A

PROJECT REPORT

ON

# CURSOR CONTROL USING EYE MOVEMENTS

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING
INFORMATION TECHNOLOGY

**BY**

**Sahil Katkamwar - 33331**
**Rahul Kamble - 33330**
**Shreyash Ghanekar - 33316**
**Swapnil Badave - 33301**

Under the guidance of
**Mrs. A. G. Ghule**

DEPARTMENT OF INFORMATION TECHNOLOGY
PUNE INSTITUTE OF COMPUTER TECHNOLOGY
SR. NO 27, PUNE-SATARA ROAD, DHANKAWADI
PUNE - 411 043.
AY: 2024-2025

SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY
DEPARTMENT OF INFORMATION TECHNOLOGY

# C E R T I F I C A T E

This is to certify that the Project work entitled
**Cursor Control Using Eye Movements**
Submitted by

**Sahil Katkamwar - 33331**
**Rahul Kamble - 33330**
**Shreyash Ghanekar - 33316**
**Swapnil Badave - 33301**

is a bonafide work carried out under the supervision of Mrs. A. G. Ghule and it is submitted towards the partial fulfillment of the requirements of Savitribai Phule Pune University, Pune for the award of the degree of Bachelor of Engineering (Information Technology).

Mrs. A. G. Ghule
Project Guide

Dr. A. S. Ghotkar
HOD IT

Dr. S. T. Gandhe
Principal

Date:
Place: Pune

# Acknowledgement

We would like to extend my deepest gratitude to Mrs. A. G. Ghule, our project guide, for her invaluable guidance, constructive critiques, and unwavering support throughout the development of this seminar. Her expertise and insights greatly shaped the direction of our research, and her encouragement pushed me to approach the topic of bias and bias mitigation in AI with a critical and thorough perspective.
We am sincerely grateful to Mr. A. M. Bagade, the project's reviewer, for his detailed feedback and valuable recommendations, which helped refine and enhance the quality of this work. His thoughtful observations provided a clear direction during the revision process. I also wish to express my appreciation to Dr. Archana Ghotkar, Head of the Department, for her administrative support and the academic resources made available during the course of this seminar. Her leadership and guidance have been key to ensuring a conducive environment for academic development.
Additionally, We would like to acknowledge the academic and technical staff of the department for their support and assistance in various capacities during the research process.Finally, We would like to thank our family and friends for their continuous encouragement and moral support throughout the duration of this project.

**Sahil Katkamwar - 33331**

**Rahul Kamble - 33330**

**Shreyash Ghanekar - 33316**

**Swapnil Badave - 33301**

# Abstract

Among the new human-computer interaction technologies such as eye tracking and gesture recognition, the human-computer interaction technology based on eye movements improves the intelligence, naturalness, and efficiency of human-computer interaction. This paper investigates the potential of using eye movements to operate a computer mouse cursor, enhancing accessibility for individuals with physical limitations or those seeking alternative input methods. By analyzing eye images to detect gaze direction and eye locations, the feasibility of converting eye movements into cursor actions using Convolutional Neural Networks and Support Vector Machines is explored. The research examines the geometric model of an eye-tracking system with a single camera under natural light, focusing on the optimal placement area of the camera when the user's head is stationary and the maximum allowable rotation range when the user's head is free to move. Key components include data collection from various user eye movements, training machine learning models, and real-time processing for precise cursor control. By addressing these challenges, the research aims to contribute to the creation of more robust and user-friendly eye-tracking systems. This investigation not only explores the technical feasibility but also highlights the transformative potential of eye-tracking technologies in creating more inclusive and accessible computing environments. Ultimately, this work aspires to pave the way for innovative applications in assistive technology, gaming, and beyond, where intuitive and efficient human-computer interaction is paramount.

**Keywords :** Convolutional Neural Networks, Support Vector Machines, Gaze Detection

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | | |
|---|---|---|
| MHI | : | Motion History Image |
| HCI | : | Human Computer Interaction |
| LED | : | Light Emitting Diodes |
| OPEN CV | : | Open Source Computer Visions |
| RGB | : | Red, Green, Blue |
| HSV | : | Hue, saturation and Values |
| GUI | : | Graphical User Interface |
| IDE | : | Integrated Development Environment |
| CNN | : | Convolutional Neural Networks |
| SVM | : | Support Vector Machines |
| KNN | : | K-Nearest Neighbors |
| HMM | : | Hidden Markov ModelsHidden Markov Models |

# 1. Introduction

## 1.1 Introduction

In computing terms, a computer mouse is a device that tracks two-dimensional movements in relation to a surface. This motion is translated into the movement of a cursor on a display, allowing users to interact with a computer's Graphical User Interface. Computers have become an indispensable part of modern life, assisting with a wide range of daily tasks. This widespread use has made HCI a significant area of study. The introduction of the laser mouse in 2004 marked a significant advancement in HCI by improving tracking accuracy. Despite these developments, physical and technical limitations persist. As touch screen technology became prevalent in mobile devices, there has been an increasing demand for similar capabilities in desktop computing. While touch screens for desktops are available, their high cost can be a barrier. This paper explores an alternative approach: using eye-tracking technology to control a computer mouse cursor. By leveraging CNN and SVM to analyze eye movements and determine gaze direction, this study examines the potential of converting these movements into precise cursor actions. Through the collection of eye movement data, training of machine learning models, and real-time processing, this research aims to advance HCI technologies and explore innovative applications across various fields. Moreover, this study seeks to push the boundaries of accessibility, providing a new level of interaction for individuals with physical limitations and offering innovative solutions for hands-free computing environments. By exploring the potential of eye-tracking technologies, this research aims to pave the way for groundbreaking advancements in HCI, shaping the future of how we interact with computers.

