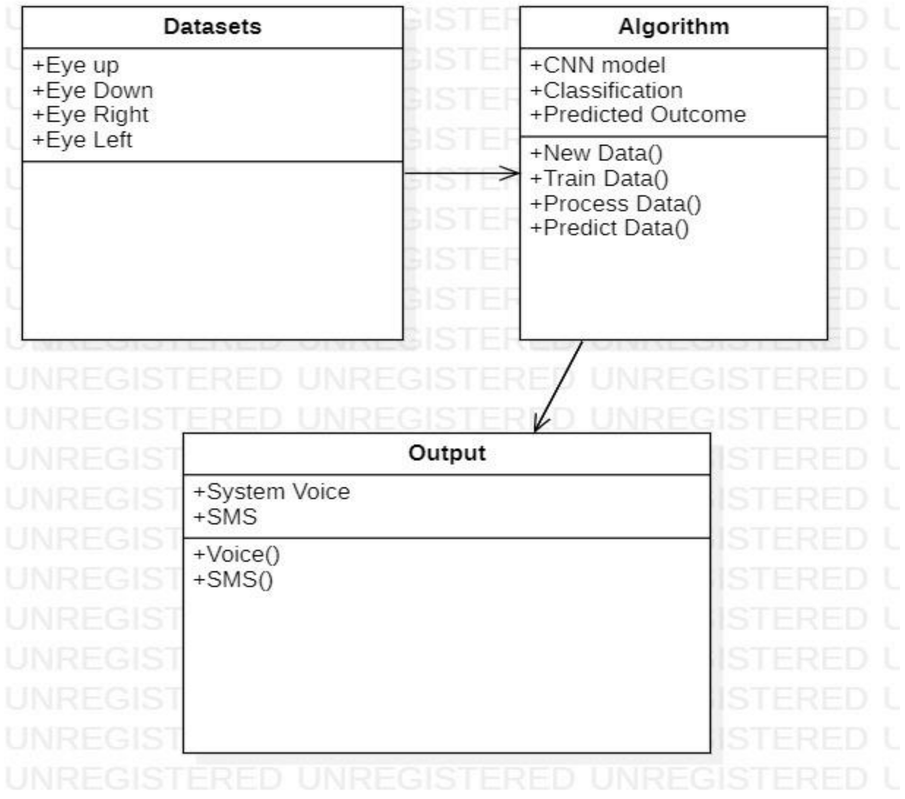


Fig 7.4 Class Diagram

CHAPTER – 8

CONVOLUTIONAL NEURAL NETWORK

8.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional neural network is one of the most popular ANN. It is widely used in the fields of image and video recognition. It is based on the concept of convolution, a mathematical concept. It is almost similar to multi-layer perceptron except it contains series of convolution layer and pooling layer before the fully connected hidden neuron layer.

Three important layers:

- Convolution layer: It is the primary building block and perform computational tasks based on convolution function.
- Pooling layer: It is arranged next to convolution layer and is used to reduce the size of inputs by removing unnecessary information so computation can be performed faster.
- Fully connected layer: It is arranged to next to series of convolution and pooling layer and classify input into various categories.

A simple CNN architecture is represented as below:

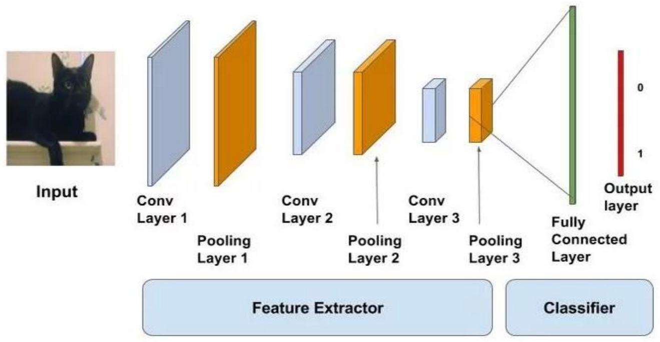


Fig 8.1 CNN Architecture Diagram

Here,

· 2 series of Convolution and pooling layer is used and it receives and process the input (e.g. image).

· A single fully connected layer is used and it is used to output the data (e.g. classification of image)

Convolutional Neural Networks are designed to process data through multiple layers of arrays. This type of neural networks is used in applications like image recognition or face recognition. The primary difference between CNN and any other ordinary neural network is that CNN takes input as a two-dimensional array and operates directly on the images rather than focusing on feature extraction which other neural networks focus on. The dominant approach of CNN includes solutions for problems of recognition. Top companies like Google and Facebook have invested in research and development towards recognition projects to get activities done with greater speed. CNN utilizes spatial correlations that exist within the input data. Each concurrent layer of a neural network connects some input neurons. This specific region is called local receptive field. Local receptive field focusses on the hidden neurons. The hidden neurons process the input data inside the mentioned field not realizing the changes outside the specific boundary. Each connection learns a weight of the hidden neuron with an associated connection with movement from one layer to another. Individual neurons perform a shift from time to time. This process is called “convolution”. The mapping of connections from the input layer to the hidden feature map is defined as “shared weights” and bias included is called “shared bias”. CNN or convolutional neural networks use pooling layers, which are the layers, positioned immediately after CNN declaration. It takes the input from the user as a feature map that comes out of convolutional networks and prepares a condensed feature map. Pooling layers helps in creating layers with neurons of previous layers.

