**Phân tích và giải thích các class, function, module của file yolodetect.py**

1. Các thư viện được import

A screen shot of a computer code

AI-generated content may be incorrect.

* Shapely: dùng để kiểm tra điểm có nằm trong polygon hay không (bằng function “isInside()”)
* Cv2: thư viện OpenCV dùng để xử lý ảnh/video.
* Ultralytics: sử dụng YOLOv8 để nhận diện object
* Telegram\_utils: chứa hàm “send\_telegram()” để gửi ảnh cảnh báo
* Datetime: để tính toán khoảng thời gian giữa các cảnh báo
* Threading: dùng để gửi Telegram cảnh báo trong thread riêng
* Torch: thư viện PyTorch dùng để kiểm tra thiết bị có GPU hay không

1. Hàm isInside()

A screen shot of a computer code

AI-generated content may be incorrect.

1. Mục đích

* Kiểm tra xem centroid( tâm bounding box) có nằm trong vùng đa giác do người dùng vẽ hay không

1. Cách hoạt động

* Polygon(points): tạo đa giác từ các đa giác điểm
* Point(centroid): tạo điểm từ tuple(x, y)
* polygon.contains(centroid): trả về true khi có centroid nằm trong polygon

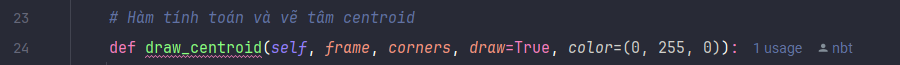
1. Class yolodetect: Quản lý mô hình YOLO và xử lý nhận diện
2. Hàm \_\_init\_\_(): khởi tạo model YOLO và thiết lập thời gian cảnh báo

A screen shot of a computer code

AI-generated content may be incorrect.

* Device : sẽ kiểm tra thiết bị có GPU, nếu có thì sử dụng GPU còn không thì sẽ sử dụng CPU. Thông qua module PyTorch
* Self.model: khởi tạo mô hình YOLO từ file yolov10n.pt trong thư mục “model”
* Self.last\_alert: lưu thời điểm cảnh báo gần nhất
* Self.alert\_interval: khoảng thời gian giữa 2 lần cảnh báo liên tiếp (Giây)

1. Hàm draw\_centroid()
2. Các tham số truyền vào



* Frame: hình ảnh trực tiếp từ camera
* Conners: hiểu đơn giản là toạ độ(x,y) của từng góc của object
* Draw : khởi tạo true để các bước sau này xét điều kiện có thể vẽ
* (0,255,0): color theo chuẩn BGR

A screenshot of a computer program

AI-generated content may be incorrect.

* Pt1,pt2,pt3,pt4: được khởi tạo như là góc của object, **pt1: top-left, pt2:top-right, pt3: bottom-right, pt4: bottom-left**
* P1[**0**]: p1.**x1**, p1[**1**]: p1.**y1,** các điểm còn lại tương tự

1. Cách sử dụng

* Hàm line\_intersection: dùng để tính toán đường chéo của object
* Cho ví dụ và giải thích hàm line\_intersection thông qua đồ thị :

A white sheet of paper with writing on it

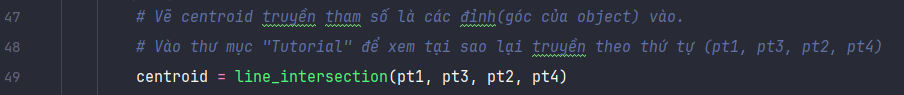
AI-generated content may be incorrect.

* Ta cho p1(2,5), p2(5,5), p3(5,3), p4(2,3)
* Tiếp theo xét 2 đường chéo p1->p3, p2->p4
* Theo công thức trên ta có:
* Đường chéo 1: A1=p2[1]-p1[1]= y2 – y1=-2, B1= -3, C1= -19
* tương tự ta có đường chéo 2: A2= -2, B2= 3, C2= 5
* =>phương trình đường thẳng: dc1: Ax + By= C ⬄ -2x -3y = -19; dc2: -2x + 3y = 5

A screenshot of a graph

AI-generated content may be incorrect.

