**AIR CANVAS**

Industry Oriented Mini Project Submitted To

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**

In partial fulfilment of the academic requirement for the award of the degree

Of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE ENGINEERING**



**UNDER THE GUIDANCE**

**OF**

**DR. G SRIDHAR**

Submitted by

**MOHD AQUIB – 20081A0534**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SHADAN COLLEGE OF ENGINEERING & TECHNOLOGY**

**PEERANCHERU, HYDERABAD-500086**

**(Affiliated to JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD)**

**HYDERABAD-500087**

**(2023-2024)**

**ACKNOWLEDGMENT**

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this report. Special thanks to our guide Dr. SHRIDHAR GUMMALLA sir whose help, stimulating suggestions and encouragement helped us in all time of fabrication process and in writing this report. I also sincerely thanks for the time spent on correcting my many of our mistakes.

Many thanks go to the all lecturer and supervisors who have given their full effort in guiding the team in achieving the goal as well as their encouragement to maintain our progress in track. My profound thanks go to all classmates, especially to my friends for spending their time in helping and giving support whenever I need it in fabricating my project.

**Submitted By:**

Shiv Nandini (20081A0554)

Mohd Aquib (20081A0534)

Moinuddin AS (200881A525)

**DECLARATION**

I hereby declare that the project work titled “AIR CANVAS” submitted to the JNTU Hyderabad, is a record of an original work done by me under the guidance of DR. SRIDHAR GUMMALLA, Head of the Department Of Computer Science & Engineering, Shadan College of Engineering &Technology, and this project work is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science& Engineering. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**SUBMITTED BY:**

Shiv Nandini (20081A0554)

Mohd Aquib (20081A0534)

Moinuddin AS (200881A525)

AIR CANVAS

**INDEX**

**DESCRIPTION PAGE NO.**

ABSTRACT viii

List of abbreviation                                                             ix

List of figures                          x

List of symbols x

List of keywords xi

**CHAPTER 1: INTRODUCTION 12-14**

1.1 Objective 13

1.2 Scope 13

1.3 Goals 14

**CHAPTER 2: LITERATURE SURVEY 15-16**

**CHAPTER 3: PROJECT DESCRIPTION 17-18**

3.1 Description 17

3.2 Methodology 17

3.3 Tools and Technology used 18

**CHAPTER 4: REQUIREMENTS 19-20**

4.1 Hardware Requirements 19

4.2 Software requirements 19

4.3 Functional requirements 20

4.4 Non-Functional requirements 20

**CHAPTER 5: SYSTEM DESIGN 21-25**

5.1 Features  21

5.2 Algorithm  21

5.3 Flow chart  22

5.4 Hand Landmark model   23

5.5 User Interface  24

5.6 Workflow  25

**CHAPTER 6: SOFTWARE SPECIFICATION 26-30**

6.1 History of python 26

6.2 Importance of python 26

6.3 Libraries used in python 27

**CHAPTER 7: IMPLEMENTATION 31-38**

7.1 Source Code 31

7.2 Outputs 38

**CHAPTER 8: SNAPSHOTS 39-44**

8.1 Snap 39

8.2 Demonstration 42

**CHAPTER 9: APPLICATIONS AND FUCTURE ENHANCEMENT 45-49**

9.1 Applications 45

9.2 Limitations 47

9.3 Functional Challenges 49

**CHAPTER 10: CONCLUSION AND REFERENCES 50-51**

**ABSTRACT**

This project presents a real-time video based digital drawing method which allows sketching or writing in air in front of camera. It mainly contributes to advancement of automated process and can improve the interface between man and machine. Object tracking is considered as important task in the field of computer vision. It involves initially, detecting the object, then tracking its movement from frame to frame and finally analyzing the behavior of the object. We will use Computer vision to track the path of finger. It will be a powerful means of communication and an effective method to reduce the use of pen and paper work.

**LIST OF ABBREVATION**

|  |  |  |
| --- | --- | --- |
| S NO. | ABBREVIATION | EXPLANATION |
| 1 | IDE | Integrated Development Environment |
| 2 | CV | Computer vision |
| 3 | NP | Numerical Python |
| 4 | MP | MediaPipe |
| 5 | HSV | Hue Saturation Value |
| 6 | BGR | Blue Green Red |
| 7 | RGB | Red Green Blue |
| 8 | VS | Visual Studio |

**LIST OF FIGURES**

* Flow chart
* User Interface workflow

**LIST OF SYMBOLS**

|  |  |  |
| --- | --- | --- |
| **NAME** | **REPRESENTATION** | **DESCRIPTION** |
| Terminator |  | Indicates the beginning or end of a program now in your diagram |
| Decision |  | Indicates the beginning or end of a program now in your diagram |
| Process |  | indicates any  processing function |
| Arrow |  | A line is a connector that  shows relationships between  the representative shapes |

**LIST OF KEYWORDS**

* Object detection
* Hand tracking
* Gesture recognition
* Paint window
* Virtual drawing
* Color selection
* OpenCV
* Motion tracking
* Computer vision

**INTRODUCTION**

In the emerging digital world. Traditional art of writing and drawing is replaced by digital art forms. Earlier, painting was done digitally either with a mouse or touch pad which was quite hectic task. Even though, we have touch screens in laptops, they are expensive. Hand tracking more specifically finger tracking techniques are used as a tool on the computer, acting as a External device similar to a keyboard and mouse. It is used in various fields like virtual reality to Sign language recognition.

Air canvas is a hands-free digital drawing canvas which utilizes camera, OpenCV and MediaPipe To recognize and map the hand gestures. The user’s finger is considered as the brush to draw or write. Also, the color can be modified by hovering on a pointer over built-in buttons. This system uses computer vision techniques and python programming language to built the code. Camera and mediapipe is used to track the finger positions.

**OBJECTIVE**

To create a simple prototype for a drawing tool that uses hand gesture recognition software to paint on screen.

Core objectives includes:

* Using OpenCV to recognize the pointer finger.
* Mapping coordinates extracted from hand recognition software
* Implementing additional features such as color change, buttons

**SCOPE**

The scope of this system is interactive user experience and it can be used as a powerful means of communication for deaf. which means implementing this system can help in effective communication method that reduces mobile and laptop usage by eliminating the need to write Physically. It can be helpful to people with hearing impairments to communicate well.

**GOALS**

* Enhanced user experience
* Creativity
* Increase productivity
* Interactivity
* Accessibility
* Innovation
* Commercial use
* Educational use

**LITERATURE SURVEY**

Humans communicate by writing, often taking notes that assist thinking. With the growing popularity of virtual reality applications, it is imperative that we better understand aspects that Effect virtual experiences. On air writing in VR is a popular writing paradigm due to its simplicity in implementation without any explicit needs for specialized hardware. A host of factors can Affect the system, we delve into investigating the same. Along these lines we investigated the effects of combination of factors on user experience and system performance, aiming to understand the circumstances under which users can both effectively and efficiently write on screen. we were interested in studying the following.

**OBJECT DETECTION**

Object detection is the computer vision task that involves identifying and locating the objects within an image or a video stream.

The system’s software would process the input from the web camera to detect the presence of object in the view.

**MOTION TRACKING**

Motion tracking also known as object tracking is the process of locating and following a moving object.

The system would track the movement of thee detected object in the air. This information is used to determine the path of the drawing as the user moves the object.

**GESTURE RECOGNITION**

Gesture recognition is a technology that enables computers and machines to interpret and understand human gestures. It involves various types of gestures including hand gestures, finger movements, facial expressions, body gestures and even sign language.

The system could incorporate gesture recognition to enable different drawing actions such as

Selecting a color or clearing the canvas, through predefined hand movements.

**USER INTERACTION**

As the user moves the object(hand) in the air, the system would interpret the movement as a drawing actions. It would replicate the path of hands index finger movement through camera on a digital canvas (static paint window) in the corresponding color.

**INTERACTIVITY**

The system might include features like drawing, writing, switching between colors based on the motion being tracked.

**USER EXPERIENCE**

The goal is to create an intuitive and engaging user experience where users can create drawings by simply moving hand in the air. It’s an interactive way to explore creativity.