8.2 Layers in CNN

There are five different layers in CNN

- Input layer
- Convo layer (Convo + ReLU)
- Pooling layer
- Fully connected(FC) layer
- Softmax/logistic layer
- Output layer

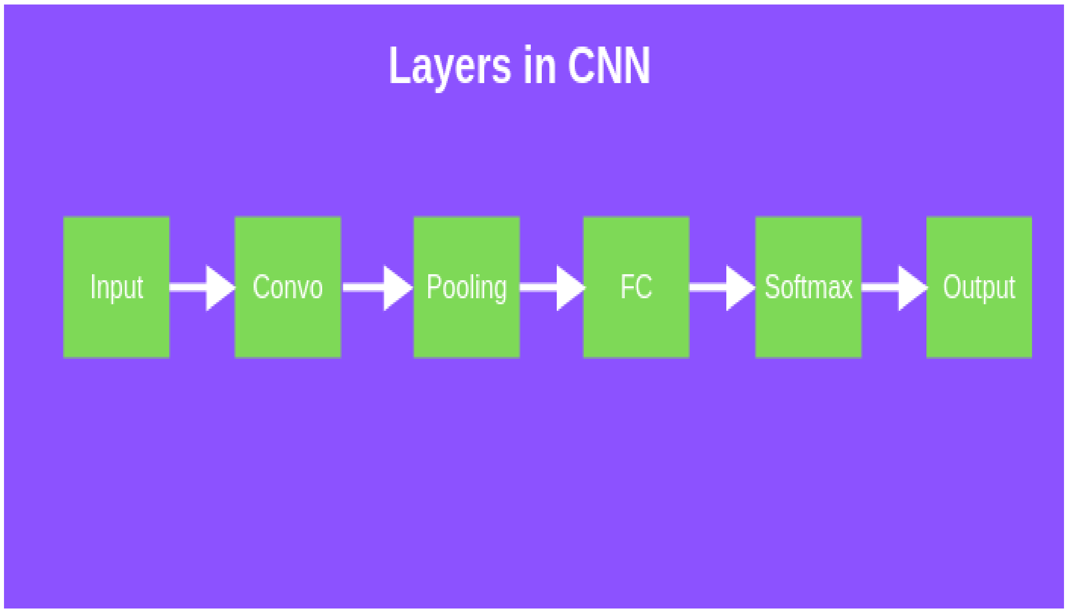


Fig 8.2 Block diagram for layers in CNN

Different layers of CNN

8.2.1 Input Layer

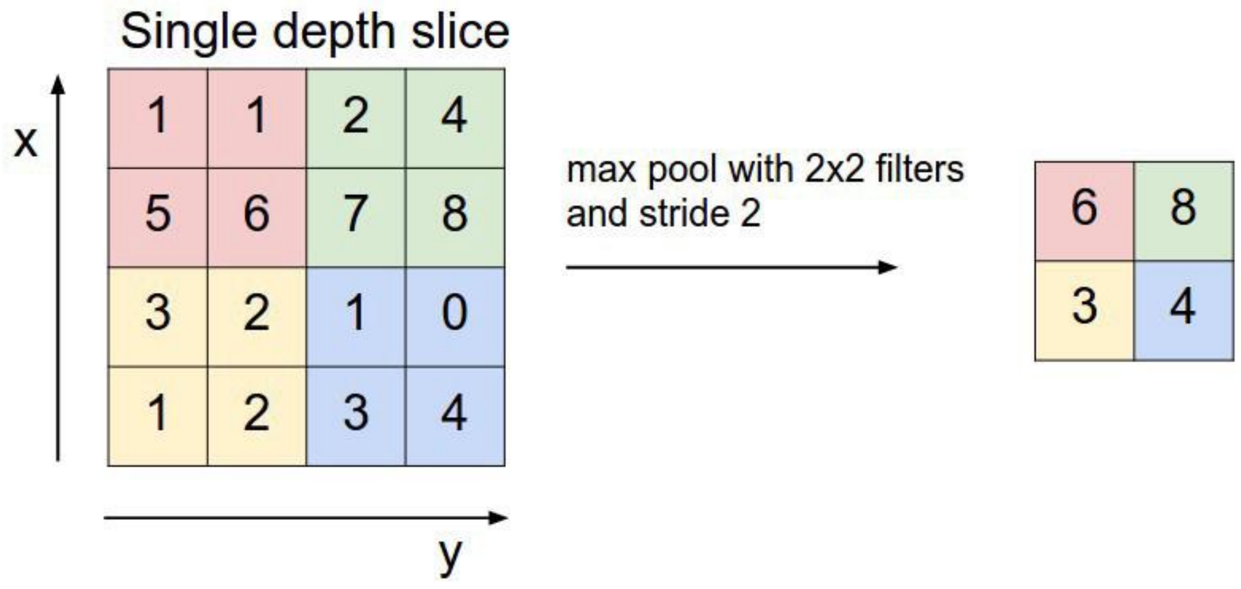
Input layer in CNN should contain image data. Image data is represented by three dimensional matrix and to reshape it into a single column. Suppose having image of dimension 28 x 28 =784, which needs to convert it into 784 x 1 before feeding into input.

