RCOEM

Shri Ramdeobaba College of Engineering and Management, Nagpur

VIRTUAL PAINTER

USING

ARTIFICIAL INTELLIGENCE AND OPENCV

Guided By-:

PROF. SHUBHANGI TIRPUDE

Created By-:

- Reema Khandelwal(A-16)
- Ayush Shete(B-29)
- Faysal Khan(B-41)
- Vinni Fengade (B-67)

TABLE OF CONTENTS

1) Problem Statement-:

To make Virtual Painter using Artificial intelligence and OpenCV

2) Abstract-:

The AI virtual painter application made using OpenCV and Mediapipe libraries. It is an application that tracks the movement of an object. Using this tracking feature, the user is able to draw on devices such as screens just by moving the object in different ways or positions.

In our project we've considered the human hand as a movable object. The hand moves in the air while the webcam is on. This motion enables the user to draw various challenging and interesting things just by using the hands.

3) Prog. Lang/ Tool /API used for the Assignment-:

In this project, tools used are

- **Python as Base Language**
- **OpenCV for Image Processing Language**
  OpenCV (Open Source Computer Vision) - it is a programming language library which consists of different types of functions for computer vision mainly. OpenCV is particularly used for Image Processing and to perform all the operations related to Images.

OpenCV can:

1. Read and Write Images.
2. Detect faces and their features.
3. Modify the quality of an image or its color.
4. Perform text recognition in images. e.g Reading posters, banners, vehicle number plates, etc.
5. Perform image processing and computer vision tasks.
6. Perform tasks such as object detection and classification, object tracking , face detection, landmark detection and a lot more.
7. Develop Augmented Reality (AR) applications.
8. Work on computer vision, image processing and machine learning.

- **Mediapipe as Framework making Media Processing Faster**
  MediaPipe is Google's open source framework (graph based) used for media processing. Mainly aims at making media processing easier for us by providing machine learning features and some integrated computer vision.
  Some of it's notable applications are as follows:
  1. Face detection
  2. Multi-hand tracking
  3. Segmenting hair
  4. Detecting and tracking an object
  5. AutoFlip - pipeline to crop videos automatically

4) Project Justification-:

Your screen is a device that does exactly what you are asking - it displays data.

The keyboard which is a traditional and widely used method to display data on the screen ,it has a keyboard composed of buttons used to create letters, numbers, and symbols, and perform additional functions but it has a few disadvantages such as it is a slow method when you need to write a long piece of writing when there are faster ways such as scanning and dictation.

The second method is the speech to text ,this is a software that works by listening to audio and delivering an editable, verbatim transcript on a given device. The software does this through voice recognition. The drawbacks of this method is that the voice recognition software won't always put your words on the screen completely accurately. Programs cannot understand the context of language the way that humans can, leading to errors that are often due to misinterpretation and it cannot always differentiate between homonyms.

Another method is a touchscreen. It is a computer screen that can be used by touching it with a finger or a stylus pen, instead of using a mouse and keyboard. It can be described as a touchpad with a screen built-in to it. A few disadvantages are that they are not suitable for inputting large amounts of data ,not very accurate - selecting detailed objects can be difficult with fingers it is more expensive than alternatives such as a mouse and can soon become faulty if misused.

5) Proposed Solution –:

Writing Hand Pose Detection

Recognizing the position of the composing hand and recognizing it through other signals is an fundamental step in initializing airborne composing. Not at all like conventional writing, when the write moves down, and the write moves up, composing within the discuss isn't laid out as a writing arrangement. Events. The framework recognizes the position of a piece hand and recognizes it from a non-writing hand by tallying the number of raised fingers.