**PROJECT DESCRIPTION**

**DESCRIPTION**

1. Import NumPy, OpenCV, mediapipe and install necessary python packages including

Deque from collections.

2. Capture frames from the web camera

3. Invoke a canvas window for drawing.

4.Detect the finger and track the colors.

5.Draw on canvas window.

**METHODOLOGY**

The methodology and stages of the system is discussed below:

1. Execute the code once all the libraries are installed. This leads to turning the camera on automatically and the OpenCV frame displays various ink buttons and eraser.
2. Webcam starts recording and convert the individual frame sends the frame to hand tracker class to track and detect the position of fingers.
3. Each frame received is compared with mediapipe hand landmarks and position of fingers is identified.
4. Performing action according to the button
5. Different buttons are selected with the help of index finger hovering over specific button. In this way different colors are chosen and also among them is the option to clear the screen.
6. Press the ‘q’ key on keyboard to exit or stop the running application.

**DEQUE**

A double ended queue is a data structure that allows elements to be added and removed from both ends efficiently. In the context of air canvas, a deque could be used to manage a history of drawing actions or strokes. This would enable user to undo and redo their drawing actions.

Here’s how a deque is implemented in the logic of air canvas

* Whenever the user makes a stroke or drawing action on canvas.it captures the necessary information about that action such as points drawn and color used.
* After capturing the users drawing actions adds it to the deque from the front end. This means that the latest drawing action will be at the beginning of deque.
* When the user wants to undo their last drawing action, remove the first element that is latest drawing action from the deque (pop operation).
* This will effectively revert the canvas to its previous state.

**TOOLS AND TECHNOLOGIES USED**

* Visual studio code IDE
* Python
* NumPy
* OpenCV
* Mediapipe
* Collections deque

**REQUIREMENTS**

Requirements of the application system are described here.

**HARDWARE REQUIREMENTS**

The minimum hardware requirements to execute the system are

* Processor – minimum intel core processor
* RAM – 4to 8 GB
* Storage
* HD web camera
* Operating system with web browser
* Monitor

**SOFTWARE REQUIREMENTS**

* Visual studio code
* Python
* NumPy
* OpenCV
* Mediapipe
* Collections-deque data structure

**FUNCTIONAL REQUIREMENTS**

* Using camera to capture input
* Detecting the hand positions
* Detecting fingertip
* Choose different tools
* Saves the work on canvas

**NON-FUNCTIONAL REQUIREMENTS**

* Reliability
* Scalability
* Maintainability
* Accessibility
* Usability

**SYSTEM DESIGN**

**FEATURES**

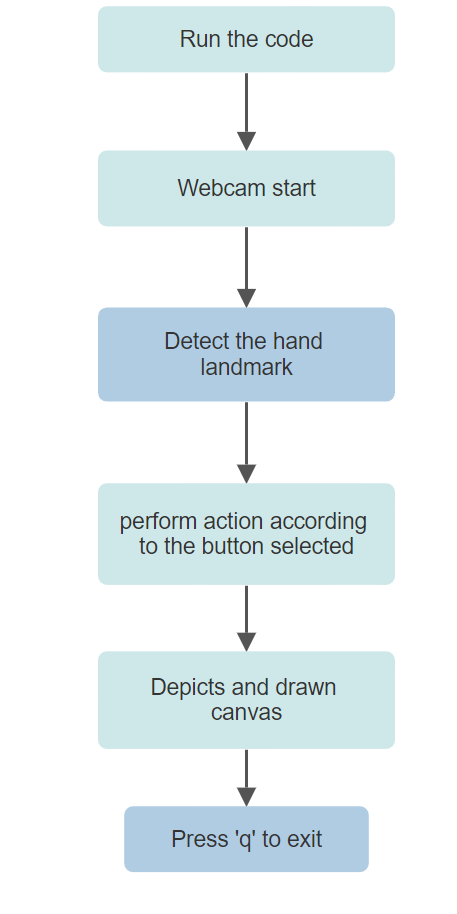
Fundamental features of air canvas

* It can track any specific color pointer
* Users can draw in four different colors and even change them without any Hassel.
* Able to clear the screen with a single location at the top of the screen.
* No need to touch the computer once the program is run
* Finger tracing using OpenCV module in python

**ALGORITHM**

1. Start reading the frames and convert the captured frames to HSV colour space.
2. Prepare the canvas frame and put the respective ink buttons on it.
3. Adjust the values of the mediapipe initialization to detect one hand only.
4. Detects the landmark by passing the RGB frame to the mediapipe hand detector.
5. Detect the hand marks, find the forefinger coordinates and keep storing them in the array of successive frames.
6. Finally draw the points stored in array on the frames and canvas.

**FLOW CHART**



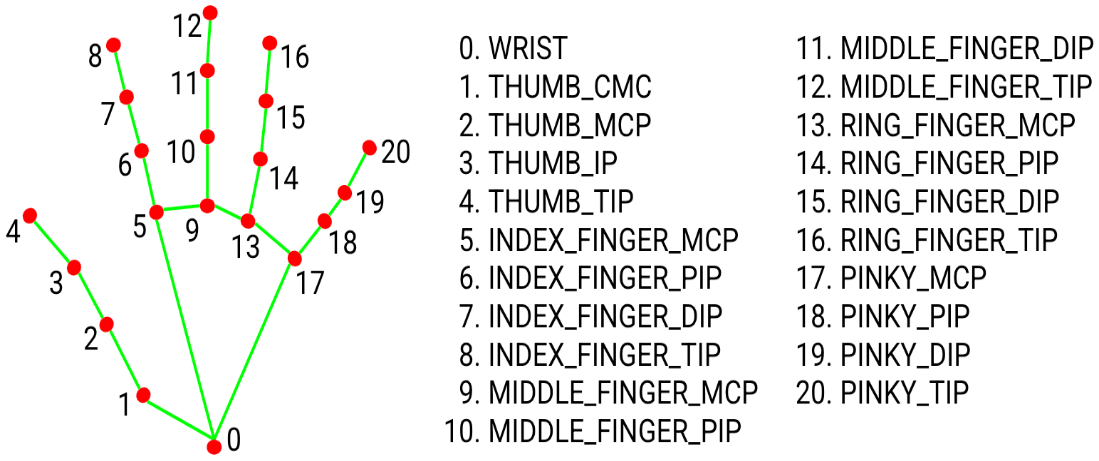
**HANDLANDMARK MODEL**

Implementation of hand tracking in the air canvas involves using computer vision and machine Learning learning techniques to accurately detect and track the user’s hand in real time.Hand tracking is the process in which a computer uses computer vision to detect a hand from an Input image and keep focus on the hand movement and orientation. The model will output. The positions of landmarks on the hands such as fingertips and knuckles. These landmarks can be used to accurately determine the hands position orientation and movements.

HAND LANDMARKS IDENTIFICATION:

Mediapipe finds the 21 hand landmarks on the image of hand.

The 21 hand points that mediapipe identifies ate shown in the image



GESTURE RECOGNITION:

This technique processes the detected hand landmarks to recognize specific gesture such as

Pinching, swiping to interact with the application.

