AIR CANVAS

A Project Report

submitted in partial fulfilment of the requirements

of

Industrial Artificial Intelligence

With

Cloud Computing

by

Abhishek Nalla,

Prince Beladiya,

Shubham Vanani,

Sujal Vadgama,

Under the Esteemed Guidance of

Jay Rathod

ACKNOWLEDGEMENT

We would like to take this opportunity to express our deep sense of gratitude to all individuals who helped us directly or indirectly during this project work. Firstly, we would like to thank our supervisor, Jay Rathod, for being a great mentor and the best adviser we could ever have. His advice, encouragement, and critiques have been a source of innovative ideas, inspiration, and are the driving forces behind the successful completion of this project. The confidence he showed in us was the biggest source of inspiration. He has always helped us during this project. Lastly, we wish to express our gratitude to all the faculty members of our department for their invaluable advice and for providing a stimulating environment for learning and research. Their input and support have been instrumental in the successful completion of this project. The resources and knowledge they provided have been essential, and we are deeply appreciative of the learning opportunities they have afforded us. Thank you all for your support and belief in us. This project would not have been possible without the collective efforts and encouragement from all of you.

ABSTRACT

This project focuses on developing a virtual painting application that leverages computer vision techniques to enable users to draw on a virtual canvas using hand gestures. The application utilizes OpenCV for real-time image processing, Mediapipe for accurate hand tracking, and a webcam to capture the user's hand movements. By detecting the positions of the user's fingers, the system allows for intuitive control over the drawing process, including selecting different colors and clearing the canvas. The project aims to provide an engaging and interactive drawing experience, showcasing the potential of computer vision and machine learning in creating innovative user interfaces. The system's design and implementation demonstrate the integration of multiple technologies to achieve seamless and responsive interaction, making it suitable for educational, entertainment, and creative applications.

Index

CHAPTER 1

INTRODUCTION

1.1. Problem Statement:

The current methods of digital drawing often rely on physical tools like a mouse, stylus, or touchscreen, which can be limiting and less intuitive compared to natural hand movements. This project addresses the need for a more engaging and intuitive drawing interface by developing a virtual painting application that uses real-time hand tracking and gesture recognition. The core objectiv is to enable users to draw on a digital canvas using simple hand gestures, select colors, and clear the canvas without the need for physical contact with any devices.

1.2. Problem Definition:

This project aims to develop an intuitive virtual painting application that allows users to draw on a digital canvas using hand gestures, without needing physical drawing tools. Utilizing OpenCV for real-time video capture, Mediapipe for hand tracking, and Python for integration, the application enables users to select colors, draw, and clear the canvas with simple gestures. The key features include real-time hand tracking and gesture-based controls, providing a seamless and engaging drawing experience that showcases the potential of computer vision and machine learning in creating innovative, touch-free user interfaces.

1.3. Expected Outcomes:

- **Accurate Hand Tracking:** The use of Mediapipe for hand tracking will result in precise detection and tracking of hand landmarks, enabling accurate interpretation of gestures.

- **Gesture-Based Controls:** Users will be able to perform various actions such as selecting colors, drawing, and clearing the canvas through simple and recognizable hand gestures, enhancing the overall usability of the application.

- **Real-Time Processing:** The application will operate in real-time with minimal latency, ensuring immediate feedback and responsiveness to user actions.

- **Enhanced Creativity:** By removing the need for physical drawing tools, the application will encourage creativity and experimentation, making digital drawing more accessible and enjoyable.

CHAPTER 2

PROPOSED METHODOLOGY

2.1 System Design

2.1.1 Functional Components:

- **Hand Tracking Module:** Details on how the system captures real-time video from the webcam, processes frames, and detects hand landmarks using Mediapipe.
- **Gesture Recognition Module:** Explanation of how the application interprets hand movements to identify gestures like selecting a color, drawing, or clearing the canvas.
- **Drawing Module:** Describes how the system translates recognized gestures into drawing actions on the canvas, including storing and rendering points.

2.1.2 Data Flow:

- **Input Data Flow:** How video frames are captured and processed.
- **Processing Data Flow:** How hand landmarks are detected and gestures are recognized.
- **Output Data Flow:** How drawing actions are rendered on the canvas.

2.1.3 Recognition:

- **Hand Tracking Algorithm:** Step-by-step explanation of the algorithm used for detecting hand landmarks using Mediapipe.
- **Gesture Recognition Algorithm:** Description of the logic for recognizing different gestures based on hand landmark positions.
- **Drawing Algorithm:** Details on how points are stored and drawn on the canvas, including handling different colors and clear actions.

