**OBJECT DETECTION AND YOLO**

Xayavong Nalongsit

K13\_CNTT

1. **Giới thiệu** 
   1. **Object detection**

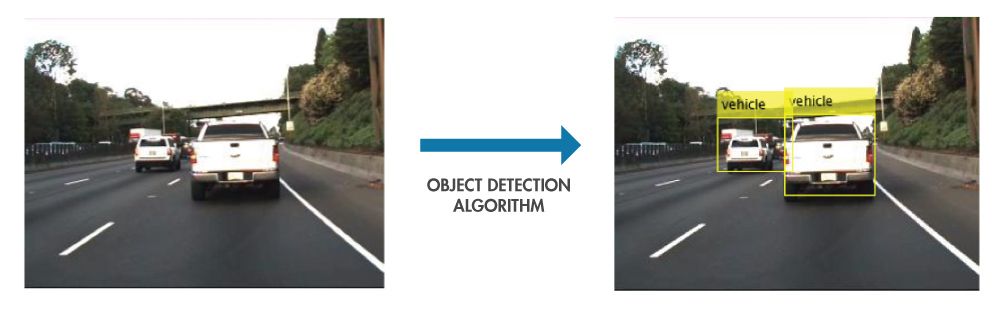
Phát hiện đối tượng là một kỹ thuật [thị giác máy tính](https://www.mathworks.com/products/computer-vision.html) để định vị các thể hiện của đối tượng trong hình ảnh hoặc video. Các thuật toán phát hiện đối tượng thường tận dụng [machine learning](https://www.mathworks.com/discovery/machine-learning.html) hoặc [deep learning](https://www.mathworks.com/discovery/deep-learning.html) để tạo ra kết quả có ý nghĩa. Khi con người nhìn vào hình ảnh hoặc video, chúng ta có thể nhận ra và định vị các đối tượng quan tâm chỉ trong chốc lát. Mục tiêu của việc phát hiện đối tượng là tái tạo trí thông minh này bằng máy tính.

Công nghệ phát hiện đối tượng (object detection) có thể được sử dụng cho nhiều mục đích và trong nhiều lĩnh vực khác nhau. Dưới đây là một số ứng dụng phổ biến:

* **Ô tô tự động (Autonomous Vehicles):**Hệ thống phát hiện đối tượng là một phần quan trọng của công nghệ lái xe tự động. Nó giúp xe tự động nhận biết và đối phó với các đối tượng như xe cộ, người đi bộ, và đèn giao thông.
* **An ninh và Quản lý Giao thông:**Sử dụng trong hệ thống giám sát an ninh để phát hiện và cảnh báo về các sự kiện đáng chú ý như xâm phạm khu vực cấm, người lạ, hoặc vật thể nghi ngờ.
* **Quản lý và An toàn Công trình:**Trong lĩnh vực xây dựng và quản lý công trình, phát hiện đối tượng có thể được sử dụng để theo dõi an toàn và quản lý những nguy cơ tiềm ẩn.
* **Dược học và Y tế:**Trong lĩnh vực y học, nó có thể được sử dụng để phân tích hình ảnh y tế, như phát hiện và đo kích thước của các đối tượng y tế trong hình ảnh chụp X-quang hoặc MRI.
* **Thương mại điện tử và Quảng cáo:**Trong môi trường thương mại điện tử, công nghệ phát hiện đối tượng có thể được sử dụng để tối ưu hóa quảng cáo, ví dụ như nhận diện và đề xuất sản phẩm tương tự dựa trên hình ảnh.
* **Giám sát Môi trường:**Sử dụng để giám sát môi trường tự nhiên, ví dụ như theo dõi và phân loại loài động vật hoặc đối tượng đặc biệt trong tự nhiên.
* **Dịch vụ xã hội và Mạng xã hội:**Trong mạng xã hội, nó có thể được sử dụng để nhận diện khuôn mặt, phân loại hình ảnh, và tạo ra tính năng nhận dạng đối tượng trong các ảnh và video.
* **Giáo dục và Nghiên cứu:**Trong giáo dục, công nghệ này có thể được sử dụng để tạo các ứng dụng giáo dục tương tác hoặc hỗ trợ nghiên cứu trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo.

