임베디드응용 및 실습

(10주차 과제)

2020161047 박종혁

1) OpenCV를 사용하여 라즈베리파이 카메라에서 받은 실시간 영상으로 얼굴 검출

• 얼굴 검출 시 사각형 박스가 표시되도록 함

• 소스 코드 및 나의 얼굴 검출 영상 제출

다음은 소스코드입니다.

```
import numpy as np
import cv2

face_cascade =
cv2.CascadeClassifier("/home/pipi/haarcascades/haarcascade_frontalface_def
ault.xml")
eye_cascade =
cv2.CascadeClassifier("/home/pipi/haarcascades/haarcascade_eye.xml")

cap = cv2.VideoCapture(0, cv2.CAP_V4L)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

while (True):
    ret, img = cap.read()
    img = cv2.flip(img, -1)

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.2, 5)
    print("Number of faces detected: " + str(len(faces)))

    for (x, y, w, h) in faces:
        img = cv2.rectangle(img, (x, y), (x + w, y + h), (0, 0, 0), 1)
        roi_gray = gray[y:y + h, x:x + w]
        roi_color = img[y:y + h, x:x + w]
        eyes = eye_cascade.detectMultiScale(roi_gray)
```

```
        for (ex, ey, ew, eh) in eyes:
            cv2.rectangle(roi_color, (ex, ey), (ex + ew, ey + eh), (0, 255, 0), 1)

    cv2.imshow('img', img)

    k = cv2.waitKey(30) & 0xff
    if k == 27:  # Esc 키를 누르면 종료
        break

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

코드해석: 나의 얼굴과 눈을 검출하기 위해서 cv2.CascadeClassifier()함수로 XML파일을 참조하여 인식할 수 있도록 하였습니다.
cv2.VideoCapture()로 카메라 영상을 불러오고 cap.set()으로 카메라 영상의 넓이, 높이를 정해줍니다.
카메라 영상을 cap.read()로 읽고 cv2.cvtColor()로 색깔을 정해줍니다.
face_cascade.detectMultiScale()로 얼굴을 탐지합니다.
탐지한 faces로 for문을 사용하여 검은색 사각형으로 표시합니다.
눈은 초록색 사각형으로 표시합니다.

다음은 실행결과입니다.

2) 첨부된 4장의 이미지를 라인 트레이서 용도로 얻었다고 가정하고, 4장의 영상에서 노란색 또는 흰색선을 추출하여 표기하시오.

• 영상 표기하는 방법은 자유롭게 한다

• 사각형, 라인, 선만 남기고 다 검게 등등…

• 제안 알고리즘을 4장의 영상에 동일하게 적용 시, 성능이 보장되도록 한다

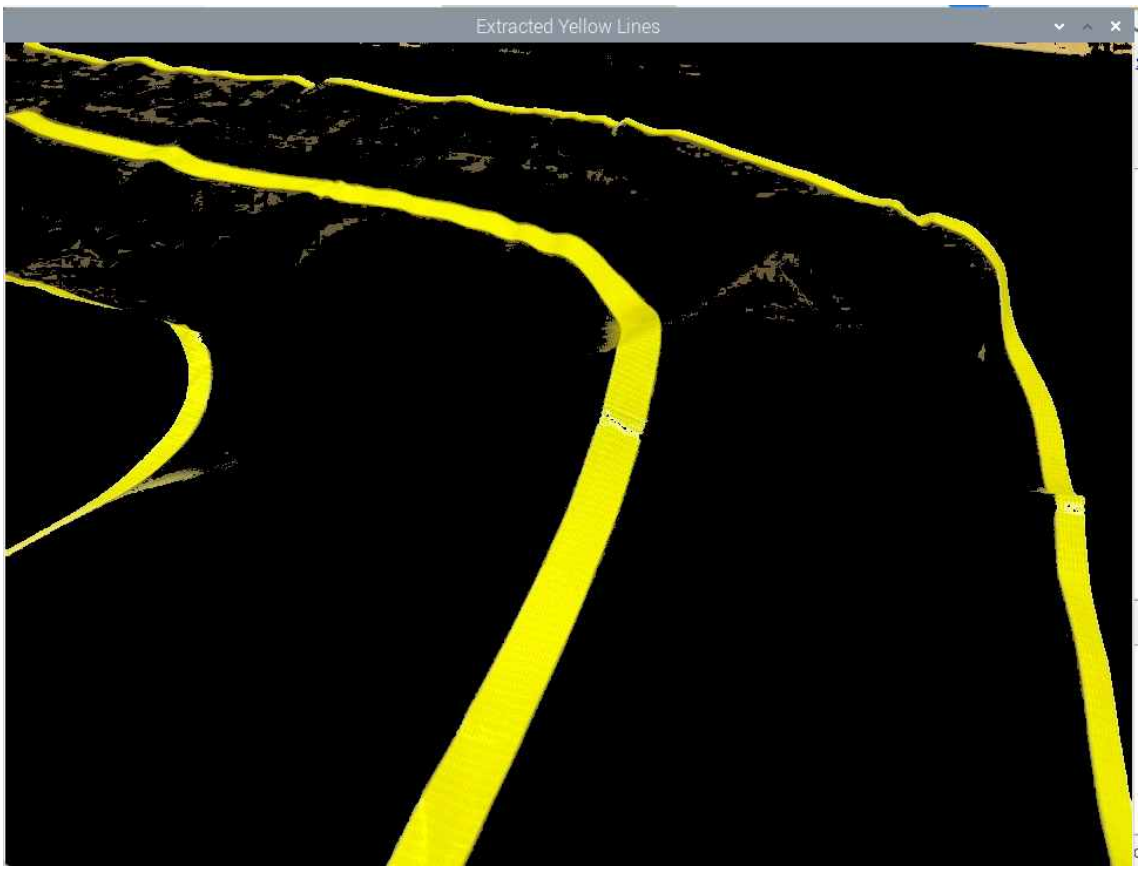
• 영상 크기 변경, 크롭, 컬러 변경 등 자유롭게 할 수 있다 • 소스 코드와 라인 표기된 4장의 영상 제출

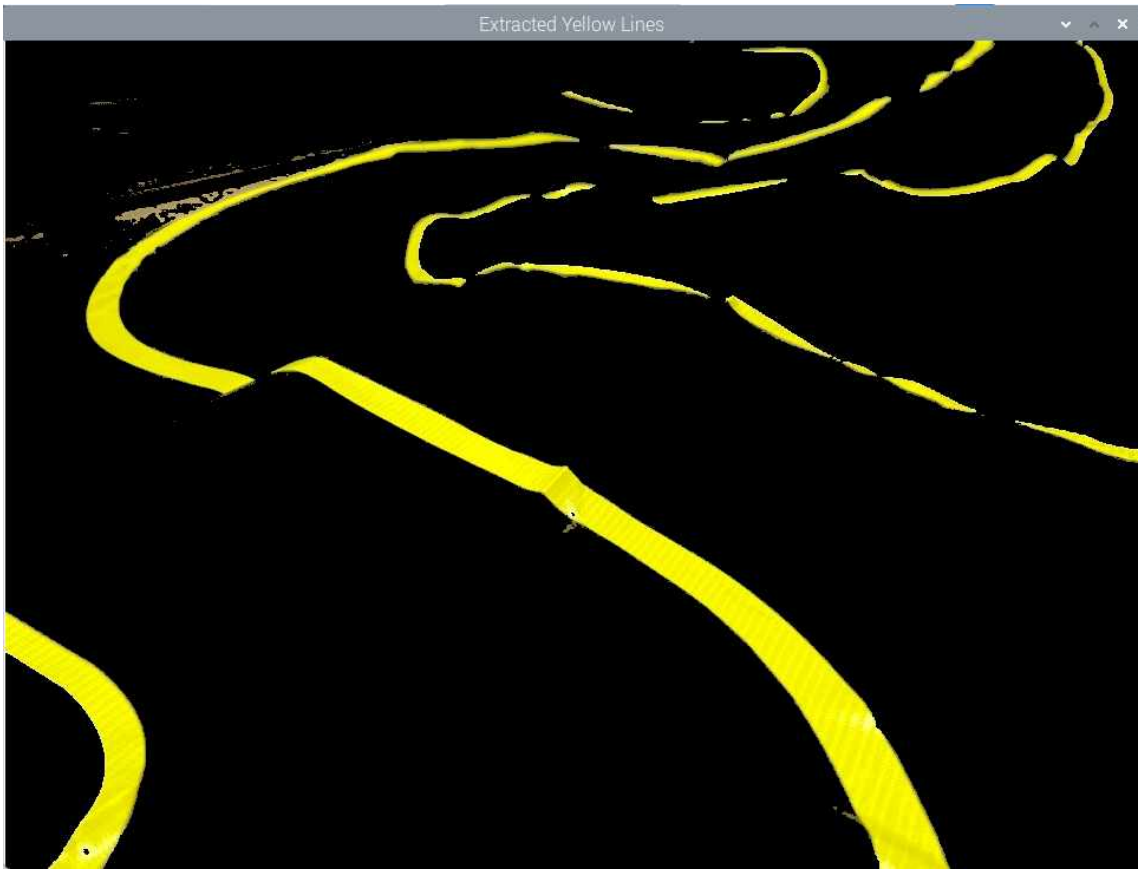
다음은 소스코드입니다.

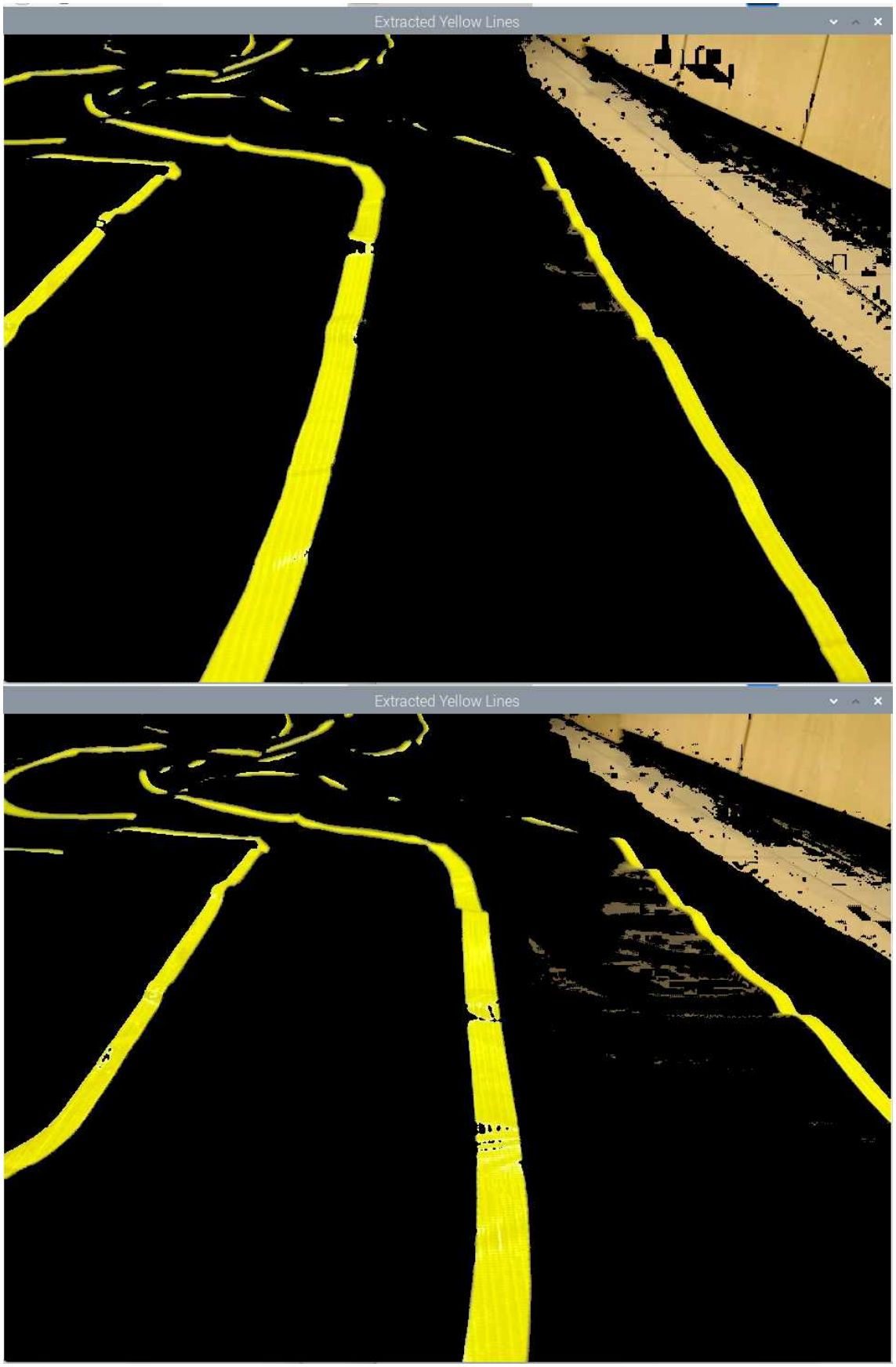
```
import cv2
import numpy as np

