Hello, we are here with our new project in our image processing journey, the GYM project.

**IMPORT LIBRARIES**

1. import cv2
2. import mediapipe as mp
3. import numpy as np

We added the OpenCv library for image processing, the mediapipe library to create body lines, and finally the numpy library because the data we will get from the mediapipe application is in matrix form.

**VİDEO FEED**

1. cap = cv2.VideoCapture(0)
2. while cap.isOpened():
3. ret, frame = cap.read()
4. cv2.imshow('Mediapipe Feed', frame)
6. if cv2.waitKey(10) & 0xFF == ord('q'):
7. break
9. cap.release()
10. cv2.destroyAllWindows()

We have opened our camera with the cv2.VideoCapture(0) command. The while cap.isOpened() command, that is, the commands to print the camera to the screen when the camera is open, and the command to exit the camera when the "q" key is pressed, were added later..

**DETERMINING THE POSE POINTS OF JOINT POINTS**

1. cap = cv2.VideoCapture(0)
2. ## Mediapipe pose noktalarını oluşturmak
3. with mp\_pose.Pose(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as pose:
4. while cap.isOpened():
5. ret, frame = cap.read()
7. # BGR’DEN RGB’YE RENK DÖNÜŞÜMÜ
8. image = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)
9. image.flags.writeable = False
11. # İNSAN VUCUDÜNDAKİ POSE YANİ NOKTALARI OLUŞTURARAK BİRLEŞTİRME
12. results = pose.process(image)
14. # RGB’DEN TEKRAR BGR’YE YANİ NORMAL GÖRÜNTÜYE DÖNÜŞTÜRME
15. image.flags.writeable = True
16. image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)
18. # Oluşturduğumuz noktaları render işlemine tabi tutmak
19. mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS,
20. mp\_drawing.DrawingSpec(color=(245,117,66), thickness=2, circle\_radius=2),
21. mp\_drawing.DrawingSpec(color=(245,66,230), thickness=2, circle\_radius=2)
22. )
24. cv2.imshow('Mediapipe Feed', image)
26. if cv2.waitKey(10) & 0xFF == ord('q'):
27. break
29. cap.release()
30. cv2.destroyAllWindows()

In the above processes, color conversion from BGR to RGB was made from the OpenCv library. Also, creating and combining Mediapipe pose points were performed..

**DETERMINATION OF JOINT POINTS**

1. cap = cv2.VideoCapture(0)
2. ## Mediapipe Pose noktaların oluşturmak
3. with mp\_pose.Pose(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as pose:
4. while cap.isOpened():
5. ret, frame = cap.read()
7. # RGB’ye dönüştürme
8. image = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)
9. image.flags.writeable = False
11. # Bulunan noktaları image yani kamera üzerinde birleştirme işlemi yaptıgımız nokta
12. results = pose.process(image)
14. # BGR’ye tekrar dönüştürme
15. image.flags.writeable = True
16. image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)
18. # EKLEM NOKTALARININ ÇIKARTILMASI
19. try:
20. landmarks = results.pose\_landmarks.landmark
21. print(landmarks)
22. except:
23. pass

26. # TESPİT EDİLEN NOKTALARIN RENDER İŞLEMİ
27. mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS,
28. mp\_drawing.DrawingSpec(color=(245,117,66), thickness=2, circle\_radius=2),
29. mp\_drawing.DrawingSpec(color=(245,66,230), thickness=2, circle\_radius=2)
30. )
32. cv2.imshow('Mediapipe Feed', image)
34. if cv2.waitKey(10) & 0xFF == ord('q'):
35. break
37. cap.release()
38. cv2.destroyAllWindows()



**Fıgure 1**

1. for lndmrk in mp\_pose.PoseLandmark:
2. print(lndmrk)

It is used to display the pose points of human body lines with the command line shown above. The correspondences of each pose point are indicated in figure 1.

1. landmarks[mp\_pose.PoseLandmark.LEFT\_SHOULDER.value].visibility

For example, when we run our code pose=LEF\_SHOULDER for our left shoulder, the mathematical equivalent of the left shoulder point is as in figure 2..



