Nguyễn Hoàng Thanh – 2151013087

BÀI NỘP: LAB 3 – COMPUTER VISION

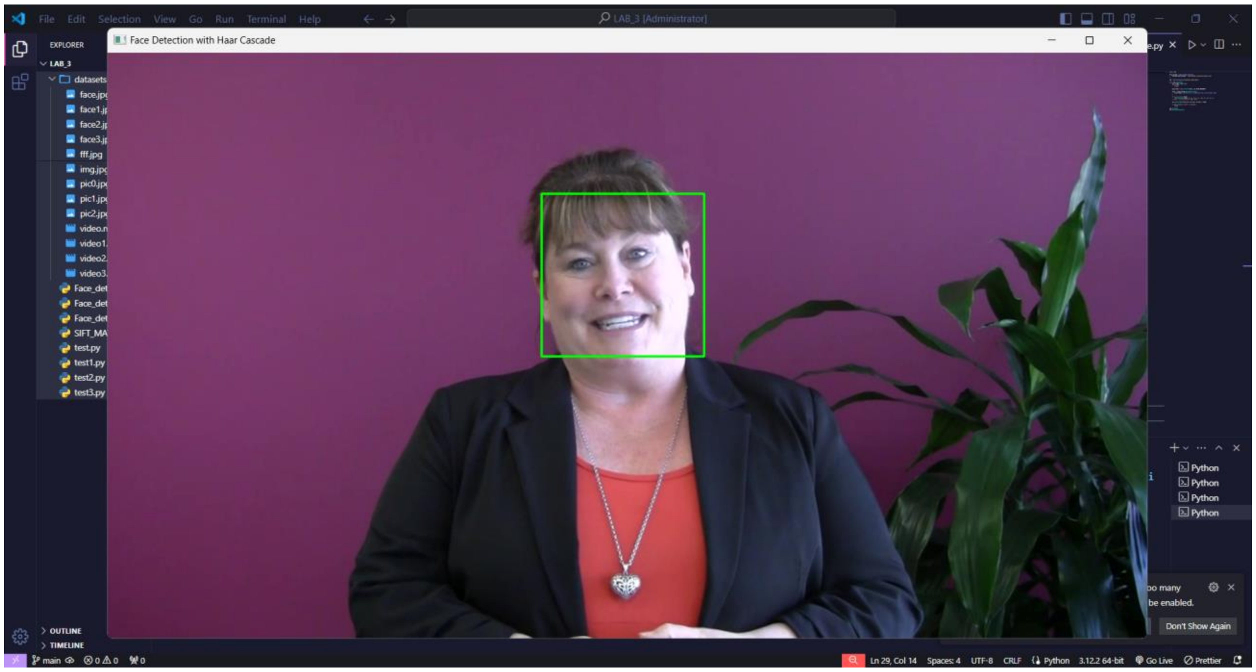
TASK: FACE DETECTION

1. Face detection with Haar-like

- Source code:

```
import cv2

face_cascade = cv2.CascadeClassifier(
    cv2.data.haarcascades +
"haarcascade_frontalface_default.xml"
)

cap = cv2.VideoCapture("datasets/video2.mp4")

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(
        gray_frame, scaleFactor=1.1, minNeighbors=20,
minSize=(100, 100)
    )

    for x, y, w, h in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0,