## 1.2 Motivation

It is safe to predict that the Virtual the Mouse will soon take the place of the conventional physical in nature mouse in the not-too-distant future, as people strive to live in a world where every technological appliance can be operated and interacted with remotely without the need for any peripheral devices, such as remote controls, keyboards, etc. Not only does it offer ease, but it also saves money.

## 1.3 Objectives

This paper investigates the application of eye-tracking technology for precise cursor control, focusing on several key goals:

1. To achieve high precision in cursor control using eye movements, leveraging advanced machine learning models such as Convolutional Neural Networks and Support Vector Machines.

2. To enhance the accessibility of computer interfaces for individuals with physical limitations by providing a hands-free method of controlling the mouse cursor.

3. To explore the potential of eye-tracking technology in facilitating tasks that require detailed precision, such as designing two-dimensional and three-dimensional models.

4. To assess the feasibility of integrating eye-tracking technology in virtual reality and augmented reality environments.

## 1.4 Scope

The scope of this seminar is to investigate the application of eye-tracking technology to develop a cursor control system that operates without the need for physical touch. In today's world, where we are adapting to life during a pandemic, a touchless cursor control system can significantly reduce the risk of spreading infections through contact with shared devices. By leveraging eye movements to control the computer mouse, this study offers an innovative solution to enhance both safety and efficiency. This system can serve as a replacement for traditional physical computer mice, providing a convenient and accurate method of interaction. The research aims to push the boundaries of HCI by exploring the potential and feasibility of eye-tracking systems, paving the way for broader applications and future developments in this field.

# 2. Literature Survey

The exploration of human-computer interaction (HCI) through eye-tracking technologies has garnered significant attention in recent research, reflecting the growing interest in understanding user behaviors and enhancing interaction methods. Zhou and Zhu conducted a comprehensive examination of various eye-tracking technologies, highlighting their applications within HCI as well as the advancements and challenges that accompany their integration into user interfaces. Their work underscores the importance of these technologies in improving usability and user satisfaction. In a notable study, Stone and Chapman developed a groundbreaking method to dynamically assess user frustration by utilizing both eye and mouse tracking data, providing significant insights into user experience. This method allows for real-time identification of frustration, which can inform adaptive system design to better meet user needs and improve overall interaction quality. Furthermore, M. S., Nirmal, and Kala proposed a novel system enabling users to control the mouse cursor through their eye movements, demonstrating both the feasibility and accuracy of eye-controlled cursor systems in controlled environments. This innovation opens up new avenues for the development of assistive technologies aimed at individuals with disabilities, enhancing accessibility and interaction capabilities. Additionally, Miah, Gulshan, and Jahan investigated the potential of eye gaze as a means for mouse cursor control, presenting a comprehensive model that aims to enhance HCI through advanced eye-tracking technology. Their findings indicate that eye-tracking can significantly improve user interactions and facilitate more intuitive navigation within interfaces. Álvarez Martín et al. introduced an intermittent control model for mouse movements, which provided new insights into the mechanics of cursor control, further contributing to the understanding of user behavior in cursor manipulation. Hucko, Moro, and Bielikova made a valuable contribution by creating a dataset specifically designed for detecting user confusion based on patterns of mouse and eye movements. This dataset is crucial for the development of adaptive user interfaces that can respond to user states, ultimately enhancing overall usability and user satisfaction. Finally, Shah, Punde, Dubey, and Sonawane explored various methods for controlling the mouse pointer through eye movements, highlighting practical applications and identifying potential improvements that could make eye-tracking technology more effective and user-friendly. Overall, this body of work illustrates the rapid advancements in eye-tracking technologies and their implications for enhancing user experience, adaptive interfaces, and the transformative potential of eye-tracking in HCI. As researchers continue to investigate and refine these technologies, it is clear that eye-tracking will play an increasingly significant role in shaping the future of human-computer interaction, fostering deeper engagement and more intuitive user experiences.

# 3. Methodologies

The proposed cursor control system using eye movements is based on frames captured by the webcam in a laptop or PC. By utilizing the Python computer vision library OpenCV, a video capture object is created, and the webcam starts capturing video. The webcam captures and passes the frames to the AI system for further processing. Each frame is captured continuously until the termination of the program. The video frames are converted from BGR to RGB color space to detect and track the user's eye movements frame by frame. This processing enables the system to analyze gaze direction, eye position, and blink patterns, translating these into precise cursor actions for hands-free computer interaction.

## 3.1 Framework/Basic Architecture

**Cursor Control using Eye Movements** is a system leveraging computer vision and machine learning to replicate mouse functions through eye gestures. The main components include the gaze detection module, which tracks the user's eye using techniques like color detection and Convolutional Neural Networks, focusing on the eye region. . The core architecture of the Virtual Mouse can be broken down into the following major components:

### 3.1.1 Input Device (Camera/Webcam)

The primary input for the virtual mouse system is a live video feed from a camera or webcam. This device captures real-time images of the user's eyes and surrounding environment, which will be processed by the system.

- The webcam captures continuous frames of the eyes.
- The video feed is transmitted for further processing.

### 3.1.2 Gaze Detection Module

The gaze detection module is responsible for identifying and tracking the user's eye from the video feed.

- **Techniques used:**
  - *Color detection* (e.g., detecting eye color).
  - *Deep learning models* such as Convolutional Neural Networks (CNNs).

- Filters out unnecessary background, focusing on the eye region.

### 3.1.3 Gesture Recognition Module

Once the eye movements are detected, the gesture recognition module identifies specific eye gestures such as single eye blink, double eye blink, and other actions that correspond to mouse functions.

- **Machine learning techniques used:**
  - Convolutional Neural Networks (CNN).
  - Support Vector Machines (SVMs).
  - Neural Networks.
- Recognizes eye gestures for actions like clicking, scrolling, and dragging.

### 3.1.4 Feature Extraction

The system extracts key features from eye movements, such as:

- Gaze direction.
- Eye position.
- Blink patterns.

