BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HCM



THÍ NGHIỆM THỊ GIÁC MÁY TÍNH

BÀI BÁO CÁO

NHÓM 5

Giảng viên : Trần Văn Hùng

Sinh viên :

Trần Công Hòa 20017891

Bùi Mỹ Trung Hiếu 20097651

TP.HCM – 2022

Đề bài:

* Nhận dạng 3 đối tượng (ảnh của chính 2 SV trong nhóm và 1 vật bất kỳ) sử dụng YOLO8.
* Điều khiển động cơ DC và servo dùng board Jetson xavier từ kết quả nhận dạng được.
* Hiển thị kết quả ra led đơn thông qua phương thức truyền I/O
* Hiển thị kết quả lên LCD thông qua phương thức truyền I2C
* Hiển thị kết quả các thông số hệ thống của board Jetson lên OLED

Đối tượng nhận diện:

Two men holding a balloon

Description automatically generated

HIẾU

BALL

HÒA

Code:

import cv2

from ultralytics import YOLO

import torch

import Jetson.GPIO as GPIO

from RPi\_GPIO\_i2c\_LCD import lcd

from time import sleep

GPIO.setmode(GPIO.BOARD)

i2c\_address = 0x27

model = YOLO("best.pt")

lcdDisplay = lcd.HD44780(i2c\_address)

cap = cv2.VideoCapture(0)

l1 = 11

l2 = 13

l3 = 15

l4 = 16

led1 = 33

led2 = 35

led3 = 36

# Thiết lập chân là OUTPUT (để điều khiển LED)

GPIO.setup(l1, GPIO.OUT)

GPIO.setup(l2, GPIO.OUT)

GPIO.setup(l3, GPIO.OUT)

GPIO.setup(l4, GPIO.OUT)

GPIO.setup(led1, GPIO.OUT)

GPIO.setup(led2, GPIO.OUT)

GPIO.setup(led3, GPIO.OUT)

while cap.isOpened():

    success, frame = cap.read()

    if success:

        results = model(frame)

        annotated\_frame = results[0].plot()

        cv2.imshow("YOLOv8 Inference", annotated\_frame)

        for result in results:

            print("class", result.boxes.cls)

            for k in result.boxes.cls:

                if k == torch.Tensor([0.]):

                    lcdDisplay.set("HOA",1)

                    GPIO.output(l1, GPIO.HIGH)

                    GPIO.output(l2, GPIO.LOW)

                    GPIO.output(l3, GPIO.LOW)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.HIGH)

                    GPIO.output(led2, GPIO.LOW)

                    GPIO.output(led3, GPIO.LOW)

                else:

                    lcdDisplay.set("                    ",1)

                if k == torch.Tensor([1.]):

                    lcdDisplay.set("HIEU",2)

                    GPIO.output(l1, GPIO.LOW)

                    GPIO.output(l2, GPIO.LOW)

                    GPIO.output(l3, GPIO.HIGH)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.LOW)

                    GPIO.output(led2, GPIO.HIGH)

                    GPIO.output(led3, GPIO.LOW)

                else:

                    lcdDisplay.set("                    ",2)

                if k == torch.Tensor([2.]):

                    lcdDisplay.set("BALL",3)

                    GPIO.output(l1, GPIO.HIGH)

                    GPIO.output(l2, GPIO.LOW)

                    GPIO.output(l3, GPIO.HIGH)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.LOW)

                    GPIO.output(led2, GPIO.LOW)

                    GPIO.output(led3, GPIO.HIGH)

                else:

                    lcdDisplay.set("                    ",3)

            #print("xyxy", result.boxes.xyxy)

            #print("conf", result.boxes.conf)

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

            break

    else:

        break

cap.release()

cv2.destroyAllWindows()

Phần đầu chương trình:Khai báo các thư viện cần dùng

A black screen with white text

Description automatically generated

* import cv2: Import thư viện OpenCV để làm việc với video và hình ảnh.
* from ultralytics import YOLO: Import mô hình YOLO từ thư viện ultralytics để thực hiện nhận dạng đối tượng.
* import torch: Import thư viện PyTorch để làm việc với các tensor và thực hiện tính toán trên GPU.
* import Jetson.GPIO as GPIO: Import thư viện GPIO của Jetson để điều khiển các chân GPIO trên bo mạch Jetson.
* from RPi\_GPIO\_i2c\_LCD import lcd: Import thư viện LCD để điều khiển màn hình LCD thông qua giao tiếp I2C.
* from time import sleep: Import hàm sleep để tạo một khoảng thời gian chờ.