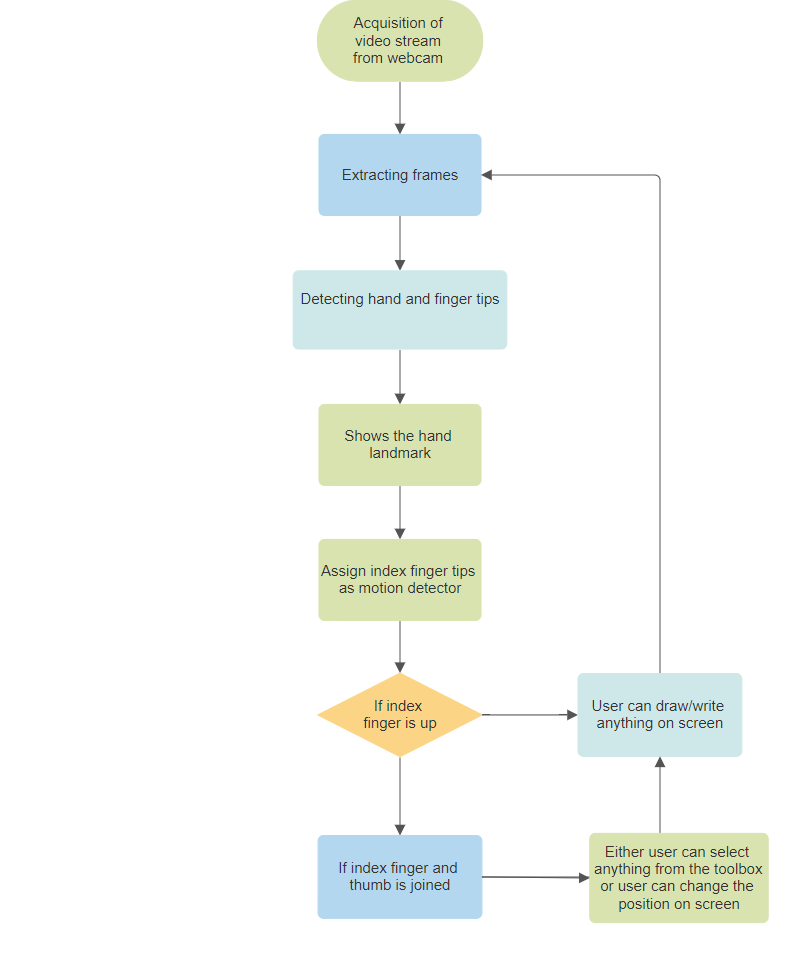
**USER INTERFACE**

After completion of hand tracking module the next step was to create the user interface where the Users will interact with the machine to achieve the desired output. We created a simple screen using web camera with various options for the user to select the required ink buttons. The positions of ink buttons are fixed. After selecting the color an automatic array list is created With the coordinates.

Whenever our hand is moving in front of camera and screen, the movements get stored in the array to give a collection of pixels as generated output.

The output generated is displayed in the static paint window also called as white canvas.

**WORKFLOW:**

****

**SOFTWARE SPECIFICATION**

**PYTHON**

Python is a high-level, general-purpose programming language that is widely used for variousApplications. Python is known for its clean and easy to use syntax. It uses indentation to defineCode blocks making it highly readable. It is an interpreted language users can write and execute code line by line making it excellent for quick prototyping and development. It is dynamic in nature.it is available on various operating systems making it highly portable. Python is open source. It is freely available and can be used, modified and distributed by anyone.

Python comes with a comprehensive standard library that provides modules and packages for wide range of applications reducing the need to write code from scratch.

**HISTORY AND IMPORTANCE OF PYTHON**

Python is widely used programming language with a history that dates back to the late1980s and early 1990s. Python’s design philosophy emphasizes code readability; it's clear and concise syntax has made it a popular choice among programmers.

Python is grown in popularity and importance for a variety of reasons

* Ease of learning
* Versatility
* Productivity
* Cross platform
* Availability
* Environment
* Opensource
* Frameworks
* Community support
* Collaborative

**LIBRARIES USED IN PYTHON**

**NUMPY**

NumPy short for” Numerical python” is a popular open-source library in python forPerforming numerical and mathematical operations. It provides support for large multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.

NumPy's primary data structure is the ndarray that is n-dimensional array. The N-Array type known as ndarray is the most significant object defined in NumPy library. The collection of identically categorized things is described. A zero-based index can be Used to access items in the collection.

A n-dimensional array’s items all take up the same amount of space as a memory block. Every item in ndarray is a data type object called dtype. A python object of one of the arrays Scalar types represent each item that is retrieved from a nd-array object via slicing. NumPy is a powerful tool for wide range of numerical and scientific computing tasks in python.

**OPENCV**

OpenCV stands for “open-source computer vision library". It is an open-source Computer vision and image processing software library. It was developed to provide A wide range of tools for various computer vision and image analysis tasks. It is A comprehensive set of computer vision algorithms such as object detection, Motion tracking, optical flow and camera calibration. It includes image processing Functions, basic operations like filtering, thresholding, morphological operations and more advanced techniques like edge detection, image segmentation and feature Extraction.

Real time computer vision applications can be created by utilizing the OpenCV library For the python programming language. OpenCV provides functions to read and write Images and videos in various formats. It can capture video from cameras and process it in Real-time.

**VISUAL STUDIO CODE**

Visual Studio Code (VS Code) is a popular free, open-source code editor developed

By Microsoft. It is designed to be a lightweight and highly customizable code editor

That is suitable for a wide range of programming and development tasks. VS Code

Is known for its excellent extensibility, a wide range of available extensions and

Strong support for various programming languages and technologies.

Built-in git support makes it easy to work with version control .VS Code provides

Robust debugging capabilities for various programming languages and platforms.

Integrated terminal allows to run commands and scripts within the editor. With the

Help of extensions visual studio code can turn into fully-fledged integrated

Development environment (IDE) for a specific language or framework, such as

Python.

**MEDIAPIPE**

Mediapipe is an open-source framework developed by Google that provides a

Wide range of Machine Learning solutions for various multimedia and

Perceptual computing tasks. It simplifies the development of applications

That involves computer vision, machine learning and image processing.

MediaPipe is particularly useful for tasks that involves analyzing and understanding Media inputs like images and videos. MediaPipe can integrate with popular machine learning frameworks. It provides pre-trained machine learning models for various computer vision and multimedia Tasks making it easier to build applications without starting from scratch. MediaPipe’s real-time capabilities, modular design and predefined models make it A valuable tool for working on a variety of multimedia and computer vision applications.

Mediapipe offers a “Hand Tracking Model” as a part of its hand package.This model enables us to track and detect hands in images and video streams,which can be used in gesture recognition.

**CAPTURING VIDEO AND PROCESSING:**

The web camera will begin recording video after the video capture object isCreated using the python computer vision package OpenCV. The virtual System receives frames from web camera and processes them.

Every frame is recorded up until the end of the application, the video frames are Converted from BGR to RGB in order to find hands in the video and display. They detected hand landmarks and connections on the screen.

**IMPLEMENTATION**

**SOURCE CODE**

**#All the imports go here**

**import cv2**

**import numpy as np**

**import mediapipe as mp**

**from collections import deque**

**#Giving different arrays to handle colour points of different colors**

**bpoints=[deque(maxlen=1024)]**

**gpoints=[deque(maxlen=1024)]**

**rpoints=[deque(maxlen=1024)]**

**ypoints=[deque(maxlen=1024)]**

**#These indexes will be used to mark the points in particular arrays of specific color**

**blue\_index=0**

**green\_index=0**

**red\_index=0**

**yellow\_index=0**

**#The kernel to be used for dilation purpose**

**kernel=np.ones((5,5),np.uint8)**

**colors=[(255,0,0),(0,255,0),(0,0,255),(0,255,255)]**

**colorIndex=0**

**#Here is code for white canvas setup**

**paintWindow=np.zeros((471,636,3))+255**

**paintWindow=cv2.rectangle(paintWindow,(40,1),(140,65),(0,0,0),2)**

**paintWindow=cv2.rectangle(paintWindow,(160,1),(255,65),(255,0,0),2)**

**paintWindow=cv2.rectangle(paintWindow,(275,1),(370,65),(0,255,0),2 )**

**paintWindow=cv2.rectangle(paintWindow,(505,1),(600,65),(0,255,255),2)**

**paintWindow=cv2.rectangle(paintWindow,(390,1),(485,65),(0,0,255),2)**

**cv2.putText(paintWindow,"CLEAR",(49,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(paintWindow,"BLUE",(185,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(paintWindow,"GREEN",(298,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(paintWindow,"RED",(420,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(paintWindow,"YELLOW",(520,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**#initialize mediapipe using the below two lines**

