AI Virtual Mouse (https://www.computervision.zone/courses/ai-virtual-mouse/)

0% COMPLETE

Previous Lesson (https://www.computervision.zone/lessons/ai-virtual-mouse-video-lesson/)

Mark Complete

AI Virtual Mouse – Video Lesson (https://www.computervision.zone/lessons/ai-virtual-mouse-video-lesson/)

Code Files (https://www.computervision.zone/lessons/code-files-17/)

Code Files

AI Virtual Mouse (https://www.computervision.zone/courses/ai-virtual-mouse/) > Code Files (https://www.c...

IN PROGRESS

AiVirtualMouseProject.py

Python

```
import numpy as np
import HandTrackingModule as htm
import time
import autopy

##########################
wCam, hCam = 640, 480
frameR = 100 # Frame Reduction
smoothening = 7
#########################

pTime = 0
plocX, plocY = 0, 0
clocX, clocY = 0, 0

cap = cv2.VideoCapture(1)
cap.set(3, wCam)
cap.set(4, hCam)
detector = htm.handDetector(maxHands=1)
wScr, hScr = autopy.screen.size()
# print(wScr, hScr)

while True:
    # 1. Find hand Landmarks
    success, img = cap.read()
    img = detector.findHands(img)
    lmList, bbox = detector.findPosition(img)
    # 2. Get the tip of the index and middle fingers
    if len(lmList) != 0:
        x1, y1 = lmList&#91;8]&#91;1:]
        x2, y2 = lmList&#91;12]&#91;1:]
        # print(x1, y1, x2, y2)

    # 3. Check which fingers are up
    fingers = detector.fingersUp()
    # print(fingers)
    cv2.rectangle(img, (frameR, frameR), (wCam - frameR, hCam - frameR),
    (255, 0, 255), 2)
    # 4. Only Index Finger : Moving Mode
                                                    1;2] == 0:
```

```
        # 5. Convert Coordinates
        x3 = np.interp(x1, (frameR, wCam - frameR), (0, wScr))
        y3 = np.interp(y1, (frameR, hCam - frameR), (0, hScr))
        # 6. Smoothen Values
        clocX = plocX + (x3 - plocX) / smoothening
        clocY = plocY + (y3 - plocY) / smoothening

        # 7. Move Mouse
        autopy.mouse.move(wScr - clocX, clocY)
        cv2.circle(img, (x1, y1), 15, (255, 0, 255), cv2.FILLED)
        plocX, plocY = clocX, clocY

    # 8. Both Index and middle fingers are up : Clicking Mode
    if fingers&#91;1] == 1 and fingers&#91;2] == 1:
        # 9. Find distance between fingers
        length, img, lineInfo = detector.findDistance(8, 12, img)
        print(length)
        # 10. Click mouse if distance short
        if length &lt; 40:
            cv2.circle(img, (lineInfo&#91;4], lineInfo&#91;5]),
            15, (0, 255, 0), cv2.FILLED)
            autopy.mouse.click()

    # 11. Frame Rate
    cTime = time.time()
    fps = 1 / (cTime - pTime)
    pTime = cTime
    cv2.putText(img, str(int(fps)), (20, 50), cv2.FONT_HERSHEY_PLAIN, 3,
    (255, 0, 0), 3)
    # 12. Display
    cv2.imshow("Image", img)
    cv2.waitKey(1)
```

HandTrackingModule.py

```
"""
Hand Tracing Module
By: Murtaza Hassan
Youtube: http://www.youtube.com/c/MurtazasWorkshopRoboticsandAI
Website: https://www.murtazahassan.com/
"""

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


class handDetector():
    def __init__(self, mode=False, maxHands=2, detectionCon=0.5, trackCon=0.5):
        self.mode = mode
        self.maxHands = maxHands
        self.detectionCon = detectionCon
        self.trackCon = trackCon

        self.mpHands = mp.solutions.hands
        self.hands = self.mpHands.Hands(self.mode, self.maxHands,
                                        self.detectionCon, self.trackCon)
        self.mpDraw = mp.solutions.drawing_utils
        self.tipIds = &#91;4, 8, 12, 16, 20]

    def findHands(self, img, draw=True):
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)
        # print(results.multi_hand_landmarks)