Các ứng dụng của công nghệ phát hiện đối tượng ngày càng được mở rộng, và nó liên quan mật thiết đến sự phát triển của nhiều lĩnh vực công nghiệp và ứng dụng khác nhau.

**Vì dụ :** Phát hiện đối tượng là công nghệ chủ chốt đằng sau hệ thống hỗ trợ người lái tiên tiến (ADAS) cho phép ô tô phát hiện làn đường lái xe hoặc thực hiện phát hiện người đi bộ để cải thiện an toàn đường bộ. Phát hiện đối tượng cũng hữu ích trong các ứng dụng như hệ thống giám sát video hoặc truy xuất hình ảnh.



Sử dụng tính năng phát hiện đối tượng để xác định và định vị phương tiện.

1. **Phương pháp nghiên cứu**

Phát hiện đối tượng bằng cách sử dụng Deep Learning Bạn có thể sử dụng nhiều kỹ thuật khác nhau để thực hiện phát hiện đối tượng. Các phương pháp tiếp cận dựa trên học sâu phổ biến sử dụng mạng thần kinh tích chập (CNN), chẳng hạn như R-CNN và YOLO v2, tự động học cách phát hiện các đối tượng trong hình ảnh.

**2.1. phương pháp chính để bắt đầu phát hiện đối tượng bằng cách sử dụng học sâu:**

1. **Tạo và đào tạo một trình phát hiện đối tượng tùy chỉnh (Create and train a custom object detector).**

Để đào tạo trình phát hiện đối tượng tùy chỉnh từ đầu, bạn cần thiết kế kiến ​​trúc mạng để tìm hiểu các tính năng của đối tượng quan tâm. Bạn cũng cần biên soạn một tập hợp dữ liệu được gắn nhãn rất lớn để huấn luyện CNN. Kết quả của một máy dò đối tượng tùy chỉnh có thể rất đáng chú ý. Điều đó có nghĩa là bạn cần thiết lập thủ công các lớp và trọng số trong CNN, việc này đòi hỏi nhiều thời gian và dữ liệu huấn luyện.

1. **Sử dụng một máy dò đối tượng được đào tạo trước(Use a pretrained object detector).**

Nhiều quy trình phát hiện đối tượng sử dụng học sâu thúc đẩy học chuyển giao , một cách tiếp cận cho phép bạn bắt đầu với mạng được huấn luyện trước và sau đó tinh chỉnh mạng đó cho ứng dụng của mình. Phương pháp này có thể mang lại kết quả nhanh hơn vì các bộ dò tìm đối tượng đã được huấn luyện trên hàng nghìn, thậm chí hàng triệu hình ảnh.



Phát hiện biển báo dừng bằng R-CNN đã được huấn luyện trước .

* 1. **Mạng của object detection**

Cho dù bạn tạo trình phát hiện đối tượng tùy chỉnh hay sử dụng trình phát hiện đối tượng được huấn luyện trước, bạn sẽ cần phải quyết định loại mạng phát hiện đối tượng nào bạn muốn sử dụng: mạng hai giai đoạn hoặc mạng một giai đoạn.

1. **Mạng hai giai đoạn (Two-Stage Networks)** Mạng hai giai đoạn là phương pháp được sử dụng rộng rãi nhất trong object detection. Mạng này thường có độ chính xác cao hơn so với mạng một giai đoạn, nhưng cũng chậm hơn.Một số mạng hai giai đoạn phổ biến bao gồm:

* R-CNN (Region-based Convolutional Neural Network): Mạng này là một trong những mạng hai giai đoạn đầu tiên được phát triển. Mạng này sử dụng thuật toán Selective Search để phát hiện các region proposal
* Fast R-CNN (Fast Region-based Convolutional Neural Network): Mạng này là một phiên bản cải tiến của R-CNN. Mạng này sử dụng thuật toán RoI Pooling để chuyển đổi các region proposal thành các kích thước cố định trước khi đưa vào mạng CNN.
* Faster R-CNN (Faster Region-based Convolutional Neural Network): Mạng này là một phiên bản cải tiến của Fast R-CNN. Mạng này sử dụng kiến trúc RPN (Region Proposal Network) để phát hiện các region proposal một cách hiệu quả hơn.