**Fıgure 2**

1. landmarks[mp\_pose.PoseLandmark.LEFT\_ELBOW.value]
2. landmarks[mp\_pose.PoseLandmark.LEFT\_WRIST.value]

The above command line shows the mathematical positions of the retrieved landmarks pose points.

**CALCULATION OF JOINT POINTS**

1. def calculate\_angle(a,b,c):
2. a = np.array(a) # First
3. b = np.array(b) # Mid
4. c = np.array(c) # End
5. radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-b[0])
6. angle = np.abs(radians\*180.0/np.pi)
8. if angle >180.0:
9. angle = 360-angle
11. return angle

Thanks to the module shown above, the joint points were converted to decimal numbers matrixically and the points were combined.

1. cap = cv2.VideoCapture(0)
3. # Hesaplama işlemi yapılması için integer tipinde değişkenler oluşturuldu
4. counter = 0
5. stage = None
7. ## Mediapipe kurulumu gerçekleştirildi
8. with mp\_pose.Pose(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as pose:
9. while cap.isOpened():
10. ret, frame = cap.read()
12. # RGB’ye renk dönüşümü yapıldı
13. image = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)
14. image.flags.writeable = False
16. # Pose noktalarının tespiti yapıldı
17. results = pose.process(image)
19. # BGR renk skalasına tekrardan dönüştürüldü
20. image.flags.writeable = True
21. image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)
23. # Landmarks pose noktaları çıkartıldı
24. try:
25. landmarks = results.pose\_landmarks.landmark
27. # Kordinatlar oluşturuldu
28. shoulder = [landmarks[mp\_pose.PoseLandmark.LEFT\_SHOULDER.value].x,landmarks[mp\_pose.PoseLandmark.LEFT\_SHOULDER.value].y]
29. elbow = [landmarks[mp\_pose.PoseLandmark.LEFT\_ELBOW.value].x,landmarks[mp\_pose.PoseLandmark.LEFT\_ELBOW.value].y]
30. wrist = [landmarks[mp\_pose.PoseLandmark.LEFT\_WRIST.value].x,landmarks[mp\_pose.PoseLandmark.LEFT\_WRIST.value].y]
32. # Hesaplama işlemi yapıldı
33. angle = calculate\_angle(shoulder, elbow, wrist)
35. # Açılar görselleştirildi
36. cv2.putText(image, str(angle),
37. tuple(np.multiply(elbow, [640, 480]).astype(int)),
38. cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE\_AA

                                )

1. # Açılardan up ve down hesaplaması için gerekli satır
2. if angle > 160:
3. stage = "down"
4. if angle < 30 and stage =='down':
5. stage="up"
6. counter +=1

9. except:
10. pass
11. # Up ve down sayaç için komutlar
12. cv2.rectangle(image, (0,0), (225,73), (245,117,16), -1)
14. # datayı alıcağımız komutlar
15. cv2.putText(image, 'SAYAC', (15,12),
16. cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE\_AA)
17. cv2.putText(image, str(counter),
18. (10,60),
19. cv2.FONT\_HERSHEY\_SIMPLEX, 2, (255,255,255), 2, cv2.LINE\_AA)
21. # Aşama verisi
22. cv2.putText(image, 'STAGE', (65,12),
23. cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE\_AA)
24. cv2.putText(image, stage,
25. (60,60),
26. cv2.FONT\_HERSHEY\_SIMPLEX, 2, (255,255,255), 2, cv2.LINE\_AA)

29. # landmark pose kavrama noktaları
30. mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS,
31. mp\_drawing.DrawingSpec(color=(245,117,66), thickness=2, circle\_radius=2),
32. mp\_drawing.DrawingSpec(color=(245,66,230), thickness=2, circle\_radius=2)
33. )
35. cv2.imshow('Mediapipe Feed', image)
37. if cv2.waitKey(10) & 0xFF == ord('q'):
38. break
40. cap.release()
41. cv2.destroyAllWindows()