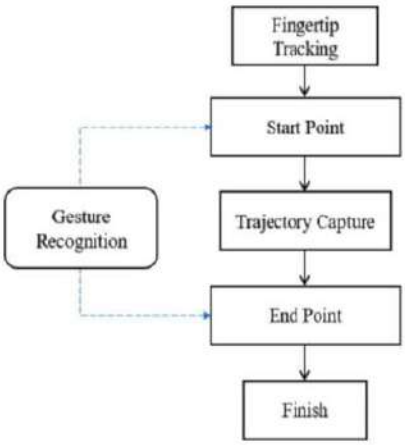
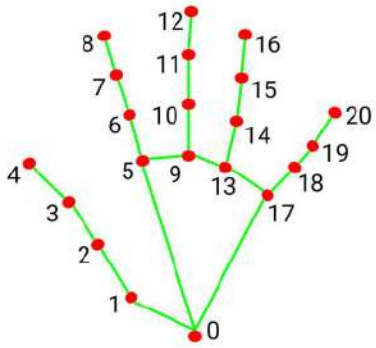


Fig: Flowchart of Proposed System

Hand Region Segmentation

Once we have precisely captured the hand utilizing the over procedure, the division of the hand zone is done employing a two-step approach, viz. The skin division and the subtraction of the foundation and the ultimate parallel picture of the hand are obtained as a conglomeration of the two. The proposed algorithm works well in real-time and gives moderately precise division. In spite of the fact that skin colors shift incredibly from breed to breed, it has been watched that skin color features a little region between distinctive skin sorts, whereas skin luminosity differs significantly.

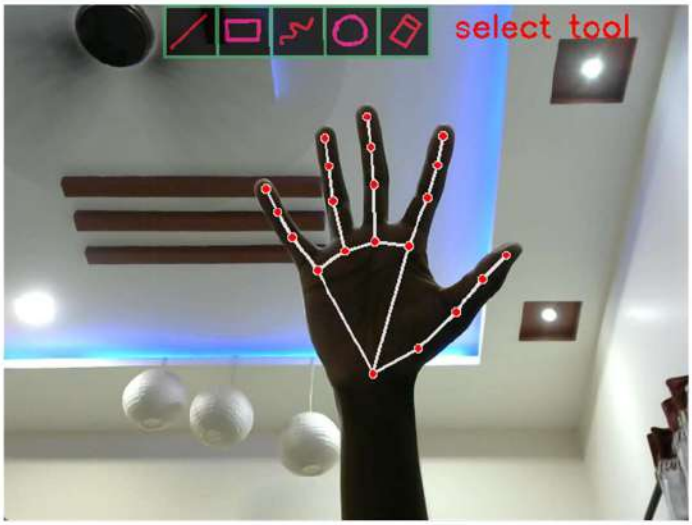


0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Background subtraction: Since precise hand location with the Speedier R-CNN handheld locator taken after by sifting of skin color at the boundary of the candidate's hand gives a sensibly great division result, the subtraction step of background is as it were utilized to evacuate skin-colored objects (not portion of the hand) that are within the bounding box of the recognized hand may be present.

Hand Centroid Localization

Since deciding the precise center of gravity of the hand is basic within the taking after steps, the framework employments two calculations to decide the initial estimates of the center of gravity, and the ultimate center of gravity is calculated as the normal of the two. The remove change strategy is utilized to obtain the first assess of the center of gravity (xc1, yc1). Within the separate change picture, each pixel is spoken to by its separate from the following edge pixel. Euclidean separate was utilized to degree the distance between a pixel and its closest edge pixel. Hence, the pixel with the most noteworthy escalated within the separate change picture is taken as the center of gravity.



Drawing Line using Position of the Contour

The real rationale behind this computer vision extend is to form a Python deque (a information structure). The deque will memorize the position of the diagram in each subsequent outline, and we are going utilize these collected focuses to form a line utilizing OpenCV's drawing capabilities. Presently utilize the layout position to choose whether to press a button or draw on the given sheet. A few of the buttons are at the best of the canvas. When the pointer enters this range, it is enacted agreeing to the strategy show in this area.