8.2.2 Convo Layer

Convo layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. A part of image is connected to Convo layer to perform convolution operation and calculating the dot product between receptive field (it is a local region of the input image that has the same size as that of filter) and the filter. Result of the operation is single integer of the output volume. Then slide the filter over the next receptive field of the same input image by a Stride and do the same operation again will repeat the same process again and again until the whole image is processed. The output will be the input for the next layer.

Convo layer also contains ReLU activation to make all negative value to zero.

8.2.3 Pooling Layer



Pooling layer is used to reduce the spatial volume of input image after convolution. It is used between two convolution layer. Apply FC after Convo layer without applying pooling or max pooling, then it will be computationally expensive. So, the max pooling is only way to reduce the spatial volume of input image. In the above example, max pooling in single depth slice with Stride of 2. By observing the 4 x 4 dimension input which is reduced to 2 x 2 dimension.

There is no parameter in pooling layer but it has two hyper parameters — Filter(F) and Stride(S).

In general, Input dimension will be W1 x H1 x D1, then

$W2 = (W1-F)/S+1$

$H2 = (H1-F)/S+1$

$D2 = D1$

Where W2, H2 and D2 are the width, height and depth of output.

8.2.4 Fully Connected Layer (FC)

Fully connected layer involves weights, biases, and neurons. It connects neurons in one layer to neurons in another layer. It is used to classify images between different categories by training.

8.2.5 Softmax / Logistic Layer

Softmax or Logistic layer is the last layer of CNN. It resides at the end of FC layer. Logistic is used for binary classification and softmax is for multi-classification.

8.2.6 Output Layer

Output layer contains the label which is in the form of one-hot encoded.

CHAPTER – 9

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, Sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectation and does not fail in an acceptable manner. There are various type of testing addresses a specific testing requirement.

TESTING STEPS

- Unit Testing
- Integration Testing
- User Acceptance Testing

TYPES OF TESTS

9.1 UNIT TESTING

Unit testing involves the design of test cases to validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies and knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

9.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is an event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactions, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problem that arise from the combination of components. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of integration test is to check that components or web applications, e.g. components in a software system or – one step up – web applications of the company level – interact without error. Test Results: All the test cases are mentioned above passed successfully. No defects encountered.

9.3 USER ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

CHAPTER – 10

ADVANTAGE AND APPLICATION

10.1 ADVANTAGES:

- Improved accuracy: AI algorithms can enhance the accuracy of eye tracking technology, making it more precise and reliable.
- Greater speed: AI-enhanced eye tracking can process eye movements faster than traditional eye tracking technology, providing real-time feedback and enabling faster interactions with digital devices.
- Increased adaptability: AI algorithms can adapt to individual differences in eye movements, providing more personalized and customized experiences.
- Better usability: AI-enhanced eye tracking can improve the usability of digital devices, making them more accessible to people with disabilities or impairments.

10.2 APPLICATION:

- An enhanced eye tracking system powered by artificial intelligence can be a game-changer for tetraplegia patients. This system can help these patients to interact with their environment and communicate with others more effectively.
- Communication: An AI-enhanced eye tracking system can be used as an alternative mode of communication for tetraplegia patients. By tracking their eye movements, the system can translate their gaze into text or speech, allowing them to communicate with others without the need for physical movement.

- Environmental control: With the help of an AI-enhanced eye tracking system, tetraplegia patients can control their environment more easily. They can use their gaze to turn on lights, adjust the temperature, or even operate a wheelchair or other assistive devices.
- Rehabilitation: Eye tracking technology can be used in rehabilitation programs to help tetraplegia patients regain control of their body. By tracking their eye movements, therapists can develop exercises that improve the patient's ability to control their gaze and focus on specific objects.
- Overall, an AI-enhanced eye tracking system has the potential to greatly enhance the quality of life for tetraplegia patients, allowing them to communicate more effectively, control their environment, and engage in new activities and experiences.