These features are then translated into corresponding mouse actions, enabling intuitive control through eye movements.

### 3.1.5 Cursor Movement and Control Module

After recognizing eye movements, this module translates them into mouse commands.

- Gaze movement is mapped to cursor movement.
- To implement cursor control using eye movements, consider using a blink of one eye for a left-click, blinking both eyes simultaneously for a right-click, and moving the eyes upward or downward for scrolling. These intuitive gestures minimize physical strain and enhance user-friendliness.

The virtual mouse behaves like a physical mouse for smooth and precise control.

### 3.1.6 Integration with Operating System (OS)

The final output of the cursor control system integrates with the OS to mimic the actions of a traditional mouse.

- Control signals generated by gesture recognition are passed to the OS.
- The OS responds by controlling the cursor, executing clicks, scrolling, or dragging.

### 3.1.7 Feedback Loop

A feedback loop can adjust the system's sensitivity and accuracy over time.

- Adjusts sensitivity based on the performance of gesture recognition.
- Improves overall system accuracy by learning from inaccuracies.

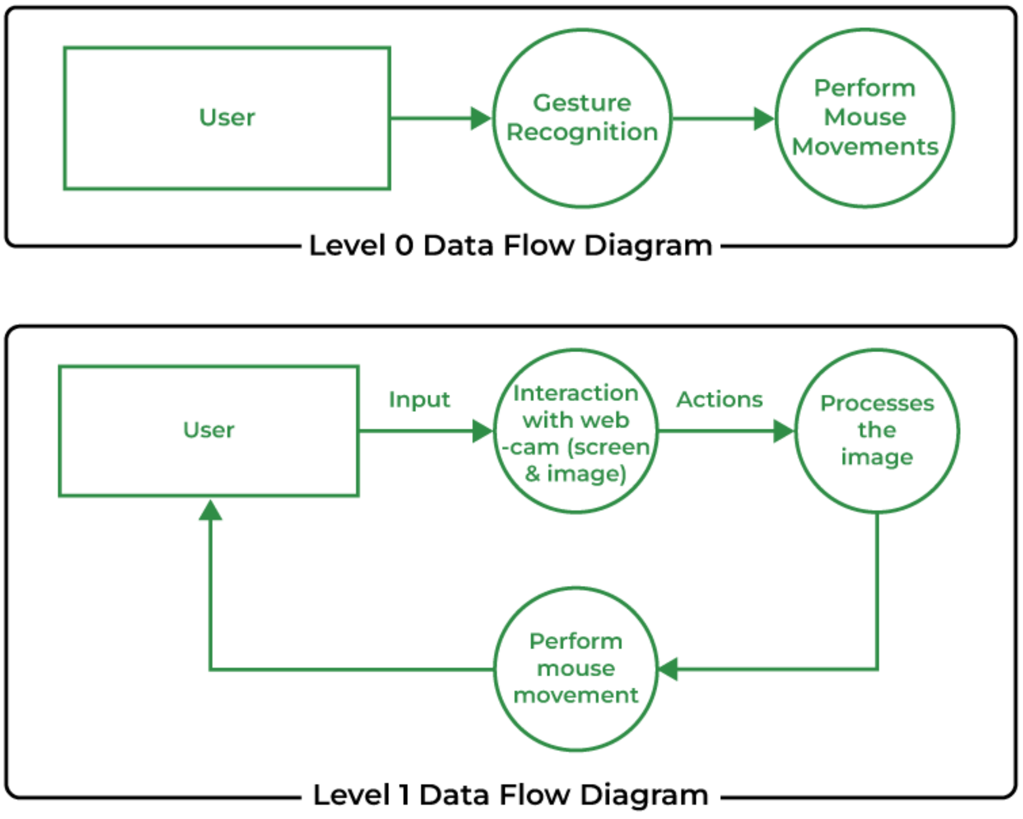

**Figure 3.1:** Work Flow in Virtual Mouse System

## 3.2 Different Approaches

In the development of the Artificial Intelligence (AI) Virtual Mouse, several approaches have been explored to enhance the accuracy and efficiency of hand gesture recognition and control.

The following are some of the prominent approaches used:

### 3.2.1 Deep Learning and Traditional Algorithms

Combining deep learning models like Convolutional Neural Networks with traditional algorithms for more nuanced and reliable eye tracking.

- Advantages:
  - Enhanced accuracy through multi-layered analysis.
  - Ability to handle complex scenarios and diverse user environments.
- Limitations:
  - Requires extensive computational power.
  - Longer training and processing times.

### 3.2.2 Electrooculography (EOG) and Image Processing

Using EOG to measure electrical activity of the eye in conjunction with image processing techniques for comprehensive eye tracking.

- Advantages:
  - Highly accurate in detecting eye movements.
  - Provides detailed data on eye activity.
- Limitations:
  - Involves attaching electrodes around the eyes, which can be cumbersome.
  - Increased complexity in implementation and user setup.

### 3.2.3 Infrared Eye Tracking

Using infrared eye tracking for precise eye movement detection combined with machine learning for interpreting gaze patterns.

- Advantages:
  - High accuracy and precision.
  - Effective in low-light conditions.
- Limitations:

  - Requires specialized hardware.
  - Higher costs and less accessibility for general users.

### 3.2.4 Hybrid Approaches

Combining multiple techniques can often yield better results:

- Integration of Image Processing and Machine Learning: Using traditional image processing for initial eye detection followed by machine learning for gaze and gesture classification.
- Advantages:
  - Increased robustness and accuracy.
  - Reduces the impact of limitations of individual methods.
- Limitations:
  - Increased complexity in system design.
  - Potentially higher processing requirements.