**mpHands=mp.solutions.hands**

**hands=mpHands.Hands(max\_num\_hands=1,min\_detection\_confidence=0.7)**

**#This is used for drawing the landmarks on the detected hand**

**mpDraw=mp.solutions.drawing\_utils**

**#Initialize the webcam**

**cap=cv2.VideoCapture(0)**

**ret,frame\_temp=cap.read()**

**while ret:**

**#Read each frame from the webcam**

**ret,frame=cap.read()**

**x,y,c=frame.shape**

**#Flip the frame vertically**

**frame=cv2.flip(frame,1)**

**#hsv=cv2.cvtColor(frame,cv2.COLOR\_BGR2HSV)**

**framergb=cv2.cvtColor(frame,cv2.COLOR\_BGR2RGB)**

**frame=cv2.rectangle(frame,(40,1),(140,65),(0,00),2)**

**frame=cv2.rectangle(frame,(160,1),(255,65),(255,0,0),2)**

**frame=cv2.rectangle(frame,(275,1),(370,65),(0,255,0),2)**

**frame=cv2.rectangle(frame,(390,1),(485,65),(0,0,255),2)**

**frame=cv2.rectangle(frame,(505,1),(600,65),(0,255,255),2)**

**cv2.putText(frame,"CLEAR",(49,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(frame,"BLUE",(185,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(frame,"GREEN",(298,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(frame,"RED",(420,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**cv2.putText(frame,"YELLOW",(520,33),cv2.FONT\_HERSHEY\_SIMPLEX,0.5,(0,0,0),2,cv2.LINE\_AA)**

**#frame=cv2.cvtColor(hsv,cv2.COLOR\_HSV@BGR)**

**#Get hand landmark prediction**

**result=hands.process(framergb)**

**#post process the result**

**if result.multi\_hand\_landmarks:**

**landmarks=[]**

**for handslms in result.multi\_hand\_landmarks:**

**for lm in handslms.landmark:**

**##printed(id,lm)**

**#print(lm,x)**

**#print(lm,y)**

**#Adjust according to your frame size**

**lmx=int(lm.x\*640)**

**lmy=int(lm.y\*480)**

**landmarks.append([lmx,lmy])**

**#Drawing landmarks on frames**

**mpDraw.draw\_landmarks(frame, handslms,mpHands.HAND\_CONNECTIONS)**

**fore\_finger=(landmarks[8][0],landmarks[8][1])**

**center=fore\_finger**

**thumb=(landmarks[4][0],landmarks[4][1])**

**cv2.circle(frame,center,3,(0,255,0),-1)**

**print(center[1]-thumb[1])**

**if(thumb[1]-center[1]<30):**

**bpoints.append(deque(maxlen=512))**

**blue\_index+=1**

**gpoints.append(deque(maxlen=512))**

**green\_index+=1**

**rpoints.append(deque(maxlen=512))**

**red\_index+=1**

**ypoints.append(deque(maxlen=512))**

**yellow\_index+=1**

**elif center[1]<=65:**

**if 40<=center[0]<=140:#clear button**

**bpoints=[deque(maxlen=512)]**

**gpoints=[deque(maxlen=512)]**

**rpoints=[deque(maxlen=512)]**

**ypoints=[deque(maxlen=512)]**

**blue\_index=0**

**green\_index=0**

**red\_index=0**

**yellow\_index=0**

**paintWindow[67:,:,:]=255**

**elif 160<=center[0]<=255:**

**colorIndex=0#blue**

**elif 275<=center[0]<=370:**

**colorIndex=1#green**

**elif 390<=center[0]<=485:**

**colorIndex=2#red**

**elif 505<=center[0]<=600:**

**colorIndex=3#yellow**

**else:**

**if colorIndex==0:**

**bpoints[blue\_index].appendleft(center)**

**elif colorIndex==1:**

**gpoints[green\_index].appendleft(center)**

**elif colorIndex==2:**

**rpoints[red\_index].appendleft(center)**

**elif colorIndex==3:**

**ypoints[yellow\_index].appendleft(center)**

**#Append the next deques when nothing is detected to avoid messing up**

**else:**

**bpoints.append(deque(maxlen=512))**

**blue\_index+=1**

**gpoints.append(deque(maxlen=512))**

**green\_index+=1**

**rpoints.append(deque(maxlen=512))**

**red\_index+=1**

**ypoints.append(deque(maxlen=512))**

**yellow\_index+=1**

**#Draw lines of all the colors on the canvas and frame**

**points=[bpoints,gpoints,rpoints,ypoints]**

**for i in range(len(points)):**

**for j in range(len(points[i])):**

**for k in range(1,len(points[i][j])):**

**if points[i][j][k-1] is None or points[i][j][k] is None:**

**continue**

**cv2.line(frame,points[i][j][k-1],points[i][j][k],colors[i],2)**

**cv2.line(paintWindow,points[i][j][k-1],points[i][j][k],colors[i],2)**

**cv2.imshow("Output",frame)**

**cv2.imshow("Paint",paintWindow)**

**if cv2.waitKey(1)==ord('q'):**

**break**

**#release the webcam and destroy all active windows**

**cap.release()**

**cv2.destroyAllWindows()**

**OUTPUT:**

**Here is the result and output of our project**

We used OpenCV module to run our code.

We executed the code in visual studio code, where we installed all the necessary packages and

Run the code on application.



In the above output image two windows are displayed.

One window is our camera page where our finger will be detected and tracked using coordinates

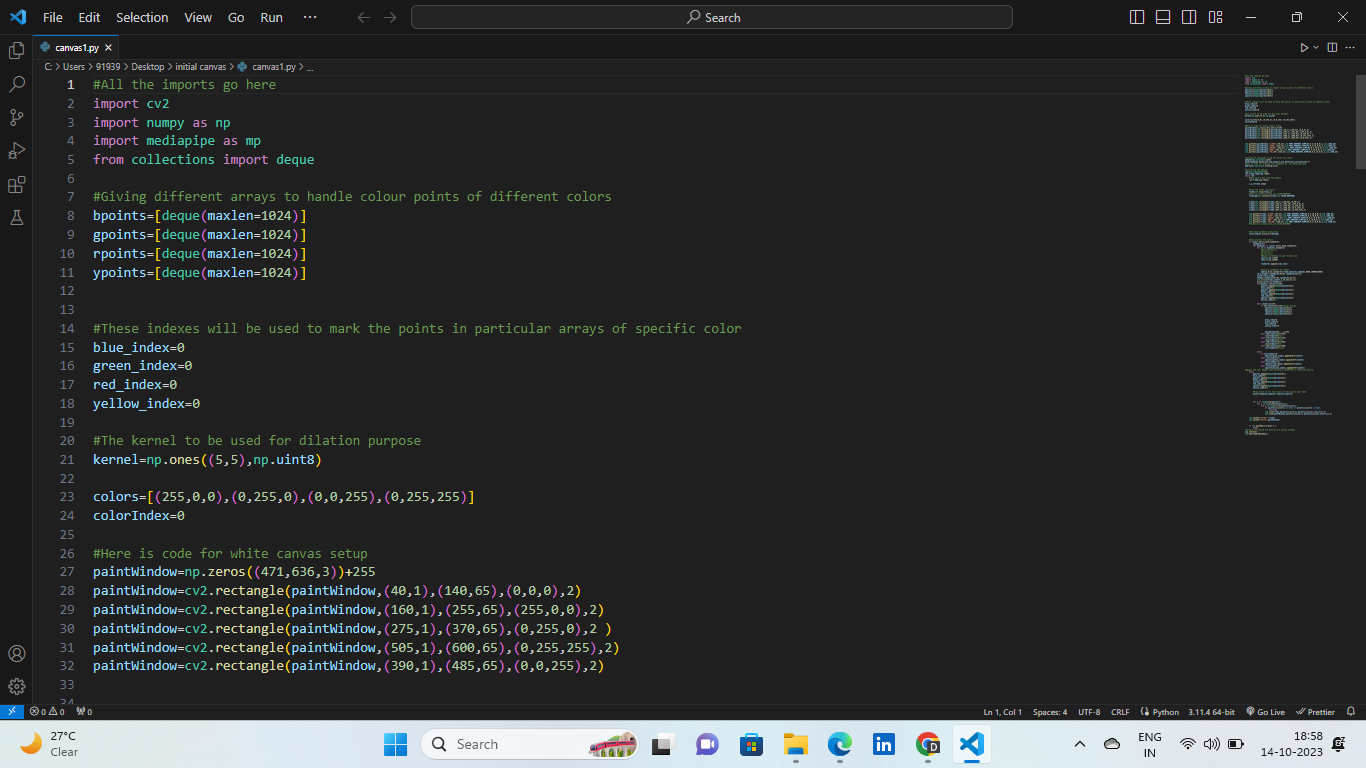
Which are specified in code.

