Practical-7

Student Name: Gauri Prabhakar
UID: 18BCS6201
Branch: 18AITAIML-2
Section/Group: B
Semester: 7
Date of Performance: 12th October, 2021
Subject Name: Computer Vision Lab
Subject Code: CSF - 432

1. Aim/Overview of the practical:

To implement face detection using HAAR cascades using mediapipe in python and OpenCV.

2. Task to be done:

To implement face detection using HAAR cascades using mediapipe in python and OpenCV.

3. Steps to be followed:

1. Importing necessary modules.

2. Importing 'mp.solutions.face_detection' to a variable 'mp_face_detection'.

3. We will use 'drawing_utils' to draw the key points.

4. Importing the cascade file into the variable 'face_cascade'.

5. Importing the image into the variable 'img'.

6. Reading the image and converting it to RGB and storing it in the variable 'gray'.

7. Now we find the faces in the image.

8. If faces are detected, we return the positions of detected faces as Rect(x,y,w,h).

9. Once we get these locations, we can create a ROI(Rectangular region of interest) for the face

10. and apply eye detection on this ROI (since eyes are always on the face !!!).

11. Returning the co-ordinates of the ROI.

12. Rendering the video with effective Facetracking to the console by using the function '.imshow()'.

13. Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.

14. Destroying all windows.

15. Creating a variable to store the video using the '.VideoCapture()' function.

16. Specifying the detection and tracking confidence uisng 'mp_pose.pose()'.

17. While the video is running.

18. Capturing the video frame by frame using the '.read()' method.

19. To improve performance, marking the image as not writable and passing by reference.

20. Reading the frames and converting them to RGB.

21. Detecting faces in the frame using the function 'face_detection.process()'.

22. Drawing the pose annotations on the image.

23. Reading the frames and converting them to RGB.

24. If a face is detected we draw a rectangle on the ROI.

25. Rendering the video with effective Posetracking to the console by using the function '.imshow()'.

26. Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.

27. '.waitkey(1)' displays a frame for 1ms after which it moves to the next frame in the video.

28. Setting 'x' as the quitting button.

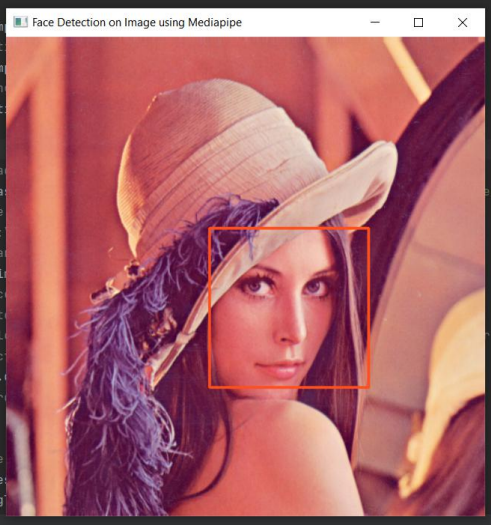
29. Releasing the variable/object 'cap'.