1. **Mạng một giai đoạn (single-Stage Networks)**

Mạng một giai đoạn là phương pháp mới hơn trong object detection. Mạng này có tốc độ nhanh hơn so với mạng hai giai đoạn, nhưng độ chính xác cũng thấp hơn một chút. Một số mạng một giai đoạn phổ biến bao gồm:

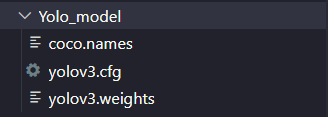
* YOLO (You Only Look Once): Mạng này sử dụng kiến trúc CNN để phát hiện các đối tượng và xác định vị trí của chúng trong một lần quét duy nhất.
* SSD (Single Shot MultiBox Detector): Mạng này sử dụng kiến trúc CNN để phát hiện các đối tượng và xác định vị trí của chúng trong nhiều kích thước khác nhau.
* RetinaNet (Focal Loss for Dense Object Detection): Mạng này sử dụng hàm mất Focal Loss để cải thiện độ chính xác của việc phát hiện các đối tượng nhỏ.
  1. **Thuật toán YOLO**

YOLO là viết tắt của "You Only Look Once", là một thuật toán phát hiện đối tượng end-to-end. Nó có nghĩa là mô hình chỉ cần nhìn vào hình ảnh một lần để phát hiện tất cả các đối tượng.

* 1. **Cách YOLO hoạt động**

1. **Chia hình ảnh thành ô:** Hình ảnh đầu vào chia thành lưới ô (grid). Mỗi ô sẽ chịu trách nhiệm dự đoán một hoặc nhiều bounding boxes và loại đối tượng tương ứng.
2. **Dự đoán bounding box:** Trong mỗi ô, YOLO dự đoán một hoặc nhiều bounding boxes. Mỗi bounding box được mô tả bởi 5 thông số: tọa độ (x, y) của góc trái trên, chiều rộng (w), chiều cao (h), và độ tin cậy của dự đoán.
3. **Phân loại đối tượng:** Ngoài các bounding box, YOLO còn dự đoán xác suất (confidence) của mỗi loại đối tượng trong bounding box. Nếu xác suất này vượt qua một ngưỡng nhất định, đối tượng sẽ được coi là đã được phát hiện.
4. **Tích hợp dự đoán:** Các dự đoán bounding box và phân loại được tích hợp lại để tạo ra kết quả cuối cùng. Mỗi bounding box sẽ đi kèm với xác suất phân loại và độ tin cậy của nó.
5. **Giảm thiểu hàm mất mát (Loss function):** YOLO sử dụng hàm mất mát đa nhiệm, bao gồm một thành phần liên quan đến dự đoán bounding box (localization loss) và một thành phần liên quan đến dự đoán loại đối tượng (classification loss). Mục tiêu là tối thiểu hóa tổng mất mát để cải thiện độ chính xác của mô hình.
6. **Non-maximum suppression (NMS):** Sau khi có được tất cả các dự đoán, YOLO sử dụng thuật toán Non-maximum Suppression để loại bỏ các bounding box trùng lặp và giữ lại những bounding box có độ tin cậy cao nhất.
7. **Đầu ra:** Kết quả cuối cùng là danh sách các bounding box với loại đối tượng tương ứng và xác suất của chúng.
8. **Thực nghiệm object detection với YOLO**

**3.1. YOLO model**

****

* 1. **Lập trình với YOLO model**

1. **Import các thư viện:**

import cv2

import numpy as np

import pandas as pd

import time

from datetime import datetime

import os

* *cv2: Thư viện OpenCV để xử lý video và hình ảnh.*
* *numpy: Thư viện xử lý mảng và ma trận.*
* *pandas: Thư viện để làm việc với DataFrame.*
* *time và datetime: Cung cấp các hàm liên quan đến thời gian.*
* *os: Thư viện hỗ trợ thao tác hệ điều hành.*