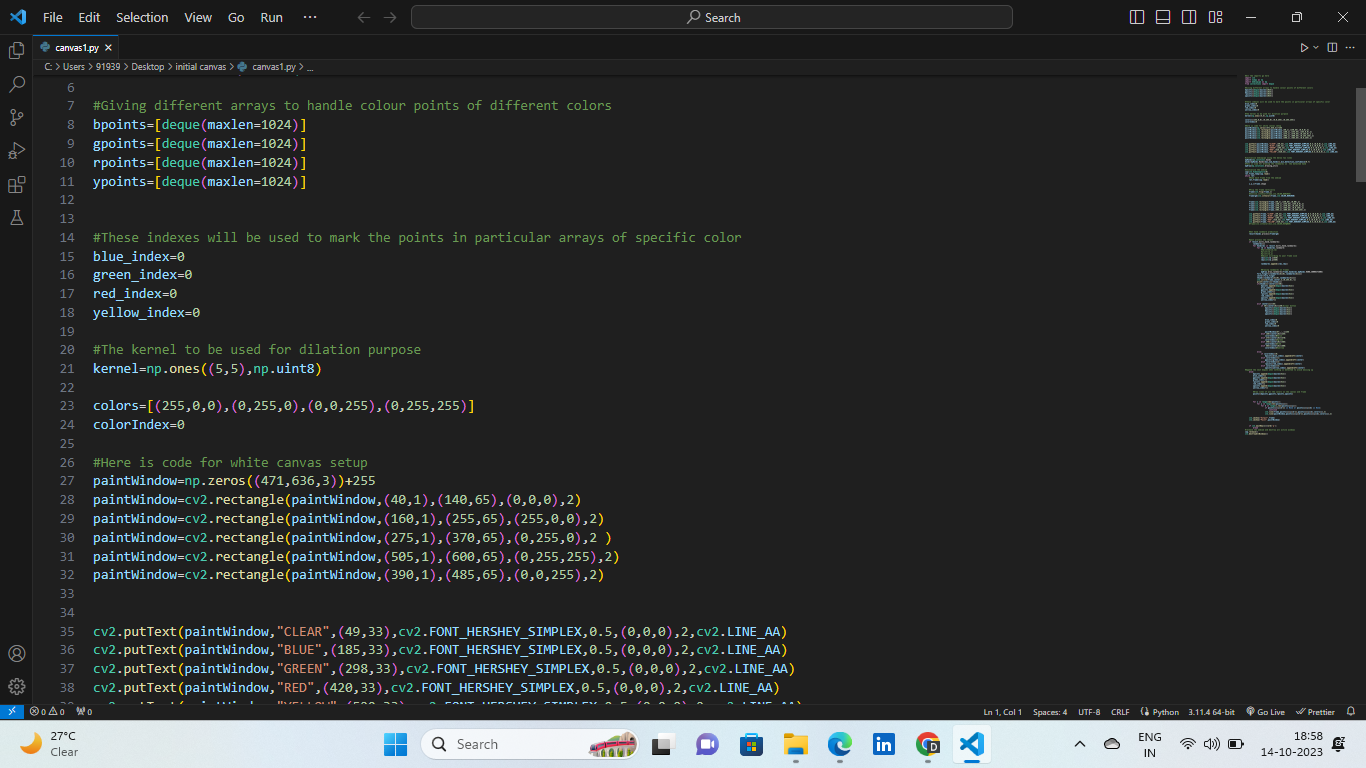
And the second static window named Paint also called feed is for drawing the recorded coordinates

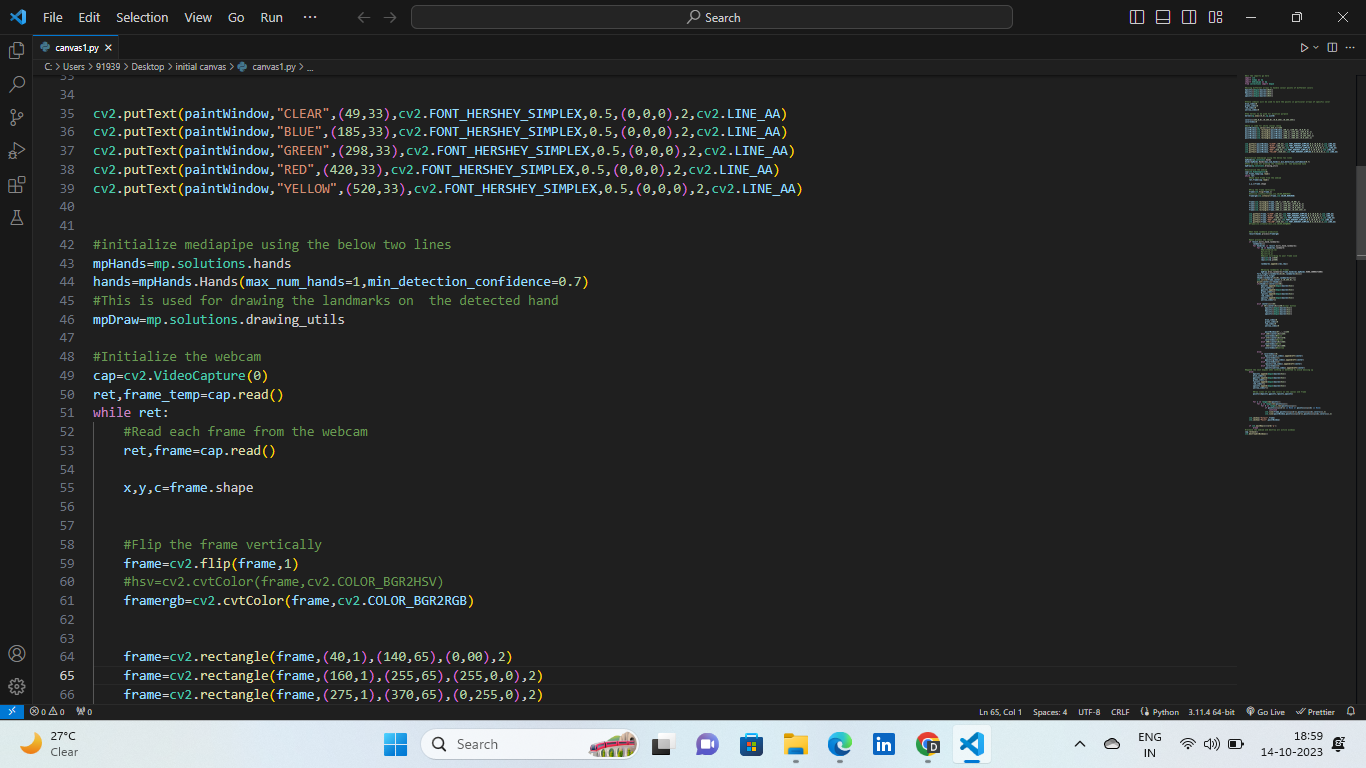
Also, user can use the components like clear and can change the color.