4. Result/Output/Writing Summary:

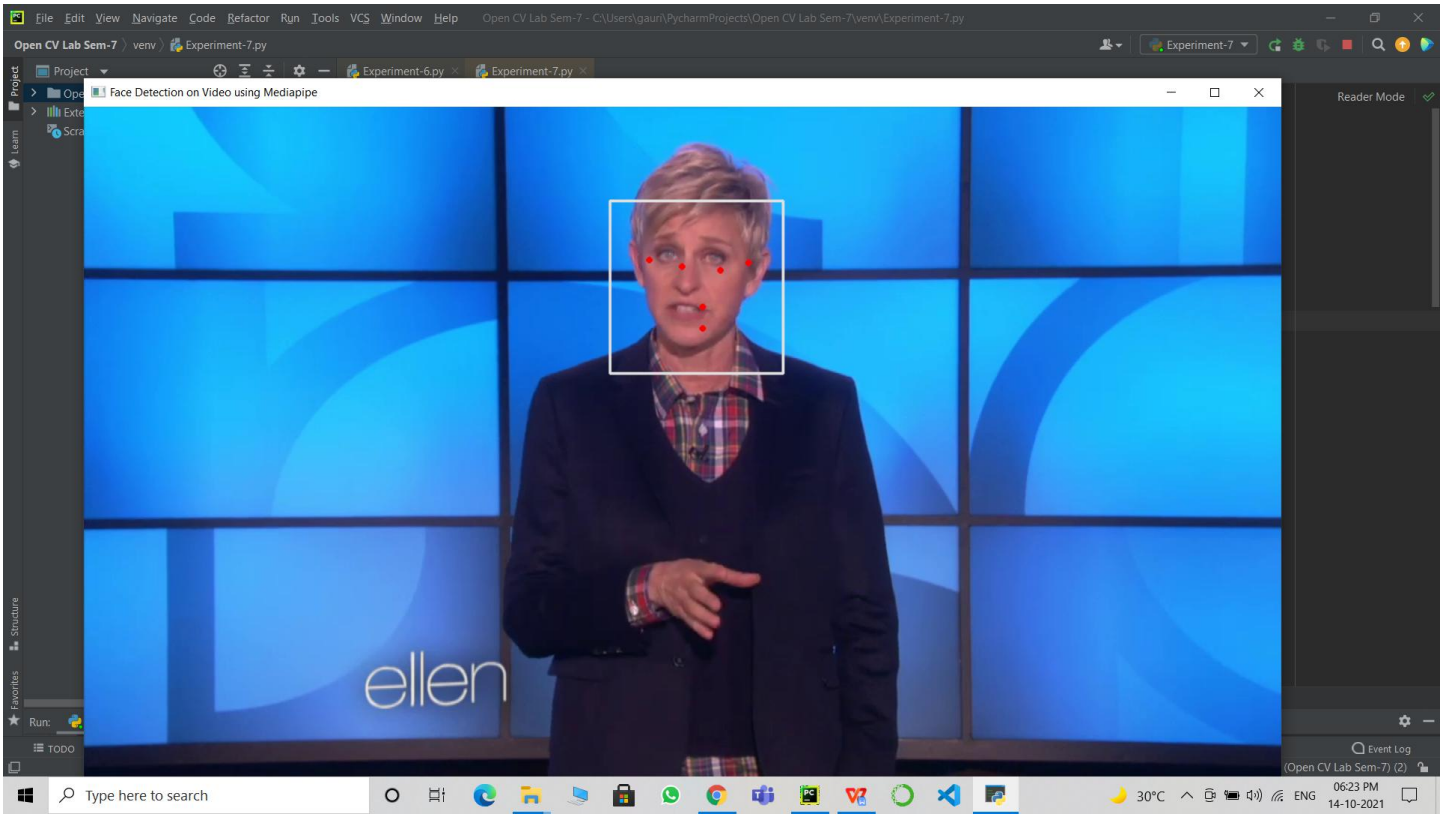
```python
# 18BCS6201-CV Practical-7 (Gauri Prabhakar) (AI-ML-2)(B)
# Aim: To implement face detction using HAAR cascades  using mediapipe in python and OpenCV.

# Importing necessary modules.
import cv2
import mediapipe as mp
# Importing 'mp.solutions.face_detection' to a variable 'mp_face_detection'.
mp_face_detection = mp.solutions.face_detection
# We will use 'drawing_utils' to draw the key points.
mp_drawing = mp.solutions.drawing_utils


# Importing the cascade file into the variable 'face_cascade'
face_cascade = cv2.CascadeClassifier(r"C:\Users\gauri\Desktop\OpenCV Media\haarcascade_frontalface_default.xml")
# Importing the image into the variable 'img'.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\lena.png")
# Reading the image and converting it to RGB and storing it in the variable 'gray'.
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Now we find the faces in the image.
# If faces are detected, we return the positions of detected faces as Rect(x,y,w,h).
# Once we get these locations, we can create a ROI(Rectangular region of interest) for the face
# and apply eye detection on this ROI (since eyes are always on the face !!! ).
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
# Returning the co-ordinates of the ROI.
print(faces)

# Drawing a rectangle on the ROI.
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(31,79,254),2)
```

```python
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(31,79,254),2)

# Rendering the video with effective Facetracking to the console by using the function '.imshow()'.
cv2.imshow('Face Detection on Image using Mediapipe',img)

# Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.
cv2.waitKey(0)
# Destroying all windows.
cv2.destroyAllWindows()


# Creating a variable to store the video using the '.VideoCapture()' function.
cap = cv2.VideoCapture(r"C:\Users\gauri\Desktop\OpenCV Media\To Plan or Not to Plan_.mp4")

# Specifying the detection and tracking confidence uisng 'mp_pose.pose()'.
with mp_face_detection.FaceDetection(
    model_selection=0, min_detection_confidence=0.5) as face_detection:
  # While the video is running.
  while cap.isOpened():
    # Capturing the video frame by frame using the '.read()' method.
    success, image = cap.read()

    # To improve performance, marking the image as not writable and passing by reference.
    image.flags.writeable = False
    # Reading the frames and converting them to RGB.
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    # Detecting faces in the frame using the function 'face_detection.process()'.
    results = face_detection.process(image)
```

```python
    # Detecting faces in the frame using the function 'face_detection.process()'.
    results = face_detection.process(image)

    # Drawing the pose annotations on the image.
    image.flags.writeable = True

    # Reading the frames and converting them to RGB.
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

    # If a face is detected we draw a rectangle on the ROI.
    if results.detections:
      for detection in results.detections:
        mp_drawing.draw_detection(image, detection)

    # Rendering the video with effective Facetracking to the console by using the function '.imshow()'.
    cv2.imshow('Face Detection on Video using Mediapipe',image)

    # Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.
    # '.waitkey(1)' displays a frame for 1ms after which it moves to the next frame in the video.
    # Setting 'x' as the quitting button.
    if cv2.waitKey(5) & 0xFF == ord('x'):
      break

# Releasing the variable/object 'cap'.
cap.release()
```




```python
# 18BCS6201-CV Practical-7 (Gauri Prabhakar) (AI-ML-2)(B)
# Aim: To implement face detction using HAAR cascades  using mediapipe in python and OpenCV.

# Importing necessary modules.
import cv2
import mediapipe as m
# Importing 'mp.solut
mp_face_detection = m
# We will use 'drawin
mp_drawing = mp.solut

# Importing the casca
face_cascade = cv2.Ca                                  _frontalface_default.xml")
# Importing the image
img = cv2.imread(r"C:
# Reading the image a
gray = cv2.cvtColor(i
# Now we find the fac
# If faces are detect
# Once we get these l                                  the face
# and apply eye detec
faces = face_cascade.
# Returning the co-or
print(faces)

# Drawing a rectangle
for (x,y,w,h) in face
    img = cv2.rectang
```



5. Learning outcomes (What I have learnt):

- Open CV modules.
- The mediapipe library.
- Detect faces using the mediapipe library and HAAR cascades.
- Face tracking a saved video.
- Face tracking a saved image.
- Highlighting key points.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			