13주차 과제

과 목 명	임베디드응용및실습
학 번	2020161088
이 름	이 순 신
제 출 일	2024년 12월 03일

- 강의노트 lec10의 마지막 장의 과제를 수행

- github에 올리고 링크를 첨부하거나, 과제란에 동영상 바로 제출

코드:

```
import cv2 as cv
import numpy as np
import threading, time
import SDcar
import sys
import tensorflow as tf
from tensorflow.keras.models import load_model

speed = 80
epsilon = 0.0001

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

def key_cmd(which_key):
    print('which_key', which_key)
    is_exit = False
    global enable_AIdrive # assignment가 있는 경우는 global 키워드로 표시
    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(30)
    elif which_key & 0xFF == 182:
        print('right')
        car.motor_right(30)
```

```
    elif which_key & 0xFF == 181:
        car.motor_stop()
        enable_AIdrive = False
        print('stop')
    elif which_key & 0xFF == ord('q'):
        car.motor_stop()
        print('exit')
        enable_AIdrive = False
        is_exit = True
        print('enable_AIdrive: ', enable_AIdrive)
    elif which_key & 0xFF == ord('e'):
        enable_AIdrive = True
        print('enable_AIdrive: ', enable_AIdrive)
    elif which_key & 0xFF == ord('w'):
        enable_AIdrive = False
        car.motor_stop()
        print('enable_AIdrive 2: ', enable_AIdrive)

    return is_exit

def detect_maskY_HSV(frame):
    crop_hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
    crop_hsv = cv.GaussianBlur(crop_hsv, (5,5), cv.BORDER_DEFAULT)
    # need to tune params
    mask_Y = cv.inRange(crop_hsv, (25, 50, 100), (35, 255, 255))
    return mask_Y

def detect_maskY_BGR(frame):
    B = frame[:,:,0]
    G = frame[:,:,1]
    R = frame[:,:,2]
    Y = np.zeros_like(G, np.uint8)
    # need to tune params
    Y = G*0.5 + R*0.5 - B*0.7 # 연산 수행 시 float64로 바뀜
    Y = Y.astype(np.uint8)
    Y = cv.GaussianBlur(Y, (5,5), cv.BORDER_DEFAULT)
    # need to tune params
    _, mask_Y = cv.threshold(Y, 100, 255, cv.THRESH_BINARY)
    return mask_Y

def line_tracing(cx):
```

```
        #print('cx, ', cx)
        #print('v_x_grid', v_x_grid)
        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
            print('cx : ', cx)
            if v_x_grid[2] <= cx < v_x_grid[4]:
                car.motor_go(speed)
                print('go')
            elif v_x_grid[3] >= cx:
                car.motor_left(30)
                print('turn left')
            elif v_x_grid[1] <= cx:
                car.motor_right(30)
                print('turn right')
            else:
                print("skip")

        else:
            car.motor_go(speed)
            print('go')
            moment = [0,0,0]

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

```
def test_fun(model):
    camera = cv.VideoCapture(0)
    camera.set(cv.CAP_PROP_FRAME_WIDTH,v_x)
    camera.set(cv.CAP_PROP_FRAME_HEIGHT,v_y)
    ret, frame = camera.read()
    frame = cv.flip(frame,-1)
    cv.imshow('camera',frame)
    crop_img = frame[int(v_y/2):,:]
    crop_img = cv.resize(crop_img, (200, 66))
    crop_img = np.expand_dims(crop_img, 0)
    a = model.predict(crop_img)
    print('okey, a: ', a)

def drive_AI(img):
    #print('id', id(model))
    img = np.expand_dims(img, 0)
    res = model.predict(img)[0]
    #print('res', res)
    steering_angle = np.argmax(np.array(res))
    print('steering_angle', steering_angle)
    if steering_angle == 0:
        print("go")
        speedSet = 60
        car.motor_go(speedSet)
    elif steering_angle == 1:
        print("left")
        speedSet = 20
        car.motor_left(speedSet)
    elif steering_angle == 2:
        print("right")
        speedSet = 20
        car.motor_right(speedSet)
    else:
        print("This cannot be entered")

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)
            # image processing start here
            crop_img = frame[int(v_y/2):,:]
            crop_img = cv.resize(crop_img, (200, 66))
            cv.imshow('crop_img ', cv.resize(crop_img, dsize=(0,0), fx=2, fy=2))

            if enable_AIdrive == True:
                drive_AI(crop_img)

            # image processing end here
            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:
        exception_type, exception_object, exception_traceback = sys.exc_info()
        filename = exception_traceback.tb_frame.f_code.co_filename
        line_number = exception_traceback.tb_lineno

        print("Exception type: ", exception_type)
        print("File name: ", filename)
        print("Line number: ", line_number)
        global is_running
        is_running = False

if __name__ == '__main__':

    v_x = 320
    v_y = 240
    v_x_grid = [int(v_x*i/10) for i in range(1, 10)]
    print(v_x_grid)
    moment = np.array([0, 0, 0])
```

```
model_path = 'lane_navigation_20241202_1135.h5'

model = load_model(model_path)
'''print('id', id(model))
print(model.summary())'''

#test_fun(model)

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

car = SDcar.Drive()

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

실행결과:
동영상으로 제출