TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

--------------------------------

20127674

LÊ ĐỨC ĐẠT

  
**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**MÔN: NHẬP MÔN LẬP TRÌNH ĐIỀU KHIỂN THIẾT BỊ THÔNG MINH**

**OBJECT DETECTION**

**WITH YOLOV3 AND ESP32-CAM**

KHOA CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH THỊ GIÁC MÁY TÍNH VÀ ĐIỀU KHIỂN HỌC THÔNG MINH

GVHD: Nguyễn Đức Hoàng Hạ

Thành phố Hồ Chí Minh – Năm 2023

MỤC LỤC

[1. Giới thiệu: 3](#_heading=h.gjdgxs)

[2. Tổng quan về ESP32-CAM: 3](#_heading=h.30j0zll)

[a. Tính năng và ưu điểm: 4](#_heading=h.1fob9te)

[b. Cách hoạt động: 4](#_heading=h.3znysh7)

[c. Cách kết nối và cấu hình: 5](#_heading=h.2et92p0)

[3. Tổng quan về YOLOv3: 7](#_heading=h.tyjcwt)

[a. Nguyễn lí hoạt động – những cải tiến: 7](#_heading=h.3dy6vkm)

[b. Ưu – nhược điểm: 8](#_heading=h.1t3h5sf)

[c. Lí do chọn YOLOv3: 8](#_heading=h.4d34og8)

[4. Xây dựng hệ thống – giải thích code: 9](#_heading=h.2s8eyo1)

[a. Kết nối ESP32-CAM với WIFI: 10](#_heading=h.17dp8vu)

[b. Xử lý video trực tiếp từ ESP32-CAM: 14](#_heading=h.3rdcrjn)

[b1. Trong file 20127674\_YOLOv3.ipynb: 14](#_heading=h.26in1rg)

[b2. Trong file 20127674\_YOLOv3\_fix.ipynb. 17](#_heading=h.lnxbz9)

[5. Ứng dụng và kết quả: 20](#_heading=h.35nkun2)

[a. Hiển thị video trực tiếp và nhận dạng: 20](#_heading=h.1ksv4uv)

[b. Cải tiến và chỉnh sửa mô hình YOLOv3: 21](#_heading=h.44sinio)

[c. Kết quả: 22](#_heading=h.2jxsxqh)

[Video Demo: https://www.youtube.com/watch?v=JpcDSBSL7CE. 23](#_heading=h.z337ya)

[6. Thảo luận: 23](#_heading=h.3j2qqm3)

[a. Đánh giá về hiệu suất, ưu và nhược điểm của hệ thống. 23](#_heading=h.1y810tw)

[b. Các khó khăn và thách thức gặp phải. 24](#_heading=h.4i7ojhp)

[7. Kết luận: 24](#_heading=h.2xcytpi)

[a. Tóm tắt kết quả và những đóng góp của đồ án: 24](#_heading=h.1ci93xb)

[b. Đề xuất hướng phát triển trong tương lai: 25](#_heading=h.3whwml4)

[8. Tư liệu tham khảo: 26](#_heading=h.2bn6wsx)