255, 0), 2)
    # frame = cv2.resize(frame, (600, 400))

    cv2.imshow("Face Detection with Haar Cascade", frame)

    if cv2.waitKey(1) & 0xFF == ord("q"):
        break

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

- Result:



2. Face detection with SIFT:
- Source code:

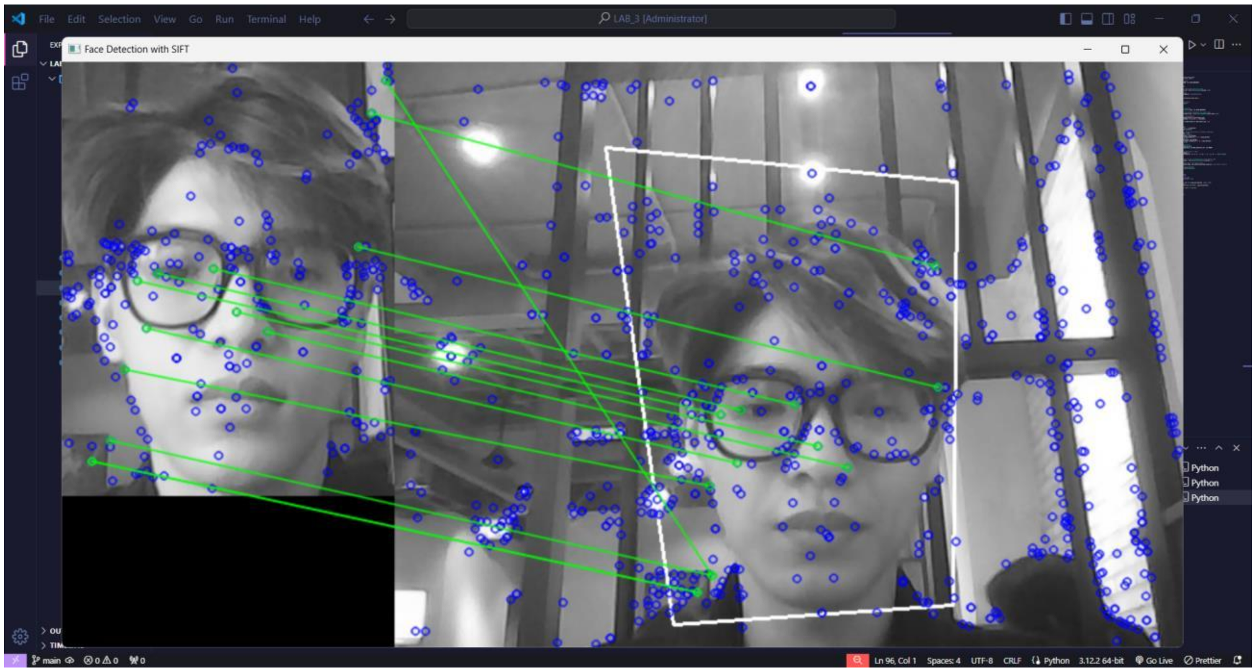
```
import cv2
import numpy as np

img0 = cv2.imread("datasets/face3.jpg")
img0 = cv2.resize(img0, (270, 356))

gray0 = cv2.cvtColor(img0, cv2.COLOR_BGR2GRAY)