CHAPTER – 11

CONCLUSION AND FUTURE ENHANCEMENTS

11.1 CONCLUSION

Tetraplegic sufferers may be given new levels of control using eye tracking devices with AI enhancements. By employing eye movements, these people can regain some degree of control over their environment and lives. This may have a substantial positive impact on their quality of life. It is important to carefully analyze the limitations and inadequacies of AI-enhanced eye tracking systems, such as the need for high-quality training data, the computing burden, and the potential for bias. If they are to be dependable and helpful, AI-enhanced eye tracking systems for people with tetraplegia must carefully address these issues. For Tetraplegic patients, AI-enhanced eye tracking systems have immense promise, but more research is needed to fully understand their benefits and limitations as well as to develop reliable and effective treatments for this population.

11.2 FUTURE ENHANCEMENTS

There are several potential future works for AI-enhanced eye tracking systems for quadriplegia patients. Here are a few ideas:

Improve accuracy: One of the primary challenges with eye tracking systems is achieving high levels of accuracy. Researchers could work on developing AI algorithms that can more accurately track eye movements, even when the user has limited mobility or involuntary eye movements.

Expand functionality: Eye tracking systems can be used for a variety of tasks, such as controlling a wheelchair, typing on a computer, or selecting objects on a screen. Researchers could explore ways to expand the functionality of eye tracking systems to allow quadriplegia patients to perform a wider range of tasks.

Develop personalized models: Each quadriplegia patient may have unique eye movements and patterns, so developing personalized AI models could help improve the accuracy and effectiveness of eye tracking systems.

Incorporate other input modalities: While eye tracking is an effective input modality, it may not be suitable for all tasks or situations. Researchers could explore ways to combine eye tracking with other input modalities, such as voice recognition or head movements, to create more robust and flexible systems.

Improve usability: Eye tracking systems can be challenging to use, particularly for those with limited mobility or cognitive impairments. Researchers could work on improving the user interface and overall usability of eye tracking systems to make them more accessible and user-friendly for quadriplegia patients.

Overall, there is a lot of potential for AI-enhanced eye tracking systems to improve the quality of life for quadriplegia patients. By continuing to innovate and improve these systems, which can help empower individuals with disabilities to lead more independent and fulfilling lives.

