Face Detection using Haar Cascades with OpenCV and Matplotlib

Aim

To write a Python program using OpenCV to perform the following image manipulations:
i) Extract ROI from an image.
ii) Perform face detection using Haar Cascades in static images.
iii) Perform eye detection in images.
iv) Perform face detection with label in real-time video from webcam.

Software Required

- Anaconda - Python 3.7 or above
- OpenCV library (`opencv-python`)
- Matplotlib library (`matplotlib`)
- Jupyter Notebook or any Python IDE (e.g., VS Code, PyCharm)

Algorithm

I) Load and Display Images

- Step 1: Import necessary packages: `numpy` , `cv2` , `matplotlib.pyplot`
- Step 2: Load grayscale images using `cv2.imread()` with flag `0`
- Step 3: Display images using `plt.imshow()` with `cmap='gray'`

II) Load Haar Cascade Classifiers

- Step 1: Load face and eye cascade XML files

III) Perform Face Detection in Images

- Step 1: Define a function `detect_face()` that copies the input image
- Step 2: Use `face_cascade.detectMultiScale()` to detect faces
- Step 3: Draw white rectangles around detected faces with thickness 10
- Step 4: Return the processed image with rectangles

IV) Perform Eye Detection in Images

- Step 1: Define a function `detect_eyes()` that copies the input image
- Step 2: Use `eye_cascade.detectMultiScale()` to detect eyes
- Step 3: Draw white rectangles around detected eyes with thickness 10

- Step 4: Return the processed image with rectangles

V) Display Detection Results on Images

- Step 1: Call `detect_face()` or `detect_eyes()` on loaded images
- Step 2: Use `plt.imshow()` with `cmap='gray'` to display images with detected regions highlighted

VI) Perform Face Detection on Real-Time Webcam Video

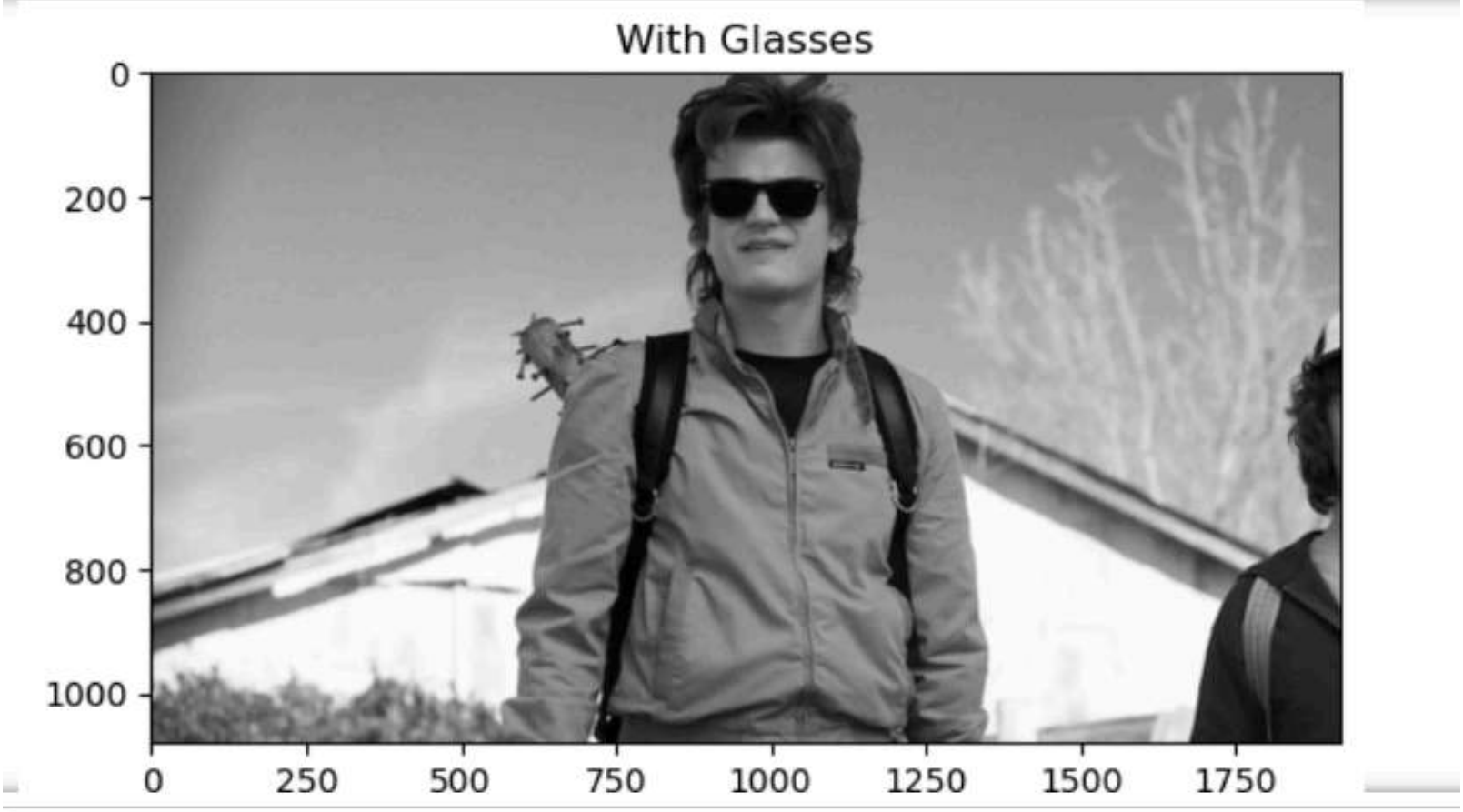
- Step 1: Capture video from webcam using `cv2.VideoCapture(0)`
- Step 2: Loop to continuously read frames from webcam
- Step 3: Apply `detect_face()` function on each frame
- Step 4: Display the video frame with rectangles around detected faces
- Step 5: Exit loop and close windows when ESC key (key code 27) is pressed
- Step 6: Release video capture and destroy all OpenCV windows