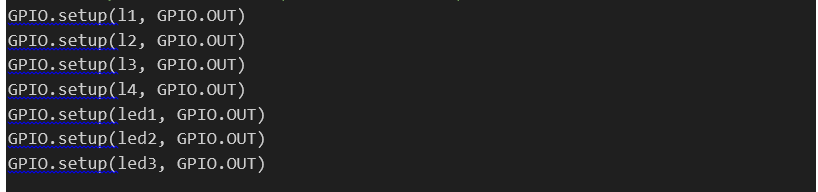
Phần tiếp theo của chương trình:Khai báo các chân GPIO và các thông số để kết nối phần cứng cần dùng.

A computer screen shot of a computer code

Description automatically generated

* GPIO.setmode(GPIO.BOARD): Thiết lập chế độ của GPIO là BOARD.
* i2c\_address = 0x27: Địa chỉ I2C của màn hình LCD.
* model = YOLO("best.pt"): Khởi tạo mô hình YOLO sử dụng tệp "best.pt".
* lcdDisplay = lcd.HD44780(i2c\_address): Khởi tạo màn hình LCD thông qua địa chỉ I2C.
* cap = cv2.VideoCapture(0): Khởi tạo một đối tượng VideoCapture để đọc video từ camera với ID 0 (camera mặc định).
* L1,l2,l3,l4: Là các biến lưu các giá trị chân GPIO tương ứng trên board Jetson Xavier lần lượt là 11,13,15,16 dùng để xuất ra tín hiệu điều khiển cho mạch cầu H để chạy động cơ.
* Led1,led2,led: Là các biến lưu giá trị chân GPIO dùng để điều khiển led.

Phần tiếp theo của chương trình:Thiết lập chức năng của các chân GPIO



Thiết lập mặc định các chân lúc bắt đầu thực thi chương chình là chế độ Output

Phần tiếp theo của chương trình:Vòng lập while chính của chương trình A computer screen shot of a computer code

Description automatically generated

* Trong vòng lặp while, chương trình tiếp tục chạy trong khi camera mở.
* success, frame = cap.read(): Đọc một khung hình từ camera và kiểm tra thành công.
* Dòng mã results = model(frame) thực hiện việc đưa khung hình frame vào mô hình YOLO (model) để thực hiện quá trình nhận dạng đối tượng.
* Nếu thành công, thực hiện nhận dạng đối tượng bằng mô hình YOLO trên khung hình (frame).
* results[0].plot(): Vẽ kết quả nhận dạng lên khung hình và gán vào biến annotated\_frame.
* cv2.imshow("YOLOv8 Inference", annotated\_frame): Hiển thị khung hình với kết quả nhận dạng lên màn hình.

Phần tiếp theo của chương trình:Thực hiện hành động với mỗi đối tượng nhận diện tương ứng

Vòng lập for này dùng để duyệt qua các đối tượng nhận dạng và in ra các thông số của đối tượng nhận dạng trong dang sách resuilt.boxes.cls được lên Terminal.Vòng lập for tiếp theo sẽ lặp qua qua danh sách các lớp của đối tượng. Mục tiêu là duyệt qua từng lớp để xem đối tượng nào thuộc lớp nào và thực hiện các hành động tùy thuộc vào lớp đó.

for result in results:

            print("class", result.boxes.cls)

            for k in result.boxes.cls:

* Xử lý với đối tượng “Hoa”

if k == torch.Tensor([0.]):

                    lcdDisplay.set("HOA",1)

                    GPIO.output(l1, GPIO.HIGH)

                    GPIO.output(l2, GPIO.LOW)

                    GPIO.output(l3, GPIO.LOW)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.HIGH)

