realtime-human-pose-estimation

January 31, 2024

0.1 Install Modules

[ ]:
```
!pip install opencv-python
!pip install mediapipe
```

0.2 Import Modules

[1]:
```
import cv2
import mediapipe as mp
```

[2]:
```
## initialize pose estimator
mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
pose = mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5)
```

0.3 Pose Estimation for Video

[20]:
```
cap = cv2.VideoCapture('test_video.mp4')
while cap.isOpened():
    # read frame
    _, frame = cap.read()
    try:
        # resize the frame for portrait video
        frame = cv2.resize(frame, (350, 600))
        # convert to RGB
        frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # process the frame for pose detection
        pose_results = pose.process(frame_rgb)
        # print(pose_results.pose_landmarks)

        # draw skeleton on the frame
        mp_drawing.draw_landmarks(frame, pose_results.pose_landmarks, mp_pose.
 ↪POSE_CONNECTIONS)
        # display the frame
        cv2.imshow('Output', frame)
    except:
```

```
        break

    if cv2.waitKey(1) == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

[19]:
```
# get landmark for a specific point
pose_results.pose_landmarks.landmark[32]
```

[19]:
```
x: 0.35414522886276245
y: 0.8367241024971008
z: 0.16406674683094025
visibility: 0.9696751236915588
```

0.4 Realtime Pose Estimation

[23]:
```
cap = cv2.VideoCapture(0)
while cap.isOpened():
    # read frame
    _, frame = cap.read()
    try:
        # resize the frame for portrait video
        # frame = cv2.resize(frame, (350, 600))
        # convert to RGB
        frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # process the frame for pose detection
        pose_results = pose.process(frame_rgb)
        # print(pose_results.pose_landmarks)

        # draw skeleton on the frame
        mp_drawing.draw_landmarks(frame, pose_results.pose_landmarks, mp_pose.
 ↪POSE_CONNECTIONS)
        # display the frame
        cv2.imshow('Output', frame)
    except:
        break

    if cv2.waitKey(1) == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

[ ]: