DIP Project

Project description:

Find the best match in a video clip. Given a contour of a character and video clip, we need to find the frame that include the frame which include the character with the closed pose to that of the contour.

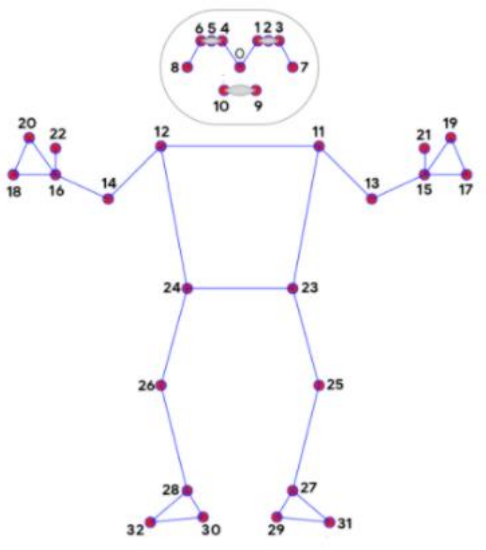
Technologies Used:

The project was written in Python 3.9 and we used the libraries below:

- CV2 - OpenCV is an open-source library which is very useful for computer vision applications.
    - **cv.matchShapes** Which enables us to compare two shapes, or two contours and returns a metric showing the similarity. The lower the result, the better match it is.
    - **cv.drawContours** It can be used to draw any shape provided you have its boundary points.
    - **cv.findContours** Function that helps in extracting the contours from the image.
- NumPy - NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices.
- Mediapipe - Mediapipe is Google's open-source framework, used for media processing. Mediapipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks and background segmentation mask on the whole body from RGB video frames utilizing our BlazePose research that also powers the ML Kit Pose Detection API.

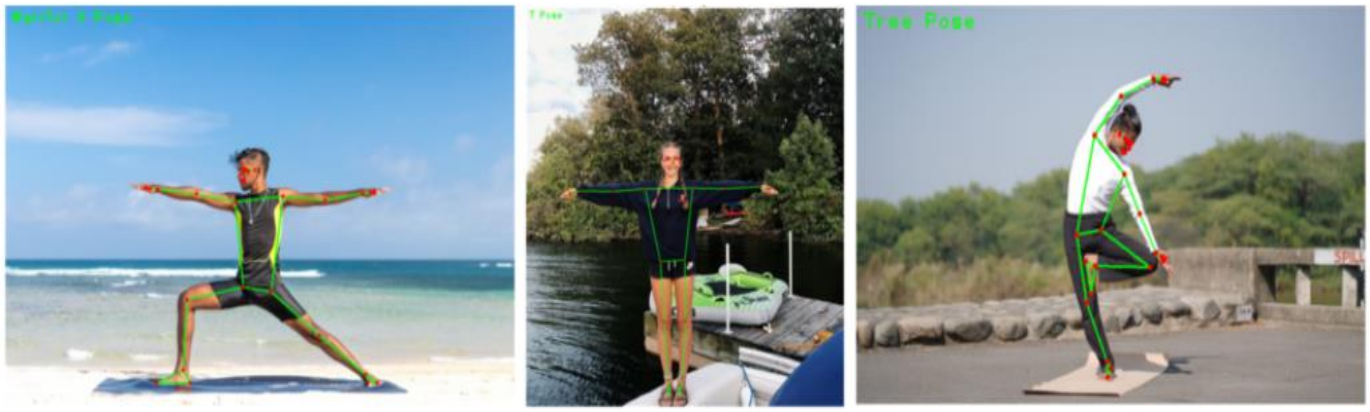
Pose Landmark Model (BlazePose GHUM 3D)

The landmark model in MediaPipe Pose predicts the location of 33 pose landmarks (see figure below).



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

Fig 4. 33 pose landmarks.



Algorithm input:

- Contour of a character - NumPy array of (x, y) coordinates of boundary points of the object.
- Video clip.

To run the code: python Project.py character_contour "Path_To_Vedio_Clip"

Algorithm Description:

The algorithm will find the frame that includes the character with the closed pose to that of the contour.

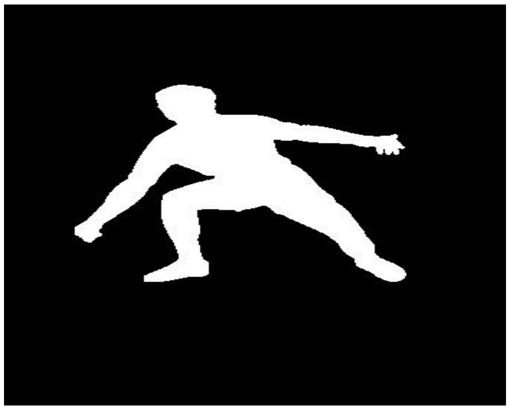
1. Read contour from input.
2. Initialize mediapipe essentials.
3. Read video from input.
4. Use mediapipe function to get poses from video frames we store the manipulated frames in a list, and return the list.
5. On the list of the manipulated frames, for each frame we extract the largest contour in the frame and store it in a list.
6. We use OpenCV matchShapes function to compare the character contour we got in the input with every contour in the list, the results are stored in the list.
7. We return the first frame with the best result(match) that we found.

Edge cases:

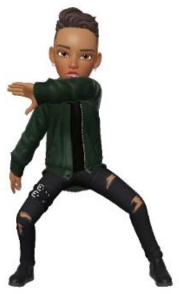
- Mediapipe is ML solution, therefore there are situation when the code will not behave as expected because there is a margin of error.
- After we used mediapipe to detect a pose, we then try to find the contour of that pose. If the pose is “complicated” the contour returned will not have enough information in it.
- MatchShapes function in OpenCV return its approximation, but there could be frames with the same result but return only one frame, with no way of telling which frame is better.

EXAMPLES:

1)

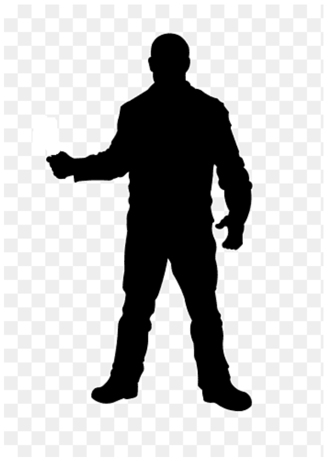


(Contour 1)

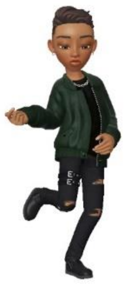


(The frame returned 1)

2)



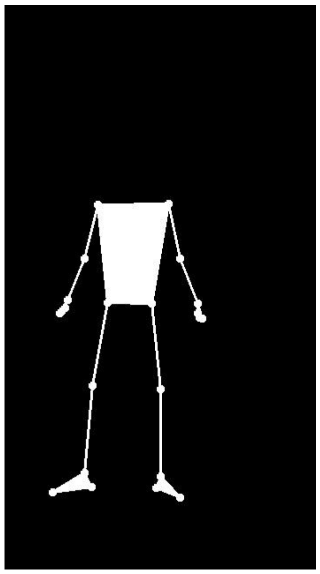
(Contour 2)



(The frame returned 2)

(Video Frame)

→



(Contour after we used mediapipe and find contour Step 4+5 from the algorithm)