                    GPIO.output(led2, GPIO.LOW)

                    GPIO.output(led3, GPIO.LOW)

                else:

                    lcdDisplay.set("                    ",1)

* Dòng mã if k == torch.Tensor([0.]): so sánh giá trị của biến k với một tensor được tạo ra bằng PyTorch, cụ thể là torch.Tensor([0.]), đối tượng “Hoa” được xử lý ở lớp thứ 0.
* Nếu nhận diện thành công sẽ cho LCD hiện thị “HOA” ở vị trí hàng 1, và chân l1 xuất ra mức cao đề gửi tín hiệu đến mạch cầu H, mạch cầu H sẽ điều khiền động cơ 1 chạy. Chân led1 sẽ được xuất ra mức cao để điều khiển led xanh sáng
* Ngược lại nếu không nhận diện được LCD sẽ không hiện thị gì ở hàng 1 và các chân GPIO sẽ được trả về mức thấp.
* Xử lý với đối tượng “Hieu”

if k == torch.Tensor([1.]):

                  lcdDisplay.set("HIEU",2)

                    GPIO.output(l1, GPIO.LOW)

                    GPIO.output(l2, GPIO.LOW)

                  GPIO.output(l3, GPIO.HIGH)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.LOW)

                    GPIO.output(led2, GPIO.HIGH)

                    GPIO.output(led3, GPIO.LOW)

                else:

                    lcdDisplay.set("                    ",2)

* Dòng mã if k == torch.Tensor([0.]): so sánh giá trị của biến k với một tensor được tạo ra bằng PyTorch, cụ thể là torch.Tensor([0.]), đối tượng “Hoa” được xử lý ở lớp thứ 1.
* Nếu nhận diện thành công sẽ cho LCD hiện thị “HIEU” ở vị trí hàng 2, và chân l3 xuất ra mức cao đề gửi tín hiệu đến mạch cầu H, mạch cầu H sẽ điều khiền động cơ 2 chạy. Chân led2 sẽ được xuất ra mức cao để điều khiển led đỏ sáng
* Ngược lại nếu không nhận diện được LCD sẽ không hiện thị gì ở hàng 2 và các chân GPIO sẽ được trả về mức thấp.
* Xử lý với đối tượng “Ball”

if k == torch.Tensor([2.]):

                    lcdDisplay.set("BALL",3)

                    GPIO.output(l1, GPIO.HIGH)

                    GPIO.output(l2, GPIO.LOW)

                    GPIO.output(l3, GPIO.HIGH)

                    GPIO.output(l4, GPIO.LOW)

                    GPIO.output(led1, GPIO.LOW)

                    GPIO.output(led2, GPIO.LOW)

                    GPIO.output(led3, GPIO.HIGH)

                else:

                    lcdDisplay.set("                    ",3)

* Dòng mã if k == torch.Tensor([0.]): so sánh giá trị của biến k với một tensor được tạo ra bằng PyTorch, cụ thể là torch.Tensor([0.]), đối tượng “Hoa” được xử lý ở lớp thứ 2.
* Nếu nhận diện thành công sẽ cho LCD hiện thị “BALL” ở vị trí hàng 3, chân l1 và chân l3 xuất ra mức cao đề gửi tín hiệu đến mạch cầu H, mạch cầu H sẽ điều khiền cả 2 động cơ 2 chạy. Chân led3 sẽ được xuất ra mức cao để điều khiển led vàng sáng
* Ngược lại nếu không nhận diện được LCD sẽ không hiện thị gì ở hàng 3 và các chân GPIO sẽ được trả về mức thấp.
* Đoạn cuối chương trình:Thực hiện tắt camera và kết thúc chương trình.

A screen shot of a computer code

Description automatically generated

* if cv2.waitKey(1) & 0xFF == ord("q"):: Nếu người dùng ấn phím "q", thoát khỏi vòng lặp.
* Cuối cùng, sau khi thoát khỏi vòng lặp, chương trình giải phóng tài nguyên, đóng video (cap.release()) và đóng cửa sổ hiển thị (cv2.destroyAllWindows()).