* Nhận thấy 2 đường chéo này không song song hay trùng nha =>THOẢ MÃN
* Tiếp theo, tính biến det để xét điều kiện xem có trùng hay song song hay không : det= -12 => THOẢ MÃN det!=0
* Det!= 0 nên trả về toạ độ (x,y) của tâm được tạo ra bởi 2 đường chéo bằng công thức trên => (3.5,4)



* Lúc này vẽ đường chéo xác định centroid trên object, truyền vào theo thứ tự (pt1, pt3, pt2, pt4) là vì YOLO xác định các góc(conners) của object theo hình đồ thị ví dụ ở trên nên từ pt1->pt3 và pt2->pt4, nên nếu truyền vào (pt1, pt2, pt3, pt4) thì sẽ lỗi
* Tiếp theo, kiểm tra tham số draw có true và có tồn tại centroid hay không. Nếu thoả điều kiện thì sử dụng hàm của thư viện cv2 để vẽ dấu chấm tròn(centroid). Sau đó trả về centroid

A screenshot of a computer screen

AI-generated content may be incorrect.

1. Hàm detect: phát hiện đối tượng và kiểm tra xâm nhập

A screen shot of a computer

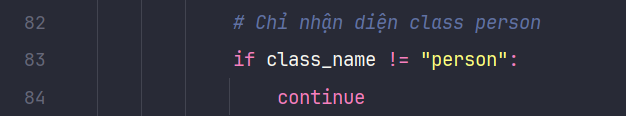
AI-generated content may be incorrect.

* Lệnh này là **gửi một ảnh (frame)** vào mô hình YOLO để dự đoán
* Self.model là mô hình YOLO đã được khỏi tạo từ file “yolov10n.pt”
* Frame: là hình ảnh truyền từ camera
* Result[0]: là 1 list các kết quả tương ứng với các ảnh đã xử lý vì đang truyền 1 ảnh duy nhất là frame thì dùng result[0]

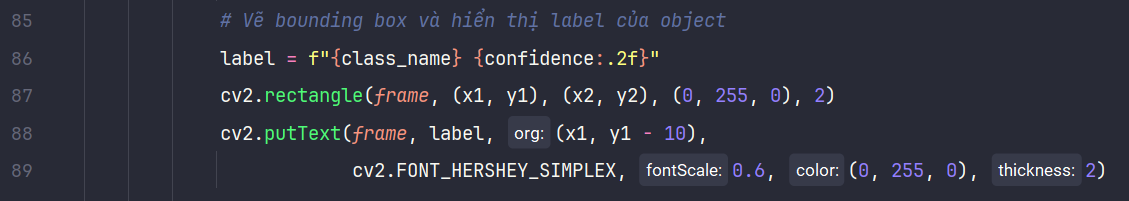
A computer screen with text

AI-generated content may be incorrect.