1. **Load pre-trained YOLO model và danh sách các lớp:**

net = cv2.dnn.readNet("./Yolo\_model/yolov3.weights", "./Yolo\_model/yolov3.cfg")

classes = []

with open("./Yolo\_model/coco.names", "r") as f:

    classes = [line.strip() for line in f.readlines()]

* ***net****: Mô hình YOLO được load từ các tệp tin trọng số (****yolov3.weights****) và cấu hình (****yolov3.cfg****).*
* ***classes****: Danh sách các lớp (đối tượng) mà mô hình YOLO có thể nhận diện.*

1. **Chọn các lớp đối tượng quan tâm:**

interested\_classes = ["person","bicycle"…,"car"]

* *Danh sách các lớp đối tượng quan tâm.*

1. **Lấy tên các lớp detection output từ mô hình:**

layer\_names = net.getUnconnectedOutLayersNames()

* *Các tên của các lớp output của mô hình YOLO.*

1. **Khởi tạo một số biến cần thiết:**

colors = np.random.uniform(0, 255, *size*=(len(classes), 3))

cap = cv2.VideoCapture(0)

moved\_objects = []

* ***colors****: Một mảng màu để đánh màu cho bounding boxes.*
* ***cap****: Đối tượng VideoCapture để quay video từ camera hoặc webcam.*
* ***moved\_objects****: Danh sách lưu trữ thông tin về các đối tượng đã di chuyển.*

1. **Tạo thư mục để lưu ảnh:**

# Create a folder to save images

image\_folder = "images"

os.makedirs(image\_folder, *exist\_ok*=True)

* *Tạo thư mục "images" để lưu các ảnh của đối tượng.*
  1. **Vòng lặp chính để thực hiện object detection:**

while True:

    ret, frame = cap.read()

    height, width, channels = frame.shape

    blob = cv2.dnn.blobFromImage(

        frame, 0.00392, (416, 416), (0, 0, 0), True, *crop*=False

    )

    net.setInput(blob)

    outs = net.forward(layer\_names)

* ***Đọc Frame từ Camera (cap.read()):******cap.read()*** *đọc một frame từ camera hoặc video capture.* ***ret*** *sẽ là* ***True*** *nếu frame được đọc thành công và* ***False*** *nếu không còn frame nào để đọc.*
* ***Chuẩn bị BLOB cho Mô hình YOLO (cv2.dnn.blobFromImage):*** *Hàm này chuyển đổi frame thành định dạng phù hợp để đưa vào mô hình YOLO.*
  + ***0.00392****: Tỉ lệ tỷ lệ pixel (được sử dụng để chuẩn hóa dữ liệu).*
  + ***(416, 416)****: Kích thước đầu vào mong muốn của mô hình YOLO.*
  + ***(0, 0, 0)****: Giá trị trung bình mỗi kênh màu.*
  + ***True****: Chỉ định liệu có cần crop ảnh hay không.*
* ***Đặt BLOB làm Input cho Mô hình (net.setInput(blob)):*** *BLOB được đặt làm đầu vào cho mô hình YOLO.*
* ***Forward Pass và Nhận Kết Quả (outs = net.forward(layer\_names)):*** *Thực hiện forward pass để nhận dự đoán từ mô hình YOLO. Kết quả được lưu trong biến* ***outs****.*
  + 1. **Xử Lý Kết Quả và Vẽ Bounding Boxes**

 for out in outs:

        for detection in out:

            scores = detection[5:]

            class\_id = np.argmax(scores)

            confidence = scores[class\_id]

            if confidence > 0.5:

                center\_x = int(detection[0] \* width)

                center\_y = int(detection[1] \* height)

                w = int(detection[2] \* width)

                h = int(detection[3] \* height)

                x = int(center\_x - w / 2)

                y = int(center\_y - h / 2)

                class\_ids.append(class\_id)

                confidences.append(float(confidence))

                boxes.append([x, y, w, h])