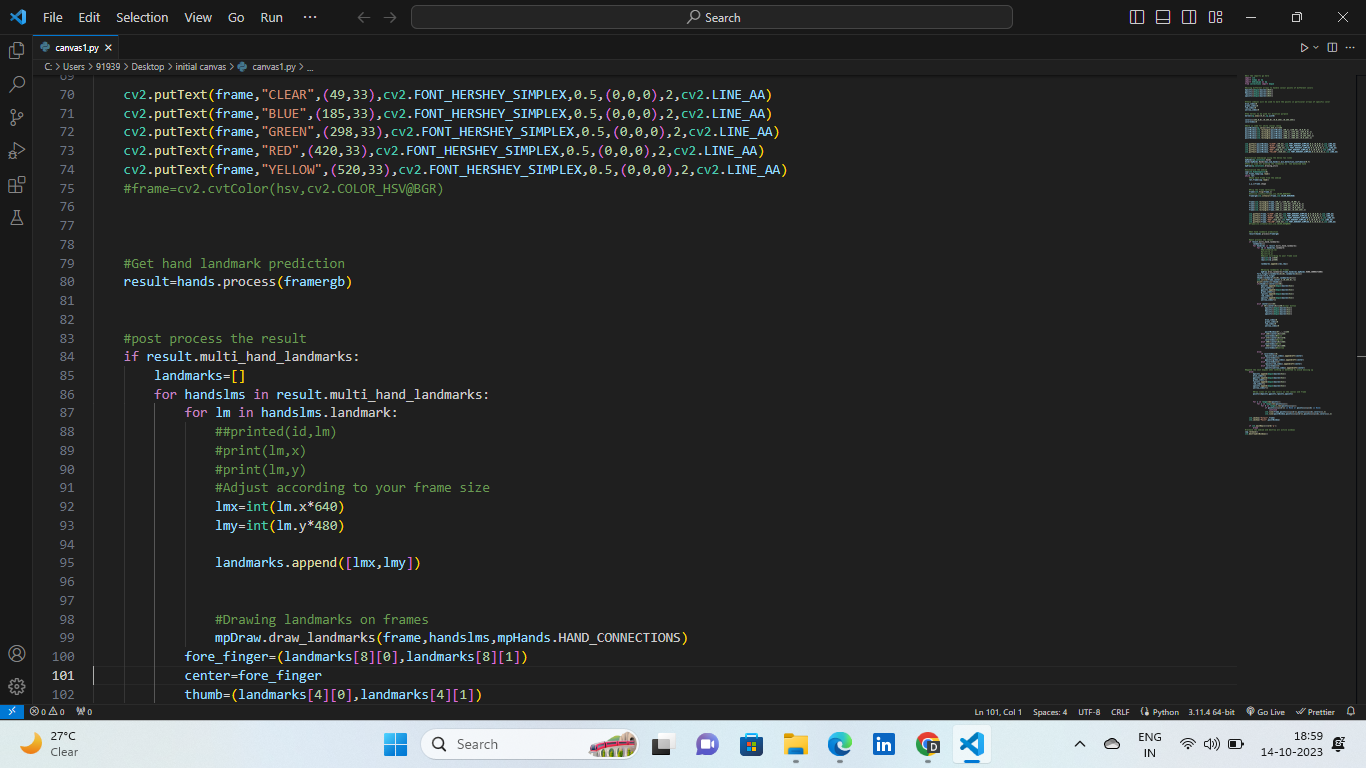
**SNAPSHOTS**

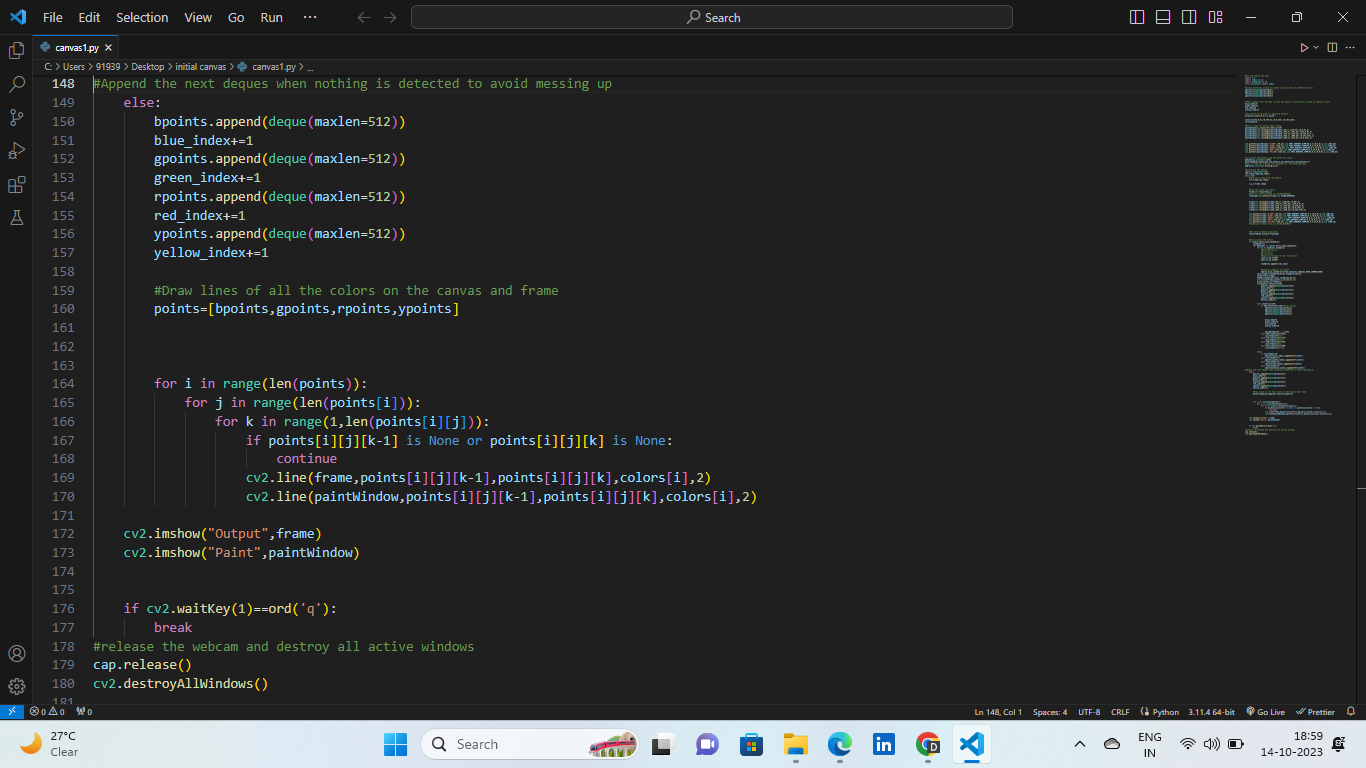
**SNAPS**

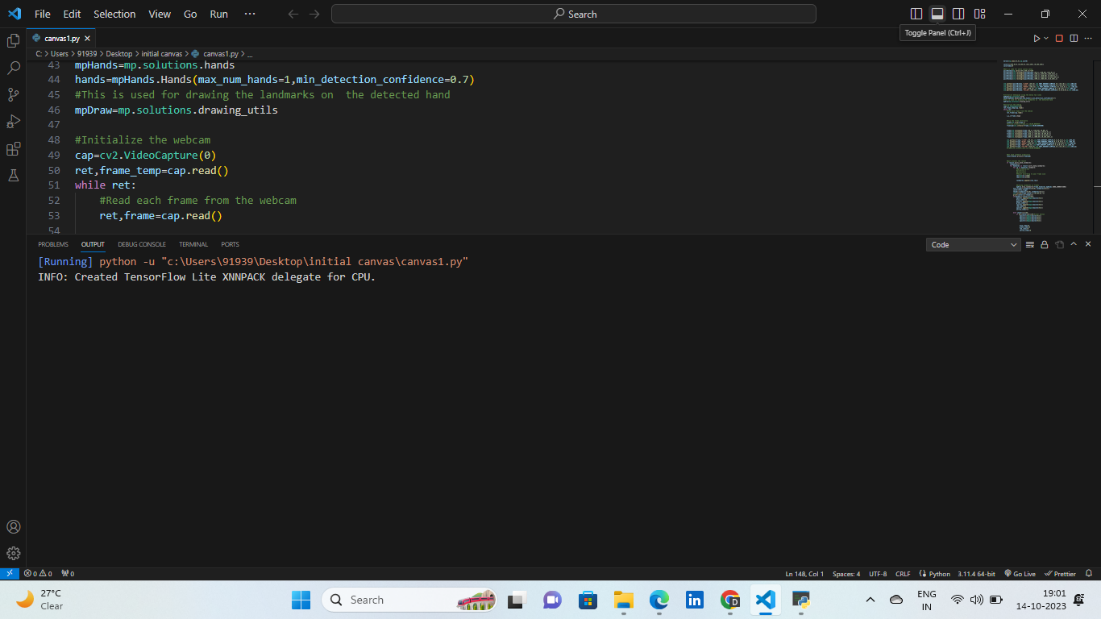




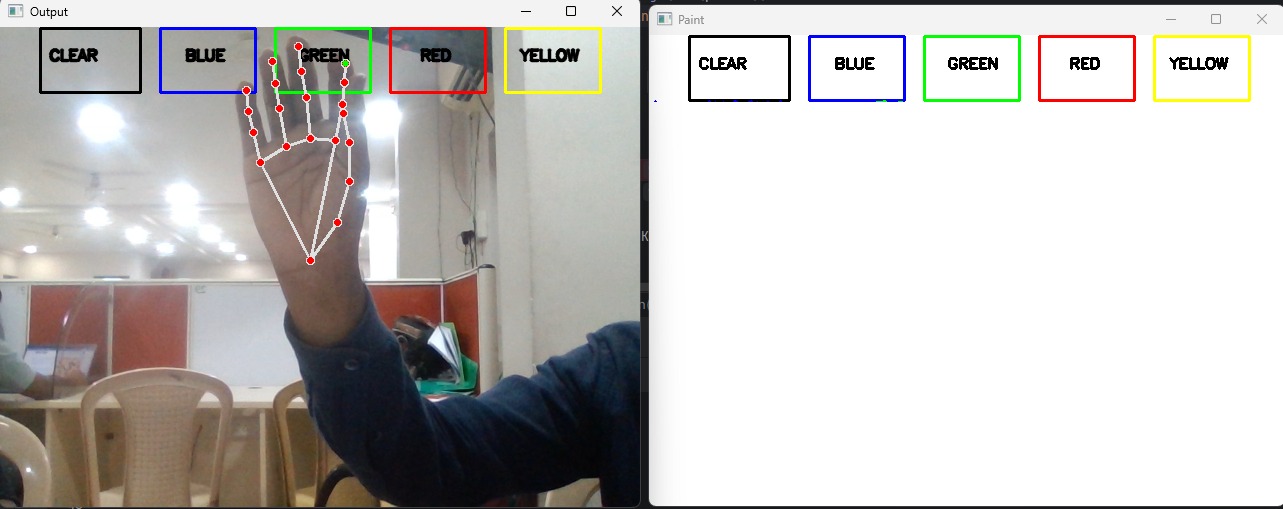




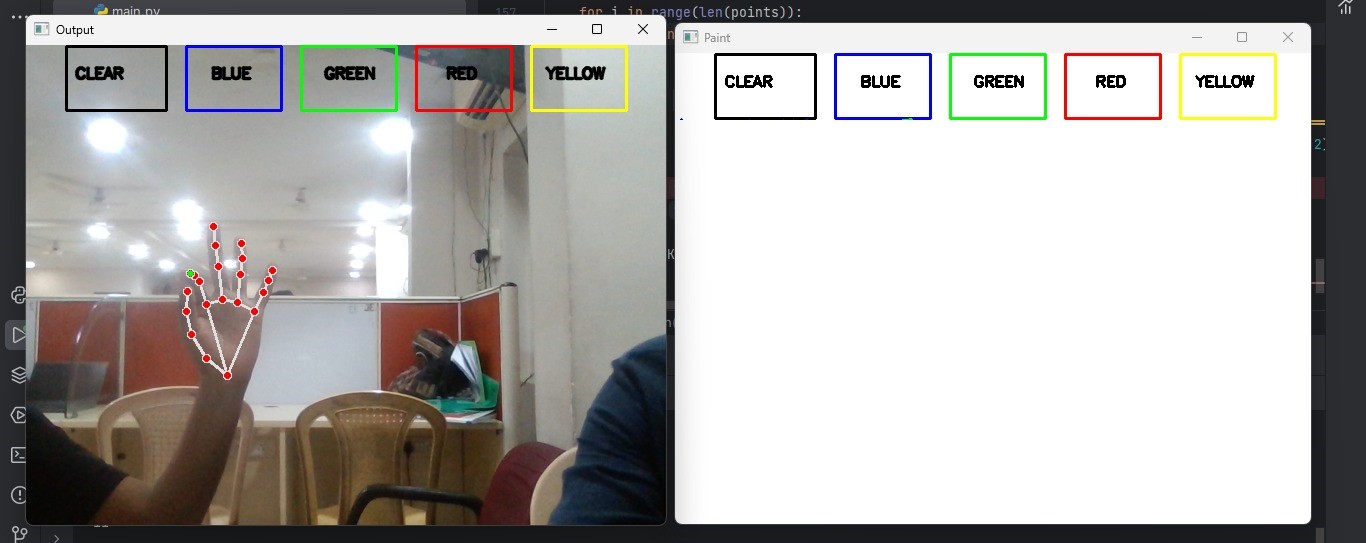




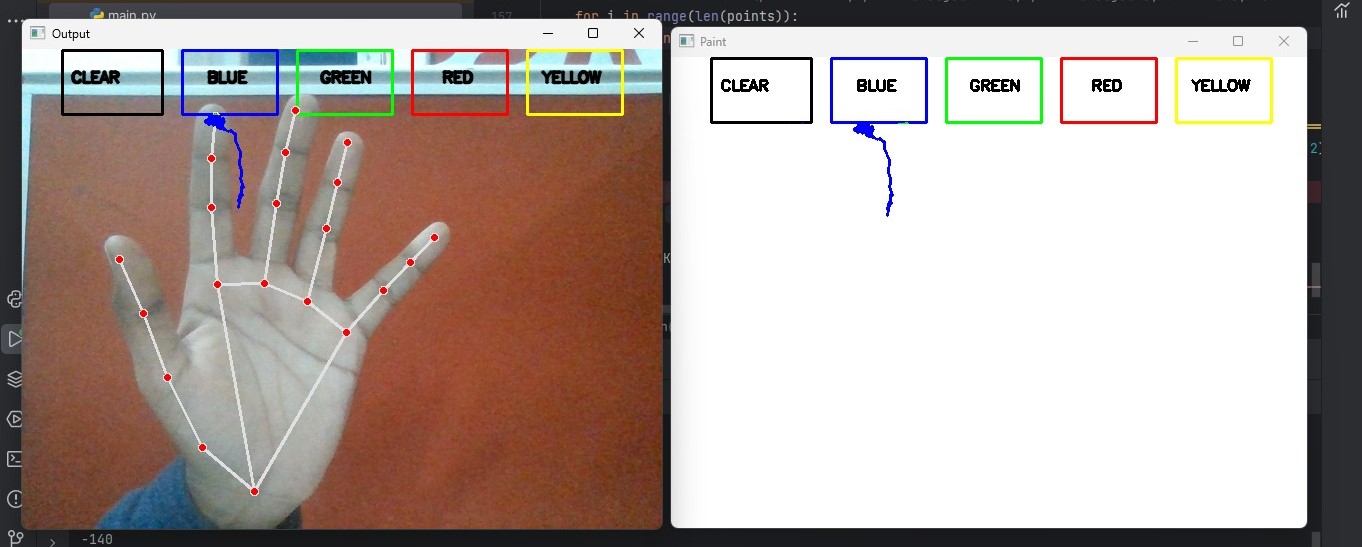
