AISaksham

edunet foundation

Air Canvas

Team:
Abhishek Nalla
Prince Beladiya
Shubham Vanani
Sujal Vadgama

Guide: Jay Rathod

OUTLINE

Abstract

Air Canvas is a tool that lets you draw using hand gestures. With Python, Mediapipe, and OpenCV, it tracks your hand movements and turns them into art on a virtual canvas. It's like painting in the air! This project makes creating art fun and intuitive, without needing fancy equipment. Give your creativity a new way to shine with Air Canvas!

Problem Statement

Ever felt like traditional drawing tools just get in the way? That's where Air Canvas comes in. It's a too that lets you draw with your hands, no fancy gadgets needed. Using Python, Mediapipe, and OpenCV, we're making art creation easy and fun for everyone. Our goal? To let you express yourself freely by turning your hand movements into beautiful drawings, no matter where you are. But the tricky part? Making sure it works smoothly for everyone, no matter their skill level or environment.

Aim

To develop Air Canvas, a tool that enables users to draw in real-time using hand gestures, thereby enhancing the spontaneity and accessibility of creative expression in digital art.

Objective

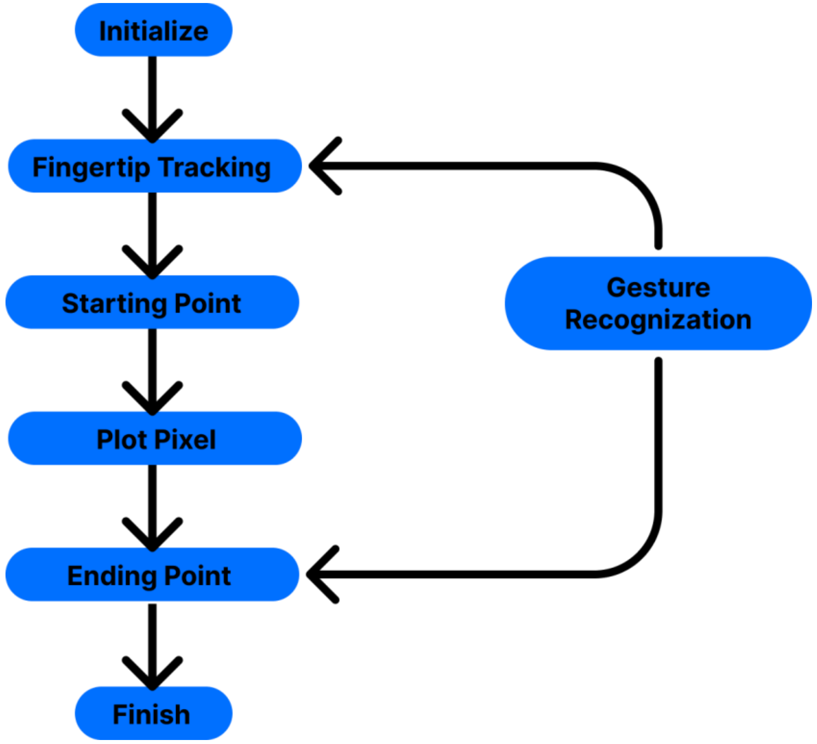
- ❑ Develop a user-friendly tool for Air Canvas.
- ❑ Implement hand gesture detection using Mediapipe and OpenCV.
- ❑ Optimize performance for smooth drawing experiences.
- ❑ Conduct user testing for feedback and improvements.
- ❑ Document development and provide user instructions.

