Software Requirements Specification

for

CLICK – AI Virtual Mouse

Version 1.0 approved

Prepared by Sparsh Dagar

KIET Group Of Institutions

22 April 2023

Table of Contents

Table of Contents .....
2 Revision History **2**
1. Introduction **3**
1.1 Purpose 3
1.2 Document Conventions 3
1.3 Intended Audience and Reading Suggestions..... 3
1.4 Product Scope 4
1.5 References 4
2. Overall Description **5**
2.1 Product Perspective 5
2.2 Product Functions 6
2.3 User Classes and Characteristics 6
2.4 Operating Environment 6
2.5 Design and Implementation Constraints 6
2.6 User Documentation 6
2.7 Assumptions and Dependencies 6
3. External Interface Requirements **7**
3.1 User Interfaces 7
3.2 Hardware Interfaces 7
3.3 Software Interfaces 7
3.4 Communications Interfaces 8
4. System Features **8**
4.1 System Feature 1..... 8
4.2 System Feature 2 (and so on) 8
5. Other Nonfunctional Requirements **8**
5.1 Performance Requirements 7
5.2 Safety Requirements 8
5.3 Security Requirements 8
5.4 Software Quality Attributes 8
5.5 Business Rules 8
6. Other Requirements **8**
Appendix A: Glossary **9**
Appendix B: Analysis Models **10**

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

CLICK – AI virtual mouse is a virtual mouse controlled using eye and lip gesture, made using OpenCV and Python. The project introduces an AI-powered virtual mouse, a transformative technology poised to redefine the landscape of human-computer interaction (HCI). Utilizing advanced Artificial Intelligence (AI) and computer vision techniques, the virtual mouse interprets eye & lip gestures and movements, enabling precise and natural control of on-screen cursor dynamics. Through artificial intelligence, the virtual mouse intelligently adapts to user behaviour, tailoring its responses and interactions to individual preferences, thereby enhancing overall usability and productivity. Emphasizing inclusivity, the virtual mouse integrates accessibility features, accommodating a diverse user base. Moreover, its cross-platform compatibility ensures a seamless user experience across various operating systems and devices.

1.2 Document Conventions

Font and Formatting: Use a consistent font and formatting throughout the document to ensure readability and consistency. For example, use a sans-serif font like Arial or Helvetica, and use bold or italic text sparingly to emphasize important points.
Section Headings: Use clear and descriptive headings for each section of the SRS to help readers quickly find the information they need. For example, use "Introduction" for the opening section, "Functional Requirements" for the section describing the system's features, and "Non-functional Requirements" for the section describing performance and quality attributes.
Requirement Numbering: Use a consistent numbering scheme for each requirement statement to facilitate traceability and management. For example, use a hierarchical numbering scheme like "1.1.2" to indicate the section, subsection, and requirement number.
Priority and Importance: Use a clear and consistent priority scheme to indicate the relative importance of each requirement. For example, use "High", "Medium", and "Low" to indicate the urgency and criticality of each requirement.
Dependencies and Relationships: Clearly state any dependencies or relationships between requirements to help readers understand the overall structure and impact of the system. For example, indicate when one requirement is dependent on another, or when a set of requirements are related to a specific feature or function.
Glossary and Definitions: Include a glossary of terms and definitions to clarify any technical or domain-specific terms used in the document. For example, include definitions for terms like "cryptocurrency", "local currency conversions", and "real time value" to ensure a common understanding among all stakeholders.

1.3 Intended Audience and Reading Suggestions

Intended Audience:

Developer: This document will be primarily used by the development team to understand the functional and non-functional requirements of the CLICK – AI Virtual Mouse

Managers: This document will be used by the project managers to plan and monitor the project's progress and ensure that the requirements are met.

Testers: This document will be used by the testing team to develop test cases and ensure that the system meets the specified requirements.

Documentation Writers: This document will be used by the documentation team to create user manuals and other instructional materials.

Reading Suggestions:

For an overview of the system's purpose and scope, start with the Introduction section. For a detailed description of the system's functional requirements, refer to the Functional Requirements section.

For a description of the system's non-functional requirements, refer to the Non-functional Requirements section.