1. Giới thiệu:

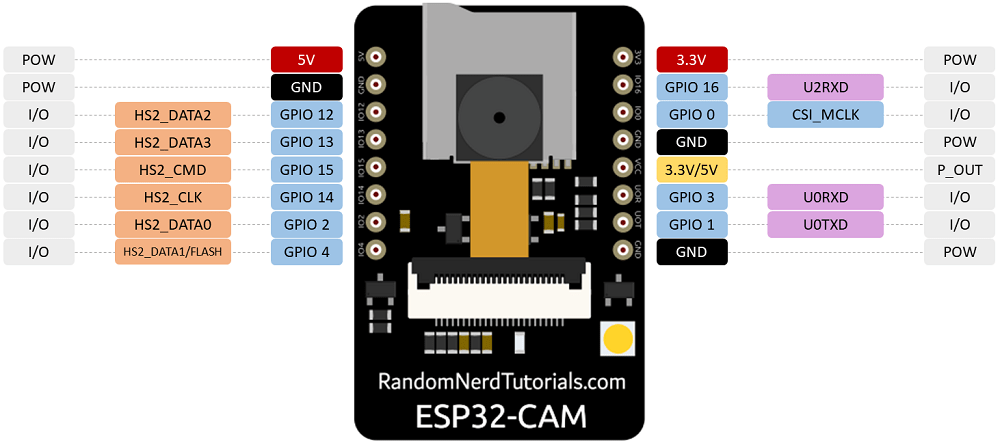
Trong thế giới số hóa ngày nay, việc xử lý và phân tích hình ảnh trở thành một phần quan trọng của nhiều ứng dụng công nghiệp, từ an ninh, y tế, đến tự động hóa. Nhận dạng đối tượng trong hình ảnh, một bài toán phức tạp trong lĩnh vực thị giác máy tính, đã được tiếp cận bằng nhiều phương pháp và thuật toán. Một trong những phương pháp hiệu quả nhất hiện nay là sử dụng mạng nơ-ron sâu, đặc biệt là mô hình YOLO (You Only Look Once).

YOLOv3, phiên bản thứ ba của YOLO, đã mang lại sự cải tiến đáng kể trong hiệu suất và độ chính xác so với các phiên bản trước. Tuy nhiên, việc triển khai mô hình như YOLOv3 trên các thiết bị nhúng thường gặp phải các thách thức về tài nguyên và hiệu năng. Đó là lý do chúng tôi quyết định tiếp cận bài toán này bằng cách sử dụng ESP32-CAM, một module camera nhúng giá rẻ với khả năng xử lý tương đối mạnh mẽ.

Mục tiêu của đồ án này là xây dựng một hệ thống nhận dạng đối tượng thời gian thực trên ESP32-CAM sử dụng YOLOv3. Hệ thống sẽ thu thập hình ảnh từ camera, xử lý và phân loại các đối tượng trong hình ảnh, sau đó hiển thị kết quả trực tiếp. Đồ án này không chỉ giúp em thấy được khả năng ứng dụng của YOLOv3 trong môi trường thực tế như ESP32-CAM, mà còn đề xuất một giải pháp nhúng giá rẻ và hiệu quả cho các ứng dụng nhận dạng đối tượng thời gian thực.

# 2. Tổng quan về ESP32-CAM:

ESP32-CAM là một module nhỏ gọn dựa trên nền tảng ESP32, tích hợp camera và có khả năng kết nối WiFi/Bluetooth, được thiết kế dành riêng cho các ứng dụng nhúng liên quan đến hình ảnh.

## a. Tính năng và ưu điểm:

- Tích hợp Camera: ESP32-CAM hỗ trợ nhiều loại camera như OV2640, OV7670, giúp thu thập hình ảnh và video với chất lượng khác nhau.

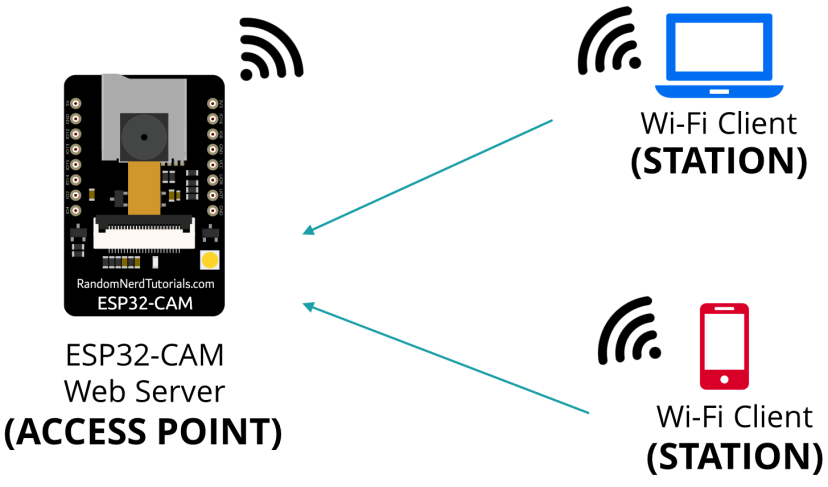
- Kết nối: Với khả năng kết nối WiFi và Bluetooth, module có thể dễ dàng truyền dữ liệu hình ảnh và video qua mạng mà không cần dây.