Program:

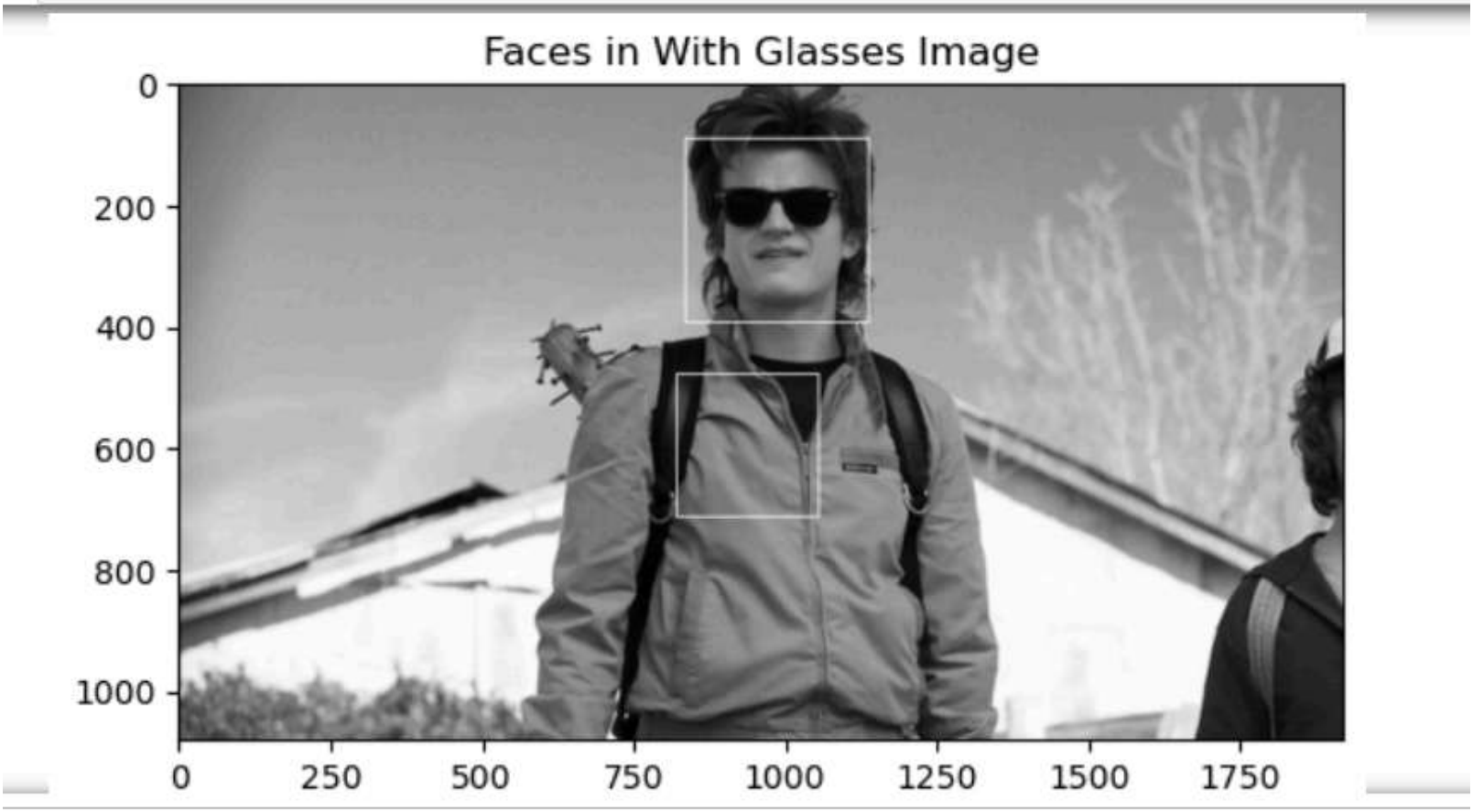
```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

withglass = cv2.imread("C:\\Users\\admin\\Downloads\\steveglass.webp", 0)
group = cv2.imread("C:\\Users\\admin\\Downloads\\group.webp", 0)

plt.imshow(withglass, cmap='gray')
plt.title("With Glasses")
plt.show()

plt.imshow(group, cmap='gray')
plt.title("Group Image")
plt.show()

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')

if face_cascade.empty():
    raise IOError("Error loading face cascade XML file")
if eye_cascade.empty():
    raise IOError("Error loading eye cascade XML file")

def detect_face(img, scaleFactor=1.1, minNeighbors=5):
    face_img = img.copy()
    face_rects = face_cascade.detectMultiScale(face_img, scaleFactor=scaleFactor,
minNeighbors=minNeighbors)
    for (x, y, w, h) in face_rects:
        cv2.rectangle(face_img, (x, y), (x + w, y + h), (255, 255, 255), 2)
    return face_img
```