Proposed Solution

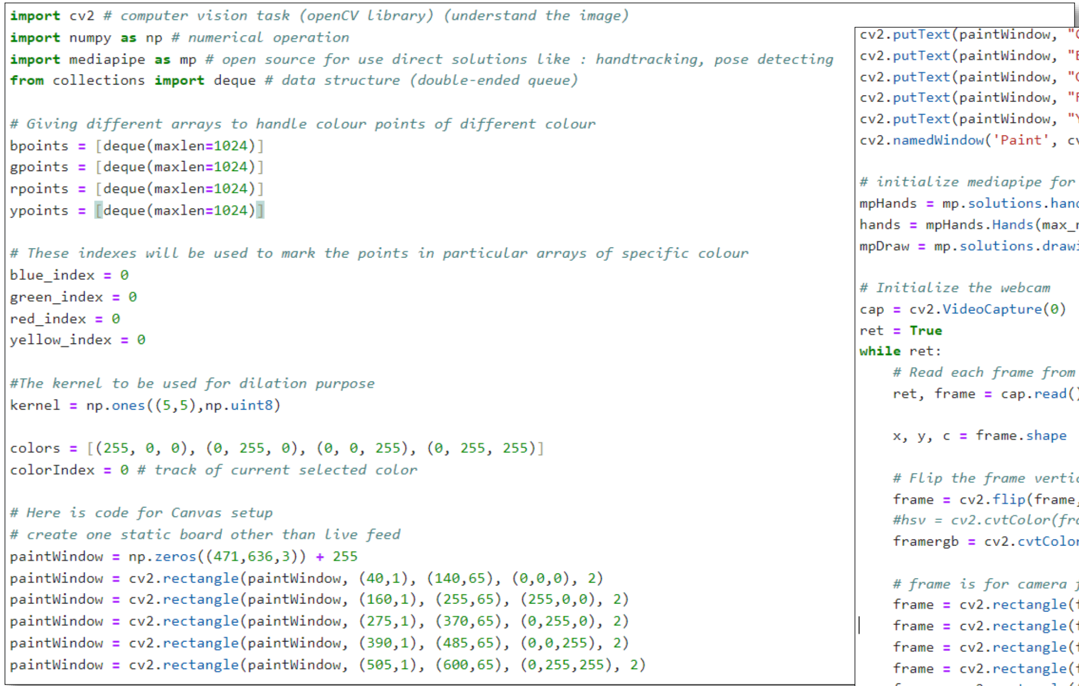
Air Canvas aims to revolutionize the digital art creation process by developing a web-based application that utilizes hand gestures for real-time drawing on a virtual canvas. The proposed solution involves the following key components:

- ❑ User-friendly tool
- ❑ Hand gesture recognition
- ❑ Real time drawing functionality
- ❑ Performance Optimization
- ❑ Documentation and Support