* *Vòng lặp này duyệt qua các dự đoán của mô hình YOLO trong từng ô của lưới.*
* *Dự đoán được chứa các thông tin như tọa độ của bounding box, xác suất của loại đối tượng.*
* *Nếu xác suất của loại đối tượng vượt qua một ngưỡng (ở đây là 0.5), thì bounding box sẽ được xác định và thông tin liên quan được lưu vào các danh sách* ***class\_ids****,* ***confidences****, và* ***boxes****.*
  + 1. **Ghi thông tin và ảnh của đối tượng đã di chuyển**

 indices = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

    if len(indices) > 0:

        for i in indices:

            box = boxes[i]

            x, y, w, h = box

            label = str(classes[class\_ids[i]])

            confidence = confidences[i]

            color = colors[class\_ids[i]]

            cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)

            cv2.putText(

                frame,

                f"{label} {confidence:.2f}",

                (x, y - 10),

                cv2.FONT\_HERSHEY\_SIMPLEX,

                0.5,

                color,

                2,

            )

            if label.lower() in interested\_classes:

                current\_time = time.time()

                if current\_time - last\_capture\_time >= capture\_interval:

                    timestamp = datetime.now().strftime("%Y%m%d%H%M%S")

                    image\_filename = f"{label}\_{timestamp}.png"

                    image\_path = os.path.join(image\_folder, image\_filename)

                    cv2.imwrite(image\_path, frame[y : y + h, x : x + w])

                    moved\_objects.append(

                        {

                            "Class": label,

                            "Confidence": confidence,

                            "StartX": x,

                            "StartY": y,

                            "EndX": x + w,

                            "EndY": y + h,

                            "Timestamp": timestamp,

                            "ImagePath": image\_path,

                        }

                    )

                    last\_capture\_time = current\_time

* *Sử dụng Non-Maximum Suppression (****cv2.dnn.NMSBoxes****) để giảm số lượng bounding boxes trùng lặp và chỉ giữ lại những bounding boxes có độ tin cậy cao nhất.*
* *Duyệt qua các bounding boxes còn lại và thực hiện các thao tác sau:*
  + *Vẽ bounding box lên frame.*
  + *Hiển thị label và xác suất trên bounding box.*
  + *Nếu loại đối tượng là một trong* ***interested\_classes*** *(các lớp đối tượng quan tâm), và đủ thời gian kể từ lần chụp trước đó, thì lưu thông tin về đối tượng vào danh sách* ***moved\_objects*** *và lưu ảnh của đối tượng vào thư mục* ***image\_folder****.*
  + *Cập nhật thời điểm lần chụp cuối cùng để đảm bảo rằng không chụp ảnh quá thường xuyên.*
    1. **Hiển thị frame:**

cv2.imshow("YOLO Object Detection", frame)

* *Hiển thị frame với các bounding boxes đã được vẽ lên.*
  + 1. **Ngắt quá trình khi nhấn 'q':**

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

        df = pd.DataFrame(moved\_objects)

        df.to\_csv("moved\_objects\_C.csv", *index*=False)

        break

* *Nếu nhấn phím 'q', kết thúc chương trình và lưu danh sách đối tượng đã di chuyển vào một file CSV.*
  + 1. **Giải phóng tài nguyên khi kết thúc:**

