임베디드 응용 및 실습

- 11주차 과제 -

학번	2020161092
이름	이재윤

(과제) 1m내외의 코스에서 직진 및 좌/우회전 라인트레이싱이 가능하도록 코딩하시오.

코드

```python
import cv2 as cv
import numpy as np
import threading, time
import SDcar

def func_thread():
    i = 0
    while True:
        print("alive!!")
        time.sleep(1)
        i = i+1
        if is_running is False:
            break

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 = cv.GaussianBlur(Y,(5,5),cv.BORDER_DEFAULT)
    _, mask_Y = cv.threshold(Y,100,255,cv.THRESH_BINARY)
    return mask_Y

```

```python
def key_cmd(which_key):
    is_exit = False
    global enable_linetracing
    if which_key & 0xFF == 184:
        print('up')
        car.motor_go(speed)
    elif which_key & 0xFF == 178:
        print('down')
        car.motor_back(speed)
    elif which_key & 0xFF == 180:
        print('left')
        car.motor_left(speed)
    elif which_key & 0xFF == 182:
        print('right')
        car.motor_right(speed)
    elif which_key & 0xFF == 181:
        car.motor_stop()
        print('stop')
    elif which_key & 0xFF == ord('q'):
        car.motor_stop()
        print('exit')
        is_exit = True
    elif which_key & 0xFF == ord('e'):
        enable_linetracing = True
        print('enable_linetracing: ', enable_linetracing)
    elif which_key & 0xFF == ord('w'):
        enable_linetracing = False
        car.motor_stop()
        print('enable_linetracing 2: ', enable_linetracing)
    return is_exit

```

```python
def show_grid(img):
    h,_,_ = img.shape
    for x in v_x_grid:
        cv.line(img, (x,0), (x,h), (0,255,0), 1, cv.LINE_4)

def line_tracing(cx):
    global moment
    global v_x
    tolerance = 0.1
    diff = 0

    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

    print('diff ={:.4f}', format(diff))

    if diff <= tolerance:

        moment[0] = moment[1]
        moment[1] = moment[2]
        moment[2] = cx

        if v_x_grid[2] <= cx < v_x_grid[3]:
            car.motor_go(speed)
            print('go')
        elif v_x_grid[3] >= cx:
            car.motor_left(speed)
            print('turn left')
        elif v_x_grid[1] <= cx:
            car.motor_right(speed)
            print('turn right')
    else:
        car.motor_go(speed)
        print('go')
        moment = [0,0,0]

```

```python
def main():

    camera = cv.VideoCapture(0)
    camera.set(cv.CAP_PROP_FRAME_WIDTH,v_x)
    camera.set(cv.CAP_PROP_FRAME_HEIGHT,v_y)

    try:
        while( camera.isOpened() ):
            ret, frame = camera.read()
            frame = cv.flip(frame,-1)
            cv.imshow('camera', frame)

            crop_img = frame[180:,:]
            maskY=detect_maskY_BGR(crop_img)

            contours,_=cv.findContours(maskY, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)

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

                cx=int(m['m10']/(m['m00']+0.000001))
                cy=int(m['m01']/(m['m00']+0.000001))
                cv.circle(crop_img, (cx,cy), 3, (0,0,255),-1)
                cv.drawContours(crop_img, contours, -1, (0,255,0),3)

                cv.putText(crop_img, str(cx), (10,10), cv.FONT_HERSHEY_DUPLEX, 0.5, (0,255,0))

                if enable_linetracing==True:
                    line_tracing(cx)

            show_grid(crop_img)
            cv.imshow('crop_img', cv.resize(crop_img, dsize=(0,0), fx=2, fy=2))

            is_exit = False
            which_key = cv.waitKey(20)
            if which_key > 0:
                is_exit = key_cmd(which_key)
            if is_exit is True:
                cv.destroyAllWindows()
                break
    except Exception as e:
        print(e)
        global is_running
        is_running = False

```

```python
if __name__ == '__main__':

    speed = 20
    v_x = 320
    v_y = 240
    v_x_grid = [int(v_x*i/10) for i in range(1,10)]

    moment = np.array([0,0,0])

    print(v_x_grid)

    t_task1 = threading.Thread(target = func_thread)
    t_task1.start()

    car = SDcar.Drive()

    is_running = True
    enable_linetracing = False
    main()
    is_running = False
    car.clean_GPIO()
    print('end vis')
```