## 3.3 State-of-the-art Algorithms

In the realm of Artificial Intelligence (AI) for cursor control using eye movements, several state-of-the-art algorithms are employed for gaze detection and interaction. Below, we outline key algorithms and provide a complexity analysis for each.

### 3.3.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks have become the backbone of image recognition tasks, including eye movement detection.

- Description: CNNs use convolutional layers to automatically extract features from images, followed by fully connected layers to classify the gestures.
- Complexity Analysis:
  - Time Complexity: The time complexity can be approximated as $O(N \cdot W^2 \cdot F^2 \cdot C)$, where $N$ is the number of training samples, $W$ is the input width and height, $F$ is the filter size, and $C$ is the number of filters.

  - Space Complexity: The space complexity primarily depends on the number of parameters, typically $O(K \cdot D)$, where $K$ is the number of filters and $D$ is the dimensions of the input.

### 3.3.2 Support Vector Machines (SVMs)

SVMs are supervised learning models commonly used for classification tasks, including gesture recognition.

- Description: SVMs find a hyperplane that best separates different classes in the feature space, maximizing the margin between classes.
- Complexity Analysis:
  - Time Complexity: The time complexity is generally $O(N^2 \cdot D)$ for training, where $N$ is the number of training samples and $D$ is the number of features. In practice, it may be $O(N^3)$ due to the use of quadratic programming.
  - Space Complexity: The space complexity is $O(N)$, as it requires storing the support vectors.

### 3.3.3 Hidden Markov Models (HMMs)

HMMs are statistical models used for time series analysis and sequential data.

- Description: HMMs model the probability of sequences of observations, making them suitable for recognizing gestures that unfold over time.
- Complexity Analysis:
  - Time Complexity: The time complexity for the Baum-Welch algorithm (training) is $O(N \cdot T \cdot K^2)$, where $N$ is the number of observations, $T$ is the number of states, and $K$ is the number of possible observations.
  - Space Complexity: The space complexity is $O(K \cdot T)$ for storing the state transition probabilities and observation probabilities.

### 3.3.4 K-Nearest Neighbors (KNN)

KNN is a simple, non-parametric algorithm used for classification based on the closest training examples in the feature space.

- Description: KNN classifies a sample based on the majority class among its $k$ nearest neighbors.

- Complexity Analysis:
  - Time Complexity: The time complexity for prediction is $O(N \cdot D)$, where $N$ is the number of training samples and $D$ is the dimensionality of the data. The training time is negligible as there is no explicit training phase.
  - Space Complexity: The space complexity is $O(N)$ for storing the training dataset.

### 3.3.5 Random Forest

Random Forest is an ensemble learning method that uses multiple decision trees for classification and regression.

- Description: It creates a multitude of decision trees during training and outputs the mode of their predictions.
- Complexity Analysis:
  - Time Complexity: The training time complexity is $O(N \cdot D \cdot \log(N))$ for each tree, multiplied by the number of trees $T$, resulting in $O(T \cdot N \cdot D \cdot \log(N))$.
  - Space Complexity: The space complexity is $O(T \cdot D)$ for storing the trees.

In summary, the choice of algorithm significantly affects the performance and efficiency of the AI system for cursor control using eye movements. Understanding these algorithms and their complexities is crucial for selecting the appropriate method for eye gesture recognition and ensuring optimal system performance.

## 3.4 Discussion

In this section, we explore the various approaches and algorithms employed in the development of AI systems for cursor control using eye movements. By analyzing the strengths and weaknesses of these methodologies, we can better understand their applicability and effectiveness in real-world scenarios.

### 3.4.1 Comparison of Algorithms

Different algorithms exhibit unique advantages and challenges, making them suitable for specific applications. Table 3.1 summarizes key features, advantages, and limitations of the discussed algorithms.

| Algorithm | Advantages | Limitations | Best Use Case |
|---|---|---|---|
| **CNN** | High accuracy in image recognition | Requires large datasets and computational resources | Gaze detection and classification |
| **SVM** | Effective in high-dimensional spaces | Sensitive to noise and outliers | Classifying distinct eye gestures |
| **HMM** | Good for sequential data | Assumes independence of observations | Real-time gaze and gesture recognition |
| **KNN** | Simple to implement | Computationally expensive at prediction time | Fast, low-complexity environments |
| **Random Forests** | Robust to overfitting | Less interpretable than other models | Classifying gaze patterns with mixed features |

**Table 3.1:** Comparison of Eye Gesture Recognition Algorithms

### 3.4.2 Analysis of Approaches

The analysis of these algorithms reveals distinct trends and considerations:

- **Data Dependency:** Algorithms like CNNs and Random Forests perform exceptionally well with large datasets, whereas simpler methods like KNN might struggle with high-dimensional data.
- **Real-time Performance:** For applications requiring immediate feedback, HMMs and CNNs are advantageous due to their sequential modeling capabilities, although CNNs can be resource-intensive.
- **Complexity vs. Interpretability:** While complex models such as CNNs and Random Forests often yield higher accuracy, they may lack the interpretability needed for certain applications. In contrast, SVMs and KNN provide clearer insights into decision-making processes.

In conclusion, the choice of algorithm for a cursor control system using eye movements should be dictated by the specific requirements of the application, including factors such as data availability, required accuracy, and system constraints. Balancing these elements will lead to more effective and efficient eye gesture recognition systems that enhance user interaction with technology.

# 4. Implementation

## 4.1 Algorithm/Methodologies

In this section, we outline the straightforward methodologies and algorithms that will be used in the development of the AI Virtual Mouse system. The aim is to create an efficient and user-friendly interface that translates eye movements into mouse commands using basic computer vision techniques.

### 4.1.1 Eye Detection Algorithm

The first step in our methodology is eye detection. We will utilize basic pupil detection and contour detection methods:

- Pupil Detection: We will implement simple techniques to detect the position of the pupil within the eye. This method involves converting the image to grayscale and using thresholding to detect the darker region of the pupil. This approach is computationally light and can operate in real-time.
- Contour Detection: Once we have identified the pupil region, we will use contour detection to outline the shape of the eye. This helps in distinguishing the eye from the background and allows for better tracking of movements.

