임베디드 응용 및 실습 11 주차 과제

2022180034 김한별

1. 코드

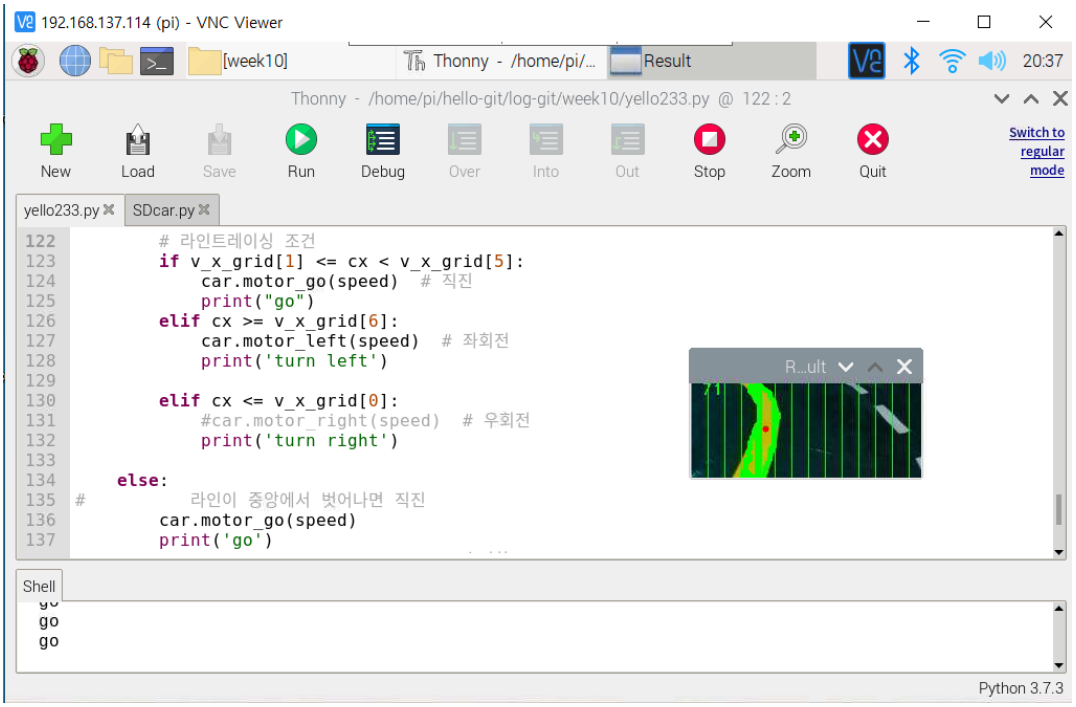
```
import cv2
import numpy as np
import SDcar

moment = np.array([0, 0, 0])
v_x = 320
v_y = 240
v_x_grid = [int(v_x * i / 20) for i in range(1, 20)]
epsilon = 1e-5  # Zero division 방지
enable_linetracing = False  # 라인트레이싱 활성화 변수

speed = 50

# 노란색 마스크 생성 함수
def detect_maskY_BGR(frame):
    B = frame[:, :, 0]
    G = frame[:, :, 1]
    R = frame[:, :, 2]
    Y = np.zeros_like(G, np.uint8)

    Y = G * 0.5 + R * 0.5 - B * 0.7
    Y = Y.astype(np.uint8)
    Y = cv2.GaussianBlur(Y, (5, 5), cv2.BORDER_DEFAULT)

    _, maskY = cv2.threshold(Y, 100, 255, cv2.THRESH_BINARY)
    return maskY

# 그리드 표시 함수
def show_grid(img):
    h, _, _ = img.shape
    for x in v_x_grid:
        cv2.line(img, (x, 0), (x, h), (0, 255, 0), 1, cv2.LINE_4)  # 초록색
그리드

# 노란색 선 검출 및 모멘트 계산 함수
def detect_yellow_and_calculate_moment():
    global enable_linetracing  # 전역 변수 참조

    cap = cv2.VideoCapture(0)
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, 320)
```

```
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 240)

    try:
        while True:
            ret, frame = cap.read()
            if not ret:
                print("카메라 프레임을 읽을 수 없습니다.")
                break

            frame = cv2.flip(frame, 0)
            frame = cv2.flip(frame, 1)
            frame = frame[180:, :-100]  # 하단 60 픽셀을 잘라내고 나머지 영역만
사용

            # 그리드 표시
            show_grid(frame)

            # 노란색 컨투어 검출
            hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
            lower_yellow = np.array([20, 100, 100])
            upper_yellow = np.array([30, 255, 255])

            # 노란색 영역 추출
            mask = cv2.inRange(hsv, lower_yellow, upper_yellow)
            contours, _ = cv2.findContours(mask, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

            if len(contours) > 0:
                c = max(contours, key=cv2.contourArea)
                m = cv2.moments(c)

                # 모멘트 계산
                cx = int(m['m10'] / (m['m00'] + epsilon))  # cx 계산
                cy = int(m['m01'] / (m['m00'] + epsilon))  # cy 계산

                # 빨간색 모멘트 위치 표시
                cv2.circle(frame, (cx, cy), 3, (0, 0, 255), -1)  # 빨간색 점

                # 초록색 컨투어 표시
                cv2.drawContours(frame, contours, -1, (0, 255, 0), 3)  # 초록색
윤곽선 그리기

                # X 좌표 출력
                cv2.putText(frame, str(cx), (10, 10), cv2.FONT_HERSHEY_DUPLEX,
0.5, (0, 255, 0))

                if enable_linetracing:
                    line_tracing(cx)  # 라인트레이싱 수행
```

```
            cv2.imshow('Result', frame)  # 화면에 결과 표시

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

    finally:
        cap.release()
        cv2.destroyAllWindows()

# 라인트레이싱 수행 함수
def line_tracing(cx):
    global moment
    global v_x
    tolerance = 0.1
    diff = 0

    # 초기 시작 시 moment 값이 제대로 설정되었는지 확인
    if moment[0] == 0 and moment[1] == 0 and moment[2] == 0:
        moment[2] = cx  # 처음 모멘트 값 설정

    if moment[0] != 0 and moment[1] != 0 and moment[2] != 0:
        avg_m = np.mean(moment)
        diff = np.abs(avg_m - cx) / v_x

    if diff <= tolerance:
        moment[0] = moment[1]
        moment[1] = moment[2]
        moment[2] = cx

        # 라인트레이싱 조건
        if v_x_grid[1] <= cx < v_x_grid[5]:
            car.motor_go(speed)  # 직진
            print("go")
        elif cx >= v_x_grid[6]:
            car.motor_left(speed)  # 좌회전
            print('turn left')
        elif cx <= v_x_grid[0]:
            car.motor_right(speed)  # 우회전
            print('turn right')

    else:
        # 라인이 중앙에서 벗어나면 직진
        car.motor_go(speed)
        print('go')
        moment = [0, 0, 0]  # moment 초기화
```

```
if __name__ == '__main__':
    car = SDcar.Drive()  # 자동차 객체 생성
    enable_linetracing=True
    detect_yellow_and_calculate_moment()  # 라인트레이싱 시작
```

2. 제어화면



3. 기존 코드에서 추가된 기능

왼쪽 노란색 선을 기준으로 라인 트레이싱이 되도록 구현하였다. 따라서 오른쪽 프레임을 추가로 100픽셀 제거하였다. 또한 더 세밀한 라인 트레이싱을 위해 기존 10등분에서 14등분으로 쪼개었다.