sift = cv2.SIFT_create()
```

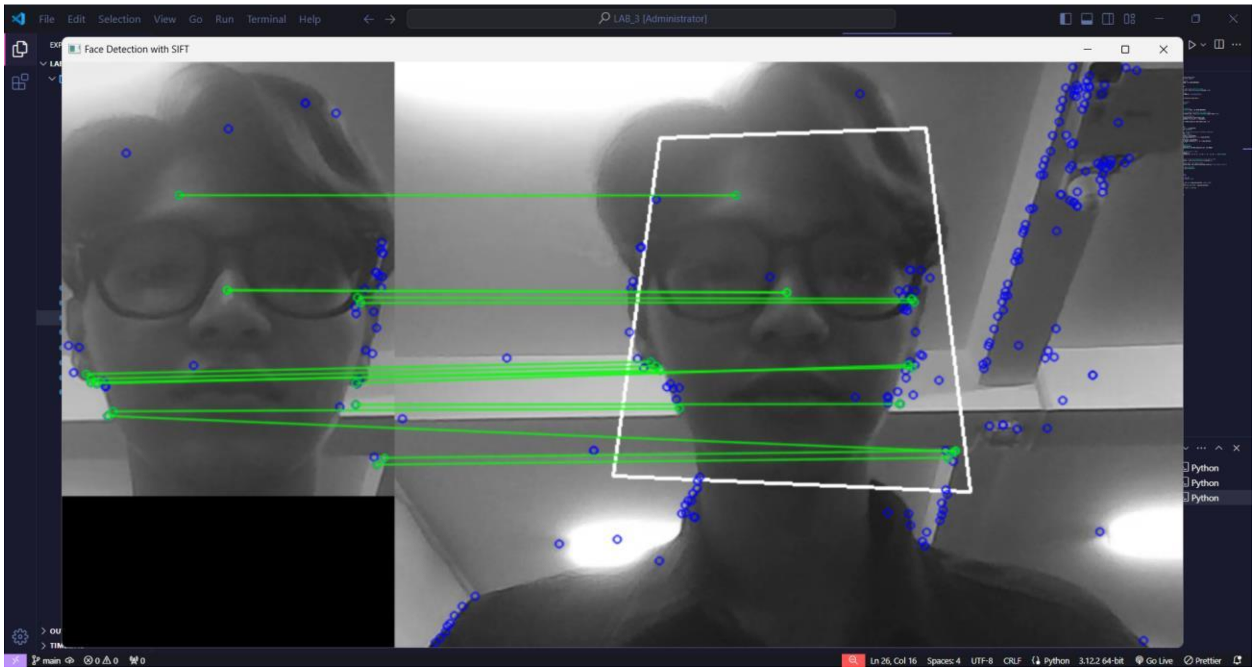
```
keypoints_0, descriptors_0 = sift.detectAndCompute(gray0,
None)

bf = cv2.BFMatcher(cv2.NORM_L2, crossCheck=False)

cap = cv2.VideoCapture("datasets/video3.mp4")
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    keypoints_cam, descriptors_cam =
sift.detectAndCompute(gray_frame, None)

    # Match descriptors using knnMatch
    matches = bf.match(descriptors_0, descriptors_cam)
    matches = sorted(matches, key=lambda x: x.distance)

    matches = bf.knnMatch(descriptors_0, descriptors_cam,
k=2)

    good_matches = []
    for m, n in matches:
        if m.distance < 0.7 * n.distance:
            good_matches.append(m)

    if len(good_matches) >= 10:
        matched_keypoints_0 = np.float32(
            [keypoints_0[m.queryIdx].pt for m in
good_matches]
        ).reshape(-1, 1, 2)
        matched_keypoints_cam = np.float32(
            [keypoints_cam[m.trainIdx].pt for m in
good_matches]
```

```
        ).reshape(-1, 1, 2)

        H, _ = cv2.findHomography(
            matched_keypoints_0, matched_keypoints_cam,
cv2.RANSAC
        )

        h, w = gray0.shape[:2]
        corners = np.float32([[0, 0], [0, h - 1], [w - 1,
h - 1], [w - 1, 0]]).reshape(
            -1, 1, 2
        )

        transformed_corners =
cv2.perspectiveTransform(corners, H)

        cv2.polylines(gray_frame,
[np.int32(transformed_corners)], True, (255, 0, 0), 2)

    img_with_matches = cv2.drawMatches(
        gray0,
        keypoints_0,
        gray_frame,
        keypoints_cam,
        good_matches[:50],
        None,
        matchColor=(0, 255, 0),
        singlePointColor=(255, 0, 0),
        flags=0,
    )
    img_with_matches = cv2.resize(img_with_matches,
(1380, 720))

    cv2.imshow("Face Detection with SIFT",
img_with_matches)
```

```
    if cv2.waitKey(20) & 0xFF == ord("q"):
        break

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

- Result:


File Edit Selection View Go Run Terminal Help
LAB_3 [Administrator]
Face Detection with SIFT
Python
Python
Python
main
Ln 96, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.2 64-bit Go Live Prettier


File Edit Selection View Go Run Terminal Help
LAB_3 [Administrator]
Face Detection with SIFT
Python
Python
Python
main
Ln 26, Col 16 Spaces: 4 UTF-8 CRLF Python 3.12.2 64-bit Go Live Prettier