APPENDIX – A

SAMPLE CODINGS

MAIN MODULE

```
import numpy as np

import os

import sys

import tensorflow as tf

from distutils.version import StrictVersion

from collections import defaultdict

from PIL import Image

from object_detection.utils import ops as utils_ops

from scipy.spatial import distance as dist

from imutils import face_utils

import imutils

import dlib

import cv2

import time

import pyttsx3

tts = pyttsx3.Engine()
```

```
def talking_tom(text):
  tts.say(text)
  tts.runAndWait()
import os
from twilio.rest import Client
account_sid = 'ACf9cbb53022d0a4c13a5686f4bda6dee3'
auth_token = '44e5fffbbf018b4779f6bfac1612a10c'
client1 = Client(account_sid, auth_token)
def sms1():
    message = client1.messages \
              .create(
                  body='cancel',
                  from_='+15856326443',
                  to='+916374005802'
                  )
    print(message.sid)
def sms2():
    message = client1.messages \
              .create(
                  body='water',
                  from_='+15856326443',
                  to='+916374005802'
                  )
    print(message.sid)
def sms3():
    message = client1.messages \
              .create(
                  body='food',
                  from_='+15856326443',
                  to='+916374005802'
                  )
    print(message.sid)
def sms4():
    message = client1.messages \
              .create(
                  body='refreshment',
                  from_='+15856326443',
                  to='+916374005802'
            )
    print(message.sid)


def camera():
  # This is needed since the notebook is stored in the object_detection folder.
  sys.path.append("..")
  if StrictVersion(tf.__version__) < StrictVersion('1.9.0'):
    raise ImportError('Please upgrade your TensorFlow installation to v1.9.* or
later!')
  from utils import label_map_util
  from utils import visualization_utils as vis_util
  MODEL_NAME = 'inference_graph'
  PATH_TO_FROZEN_GRAPH = MODEL_NAME +
'/frozen_inference_graph.pb'
  PATH_TO_LABELS = 'training/labelmap.pbtxt'
  detection_graph = tf.Graph()
  with detection_graph.as_default():
    od_graph_def = tf.GraphDef()
    with tf.gfile.GFile(PATH_TO_FROZEN_GRAPH, 'rb') as fid:
      serialized_graph = fid.read()
      od_graph_def.ParseFromString(serialized_graph)

      tf.import_graph_def(od_graph_def, name='')

  category_index =
label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,
use_display_name=True)

  def run_inference_for_single_image(image, graph):

      if 'detection_masks' in tensor_dict:

          # The following processing is only for single image

          detection_boxes = tf.squeeze(tensor_dict['detection_boxes'], [0])

          detection_masks = tf.squeeze(tensor_dict['detection_masks'], [0])

          # Reframe is required to translate mask from box coordinates to image
coordinates and fit the image size.

          real_num_detection = tf.cast(tensor_dict['num_detections'][0], tf.int32)

          detection_boxes = tf.slice(detection_boxes, [0, 0], [real_num_detection, -1])

          detection_masks = tf.slice(detection_masks, [0, 0, 0], [real_num_detection, -
1, -1])

          detection_masks_reframed =
utils_ops.reframe_box_masks_to_image_masks(

              detection_masks, detection_boxes, image.shape[0], image.shape[1])

          detection_masks_reframed = tf.cast(
              tf.greater(detection_masks_reframed, 0.5), tf.uint8)
          # Follow the convention by adding back the batch dimension
          tensor_dict['detection_masks'] = tf.expand_dims(
              detection_masks_reframed, 0)
      image_tensor = tf.get_default_graph().get_tensor_by_name('image_tensor:0')
      # Run inference
      output_dict = sess.run(tensor_dict,
                             feed_dict={image_tensor: np.expand_dims(image, 0)})
      # all outputs are float32 numpy arrays, so convert types as appropriate
      output_dict['num_detections'] = int(output_dict['num_detections'][0])
      output_dict['detection_classes'] = output_dict[
          'detection_classes'][0].astype(np.uint8)
      output_dict['detection_boxes'] = output_dict['detection_boxes'][0]
      output_dict['detection_scores'] = output_dict['detection_scores'][0]
      global a2
      if 'detection_masks' in output_dict:
          output_dict['detection_masks'] = output_dict['detection_masks'][0]
      return output_dict
  a1=0
```

```
 a2=0

 import cv2

 cap = cv2.VideoCapture(0)

 try:

     with detection_graph.as_default():

         with tf.Session() as sess:

                 # Get handles to input and output tensors

                 ops = tf.get_default_graph().get_operations()

                 all_tensor_names = {output.name for op in ops for output in
op.outputs}

                 tensor_dict = {}

                 for key in [

                   'num_detections', 'detection_boxes', 'detection_scores',

                   'detection_classes', 'detection_masks'

                 ]:

                     tensor_name = key + ':0'

                     if tensor_name in all_tensor_names:

                         tensor_dict[key] = tf.get_default_graph().get_tensor_by_name(

                       tensor_name)
```

```
co=0

l=[]

while True:

  (__, image_np) = cap.read()

  # Expand dimensions since the model expects images to have shape:
[1, None, None, 3]

  image_np_expanded = np.expand_dims(image_np, axis=0)

  cv2.imwrite('capture.jpg',image_np)

  # Actual detection.

  output_dict = run_inference_for_single_image(image_np,
detection_graph)

  if output_dict['detection_scores'][0] > 0.80:

   l.append(output_dict['detection_classes'][0])

   co+=1

  # Visualization of the results of a detection.

  vis_util.visualize_boxes_and_labels_on_image_array(

    image_np,

    output_dict['detection_boxes'],

    output_dict['detection_classes'],

    output_dict['detection_scores'],
    category_index,
    instance_masks=output_dict.get('detection_masks'),
    use_normalized_coordinates=True,
    line_thickness=8)
cv2.imshow('object_detection', cv2.resize(image_np,(800,600)))
if co>5:
  temp = max(l,key=l.count)
  if temp == 1:
    print("Down eye")
    talking_tom("cancel")
    sms1()
    blink()
  if temp == 2:
    print("left eye")
    talking_tom("water")
    sms2()
    blink()
  if temp ==3:
    print("right eye")
```

```
                  talking_tom("food")
                  sms3()
                  blink()
                 if temp == 4:
                  print("Up eye")
                  talking_tom("refreshment")
                  sms4()
                  blink()
                 co=0
                 l=[]
                if cv2.waitKey(1)& 0xFF == ord('q'):
                  cap.release()
                  cv2.destroyAllWindows()
                  break
except Exception as e:
    print(e)
    #cap.release()
```

```
def blink():
  def eyeAspectRatio(eye):
      A = dist.euclidean(eye[1], eye[5])
      B = dist.euclidean(eye[2], eye[4])
      C = dist.euclidean(eye[0], eye[3])
      ear = ((A + B) / (C))
      return ear
  y, x = [], []
  i=0
  j=0
  count = 0
  earThresh = 1 #distance between vertical eye coordinate Threshold
  earFrames = 3 #consecutive frames for eye closure
  shapePredictor = "shape_predictor_68_face_landmarks.dat"
  cam = cv2.VideoCapture(0)
  print("blink your eyes")
  detector = dlib.get_frontal_face_detector()
```

```
predictor = dlib.shape_predictor(shapePredictor)


#get the coord of left & right eye
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
while True:
    _, frame = cam.read()
##    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    rects = detector(gray, 0)
    for rect in rects:
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        leftEAR = eyeAspectRatio(leftEye)
        rightEAR = eyeAspectRatio(rightEye)
        ear = (leftEAR + rightEAR)
        leftEyeHull = cv2.convexHull(leftEye)
```

```
            rightEyeHull = cv2.convexHull(rightEye)

            cv2.drawContours(frame, [leftEyeHull], -1, (0, 0, 255), 1)

            cv2.drawContours(frame, [rightEyeHull], -1, (0, 0, 255), 1)

            if ear < earThresh:

                count += 1

                print(count)

                if count == 3:

                    camera()

                    break

    ##          else:

    ##              count = 0

        cv2.imshow("Frame", frame)

        key = cv2.waitKey(1) & 0xFF

        if key == ord("q"):

            break

    cam.release()

    cv2.destroyAllWindows()

blink()
```