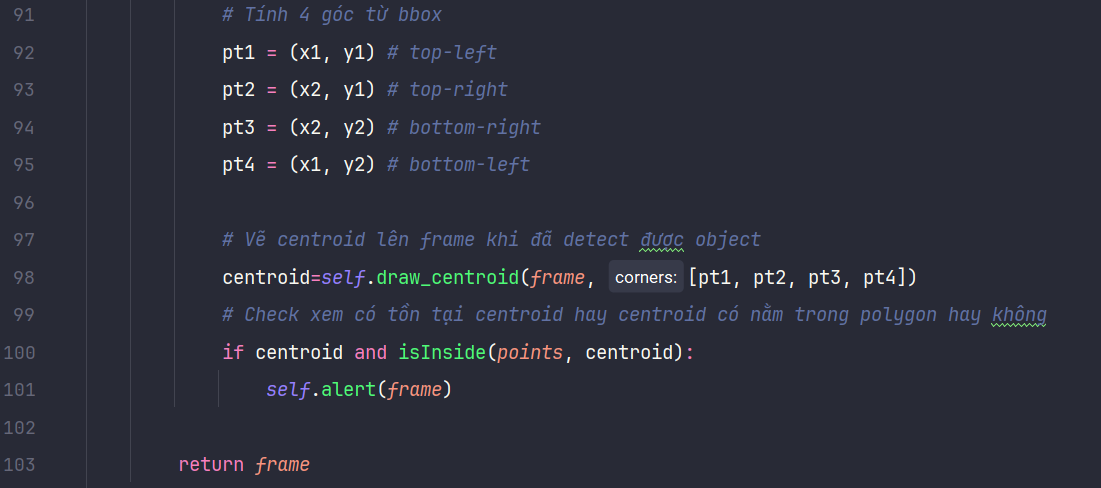
* Bên trong results[0] có thuộc tính .boxes, chứa **danh sách các object (bounding boxes)** mà YOLO phát hiện được trên frame đó. Mỗi box trong vòng for là 1 object(ví dụ như: { 0: ‘person’, 60: ‘chair’,…})
* X1, x2, x3, x4 là các tham số tương ứng với toạ độ góc(x,y) của object được lấy từ box.xyxy[0] : chứa toạ độ 4 góc của bounding box. Tolist() là hàm dùng để chuyển các số liệu lấy được từ kiểu tensor sang list bình thường vì OpenCV yêu cầu toạ độ phải là số nguyên hoặc tuple/list, ép kiểu các số liệu đó để dùng để vẽ hình
* Class\_id: là Id của object mà YOLO nhận diện được, và được lấy bằng hàm box.cls[0], ở đây là cls[0]: {id: 0 ,class\_name: ’person’}
* Class\_name cũng được lấy ra từ class\_id
* Tương tự, confidence score(độ tin cậy) mà YOLO dự đoán cho object đó, được lấy bằng hàm box.conf[0]



* Lọc ra các box(object) và chỉ lấy object có class\_name là “person”



* Vẽ bounding box( khung chữ nhật xung quanh object) bằng các số liệu đã lấy được trước đó và hiển thị class\_name và confidence score lên ngay góc trên trái của khung chữ nhật



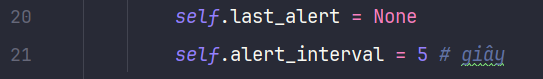
* Tiếp theo, xác định các góc của bounding box sau đó vẽ centroid cho bounding box đó qua hàm draw\_centroid
* Cuối cùng là kiểm tra điều kiện có tồn tại centroid và centroid đó có nằm trong polygon hay không, nếu có thì gọi hàm alert để cảnh báo qua telegram

1. Hàm alert

A screenshot of a computer program

AI-generated content may be incorrect.

* Đầu tiên, nếu centroid nằm trong polygon thì hàm alert sẽ được gọi và hiển thị chữ “ALERTTTT!!!!” lên góc trái trên của frame
* Tiếp theo, dùng 1 biến now để đo khoảng thời gian hiện tại theo chuẩn quốc tế (UTC)
* Kiểm tra điều kiện *self*.last\_alert is None (xem lần cảnh báo gần nhất là lần đầu tiên có none hay không) hoặc khoảng thời gian từ lần cảnh báo gần nhất đến hiện tại có lớn hơn hoặc bằng khoảng thời gian người dùng đặt là 5 giây hay không, nếu thoả điều kiện thì đặt self.last\_alert =now (đặt khoảng thời gian cảnh báo gần nhất là hiện tại)



* Sau đó lưu ảnh (frame) cảnh báo vào thư mục "Alert\_nofications/alert.png", ảnh (frame) này sẽ được cập nhật sau mỗi lần cảnh báo
* Cuối cùng là sử dụng hàm thread trong module threading để gửi cảnh báo qua telegram thông qua hàm send\_telegram, sử dụng thread để không bị làm chậm video hay hiển thị frame khi gửi ảnh qua mạng.

A screen shot of a computer code

AI-generated content may be incorrect.

* Sử dụng thêm thư viện pygame để phát âm thanh khi có object trong polygon