For a description of the system's design and architecture, refer to the Design section. For a description of the system's testing approach and methodology, refer to the Testing section.

For a list of known issues and limitations, refer to the Bugs section.

It is recommended that readers start with the overview sections and then proceed to the sections that are most pertinent to their role or interest. Developers should focus on the functional and non-functional requirements, while testers should focus on the testing section. Project managers should review the entire document to ensure that the project is on track and that all requirements are met.

1.4 Product Scope

AI virtual mouse represents a transformative advancement in human computer interaction (HCI). Through the integration of advanced artificial intelligence and computer vision technologies, it offers users an intuitive, precise, and adaptable means of interacting with digital interfaces. Product has scope for Daily use for disabled persons, Accessibility and inclusivity enhancement, Immersive virtual reality and augmented reality experiences, Productivity boost in various work healthcare, Integration into everyday devices for seamless interaction.

1.5 References

[1] S. Shriram, ”Deep Learning-Based Real-Time AI Virtual Mouse System Using Computer Vision to Avoid COVID-19 Spread,” 25 Oct 2021.

[2] S. U. Dudhane, “Cursor control system using hand gesture recognition,” IJARCCE, vol. 2, no. 5, 2013.

[3] L. Somas, “Virtual mouse using hand gesture,” International Research Journal of Engineering and Technology (IRJET, vol. 5, no. 4, 2018.
[4] J. Katona, “A review of human–computer interaction and virtual reality research fields in cognitive InfoCommunications,” Applied Sciences, vol. 11, no. 6, p. 2646, 2021.
[5] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, “Realtime computer vision with openCV,” Queue, vol. 10, no.4, pp. 40–56, 2012.]
[6] J. Jaya and K. Thanushkodi, “Implementation of classification system for medical images,” European Journal of Scientific Research, vol. 53, no. 4, pp. 561–569, 2011.
[7] L. Thomas, “Virtual mouse using hand gesture,” International Research Journal of Engineering and Technology (IRJET, vol. 5, no. 4, 2018.
[8] V. Bazarevsky and G. R. Fan Zhang. On-Device, MediaPipe for Real-Time Hand Tracking.
[9] J. T. Camillo Lugaresi, “MediaPipe: A Framework for Building Perception Pipelines,” 2019, https://arxiv.org/abs/1906.08172.
[10] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, “Realtime computer vision with openCV,” Queue, vol. 10, no. 4, pp. 40–56, 2012.
[11] K. P. Vinay, “Cursor control using hand gestures,” International Journal of Critical Accounting, vol. 0975–8887, 2016.
[12] V. Bazarevsky and G. R. Fan Zhang. On-Device, MediaPipe for Real-Time Hand Tracking.
[13] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, “Realtime computer vision with openCV,” Queue, vol. 10, no. 4, pp. 40–56, 2012.
[14] A. Haria, A. Subramanian, N. Asokkumar, S. Poddar, and J. S. Nayak, “Hand gesture recognition for human computer interaction,” Procedia Computer Science, vol. 115, pp. 367– 374, 2017.
[15] K. H. Shibly, S. Kumar Dey, M. A. Islam, and S. Iftekhar Showrav, “Design and development of hand gesture based virtual mouse,” in Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), pp. 1– 5, Dhaka, Bangladesh, May 2019.

2. Overall Description

2.1 Product Perspective

The main perspective of this project is to developing an AI virtual mouse system encompasses several key features aimed at revolutionizing user interaction with computing devices. Firstly, precision is paramount; the system must offer unparalleled accuracy in pointing and clicking, surpassing conventional input methods. Moreover, accessibility is crucial, ensuring inclusivity for users with physical disabilities through customizable input options and compatibility with alternative devices. Incorporating gesture recognition elevates user experience, enabling intuitive control via eye and lip movements for common mouse functions. Adaptive sensitivity further enhances versatility, dynamically adjusting to user preferences and task requirements, whether it be graphic design, gaming, or text editing. Finally, prioritizing natural interaction minimizes user fatigue and discomfort during prolonged use by mimicking organic eye and lip movements, fostering a seamless and ergonomic computing experience.