TRAIN MODULE

```
import functools

import json

import os

import tensorflow as tf


from object_detection.builders import dataset_builder

from object_detection.builders import graph_rewriter_builder

from object_detection.builders import model_builder

from object_detection.legacy import trainer

from object_detection.utils import config_util


tf.logging.set_verbosity(tf.logging.INFO)


flags = tf.app.flags

flags.DEFINE_string('master', '', 'Name of the TensorFlow master to use.')

flags.DEFINE_integer('task', 0, 'task id')

flags.DEFINE_integer('num_clones', 1, 'Number of clones to deploy per worker.')
flags.DEFINE_boolean('clone_on_cpu', False,
                     'Force clones to be deployed on CPU.  Note that even if '
                     'set to False (allowing ops to run on gpu), some ops may '
                     'still be run on the CPU if they have no GPU kernel.')
flags.DEFINE_integer('worker_replicas', 1, 'Number of worker+trainer '
                     'replicas.')
flags.DEFINE_integer('ps_tasks', 0,
                     'Number of parameter server tasks. If None, does not use '
                     'a parameter server.')
flags.DEFINE_string('train_dir', '',
                    'Directory to save the checkpoints and training summaries.')

flags.DEFINE_string('pipeline_config_path', '',
                    'Path to a pipeline_pb2.TrainEvalPipelineConfig config '
                    'file. If provided, other configs are ignored')

flags.DEFINE_string('train_config_path', '',
                    'Path to a train_pb2.TrainConfig config file.')
flags.DEFINE_string('input_config_path', '',
                    'Path to an input_reader_pb2.InputReader config file.')
flags.DEFINE_string('model_config_path', '',
                    'Path to a model_pb2.DetectionModel config file.')

FLAGS = flags.FLAGS


@tf.contrib.framework.deprecated(None, 'Use object_detection/model_main.py.')
def main(_):
  assert FLAGS.train_dir, '`train_dir` is missing.'
  if FLAGS.task == 0: tf.gfile.MakeDirs(FLAGS.train_dir)
  if FLAGS.pipeline_config_path:
    configs = config_util.get_configs_from_pipeline_file(
        FLAGS.pipeline_config_path)
    if FLAGS.task == 0:
      tf.gfile.Copy(FLAGS.pipeline_config_path,
                    os.path.join(FLAGS.train_dir, 'pipeline.config'),
                    overwrite=True)
  else:
```

```
  configs = config_util.get_configs_from_multiple_files(
      model_config_path=FLAGS.model_config_path,
      train_config_path=FLAGS.train_config_path,
      train_input_config_path=FLAGS.input_config_path)
  if FLAGS.task == 0:
    for name, config in [('model.config', FLAGS.model_config_path),
                         ('train.config', FLAGS.train_config_path),
                         ('input.config', FLAGS.input_config_path)]:
      tf.gfile.Copy(config, os.path.join(FLAGS.train_dir, name),
                    overwrite=True)

model_config = configs['model']
train_config = configs['train_config']
input_config = configs['train_input_config']

model_fn = functools.partial(
    model_builder.build,
    model_config=model_config,
    is_training=True)
```