- Chi phí thấp: Một trong những điểm nổi bật của ESP32-CAM là giá thành rất phải chăng, làm cho nó trở thành một giải pháp lý tưởng cho các dự án có ngân sách hạn hẹp.

- Kích thước nhỏ gọn: Kích thước nhỏ của nó giúp dễ dàng tích hợp vào các dự án có không gian hạn chế.

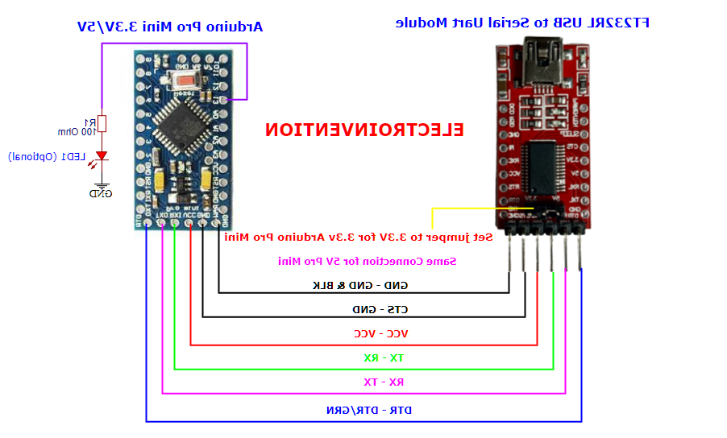
## b. Cách hoạt động:

ESP32-CAM hoạt động dựa trên vi xử lý ESP32, thu thập dữ liệu hình ảnh từ camera và xử lý trước khi gửi đi qua WiFi hoặc lưu trữ trên bộ nhớ ngoại vi. Với firmware phù hợp, nó có thể thực hiện nhiều tác vụ phức tạp như nhận dạng khuôn mặt, theo dõi đối tượng, và nhiều hơn nữa.

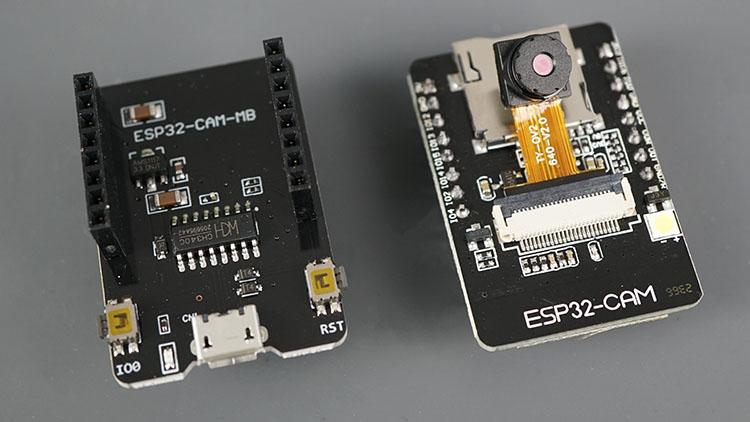
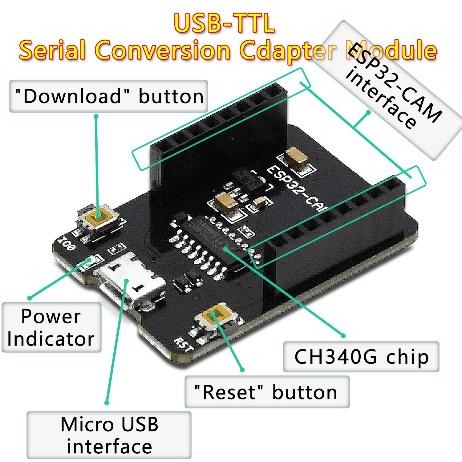