6) Code –:

```
import mediapipe as mp
import cv2
import numpy as np
import time

#constants
ml = 150
max_x, max_y = 250+ml, 50
curr_tool = "select tool"
time_init = True
rad = 40
var_inits = False
thick = 4
prevx, prevy = 0,0

#get tools function
def getTool(x):
    if x < 50 + ml:
        return "line"

    elif x<100 + ml:
        return "rectangle"

    elif x < 150 + ml:
        return"draw"

    elif x<200 + ml:
        return "circle"

    else:
        return "erase"

def index_raised(yi, y9):
    if (y9 - yi) > 40:
        return True
```

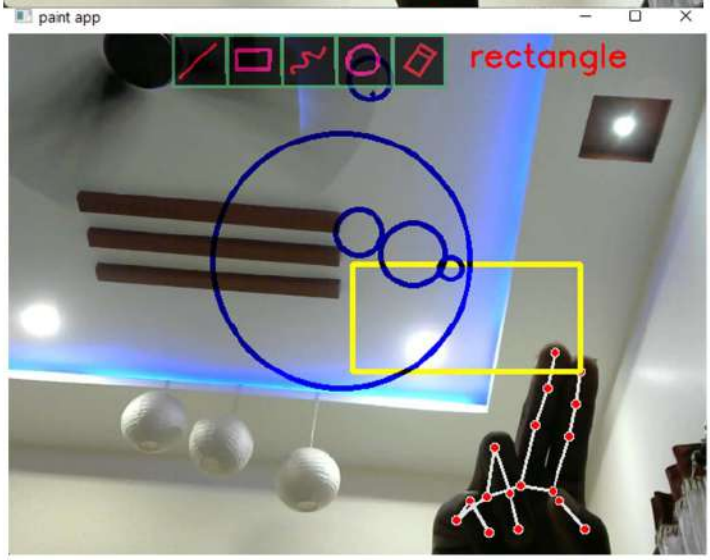
```
    return False

hands = mp.solutions.hands
hand_landmark = hands.Hands(min_detection_confidence=0.6, min_tracking_confidence=0.6,
max_num_hands=1)
draw = mp.solutions.drawing_utils


# drawing tools
tools = cv2.imread("tools.png")
tools = tools.astype('uint8')

mask = np.ones((480, 640))*255
mask = mask.astype('uint8')
'''
tools = np.zeros((max_y+5, max_x+5, 3), dtype="uint8")
cv2.rectangle(tools, (0,0), (max_x, max_y), (0,0,255), 2)
cv2.line(tools, (50,0), (50,50), (0,0,255), 2)
cv2.line(tools, (100,0), (100,50), (0,0,255), 2)
cv2.line(tools, (150,0), (150,50), (0,0,255), 2)
cv2.line(tools, (200,0), (200,50), (0,0,255), 2)
'''

cap = cv2.VideoCapture(0)
while True:
    _, frm = cap.read()
    frm = cv2.flip(frm, 1)

    rgb = cv2.cvtColor(frm, cv2.COLOR_BGR2RGB)

    op = hand_landmark.process(rgb)

    if op.multi_hand_landmarks:
        for i in op.multi_hand_landmarks:
            draw.draw_landmarks(frm, i, hands.HAND_CONNECTIONS)
            x, y = int(i.landmark[8].x*640), int(i.landmark[8].y*480)
```