### 4.1.2 Gaze Tracking Algorithm

For tracking gaze direction, we will use basic techniques that do not require complex models:

- Hough Circle Transform: We will employ Hough Circle Transform to accurately detect and track the circular shape of the pupil.
- Simple Rule-Based Approach: Gaze direction will be recognized based on the relative position of the pupil within the eye. For example, if the pupil is centered, it indicates a neutral gaze, while deviation to the sides indicates looking left or right.

### 4.1.3 Feature Extraction

Basic feature extraction will be implemented to capture essential characteristics of eye movements:

- Pupil Coordinates: The coordinates of the pupil will be extracted from the detected contours, providing information about gaze direction.
- Eye Position: The overall position of the eyes will be tracked to help determine the screen area being controlled.

### 4.1.4 Mapping Gaze to Mouse Commands

This step involves defining how recognized gaze directions translate into mouse actions:

- Cursor Movement: The position of the pupil will be directly mapped to the cursor position on the screen.
- Click Actions: Blinking or holding the gaze on a specific point will be interpreted as a click action.
- Scrolling: Simple eye movements, such as looking up or down, can be mapped to scroll actions.

In summary, this approach emphasizes the use of simple algorithms and methodologies to create an effective AI Virtual Mouse system. By leveraging straightforward techniques, we can ensure a user-friendly experience while maintaining responsiveness and accuracy in gaze tracking.

## 4.2 Proposed Solution

In light of the insights gained from the previous discussion on existing approaches, we propose a solution that leverages straightforward algorithms to develop an efficient AI Virtual Mouse system. Our implementation will focus on using computer vision techniques for real-time eye movement recognition, ensuring accessibility and ease of use.

### 4.2.1 Implementation Steps

The implementation of the AI Virtual Mouse will proceed through the following steps:

**1. Eye Detection**

Using a webcam, we will implement basic pupil detection to identify eye regions within the video feed:

- Convert the video frames to grayscale to facilitate effective pupil detection.
- Apply thresholding to isolate the darker region of the pupil, resulting in a binary mask.
- Utilize contour detection on the binary mask to find and track the pupil's position and shape.

**2. Gaze Tracking**

Tracking gaze direction will involve a combination of Hough Circle Transform and a rule-based approach:

- Apply Hough Circle Transform to detect and track the circular shape of the pupil.
- Implement a simple rule-based system to determine gaze direction based on the relative position of the pupil within the eye.

**3. Feature Extraction**

The system will extract critical features from the detected gaze:

- Capture the coordinates of the pupil to map movements directly to cursor actions on the screen.
- Determine the overall position of the eyes to facilitate accurate mapping of gaze direction to screen locations.

**4. Gaze to Action Mapping**

This step involves translating recognized gaze directions into corresponding mouse actions:

- Map the position of the pupil to the cursor position, allowing for direct movement across the screen.
- Define specific actions (e.g., blinking or holding gaze on a point) to perform mouse clicks.
- Implement simple scrolling actions based on vertical eye movements.

### 4.2.2 User Testing and Feedback

Once the system is implemented, we will conduct user testing to evaluate its performance. Feedback will be gathered on:

- Ease of use and intuitiveness of the gaze recognition system.
- Responsiveness and accuracy of cursor movements and click actions.
- Overall user satisfaction and suggestions for improvement.

### 4.2.3 Future Enhancements

Based on user feedback, future enhancements may include:

- Incorporating machine learning techniques to improve gaze recognition accuracy and adaptability over time.
- Adding support for additional gaze-based actions to expand functionality.
- Developing a mobile application version of the AI Virtual Mouse for greater accessibility.

## 4.3 Software Requirement Specification

## 4.4 Hardware and Software Requirement

### 4.4.1 Hardware and Software Requirement

#### Hardware Requirement

The hardware required to run and create the Virtual Mouse program is as follows:

- **Computer desktop or laptop:** A desktop or laptop will be used to run the visual program that displays what the camera captures. A notebook, which is a small, lightweight, and affordable laptop computer, can be used to promote mobility.
- **System Specifications:**
  - Processor: Core2Dual
  - Main Memory: 4GB RAM
  - Hard Disk: 320GB
  - Display: 14” Monitor
- **Webcam:** A webcam will be used for image processing, allowing the application to process images and determine the positions of individual pixels.

#### Software Requirement

The following software is required for the development of the Virtual Mouse application:

- **Python Language:** The Virtual Mouse application will be coded using Python in Microsoft Visual Studio, an integrated development environment (IDE). Python provides numerous operators, including comparisons, logical operations, bit manipulation, and basic arithmetic.
- **OpenCV Library:** OpenCV is a collection of programming functions for real-time computer vision. It can read image pixel values and recognize eye movements and blinks in real-time.

**Software Specifications:**

- OS: Windows 10 Ultimate 64-bit
- Language: Python
- Tools Used: OpenCV and MediaPipe

### 4.4.2 Verification Plan

The Virtual Mouse system must accurately and consistently identify eye movements provided by users without affecting the performance of other activities. Recognition outcomes may vary depending on environmental conditions like lighting or background colors. Below are factors that may cause errors in the recognition phase:

- a) Real-time images are captured in either dark or overly bright environments.
- b) Real-time images have conflicting background colors.
- c) Users can interact with the software from varying distances (near or far).
- d) Real-time images are rotated either clockwise or counterclockwise.