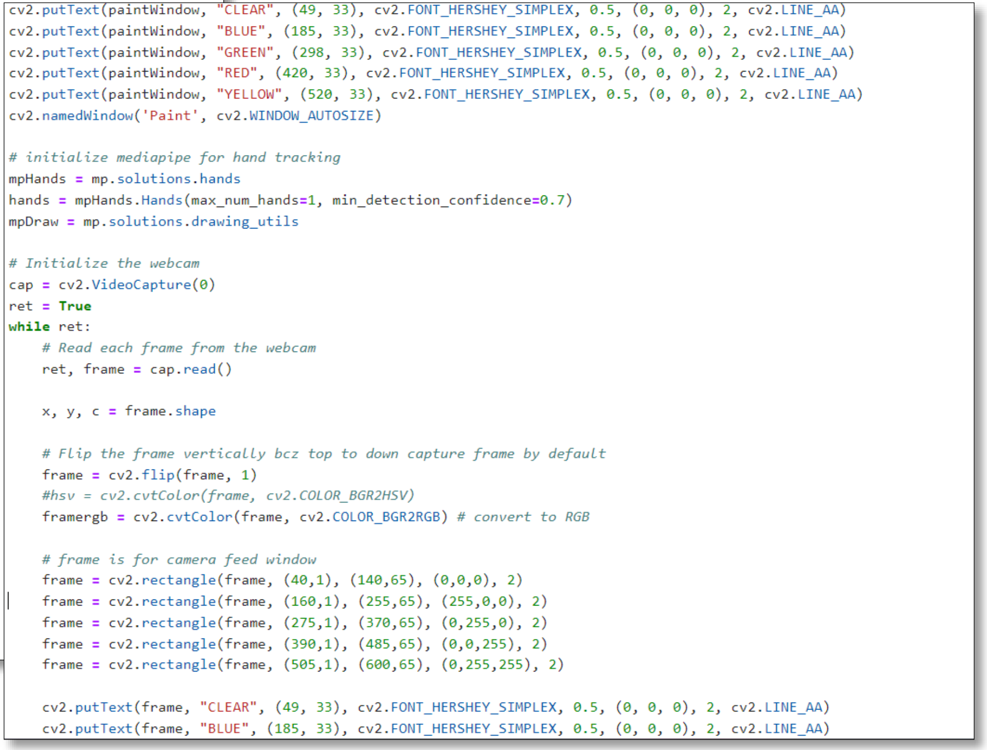
System Architecture



Algorithm & Deployment

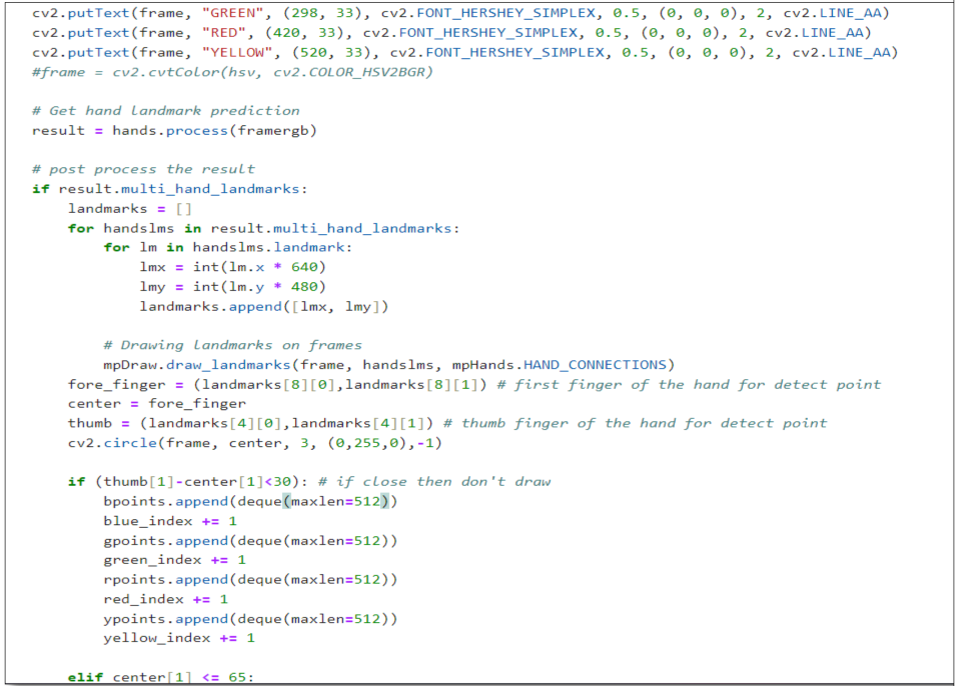


```python
import cv2 # computer vision task (openCV library) (understand the image)
import numpy as np # numerical operation
import mediapipe as mp # open source for use direct solutions like : handtracking, pose detecting
from collections import deque # data structure (double-ended queue)

# Giving different arrays to handle colour points of different colour
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 colour
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 # track of current selected color

# Here is code for Canvas setup
# create one static board other than live feed
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, (390,1), (485,65), (0,0,255), 2)
paintWindow = cv2.rectangle(paintWindow, (505,1), (600,65), (0,255,255), 2)
```

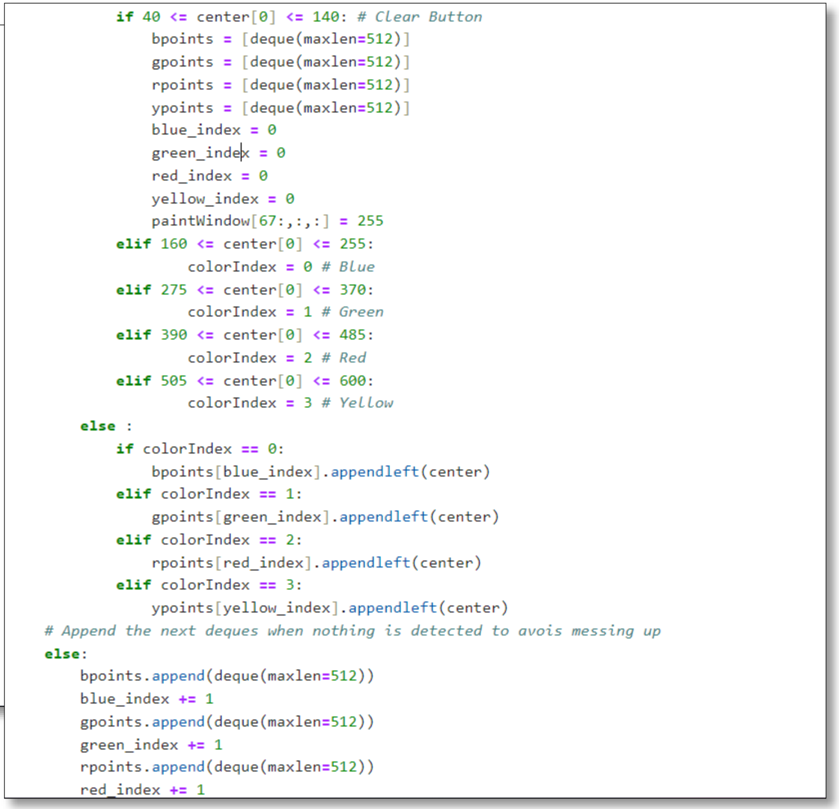


```python
cv2.putText(paintWindow, "CLEAR", (49, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
cv2.putText(paintWindow, "BLUE", (185, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
cv2.putText(paintWindow, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 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)
cv2.namedWindow('Paint', cv2.WINDOW_AUTOSIZE)

# initialize mediapipe for hand tracking
mpHands = mp.solutions.hands
hands = mpHands.Hands(max_num_hands=1, min_detection_confidence=0.7)
mpDraw = mp.solutions.drawing_utils

# Initialize the webcam
cap = cv2.VideoCapture(0)
ret = True
while ret:
    # Read each frame from the webcam
    ret, frame = cap.read()

    x, y, c = frame.shape

    # Flip the frame vertically bcz top to down capture frame by default
    frame = cv2.flip(frame, 1)
    #hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    framergb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB) # convert to RGB

    # frame is for camera feed window
    frame = cv2.rectangle(frame, (40,1), (140,65), (0,0,0), 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)
```