**DEMONSTRATION**

****

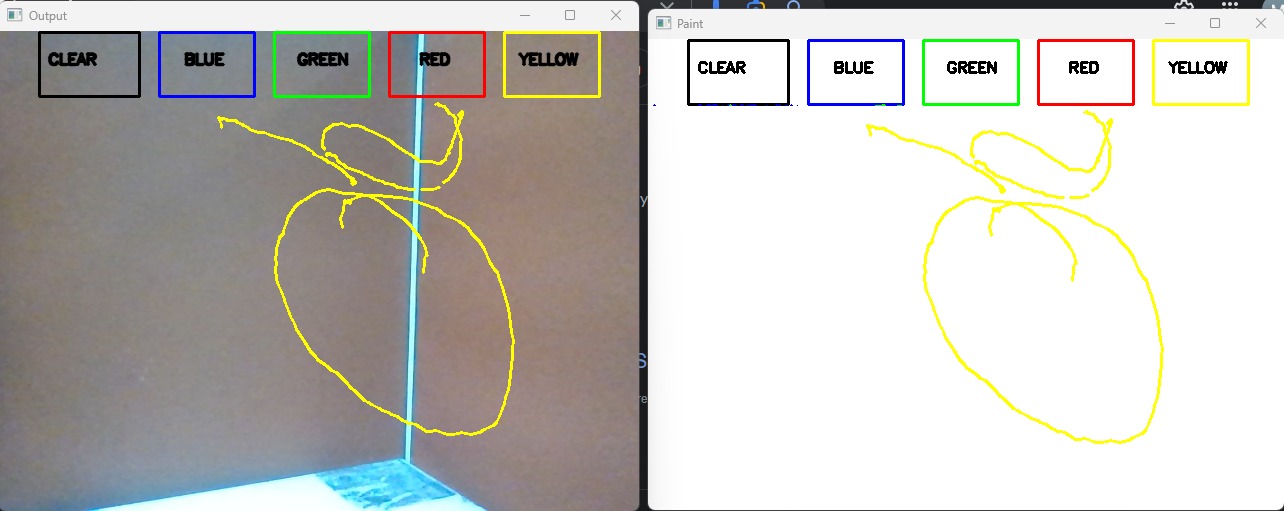
**COLOR SELECTION: User can choose colors for their work by selecting from colored ink buttons using gestures. This involve pointing at a color make a specific gesture to pick it.**