## 4.5 Result

### 4.5.1 Overview

Using a virtual mouse, the system's webcam is used for tracking eye movements, and eye gesture recognition enables users to control the mouse with the help of eye gestures. Gesture recognition is accomplished using computer vision techniques. To collect data from a live video, OpenCV includes a package called video capture. This project can be applied in a variety of real-time applications, such as computer cursor control and smart televisions running Android, among others. The work is so straightforward that it minimizes the use of external hardware in such a way that eye movements in front of a camera will cause the necessary operation on the screen, even though there are tools like mouse and laser remotes for the same purpose.

### 4.5.2 Moving Environment

When the project runs, the cursor is already in moving mode. Based on the user's eye movements, the cursor will move to perform brightness-increasing and decreasing operations. When the distance between two pupils is at its maximum, the cursor is considered to be in moving mode, signifying the required operation.

The proposed AI virtual mouse system utilizes the camera of a laptop or PC. The webcam begins recording video when the video collection object is created using the machine vision package OpenCV. The recorded footage is processed and sent through the AI Virtual Mouse system.

In this context, **accuracy** refers to the percentage of accurately classified data samples over all the data. It can be calculated using the following equation:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

Where:

- $TP$ = True Positives
- $TN$ = True Negatives
- $FP$ = False Positives
- $FN$ = False Negatives

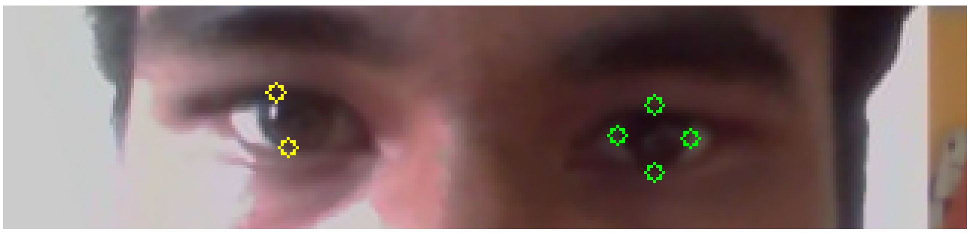

**Figure 4.1:** Mouse Cursor Moving around The computer view

### 4.5.3 Opening File

In our program, we already specify a minimum distance between two pupils. When the user fulfills this, it will mean the cursor is now in clickable mode. Now the user can open a file or folder. The computer is programmed to activate the left mouse button if both pupils are up and the gap between them is less than a specified distance. The virtual mouse performed exceptionally well in terms of precision. The suggested model is unique in that it can execute most mouse tasks and mouse cursor movement using eye detection, and it is also useful in operating the PC in virtual mode like a hardware mouse.

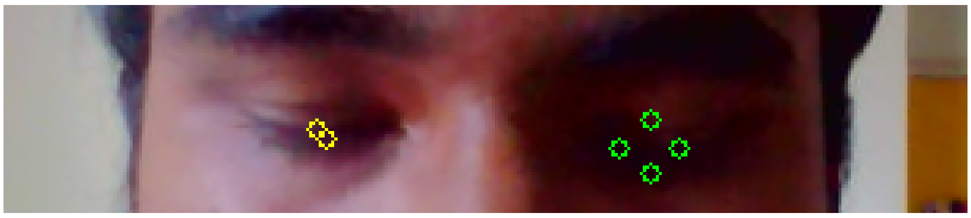

**Figure 4.2:** Button Click Performance

We have tested our Virtual Mouse many times and got a good accuracy which is shown in the following table:

| **AI Virtual Mouse** | **Hand Gesture** | **Accuracy** |
|---|---|---|
| Cursor Move | 199/200 | 99.5% |
| Left Click | 195/200 | 97.5% |
| Right Click | 195/200 | 97.5% |
| Double Click | 190/200 | 95% |

**Table 4.1:** Average accuracy of the defined Virtual Mouse.

### 4.5.4 Accuracy

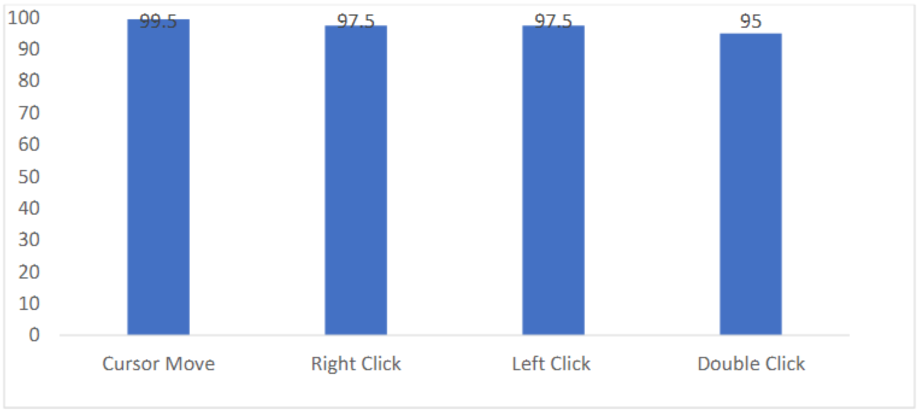

**Figure 4.3:** Graphic Chart of AI Virtual Mouse Accuracy

### 4.5.5 Comparison

Optical mouse illuminates the surface with infrared LED light. While the Laser mouse illuminates the surface with a laser beam. Optical mouse senses the top of the surface it's on. Laser mouse senses peaks and valleys in a surface. Optical mouse works better on mouse pads and non-glossy surface. Laser mouse has no such preferred surfaces.