Algorithm & Deployment

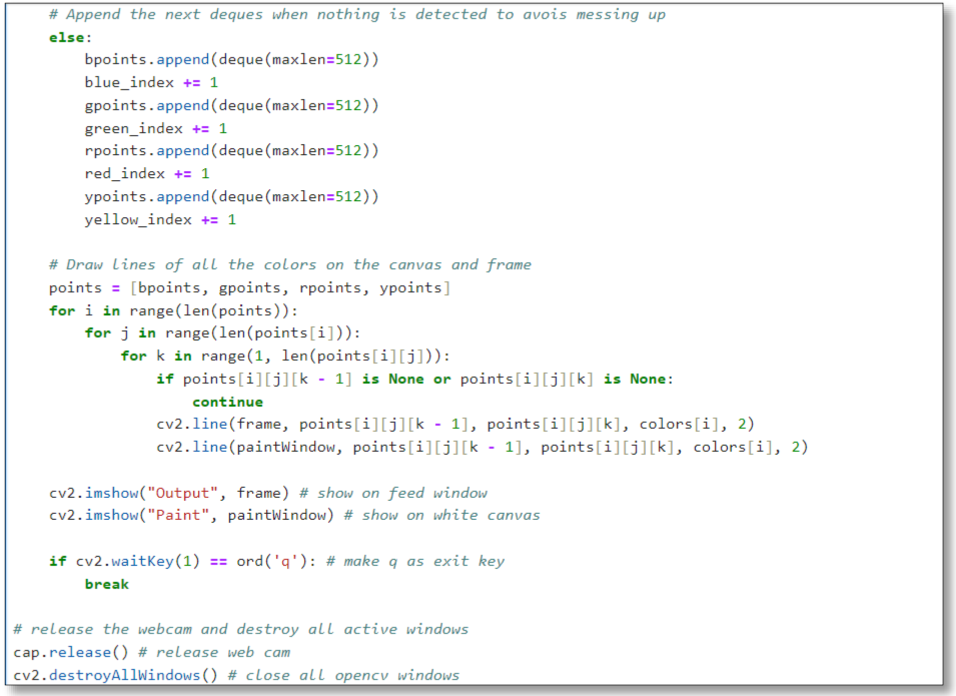


```
cv2.putText(frame, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
cv2.putText(frame, "RED", (420, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
cv2.putText(frame, "YELLOW", (520, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
#frame = cv2.cvtColor(hsv, cv2.COLOR_HSV2BGR)

# 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:
            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]) # first finger of the hand for detect point
    center = fore_finger
    thumb = (landmarks[4][0],landmarks[4][1]) # thumb finger of the hand for detect point
    cv2.circle(frame, center, 3, (0,255,0),-1)

    if (thumb[1]-center[1]<30): # if close then don't draw
        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 avois 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
```

Algorithm & Deployment



```
    # Append the next deques when nothing is detected to avois 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) # show on feed window
    cv2.imshow("Paint", paintWindow) # show on white canvas

    if cv2.waitKey(1) == ord('q'): # make q as exit key
        break

# release the webcam and destroy all active windows
cap.release() # release web cam
cv2.destroyAllWindows() # close all opencv windows
```

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.