2.2 Modules Used

- **OpenCV:** OpenCV (Open Source Computer Vision Library) is utilized for real-time image processing tasks, such as capturing video frames from the webcam and performing operations like edge detection and color manipulation.

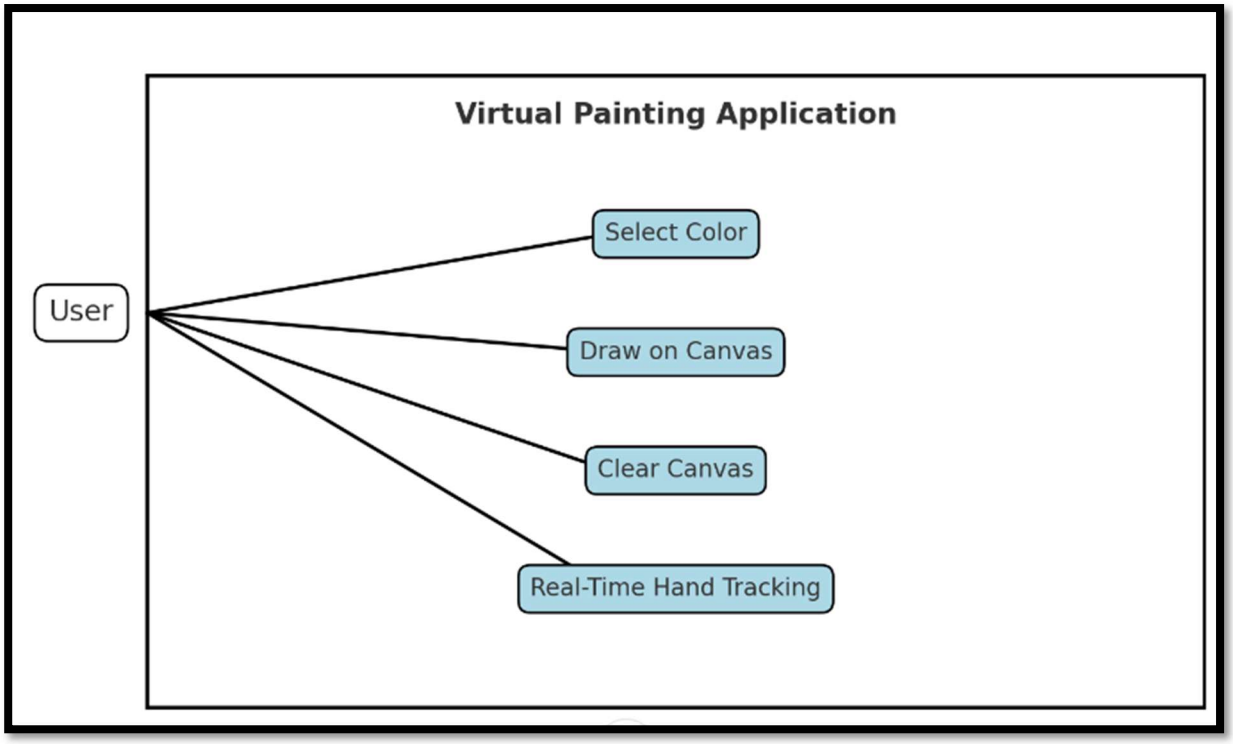
- **Mediapipe:** Mediapipe is employed for accurate hand tracking, enabling the application to detect and locate key landmarks on the user's hand in real-time. These landmarks serve as reference points for interpreting hand gestures.

- **Python:** Python programming language serves as the primary platform for implementing the project. Python's versatility and extensive libraries make it well-suited for integrating various components of the application, from image processing to gesture recognition.

- **NumPy:** NumPy, a fundamental package for scientific computing with Python, is used for numerical operations and data manipulation. It facilitates handling and processing of image data and hand landmark coordinates efficiently**.**

- **Deque (from collections):** Deque (double-ended queue) is a Python module from the collections library. It provides an optimized data structure that allows fast and memory-efficient appends and pops from either end of the deque. You've used deques to store drawing points for different colors in your application.

2.3 Use Case Diagram



2.4 Advantages

- Intuitive interaction with natural hand gestures.
- Real-time feedback enhances user experience.
- Touch-free interface promotes hygiene and safety.
- Gesture-based controls simplify usability.
- Cross-platform compatibility ensures broad accessibility.
- Customization and extensibility support tailored features.
- Educational tool for computer vision and machine learning concepts.
- Therapeutic value for motor skills development and creative expression.

2.5 Requirement Specification

2.5.1 Hardware Requirements:

- **Webcam:** A webcam capable of capturing video input is required for real-time hand tracking. Higher-resolution webcams can enhance the accuracy of hand detection and tracking.