2.2 Product Functions

The system opens the camera of desktop and detects the eye and lip gesture of user using OpenCV modules. Then, the cursor which is connected to system is traced and moves accordingly as the user moves. If the user moves his/her lips and says “click”, the left click function of mouse is performed. Finally, if the user moves out of bound, the system stops.

2.3 User classes and Characteristics

General Users: Anyone looking to use a hands-free method to control their computer.
Users with Disabilities: Individuals who have difficulty using traditional input devices like a mouse or keyboard.

2.4 Operating Environment

Operating Environment:
Hardware: Standard computer with a webcam.
Software: Runs on Windows, macOS, or Linux with Python installed.
Dependencies: Requires OpenCV, Mediapipe, and PyAutoGUI Python libraries.Operating System: The Cryptocurrency Analysis WebApp will be compatible with the major operating system, including Windows. The minimum required operating system version is Windows 7.

2.5 Design and Implementation Constraints

Must have a functional webcam.

Python and required libraries must be installed.

The system should operate with minimal latency for real-time interaction.

2.6 User Documentation

User Guide: Instructions on how to install and use CLICK.
Troubleshooting Guide: Common issues and solutions.
Developer Documentation: Detailed code explanations and setup instructions for developers.

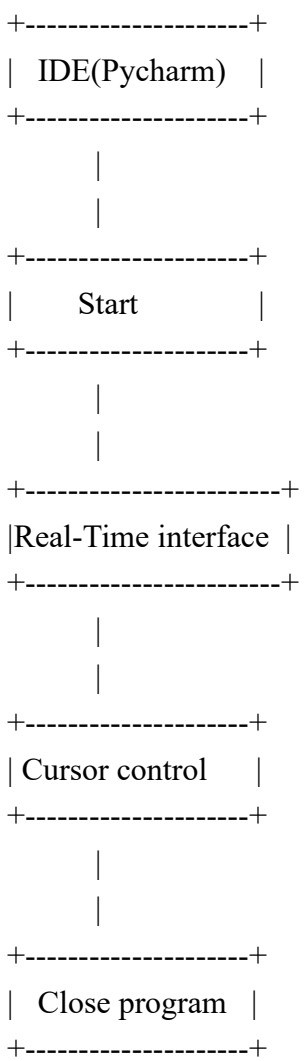
2.7 Assumptions and Dependencies

Assumptions:
Assumes users have basic computer skills.
Depends on the availability and proper functioning of a webcam and the installation of necessary Python libraries.

3. External Interface Requirements

3.1 User Interfaces

Main Interface: Displays the webcam feed with visual markers for detected facial landmarks.
Top Performing Currencies Page: This page displays the top-performing cryptocurrencies based on market capitalization, price, and other performance metrics. Here's a diagram illustrating the different components of the User Interface:



3.2 Hardware Interfaces

Webcam: Used to capture the user’s face and detect gestures.

3.3 Software Interfaces

OpenCV: For capturing and processing video frames.

Mediapipe: For facial landmark detection.
PyAutoGUI: For simulating mouse movements and clicks.

3.4 Communications Interfaces

Not applicable as the system runs locally on the user's computer.

4. System Features

4.1 System Feature

Cursor Movement

Description: The cursor moves based on the detected position of the eyes.
Functional Requirements:
Detect eye position using Mediapipe.
Map eye movement to cursor movement on the screen.

4.2 System Feature

Click Detection

Description: Executes mouse clicks based on lip gestures.
Functional Requirements:
Detect lip distance using Mediapipe.
Execute a click when the lips are detected to be apart beyond a certain threshold.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system should process video frames and update the cursor position with minimal latency (ideally within 50 milliseconds).

5.2 Safety Requirements

Ensure the system does not store or transmit any personal data.
The system should not cause any harm or discomfort to the user.

5.3 Security Requirements

The application should run in a secure environment without requiring elevated permissions.

5.4 Software Quality Attributes

Reliability: The system should function correctly under normal conditions.
Usability: The system should be easy to use with a clear and intuitive interface.
Efficiency: The system should not consume excessive computational resources.