def process_image(image_path):
    img = cv2.imread(image_path)

    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    lower_yellow = np.array([20, 100, 100])
    upper_yellow = np.array([40, 255, 255])

    yellow_mask = cv2.inRange(hsv, lower_yellow, upper_yellow)

    result = cv2.bitwise_and(img, img, mask=yellow_mask)

    cv2.imshow("Extracted Yellow Lines", result)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

image_paths = ['image1.jpg', 'image2.jpg', 'image3.jpg', 'image4.jpg']
for image_path in image_paths:
    process_image(image_path)
```

코드해석: 이미지를 읽어오는 함수 process_image()를 선언하여 이미지를 cv2.imread()로 읽어오고 cv2.cvtColor()로 HSV 색 공간으로 변환합니다.
lower_yellow와 upper_yellow로 노란색 값의 최소와 최대를 정의하고 노란색 영역을 cv2.inRange()로 범위를 정합니다.
result = cv2.bitwise_and(img, img, mask=yellow_mask)로 원본이미지에서 노란색 선만 추출합니다.
결과를 cv2.imshow()로 보입니다.
마지막으로 이미지 경로를 설정해주고 for문으로 함수를 실행합니다.

다음은 실행결과입니다.




Extracted Yellow Lines


Extracted Yellow Lines
Extracted Yellow Lines