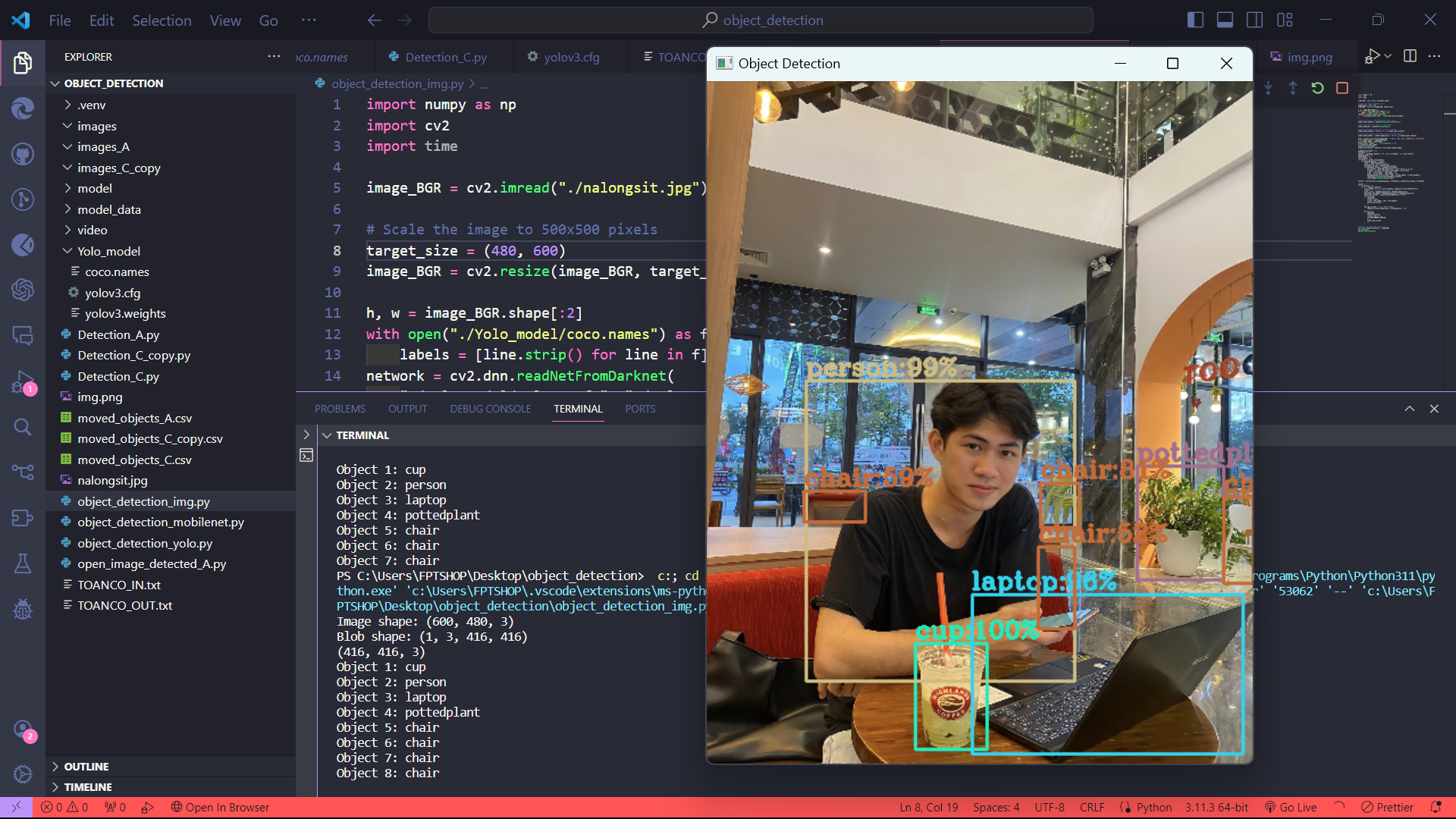
**cap.release()**

cv2.destroyAllWindows()