Future Scope

- ❑ **Improved Gesture Recognition**
- ❑ **Collaborative Features** - Enable multiple users to draw together on the same canvas.
- ❑ **Integration with Digital Art Platforms** - Seamlessly export creations to professional software.
- ❑ **Accessibility Features** - Ensure usability for users with disabilities.
- ❑ **Mobile and Web Application** - Develop apps or web app for drawing on the go.
- ❑ **Community Engagement** - Foster a vibrant user community for sharing and collaboration.

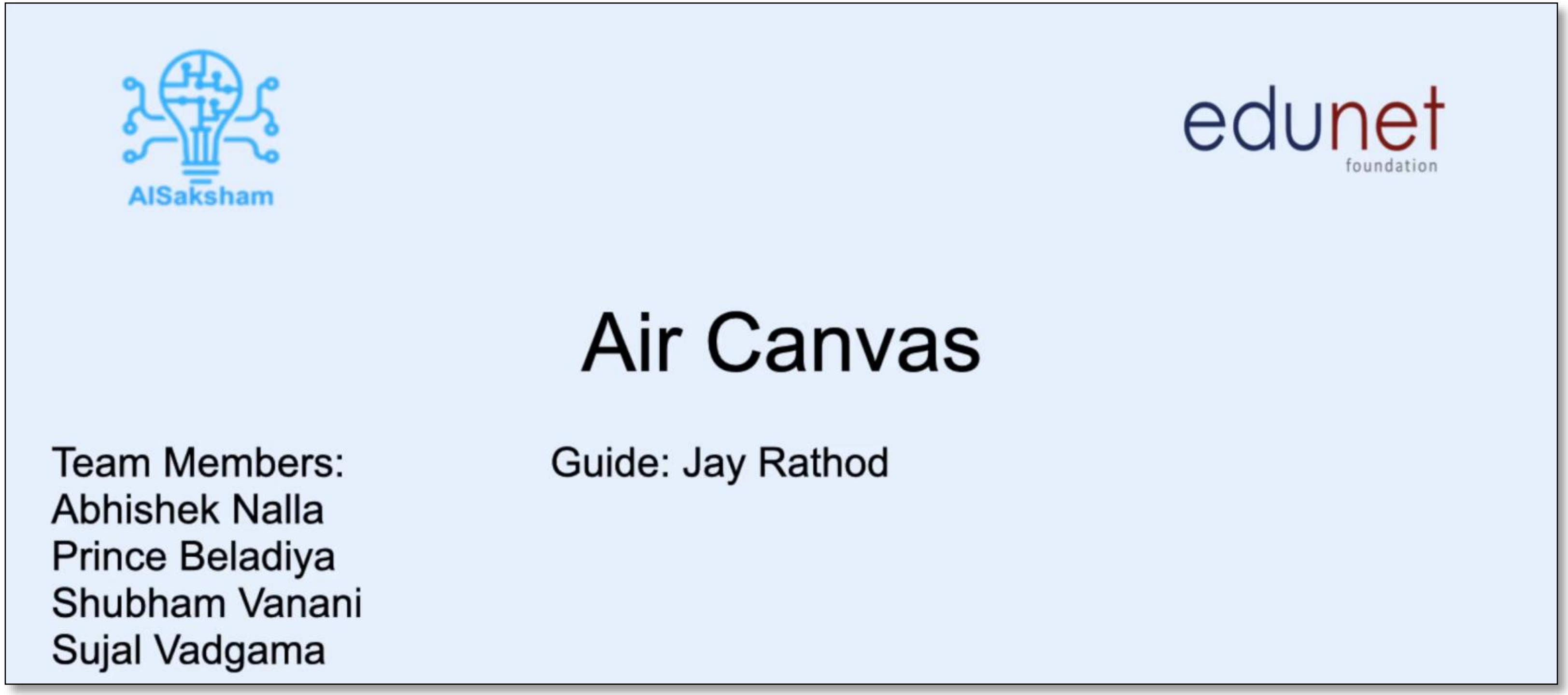
Air Canvas

Reference

- https://stackoverflow.com/questions/tagged/python
- https://www.youtube.com/playlist?list=PLTXuqKbKkxkTy764PhX1yil5hj-5va099
- https://docs.python.org/3/
- https://pypi.org/project/mediapipe/
- https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html

Air Canvas

- Demo Video



Air Canvas

Team Members:
Abhishek Nalla
Prince Beladiya
Shubham Vanani
Sujal Vadgama

Guide: Jay Rathod

Thank you!