****

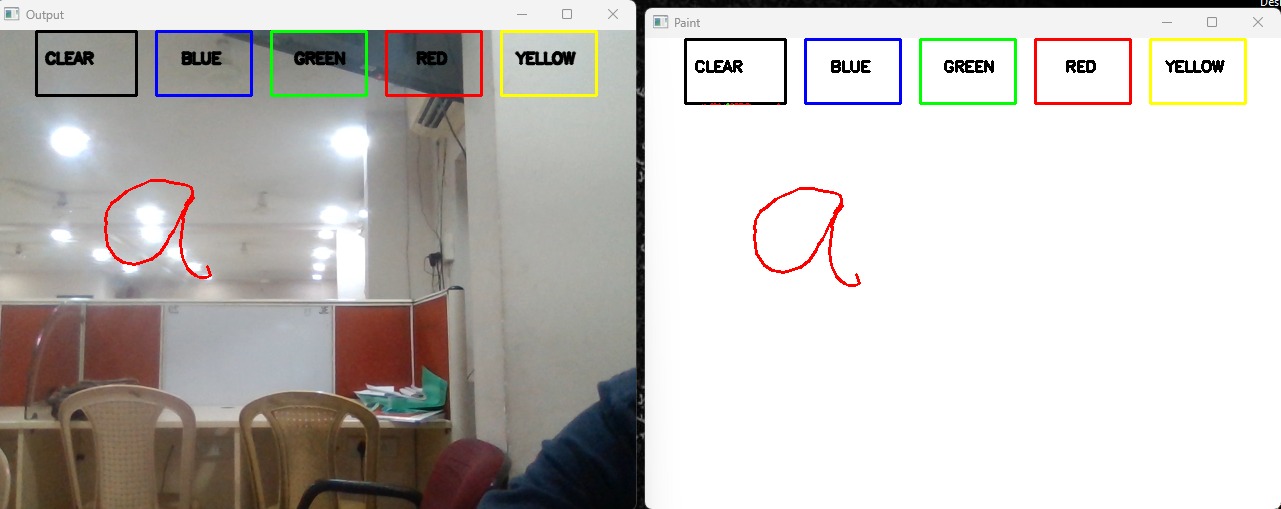
**OBJECT DETECTION: The system's software would process the input from the webcam to detect the presence of object(hand) in the field of view.**

****

**MOTION TRACKING: The system would track the movement of detected hand in the air. This information is used to determine the path of drawing as the user moves the colored pointer around.**

****

**DRAWING INTERACTION: As the user moves the hand in the air, the system would interpret the movements as drawing actions. It would replicate the path of the hand's index finger through camera on a digital canvas.**

****

**INTERACTIVITY: the system includes features like drawing, writing, switching, between colors based on the motion being tracked.**

****

**GESTURE RECOGNITION: the system could also incorporate gesture recognition to enable different drawing actions, such as selecting a color or clearing the canvas using eraser through predefined hand movements.**

**FUTURE   ENHANCEMENT**

**APPLICATIONS**

Air canvas have a wide range of applications across various industries. These tools offer a

Dynamic and interactive platform for creativity, collaboration and communication.

Some of the applications of digital air canvas are:

* **Interactive Teaching:**

Teachers can use digital air canvas to create engaging and interactive lessons, to

        Write, draw and share content with students in real-time.

* **Distance Learning:**

Facilitates remote or online learning by providing a platform for teachers an students

to interact as if they were in a physical classroom.

* **Business and Collaboration:**

Digital air canvas are used in business meetings for interactive , engaging presentations.

        It enables group work, brainstorming and collaborative projects.

* **Virtual Collaboration:**

Supports remote and distributed teams by providing a shared digital workspace for

           Collaboration.

* **Design and Creativity:**

Artists and designers use digital canvases to create digital art, illustrations and designs

with various tools.

* **Animation:**

Animators use digital air canvas for drawing and animating.

* **Conferences and Seminars:**

Presenters use digital air canvases to engage with the audience through interactive

         Presentations.

* **Video Storyboarding:**

Media producers and content creators can use digital air canvases for storyboarding,

             Educating and entertaining.

* **Language Learning:**

Language instructors can use digital air canvas to illustrate language concepts.

**LIMITATIONS**

When digital air canvas can offer many advantages and applications, they also have certain

Limitations. Understanding these limitations can help users make informed decisions about

There usage. Some limitations of air canvas are:

* **Technical Issues:**

Like all digital technology, air canvas can experience technical problems, including

           Software glitches, malfunctions and compatibility issues with the operating system

          or devices.

* **Dependency on Power and Connectivity:**

Digital air canvas rely on a stable power source and internet connectivity. Making them

          Less reliable in areas with frequent power outages or limited internet access.

* **Privacy and Security Concerns:**

Collaborative digital canvas may raise privacy and security concerns when sensitive or

           Confidential information is shared.

* **Tactile Feedback:**

Digital air canvas lacks the tactile feedback of traditional media like papers and

            Whiteboards. Users may miss the sensation of writing, drawing or interacting with

           Physical materials.

* **Short Lifespan:**

Like many digital devices, digital canvases may have short lifespan compared to

          Traditional tools. This can lead to electronic waste and environmental concerns.

* **Collaboration challenges:**

Collaborative work on digital canvas may be less intuitive  or familiar to some users.

* **Limited Workspace size:**

The physical size of canvas can limit the space available for creating and collaborating.

           Which can be restrictive for large project and group activities.

**FUNCTIONAL CHALLENGES**

Challenges identified in the air can application includes:

* Camera’s ability to detect and capture in various lighting conditions
* Precise motion tracking
* Minimize lag between the real-world actions and digital representation
* Faster movements are not detected by camera which leads to inefficient outputs
* Distance from camera

**CONCLUSION**

This program has potential to challenge traditional writing methods. In this technology gesture-

Based interactions facilitate a more dynamic and expressive communication, collaborations and

Sharing information in both personal and professional contexts. Augmented reality can bring text to life. By extending the functionality the system can be used for quick control of internet of things (IOT) these devices allow users to contol and manipulate physical objects and data in

Real-time bridging the gap between the digital world and real world. In future implementation

Of You Only Look Once YOLO a deep learning algorithm. Used for real-time object detection

In images and videos. YOLO has several versions such as YOLOv3 and YOLOv4. YOLO

Model can be integrated with digital air canvas application. These future enhancements are

Likely to transform digital air canvas into versatile tools that cater to a broader range of

Industries and user needs enhancing productivity, creativity and collaboration in the

Process. Advancement of Artificial Intelligence will improve the efficiency of writing in air.

As technology continues to evolve, we can expect more innovative features and functionalities

In air canvas.

**REFERENCES**

* Motivation: <https://courses.ece.cornell.edu/ece5990/ECE5725_Spring2019_Projects/jaj263_scl97/index.html>
* Object detection:
* <https://paperswithcode.com/task/object-detection>
* <https://opencv.org/>
* <https://developers.google.com/mediapipe>
* Research gate
* <https://www.researchgate.net/>
* https://github.com/
* Internet
* Wikipedia