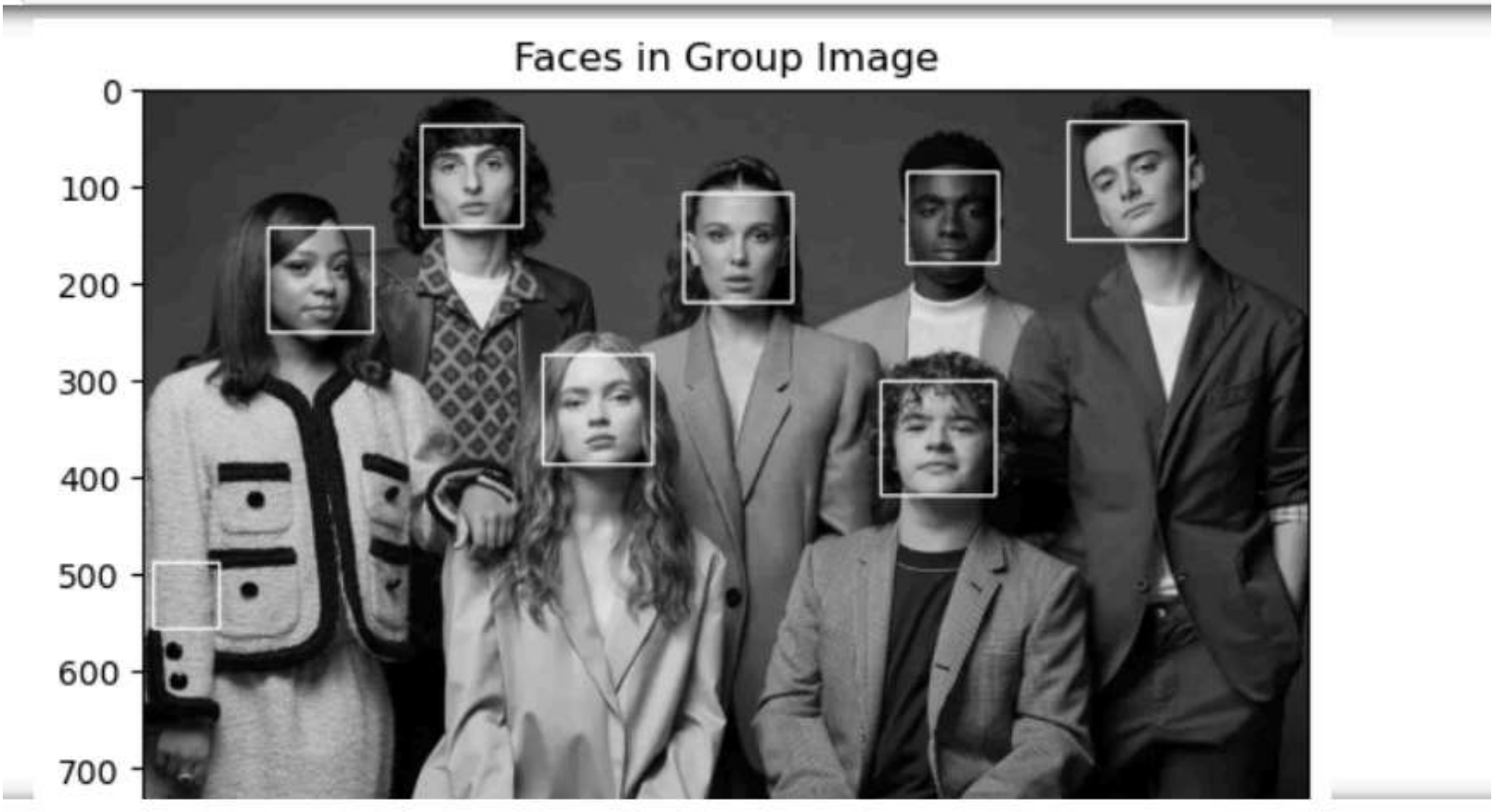
```
def detect_eyes(img):
    face_img = img.copy()
    eyes = eye_cascade.detectMultiScale(face_img)
    for (x, y, w, h) in eyes:
        cv2.rectangle(face_img, (x, y), (x + w, y + h), (255, 255, 255), 2)
    return face_img

result_withglass_faces = detect_face(withglass)
plt.imshow(result_withglass_faces, cmap='gray')
plt.title("Faces in With Glasses Image")
plt.show()

result_group_faces = detect_face(group)
plt.imshow(result_group_faces, cmap='gray')
plt.title("Faces in Group Image")
plt.show()

result_withglass_eyes = detect_eyes(withglass)
plt.imshow(result_withglass_eyes, cmap='gray')
plt.title("Eyes in With Glasses Image")
plt.show()

result_group_eyes = detect_eyes(group)
plt.imshow(result_group_eyes, cmap='gray')
plt.title("Eyes in Group Image")
plt.show()
```

Output:


With Glasses
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750


Group Image
0
100
200
300
400
500
600
700





Results:

Face Detection using Haar Cascades with OpenCV and Matplotlib executed sucessfully