5.5 Business Rules

The software is intended for non-commercial use and should be freely available to users.
This template provides a structured and detailed format for an SRS document, ensuring all critical aspects of the project are covered. You can further customize it based on specific project needs or additional requirements.

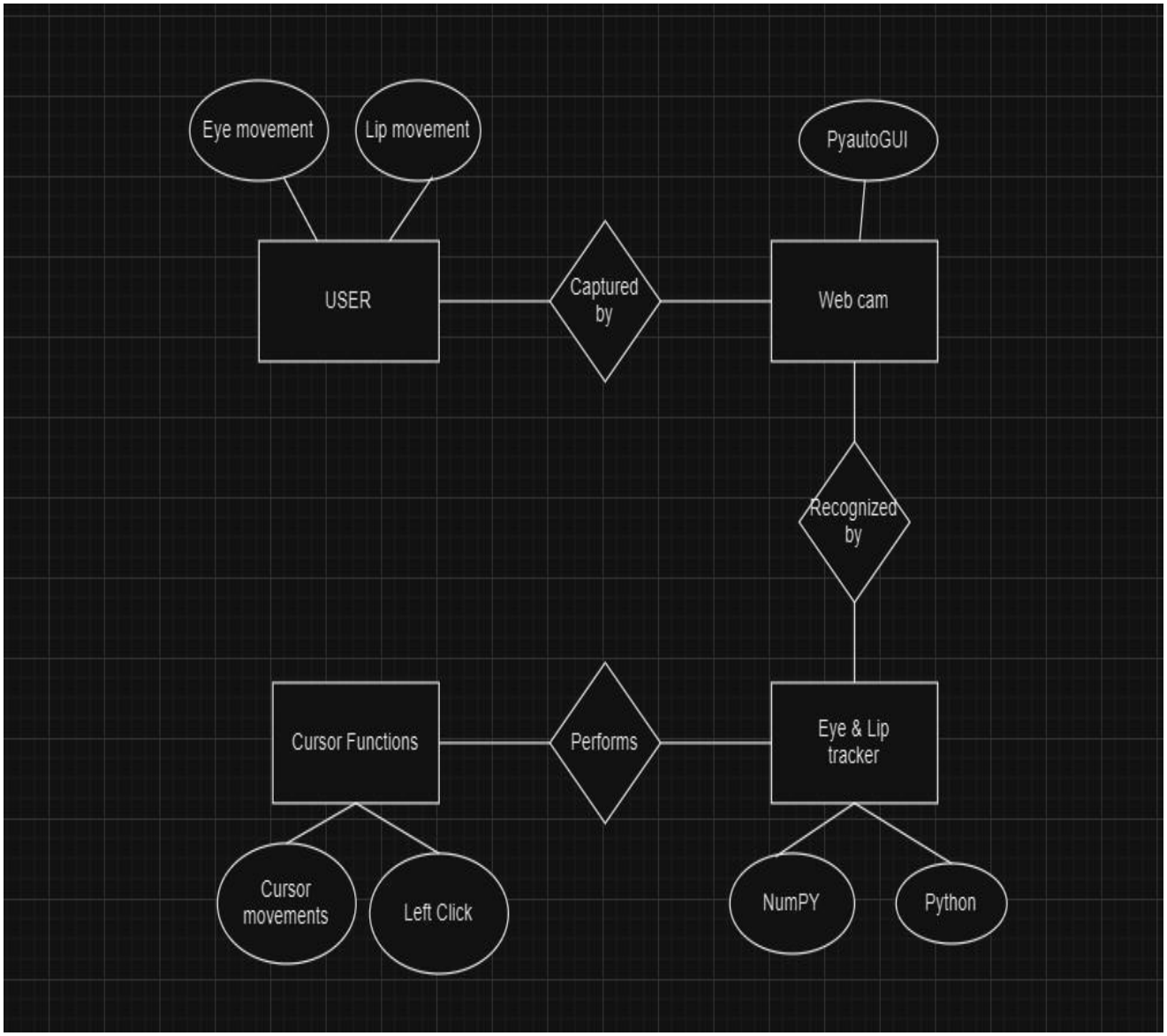
Appendix A: Glossary

stakeholders: Individuals or groups who have a vested interest in the virtual devices, developers, project managers, testers, and documentation writers.