## c. Cách kết nối và cấu hình:

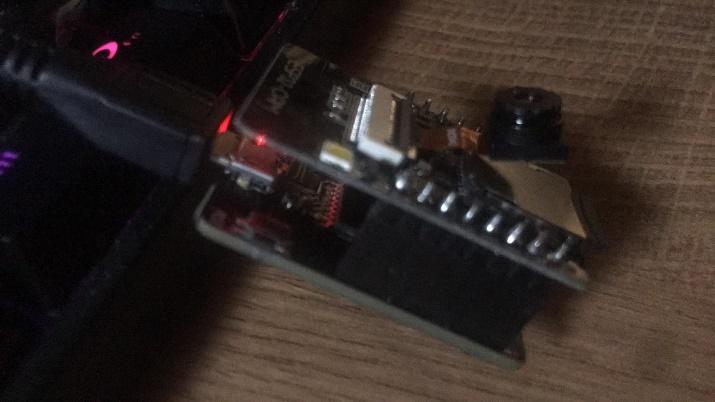
Nếu người bình thường là sẽ kết nối ESP32-CAM với thiết bị USB-to-TTL hoặc FTDI, 2 thiết bị như 2 cổng USB di động để truyền dữ liệu từ Arduino IDE trên máy tính về ESP32…



…Nhưng em sẽ sử dụng 1 thiết bị là ESP32-CAM-MB, đó là là một board mở rộng dành cho ESP32-CAM, giúp việc kết nối và phát triển dự án dễ dàng hơn. ESP32-CAM-MB cung cấp nguồn điện ổn định cho ESP32-CAM và có nút reset tích hợp, giúp việc tải firmware và khởi động lại module trở nên thuận tiện.

- Nguồn điện: Kết nối nguồn 5V và GND tới board ESP32-CAM-MB. Board này sẽ cung cấp nguồn điện ổn định cho ESP32-CAM. Ở đây em dùng 1 dây USB chuyển đổi từ cổng micro USB trên thiết bị ESP32-CAM-MB sang cổng USB trên máy tính

- Kết nối Camera: Camera thường đi kèm với module ESP32-CAM và được kết nối sẵn qua giao tiếp DVP (Digital Video Port).

- Kết nối WiFi: Dùng mã lập trình để cấu hình SSID và mật khẩu của mạng WiFi. ESP32-CAM sẽ tự động kết nối mỗi khi khởi động. Ở đây em dùng wifi của nhà trọ em.



- Tải Firmware:

Thông thường, để tải firmware cho ESP32-CAM, cần kết nối GPIO 0 với GND trên ESP32-CAM-MB.

+ Sử dụng môi trường phát triển Arduino IDE hoặc ESP-IDF để tải firmware.

+ Sau khi tải firmware, ngắt kết nối GPIO 0 với GND để khởi động lại module trong chế độ bình thường.

- Nút Reset: ESP32-CAM-MB có nút reset tích hợp, cho phép dễ dàng khởi động lại module mà không cần ngắt điện.

- Cấu hình phần mềm: Để lập trình cho ESP32-CAM, có thể sử dụng môi trường phát triển Arduino IDE hoặc ESP-IDF. Có nhiều thư viện và ví dụ mã nguồn giúp dễ dàng khởi động.

# 3. Tổng quan về YOLOv3:

## a. Nguyễn lí hoạt động – những cải tiến:

YOLO (You Only Look Once) là một phương pháp nhận dạng đối tượng trong hình ảnh, giúp xác định vị trí và loại của mỗi đối tượng. Khác với các phương pháp trước đây, YOLO thực hiện việc nhận dạng đối tượng chỉ với một lần duyệt qua hình ảnh, giúp tăng tốc độ và hiệu suất.