```

```
33          if self.results.multi_hand_landmarks:
34              for handLms in self.results.multi_hand_landmarks:
35                  if draw:
36                      self.mpDraw.draw_landmarks(img, handLms,
37                                                 self.mpHands.HAND_CONNECTIONS)
38
39          return img
40
41      def findPosition(self, img, handNo=0, draw=True):
42          xList = &#91;]
43          yList = &#91;]
44          bbox = &#91;]
45          self.lmList = &#91;]
46          if self.results.multi_hand_landmarks:
47              myHand = self.results.multi_hand_landmarks&#91;handNo]
48              for id, lm in enumerate(myHand.landmark):
49                  # print(id, lm)
50                  h, w, c = img.shape
51                  cx, cy = int(lm.x * w), int(lm.y * h)
52                  xList.append(cx)
53                  yList.append(cy)
54                  # print(id, cx, cy)
55                  self.lmList.append(&#91;id, cx, cy])
56                  if draw:
57                      cv2.circle(img, (cx, cy), 5, (255, 0, 255), cv2.FILLED)
58
59              xmin, xmax = min(xList), max(xList)
60              ymin, ymax = min(yList), max(yList)
61              bbox = xmin, ymin, xmax, ymax
62
63              if draw:
64                  cv2.rectangle(img, (xmin - 20, ymin - 20), (xmax + 20, ymax + 20),
65                                (0, 255, 0), 2)
66
67          return self.lmList, bbox
68
69      def fingersUp(self):
70          fingers = &#91;]
71          # Thumb
72          if self.lmList&#91;self.tipIds&#91;0]]&#91;1] > self.lmList&#91;self.tipIds&#91;
73              fingers.append(1)
74          else:
75              fingers.append(0)
76
77          # Fingers
78          for id in range(1, 5):
79
80              if self.lmList&#91;self.tipIds&#91;id]]&#91;2] &lt; self.lmList&#91;self.tip
81                  fingers.append(1)
82              else:
83                  fingers.append(0)
84
85          # totalFingers = fingers.count(1)
86
87          return fingers
88
89      def findDistance(self, p1, p2, img, draw=True,r=15, t=3):
90          x1, y1 = self.lmList&#91;p1]&#91;1:]
91          x2, y2 = self.lmList&#91;p2]&#91;1:]
92          cx, cy = (x1 + x2) // 2, (y1 + y2) // 2
93
94          if draw:
95              cv2.line(img, (x1, y1), (x2, y2), (255, 0, 255), t)
96              cv2.circle(img, (x1, y1), r, (255, 0, 255), cv2.FILLED)
97              cv2.circle(img, (x2, y2), r, (255, 0, 255), cv2.FILLED)
98              cv2.circle(img, (cx, cy), r, (0, 0, 255), cv2.FILLED)
99          length = math.hypot(x2 - x1, y2 - y1)
100
101         return length, img, &#91;x1, y1, x2, y2, cx, cy]
```

```
102
103
104 def main():
105     pTime = 0
106     cTime = 0
107     cap = cv2.VideoCapture(1)
108     detector = handDetector()
109     while True:
110         success, img = cap.read()
111         img = detector.findHands(img)
112         lmList, bbox = detector.findPosition(img)
113         if len(lmList) != 0:
114             print(lmList&#91;4])
115
116         cTime = time.time()
117         fps = 1 / (cTime - pTime)
118         pTime = cTime
119
120         cv2.putText(img, str(int(fps)), (10, 70), cv2.FONT_HERSHEY_PLAIN, 3,
121                     (255, 0, 255), 3)
122
123         cv2.imshow("Image", img)
124         cv2.waitKey(1)
125
126
127 if __name__ == "__main__":
128     main()
```

Mark Complete ✔

Back to Course (https://www.computervision.zone/courses/ai-virtual-mouse/)

Previous Lesson (https://www.computervision.zone/lessons/ai-virtual-mouse-video-lesson/)