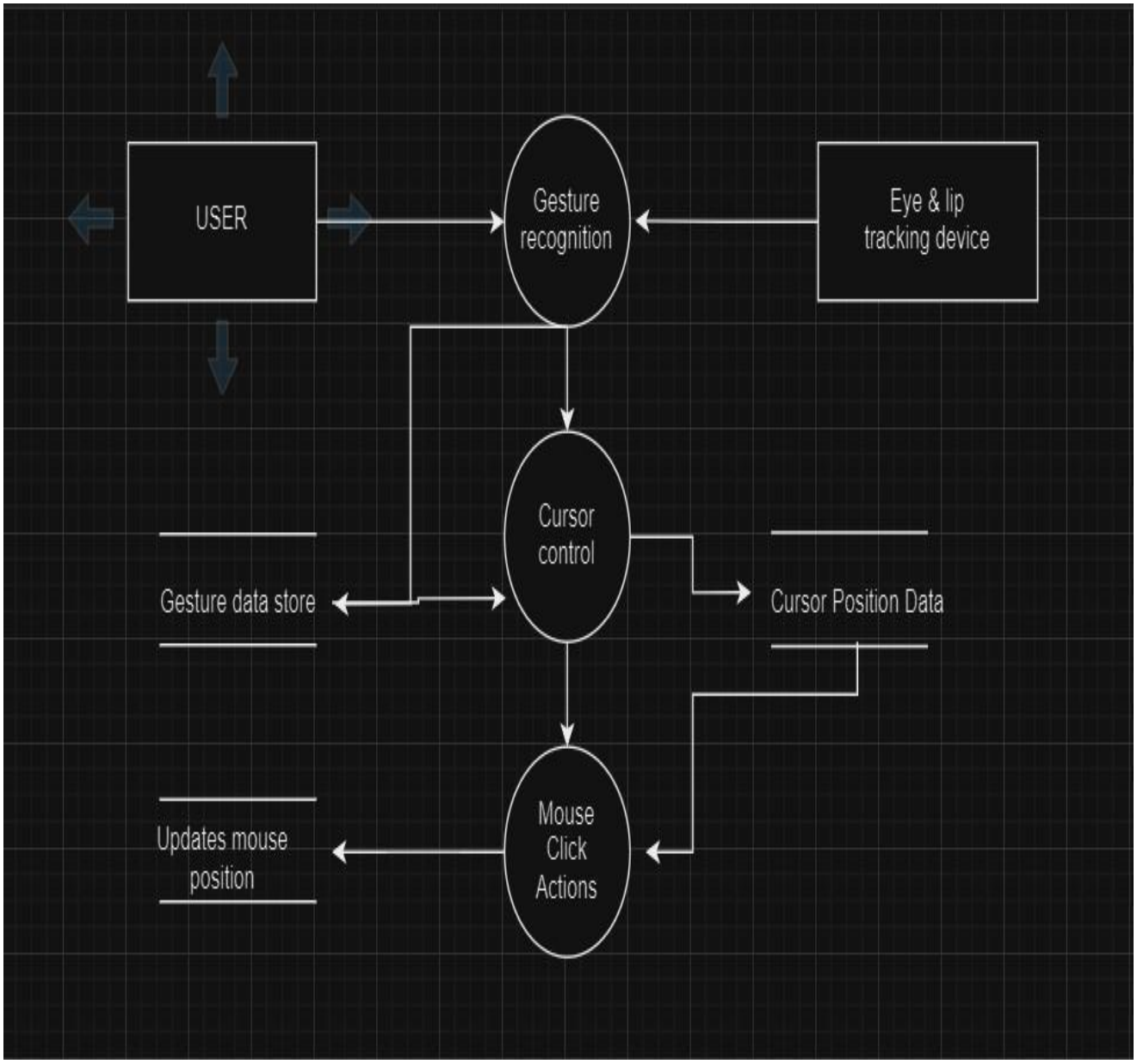
Response time: The time it takes for the web application to respond to user actions, typically 30 milliseconds