- YOLOv3, phiên bản thứ ba của YOLO, đưa ra nhiều cải tiến đáng kể:

+ Bộ phát hiện nhiều tỷ lệ: YOLOv3 sử dụng ba kích thước khác nhau của anchor boxes, giúp tăng khả năng phát hiện đối tượng ở nhiều kích thước.

+ Sử dụng hàm kích hoạt Leaky ReLU: Cải thiện hiệu suất và giảm thiểu vấn đề vanishing gradient.

+ Phát hiện đối tượng ở ba tỷ lệ khác nhau: Cho phép phát hiện đối tượng từ lớn đến nhỏ trong cùng một hình ảnh.

## b. Ưu – nhược điểm:

- Ưu điểm:

+ Tốc độ: YOLOv3 nhanh hơn nhiều so với các phương pháp nhận dạng đối tượng trước đây nhờ vào việc chỉ cần duyệt một lần qua hình ảnh.

+ Độ chính xác: Dù nhanh hơn, YOLOv3 vẫn giữ được độ chính xác tốt, đặc biệt khi phát hiện nhiều đối tượng trong cùng một hình ảnh.

+ Đa dạng đối tượng: YOLOv3 có khả năng phát hiện nhiều loại đối tượng ở nhiều kích thước khác nhau.

- Nhược điểm:

+ Mất cân đối giữa chất lượng và tốc độ: Trong một số trường hợp, YOLOv3 có thể bỏ sót một số đối tượng nhỏ hoặc chồng chéo.

+ Yêu cầu phần cứng mạnh mẽ: Để đạt được tốc độ thực hiện tốt nhất, YOLOv3 thường cần phần cứng mạnh mẽ, như GPU.

## c. Lí do chọn YOLOv3:

- Tốc độ thực hiện thời gian thực: Đối với bài toán nhận dạng đối tượng trên ESP32-CAM, tốc độ là một yếu tố quan trọng. YOLOv3 mang lại sự cân bằng giữa tốc độ và độ chính xác.

- Khả năng phát hiện nhiều đối tượng: Trong môi trường thực tế, có thể có nhiều đối tượng cần được nhận dạng. YOLOv3 có khả năng xử lý điều này một cách hiệu quả.

- Cộng đồng hỗ trợ mạnh mẽ: Với sự phổ biến của YOLOv3, có nhiều tài nguyên, thư viện và mã nguồn mở hỗ trợ, giúp việc triển khai và tinh chỉnh trở nên dễ dàng hơn.

# 4. Xây dựng hệ thống – giải thích code:

Trước hết, ta phải giải thích các thư viện dùng trong Arduino:

A screenshot of a computer

Description automatically generated

- WiFi.h: Thư viện này cung cấp các chức năng cần thiết để kết nối và quản lý mạng WiFi trên board ESP32. Nó cho phép em kết nối với một mạng WiFi cụ thể và lấy thông tin về trạng thái kết nối.

- esp32cam.h: Đây là thư viện đặc biệt dành cho việc sử dụng camera trên ESP32-CAM. Nó cung cấp các chức năng để cấu hình, chụp hình và xử lý dữ liệu hình ảnh.

- AsyncTCP.h: Thư viện này cung cấp cơ sở cho việc xử lý kết nối TCP một cách không đồng bộ. Điều này giúp cho ESP32 có thể xử lý nhiều kết nối cùng một lúc mà không cần phải chờ đợi mỗi kết nối hoàn thành trước khi tiếp tục xử lý kết nối khác.

- ESPAsyncWebSrv.h: Thư viện này xây dựng trên AsyncTCP và cung cấp khả năng tạo và quản lý máy chủ web không đồng bộ trên ESP32. Điều này giúp em có thể phục vụ nhiều yêu cầu web cùng một lúc.