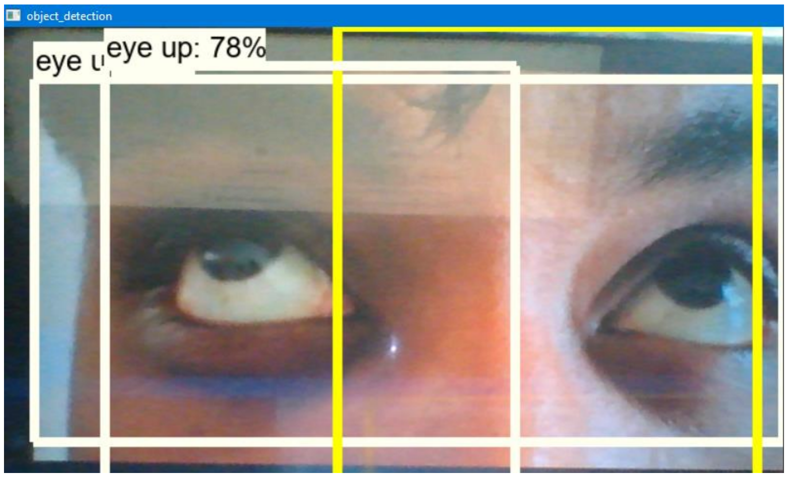
```
def get_next(config):

  return dataset_builder.make_initializable_iterator(

      dataset_builder.build(config)).get_next()


create_input_dict_fn = functools.partial(get_next, input_config)


env = json.loads(os.environ.get('TF_CONFIG', '{}'))

cluster_data = env.get('cluster', None)

cluster = tf.train.ClusterSpec(cluster_data) if cluster_data else None

task_data = env.get('task', None) or {'type': 'master', 'index': 0}

task_info = type('TaskSpec', (object,), task_data)


# Parameters for a single worker.

ps_tasks = 0

worker_replicas = 1

worker_job_name = 'lonely_worker'

task = 0

is_chief = True
```

```
master = ''


if cluster_data and 'worker' in cluster_data:
  # Number of total worker replicas include "worker"s and the "master".
  worker_replicas = len(cluster_data['worker']) + 1
if cluster_data and 'ps' in cluster_data:
  ps_tasks = len(cluster_data['ps'])


if worker_replicas > 1 and ps_tasks < 1:
  raise ValueError('At least 1 ps task is needed for distributed training.')


if worker_replicas >= 1 and ps_tasks > 0:
  # Set up distributed training.
  server = tf.train.Server(tf.train.ClusterSpec(cluster), protocol='grpc',
                           job_name=task_info.type,
                           task_index=task_info.index)
  if task_info.type == 'ps':
    server.join()
    return
  worker_job_name = '%s/task:%d' % (task_info.type, task_info.index)
  task = task_info.index
  is_chief = (task_info.type == 'master')
  master = server.target

graph_rewriter_fn = None
if 'graph_rewriter_config' in configs:
  graph_rewriter_fn = graph_rewriter_builder.build(
      configs['graph_rewriter_config'], is_training=True)

trainer.train(
    create_input_dict_fn,
    model_fn,
    train_config,
    master,
    task,
    FLAGS.num_clones,
    worker_replicas,
      FLAGS.clone_on_cpu,

      ps_tasks,

      worker_job_name,

      is_chief,

      FLAGS.train_dir,

      graph_hook_fn=graph_rewriter_fn)



if __name__ == '__main__':

  tf.app.run()
```

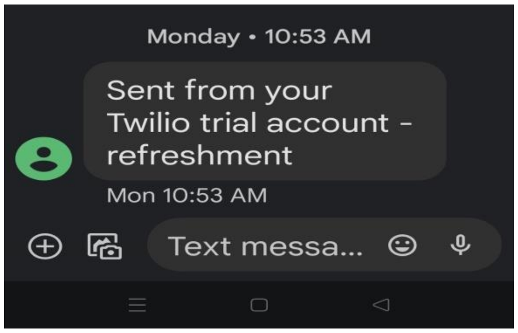
APPENDIX - B

OUTPUT

1) Eye Detection:



2) SMS Output:



Monday • 10:53 AM

Sent from your Twilio trial account - refreshment

Mon 10:53 AM

REFERENCES

[1] An, Y., Wang, L., Xiong, C., & Zhang, X. (2019). A survey of deep learning-based methods for gaze estimation. Journal of Ambient Intelligence and Humanized Computing, 10(1), 107-124.

[2] Benini, L., & De Rossi, D. (2016). Eyetracking for human-computer interaction: A comprehensive survey. ACM Computing Surveys (CSUR), 49(2), 1-40.

[3] Chen, Y., & So, K. (2019). Deep learningbased eye tracking for human-computer interaction. Multimedia Tools and Applications, 78(10), 12473-12498.