Appendix B: Analysis Models

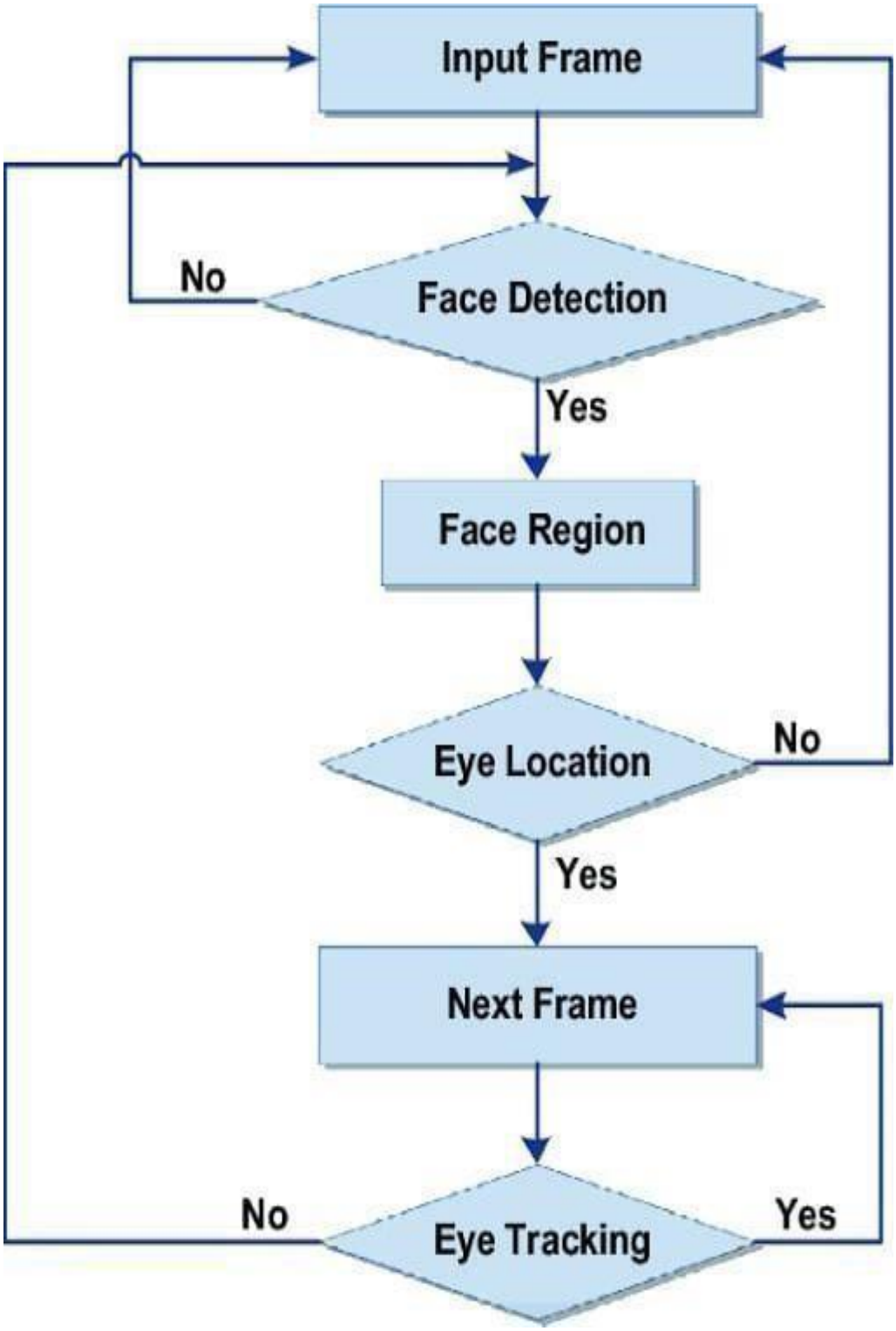
ER Diagram



DATA FLOW DIAGRAMS (DFD)



Flowchart


Input Frame
No
Face Detection
Yes
Face Region
Eye Location
No
Yes
Next Frame
No
Eye Tracking
Yes