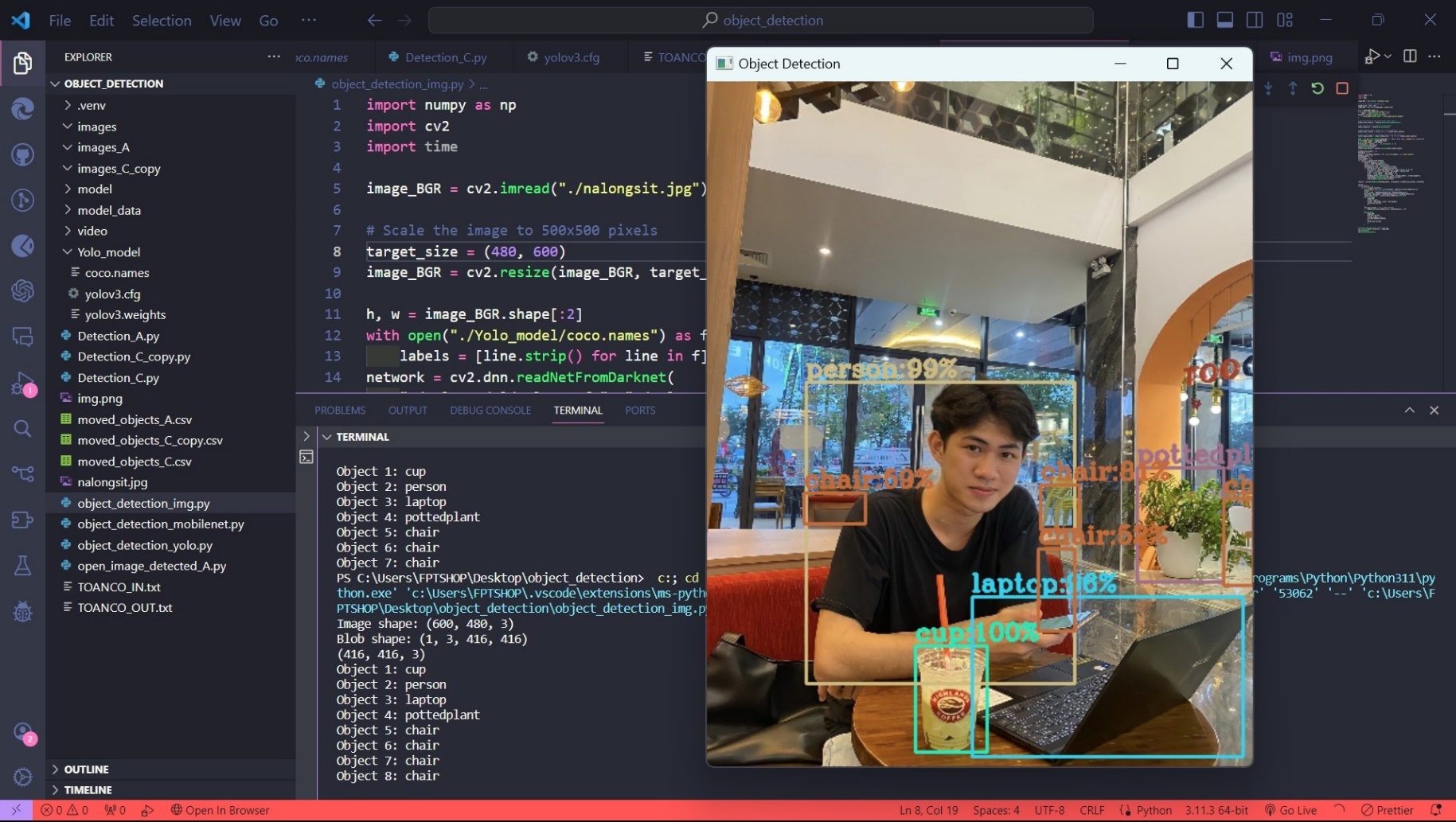
* *Giải phóng tài nguyên của camera và đóng các cửa sổ hiển thị.*

1. **kết quả thực nghiệm**

* **resualt of image**

A person sitting at a table with a computer and a drink

Description automatically generated



* **resualt of webcom**
* **lưu dữ liệu vào file.csv**

**A screenshot of a computer

Description automatically generated**

* **chụp một object trong 1 sec**

( có thể kiểm tra được theo tên của ảnh là cell phone 2023-12-04 16:56:35, cell phone 2023-12-04 16:56:36, 2023-12-04 16:56:37…)

**A collage of a person taking a selfie

Description automatically generated**

1. **Kết luận Object detection và Yolo**

Object detection và YOLO đều đóng góp lớn trong việc giải quyết vấn đề xác định và phân loại đối tượng trong ảnh và video. Trong khi object detection là một lĩnh vực tổng quát, YOLO đại diện cho một phương pháp cụ thể và hiệu quả trong thực tế, đặc biệt là khi cần xử lý dữ liệu thời gian thực. Sự tiếp tục nghiên cứu và phát triển trong cả hai lĩnh vực này sẽ đóng vai trò quan trọng trong tương lai của thị giác máy tính và trí tuệ nhân tạo.

1. **tài liệu tham khảo**

### <https://www.mathworks.com/discovery/object-detection.htm>

### <https://viblo.asia/p/tim-hieu-ve-yolo-trong-bai-toan-real-time-object-detection-yMnKMdvr57P>

### <https://github.com/pjreddie/darknet/blob/master/data/coco.names>

### <https://medium.com/@chamkung1412> ( còn nhiều bài để có thể tìm hiểu thêm trong link )