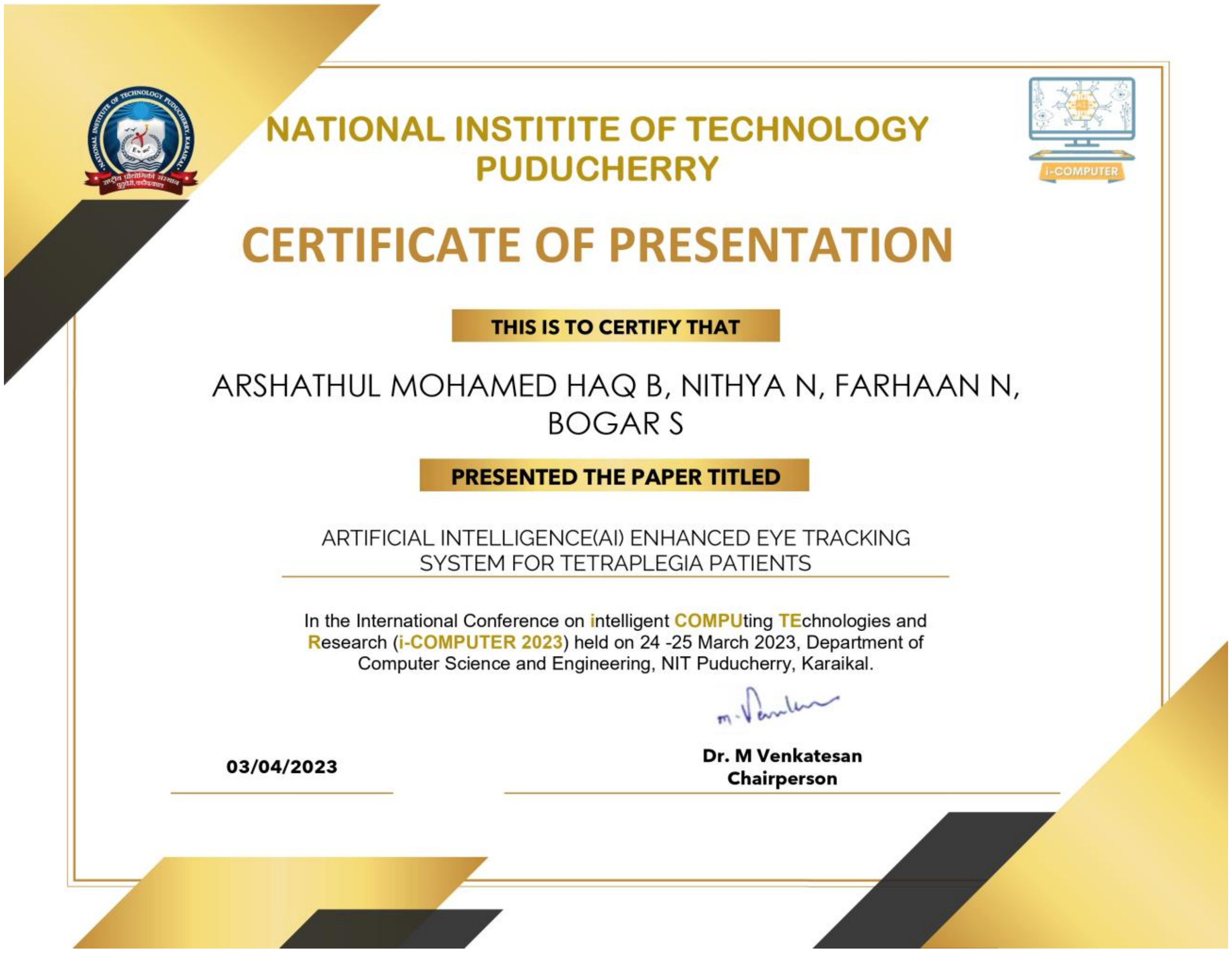
[4] Funes, M. O., & Serrano-Gotarredona, T. (2010). A 128 x 128 1.5

[5] Gaze Flow: A New Quality Metric for Eye Tracking in User Studies (2017) by Mar´ıa Mart´ın-Garc´ıa, Manuel Mart´ınez-Ramon, and ´ Vicente Alavaro-Benavent.

[6] Andersson, R., ¨ Dewhurst, R., Holmqvist, K., Jarodzka, H., Nystrom, M., & Van de Weijer, J. (2011). Eye tracking methodology: Theory and practice. Psychology Press.

[7] An, Y., Wang, L., Xiong, C., & Zhang, X., (2018). Deep learning-based gaze estimation with face alignment and adaptive gain control. In 2018 ACM on Multimedia Conference (MM) (pp. 571-579). ACM.

CERTIFICATE



NATIONAL INSTITITE OF TECHNOLOGY PUDUCHERRY

i-COMPUTER

CERTIFICATE OF PRESENTATION

THIS IS TO CERTIFY THAT

ARSHATHUL MOHAMED HAQ B, NITHYA N, FARHAAN N, BOGAR S

PRESENTED THE PAPER TITLED

ARTIFICIAL INTELLIGENCE(AI) ENHANCED EYE TRACKING SYSTEM FOR TETRAPLEGIA PATIENTS

In the International Conference on intelligent COMPUting TEchnologies and Research (i-COMPUTER 2023) held on 24 -25 March 2023, Department of Computer Science and Engineering, NIT Puducherry, Karaikal.

03/04/2023

[illegible]

Dr. M Venkatesan
Chairperson