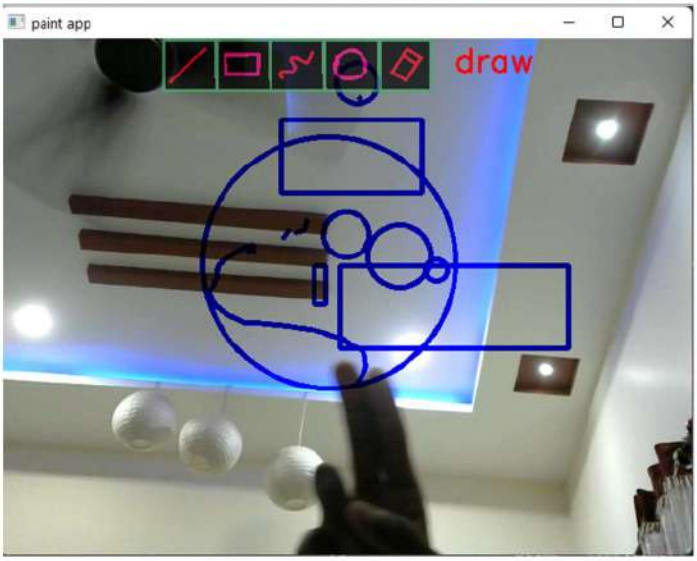
```
if x < max_x and y < max_y and x > ml:
    if time_init:
        ctime = time.time()
        time_init = False
    ptime = time.time()

    cv2.circle(frm, (x, y), rad, (0,255,255), 2)
    rad -= 1

    if (ptime - ctime) > 0.8:
        curr_tool = getTool(x)
        print("your current tool set to : ", curr_tool)
        time_init = True
        rad = 40

else:
    time_init = True
    rad = 40

if curr_tool == "draw":
    xi, yi = int(i.landmark[12].x*640), int(i.landmark[12].y*480)
    y9  = int(i.landmark[9].y*480)

    if index_raised(yi, y9):
        cv2.line(mask, (prevx, prevy), (x, y), 0, thick)
        prevx, prevy = x, y

    else:
        prevx = x
        prevy = y

elif curr_tool == "line":
    xi, yi = int(i.landmark[12].x*640), int(i.landmark[12].y*480)
    y9  = int(i.landmark[9].y*480)

    if index_raised(yi, y9):
```

```
            if not(var_inits):
                xii, yii = x, y
                var_inits = True

            cv2.line(frm, (xii, yii), (x, y), (50,152,255), thick)

        else:
            if var_inits:
                cv2.line(mask, (xii, yii), (x, y), 0, thick)
                var_inits = False

    elif curr_tool == "rectangle":
        xi, yi = int(i.landmark[12].x*640), int(i.landmark[12].y*480)
        y9  = int(i.landmark[9].y*480)

        if index_raised(yi, y9):
            if not(var_inits):
                xii, yii = x, y
                var_inits = True

            cv2.rectangle(frm, (xii, yii), (x, y), (0,255,255), thick)

        else:
            if var_inits:
                cv2.rectangle(mask, (xii, yii), (x, y), 0, thick)
                var_inits = False

    elif curr_tool == "circle":
        xi, yi = int(i.landmark[12].x*640), int(i.landmark[12].y*480)
        y9  = int(i.landmark[9].y*480)

        if index_raised(yi, y9):
            if not(var_inits):
                xii, yii = x, y
                var_inits = True

            cv2.circle(frm, (xii, yii), int(((xii-x)**2 + (yii-y)**2)**0.5), (255,255,0), thick)
```

```
            else:
                if var_inits:
                    cv2.circle(mask, (xii, yii), int(((xii-x)**2 + (yii-y)**2)**0.5), (0,255,0), thick)
                    var_inits = False

        elif curr_tool == "erase":
            xi, yi = int(i.landmark[12].x*640), int(i.landmark[12].y*480)
            y9  = int(i.landmark[9].y*480)

            if index_raised(yi, y9):
                cv2.circle(frm, (x, y), 30, (0,0,0), -1)
                cv2.circle(mask, (x, y), 30, 255, -1)



op = cv2.bitwise_and(frm, frm, mask=mask)
frm[:, :, 1] = op[:, :, 1]
frm[:, :, 2] = op[:, :, 2]

frm[:max_y, ml:max_x] = cv2.addWeighted(tools, 0.7, frm[:max_y, ml:max_x], 0.3, 0)

cv2.putText(frm, curr_tool, (270+ml,30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)
cv2.imshow("paint app", frm)

if cv2.waitKey(1) == 27:
    cv2.destroyAllWindows()
    cap.release()
    break
```

7) Output –:





8) Conclusion–:

This user-friendly application is built using python, NumPy and OpenCV. Just by using a single finger the user can draw anything of their choice and by using two fingers they can make a choice regarding which tool to pick from the available tools. It becomes easy for the user to operate and even a naive user can easily handle it. This application principally demonstrates the interaction between man and machine and is a very effective tool as it can be used in various fields.

9) References–:

[1] https://ieeexplore.ieee.org/abstract/document/9742419

[2] https://ieeexplore.ieee.org/abstract/document/9382703

[3] https://ieeexplore.ieee.org/abstract/document/9340297