- **Computer:** A computer with sufficient processing power and memory to handle real-time image processing tasks is necessary. While the application can run on most modern computers, higher-end systems may provide smoother performance, especially when dealing with multiple hand tracking and drawing operations simultaneously.

- **Display:** A display screen, such as a monitor or laptop screen, is needed to visualize the virtual canvas and drawing interactions. A larger screen size may offer a more comfortable drawing experience, particularly for detailed artwork.

2.5.2 Software Requirements:

- **Python Interpreter:** The application requires a Python interpreter to execute the code. It is recommended to use Python 3.x, preferably the latest stable version available at the time of development, to ensure compatibility with the required libraries and dependencies.

- **OpenCV (cv2):** OpenCV is a core dependency for the application, providing essential functions for real-time image processing, webcam access, and drawing on the canvas.

- **NumPy (np):** NumPy is another fundamental dependency for scientific computing in Python, utilized for efficient handling and manipulation of numerical data, particularly for image processing tasks.

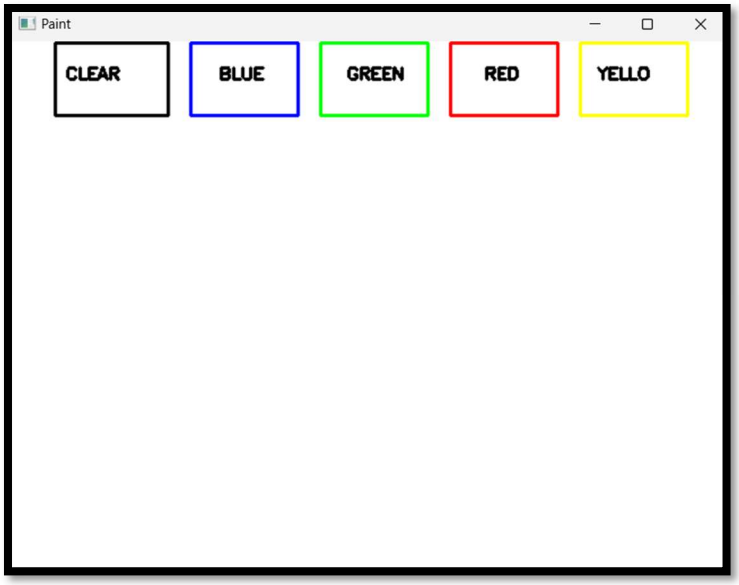
- **Mediapipe:** Mediapipe is required for hand tracking functionality, providing pre-trained models and algorithms for detecting and tracking hand landmarks in real-time.