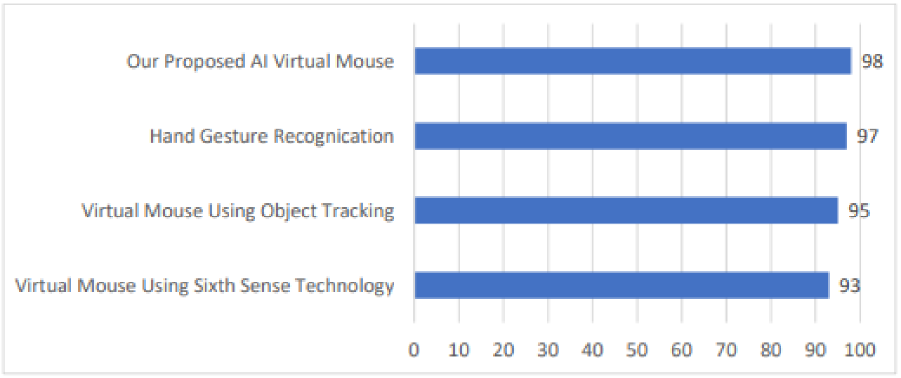

**Figure 4.4:** Graph for Compression between The Models

# 5. Applications

## 5.1 State-of-the-art Applications

Artificial Intelligence (AI) and computer vision technologies have enabled innovative applications of eye-tracking systems for cursor control. By analyzing eye movements, these systems provide touchless, intuitive, and efficient ways for users to interact with technology. Below are some state-of-the-art applications of eye-tracking systems:

### 5.1.1 Touchless Computing Interfaces

Eye-tracking systems offer touchless interfaces, allowing users to control computers without physical contact. In sterile environments like hospitals, this is particularly useful, as it reduces the risk of contamination. For example, surgeons can control medical imaging and diagnostic displays during surgery through eye movements, without needing to break sterility.

### 5.1.2 Gaming and Virtual Reality (VR)

In the gaming and virtual reality industry, eye-tracking technology provides immersive, hands-free interaction. Gamers can aim, navigate, and select objects within a virtual world simply by looking at them. This creates a more natural and fluid experience, enhancing player engagement in VR environments. Eye movements in VR can also enhance user presence by adjusting focus and depth of field based on the player's gaze.

### 5.1.3 Assistive Technology for Disabled Users

Eye-tracking technology offers substantial benefits for individuals with physical disabilities, allowing users to control a computer by directing the cursor with their gaze. This technology enables those with limited mobility to perform tasks such as typing, web browsing, and operating devices without the need for traditional input methods, thus fostering independence and improving quality of life.

### 5.1.4 Smart Home Control

In smart home environments, eye-tracking systems provide users with the ability to control various IoT (Internet of Things) devices such as lighting, climate control, and entertainment systems using their gaze. This interaction offers convenience and hands-free control, making smart homes more intuitive and accessible.

### 5.1.5 Presentation Control in Business and Education

Eye-tracking technology enhances presentations by allowing users to control slideshows or highlight content with simple gaze movements. Presenters can navigate through slides or zoom in on key points without physical devices, which adds a more engaging and hands-free dynamic to business meetings and educational lectures.

### 5.1.6 Medical and Clinical Applications

In clinical environments, eye-tracking technology plays a crucial role in diagnosing neurological disorders, monitoring cognitive functions, and evaluating visual attention. The ability to detect and analyze eye movements can aid in diagnosing conditions such as Parkinson's disease or Alzheimer's, as well as in monitoring patients with cognitive impairments or motor disabilities.

### 5.1.7 Interactive Art Installations

Eye-tracking is also making an impact in the world of interactive art, where artists are creating installations that respond to a viewer's gaze. By focusing on different areas of the artwork, users can influence the visual dynamics, making each interaction unique and providing an engaging, personalized experience.

### 5.1.8 Augmented Reality (AR) Applications

In augmented reality, eye-tracking systems enhance interactions by enabling users to interact with digital content overlaid on the real world. For instance, users can select, move, or adjust digital objects in AR environments just by looking at them. This makes the interaction between physical and virtual worlds more fluid and intuitive, enhancing user experience with AR devices like smart glasses.

### 5.1.9 Retail and Marketing

Eye-tracking systems are being integrated into retail environments to offer customers a personalized and interactive shopping experience. In physical stores, users can browse through products and receive information based on where they look. Online, similar systems allow shoppers to control the interface through gaze, improving convenience and interactivity in e-commerce.

## 5.2 Challenges

Despite the significant advancements in eye-tracking systems for cursor control, several challenges must be addressed to enhance their efficiency, accuracy, and usability. Below are some of the key challenges:

### 5.2.1 Accuracy of Eye Movement Detection

Achieving high accuracy in detecting eye movements is crucial for effective cursor control. Factors such as variations in lighting conditions, reflections, and user fatigue can affect the reliability of eye-tracking algorithms. Ensuring precise detection under diverse conditions remains a significant hurdle.

### 5.2.2 Calibration and User Variability

Eye-tracking systems often require initial calibration to account for individual user differences in eye shape, position, and gaze behavior. Variability among users can lead to discrepancies in detection accuracy, necessitating robust calibration methods that are user-friendly and adaptable.

### 5.2.3 Real-time Processing and Latency

For a seamless user experience, eye-tracking systems must process data in real time. This involves translating eye movements into cursor actions quickly and accurately. However, achieving low latency while maintaining high processing accuracy can be challenging, especially when using complex algorithms.

### 5.2.4 Environmental Influences

Environmental factors, such as ambient light and background distractions, can significantly impact the performance of eye-tracking systems. Consistent lighting conditions are often required for optimal performance, and variations in the environment can lead to unreliable tracking results.

### 5.2.5 User Fatigue and Comfort

Extended use of eye-tracking systems can lead to user fatigue, particularly if the system demands continuous focus or sustained gaze. Enhancing user comfort during prolonged interaction is essential for promoting adoption and ensuring usability in real-world applications.

### 5.2.6 Integration with Existing Interfaces

Integrating eye-tracking systems with existing software applications and operating systems presents challenges in ensuring compatibility and seamless interaction. Developing effective integration solutions is crucial for user acceptance and functionality.

