Computer Vision Final Project

Name: Malak Mahmoud Medhat Aref

ID: 20221445867

Department: Intelligent Systems (Level Two)

Ai action recognition and pose estimation with Mediapipe:

The main idea of the project is to recognize human activities from video sequences or live camera, and we will try both on our project.

- First, we will import the needed libraries. And the most important library is Mediapipe → to give us all of our pose estimation libraries and all our different mediapipe solutions. It is a cross-platform library developed by Google that provides amazing ready-to-use ML solutions for computer vision tasks.

- Then we created two variables
    1. mp_drawing → give us all of our drwaing utilities so when we actually visualize our poses we will use this drawing utilities.
    2. mp_pose → import our pose estimation model.

- Then setup the video capture device.

- Setting up a new instance of our mediapipe feed and we use the with statement.

- Recolor our image feed from our camera from BGR to RGB and we set a writable status to false.

- Make detection using pose.process.

- Set our writable status back to true so we can draw on it and then we converte it back from RGB to BGR so that it works with opencv.

- Draw our detection by mp_drawing.draw_landmarks it takes several parameters, as results.pose_landmarks and mp_pose.POSE_CONNECTIONS
  Output of result .landmarks → each one of this represents a different coordinates or a different joint within our pose estimation model.

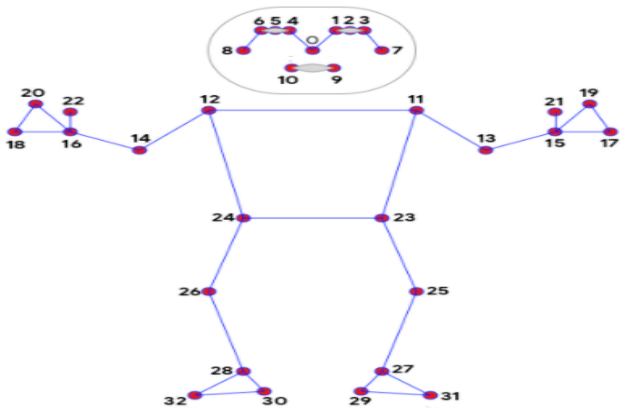
```
landmark {
  x: 0.36683154106140137
  y: 0.13542136549949646
  z: -0.21010589599609375
  visibility: 0.999544084072113
}
landmark {
  x: 0.3665555715560913
  y: 0.12118399888277054
  z: -0.19909575581550598
  visibility: 0.9989718794822693
}
landmark {
  x: 0.3666929304599762
  y: 0.1211947649717331
  z: -0.1990395486354828
  visibility: 0.9988675117492676
}
```

Output of mp_pose.POSE_CONNECTIONS → show which landmarks are connected to eachother.

```
frozenset({(<PoseLandmark.NOSE: 0>, <PoseLandmark.LEFT_EYE_INNER: 1>),
           (<PoseLandmark.NOSE: 0>, <PoseLandmark.RIGHT_EYE_INNER: 4>),
           (<PoseLandmark.LEFT_EYE_INNER: 1>, <PoseLandmark.LEFT_EYE: 2>),
           (<PoseLandmark.LEFT_EYE: 2>, <PoseLandmark.LEFT_EYE_OUTER: 3>),
           (<PoseLandmark.LEFT_EYE_OUTER: 3>, <PoseLandmark.LEFT_EAR: 7>),
           (<PoseLandmark.RIGHT_EYE_INNER: 4>, <PoseLandmark.RIGHT_EYE: 5>),
           (<PoseLandmark.RIGHT_EYE: 5>, <PoseLandmark.RIGHT_EYE_OUTER: 6>),
           (<PoseLandmark.RIGHT_EYE_OUTER: 6>, <PoseLandmark.RIGHT_EAR: 8>),
           (<PoseLandmark.MOUTH_RIGHT: 10>, <PoseLandmark.MOUTH_LEFT: 9>),
           (<PoseLandmark.LEFT_SHOULDER: 11>, <PoseLandmark.LEFT_ELBOW: 13>),
           (<PoseLandmark.LEFT_SHOULDER: 11>, <PoseLandmark.LEFT_HIP: 23>),
```

- Moreover, the nice thing about mediapipe is that it's got awesome visualization that allows you to easily draw these results, so no need to draw point by point we can use those drawing utilities.

- Joints



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

- The program output on a video

Reference of helper resources → https://youtu.be/06TE_U21FK4

Ps: the code file doesn't contain that video because it's very large to download so the program will work on the live camera.