CHAPTER 3

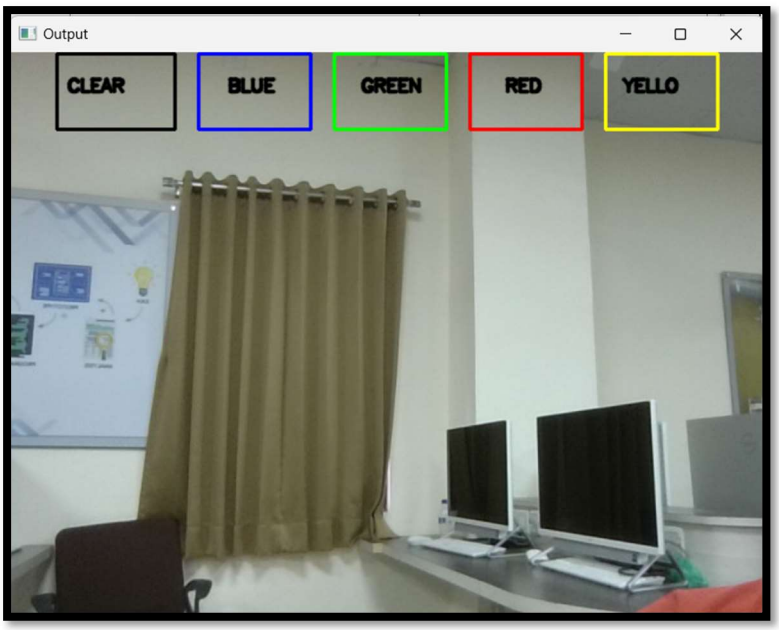
IMPLEMENTATION AND RESULT

3.1 UI of Canvas

❖ UI of a Painting Window

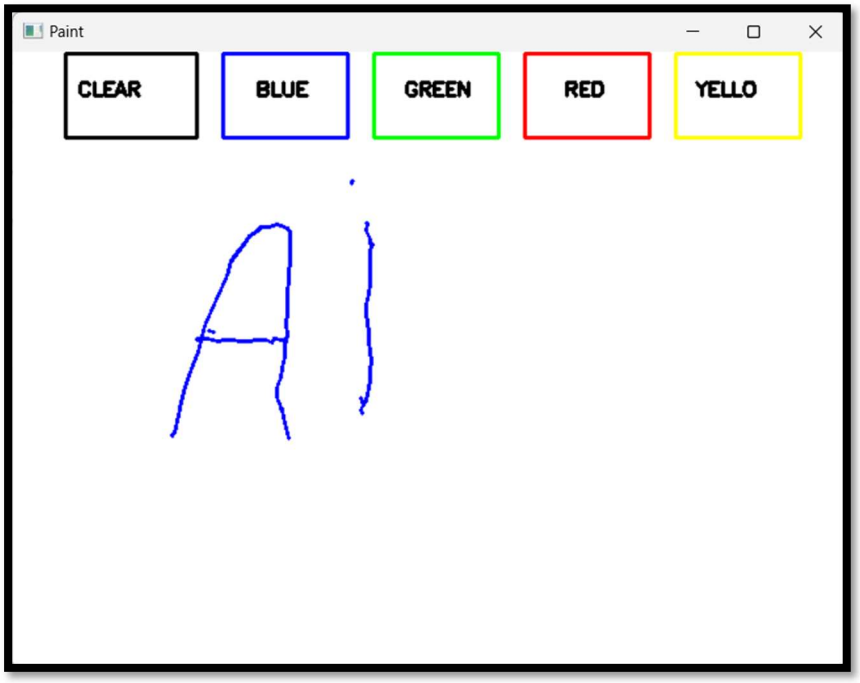


❖ UI of a Webcam Window

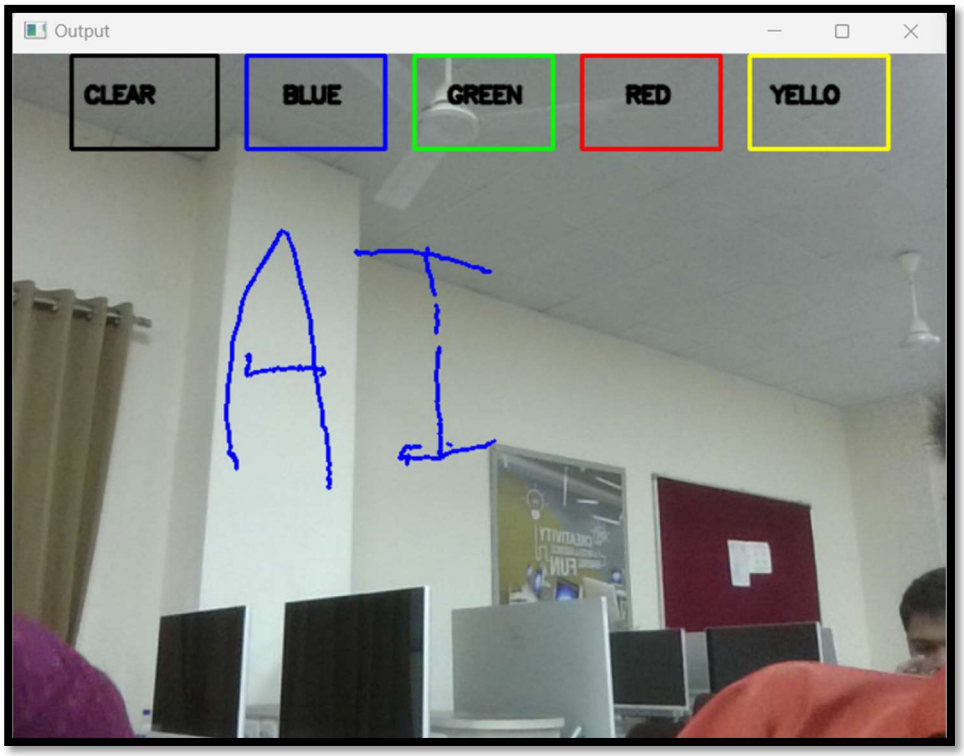


3.2 Result of Canvas

❖ Result of a Painting Window



❖ Result of a Webcam Window



CHAPTER 4

CONCLUSION

In conclusion, Air Canvas is a breakthrough tool for digital art creation. By using hand gestures, it lets you draw naturally and easily. We've focused on making it simple to use, with features like real-time drawing and smooth performance.

Air Canvas opens up new possibilities for artists of all levels. We're dedicated to improving it based on your feedback, so you can keep creating without limits.