### 5.2.7 Hardware Limitations

The effectiveness of eye-tracking systems can be limited by hardware capabilities, such as camera resolution and processing power. Ensuring that the system performs well across various hardware configurations, from high-end setups to more affordable options, is vital for widespread adoption.

### 5.2.8 Learning Curve and User Acceptance

Users accustomed to traditional input methods, such as keyboards and mice, often encounter significant challenges when transitioning to eye-tracking systems. The learning curve associated with mastering eye movements for effective cursor control can vary among individuals, influenced by prior experience with assistive technologies and the complexity of the interface. Initially, users must develop new motor skills and cognitive strategies to translate their eye movements into accurate cursor actions, which can lead to frustration and resistance, especially for those unfamiliar with such technology. The design of the eye-tracking system plays a crucial role in user acceptance; intuitive interfaces and responsive feedback can facilitate a smoother transition. Additionally, comprehensive training and support resources are essential for helping users navigate the learning curve effectively. By addressing these challenges and fostering a supportive learning environment, developers can enhance user confidence, promote long-term engagement, and improve overall satisfaction with eye-tracking technology in various applications, including gaming, education, and accessibility.

# 6. Conclusion and Future Scope

## 6.1 Overview

In the field of Human-Computer Interfaces (HCI), innovative technologies are redefining interactions, gradually replacing traditional input methods with immersive alternatives. This project developed a machine learning-based cursor control system utilizing eye movements, enabling effective hands-free interaction for users. The primary objective was to allow cursor control through eye movements while maintaining precision and efficiency, achieved by analyzing eye images with algorithms such as Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs). The system optimizes cursor control by accurately detecting gaze direction and eye position, translating these inputs into precise cursor actions and minimizing unintended movements.

Furthermore, the design incorporates a geometric model for optimal camera placement, critical for achieving accurate tracking of eye movements. A calibration phase enhances adaptability by allowing users to personalize the system based on their preferences. In conclusion, the implementation of this machine learning-based cursor control system showcases the potential for advancements in HCI and lays the groundwork for future developments in accessible, touchless user interfaces, ultimately enhancing usability for individuals with physical limitations.

## 6.2 Limitation

Despite its potential, the proposed system has several limitations. The main issue lies in the environment where the color recognition process takes place. The system is highly sensitive to changes in brightness, and extreme lighting conditions (too bright or too dark) can affect the visibility of the targeted colors, thus impacting the accuracy of the color recognition process.

Additionally, distance is another significant limitation. The current color detection mechanism is limited to a 25 cm radius, meaning any color display outside this detection zone is considered noise and ignored. This restriction reduces the flexibility of the system and limits its effectiveness in more dynamic or larger spaces.

## 6.3 Future Work

To enhance the functionality and user experience of the eye-tracking-based cursor control system, several improvements and features should be considered for future iterations of the project:

### 6.3.1 Improved Tracking Algorithms

Implementing advanced tracking algorithms could significantly enhance the accuracy and responsiveness of the system. For instance, incorporating machine learning techniques to better interpret eye movement patterns would allow for more precise cursor positioning and smoother interactions. Additionally, refining the calibration process to accommodate a broader range of user variations could further enhance performance.

### 6.3.2 Integration with Other Input Methods

Future developments could explore integrating eye tracking with other input modalities, such as voice commands or gesture recognition. This multimodal approach would allow users to combine different methods of interaction, making the system more versatile and accessible, particularly for individuals with varying levels of mobility.

### 6.3.3 User Interface and Accessibility Enhancements

Improving the user interface (UI) to be more intuitive and customizable is essential for enhancing usability. Features such as adjustable cursor speed, sensitivity settings, and personalized calibration profiles would allow users to tailor the experience to their preferences. Furthermore, providing tutorials or visual aids could help new users become familiar with the system quickly.

# Bibliography

[1] Jingjing Zhou and Yi Zhu. 2024. Research on the technologies for Human-computer interaction Based on the Eye-tracking. In 5th International Conference on Computer Information and Big Data Applications (CIBDA 2024), April 26–28, 2024, Wuhan, China. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3671151.3671164.

[2] Scott A. Stone and Craig S. Chapman. 2023. Unconscious Frustration: Dynamically Assessing User Experience using Eye and Mouse Tracking. Proc. ACM Hum.-Comput. Interact. 7, ETRA, Article 168 (May 2023), 17 pages. https://doi.org/10.1145/3591137.

[3] M. S., S. Nirmal, and L. S. Kala, ”Eye-Controlled Mouse Cursor,” International Journal of Engineering Technology and Management Sciences, vol. 7, no. 3, pp. 1-6, May-Jun. 2023, doi: 10.46647/ijetms.2023.v07i03.051.

[4] P. Miah, M. R. Gulshan, and N. Jahan, ”Mouse Cursor Movement and Control using Eye Gaze- A Human Computer Interaction,” 2022 International Conference on Artificial Intelligence of Things (ICAIoT), Dhaka, Bangladesh, 2022, pp. 1-6, doi: 10.1109/ICAIoT57170.2022.10121742.

[5] J. Alberto Álvarez Martín, Henrik Gollee, Jörg Müller, and Roderick Murray-Smith. 2021. Intermittent Control as a Model of Mouse Movements. ACM Trans. Comput.-Hum. Interact. 28, 5, Article 35 (August 2021), 46 pages. https://doi.org/10.1145/3461836.

[6] Michal Hucko, Robert Moro, and Maria Bielikova. 2020. Confusion Detection Dataset of Mouse and Eye Movements. In 28th ACM Conference on User Modeling, Adaptation and Personalization (UMAP ’20 Adjunct), July 14–17, 2020, Genoa, Italy. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3386392.3399289.

[7] V. Shah, P. Punde, S. Dubey, and B. Sonawane, ”Mouse Pointer Control Using Eye Movements,” JETIR, vol. 6, no. 4, pp. 1-6, Apr. 2019.