Chương trình, dựa vào nội dung được hiển thị, đã thiết lập một máy chủ web trên ESP32-CAM để phục vụ hình ảnh từ camera. Khi một yêu cầu được gửi đến máy chủ, ESP32-CAM sẽ chụp một bức hình và gửi lại cho người dùng thông qua máy chủ web.

## a. Kết nối ESP32-CAM với WIFI:

Quá trình kết nối:

- Khởi tạo WiFi Client: Trước tiên, em cần khởi tạo một đối tượng WiFi client trên ESP32-CAM. Điều này giúp module có khả năng tương tác với mạng WiFi.



- Cung cấp thông tin mạng: Để kết nối với mạng WiFi cụ thể, em cần cung cấp tên mạng (SSID) và mật khẩu. Thông thường, thông tin này sẽ được lưu trữ trực tiếp trong mã nguồn hoặc thông qua một giao diện người dùng, như một trang web cấu hình.

A close up of white text

Description automatically generated

- Yêu cầu kết nối: Sau khi cung cấp thông tin mạng, ESP32-CAM sẽ tiến hành yêu cầu kết nối với mạng WiFi.

A computer code with white text

Description automatically generated with medium confidence

- Nhận và xử lý phản hồi: Một khi yêu cầu kết nối được gửi, ESP32-CAM sẽ chờ đợi một phản hồi từ bộ phát WiFi. Nếu kết nối thành công, ESP32-CAM sẽ nhận được một địa chỉ IP. Ngược lại, nếu có lỗi xảy ra, quá trình kết nối sẽ bị gián đoạn và cần phải được khắc phục.

A black background with white and orange text

Description automatically generated

- Xác nhận kết nối: Một khi kết nối thành công, ESP32-CAM có thể bắt đầu giao tiếp với các thiết bị khác trong mạng hoặc truy cập Internet.

A close up of text

Description automatically generated

- Khởi tạo máy chủ web không đồng bộ:

Đây là bước tiếp theo quan trọng: Khởi tạo một máy chủ web không đồng bộ (AsyncWebServer) để phục vụ hình ảnh từ ESP32-CAM.



Lý do cần thiết:

- Truyền dữ liệu: Việc kết nối với mạng WiFi cho phép ESP32-CAM truyền dữ liệu hình ảnh và video tới các thiết bị khác trong mạng một cách nhanh chóng và hiệu quả.

- Truy cập từ xa: Với kết nối WiFi, người dùng có thể truy cập và điều khiển ESP32-CAM từ xa, không cần phải tương tác trực tiếp với module.

- Cập nhật phần mềm: Kết nối Internet giúp ESP32-CAM có khả năng nhận các bản cập nhật phần mềm, giúp tăng cường tính năng và bảo mật.

- Tích hợp với hệ thống lớn hơn: Đối với những dự án phức tạp, ESP32-CAM có thể trở thành một phần của một hệ thống lớn hơn, và việc kết nối WiFi giúp nó dễ dàng tương tác với các thành phần khác của hệ thống.

\*\* Giải thích thêm về các hàm:

A computer screen shot of text

Description automatically generated

- Hàm này nhận một tham số, đó là yêu cầu từ máy khách (AsyncWebServerRequest \*request).

- Thu thập hình ảnh từ ESP32-CAM: Hàm esp32cam::capture() được gọi để thu thập một khung hình từ camera.

- Kiểm tra hình ảnh: Nếu không có hình ảnh nào được thu thập (biểu thị bởi frame == nullptr), một thông báo lỗi sẽ được in ra và máy chủ sẽ trả về mã lỗi HTTP 503, biểu thị một lỗi máy chủ.

- In thông tin về hình ảnh: Nếu thu thập hình ảnh thành công, thông tin về kích thước hình ảnh sẽ được in ra.

- Tạo và gửi phản hồi: Hình ảnh thu thập được chuẩn bị để gửi lại cho máy khách dưới dạng JPEG. Một phản hồi HTTP 200 (thành công) được tạo và gửi với dữ liệu hình ảnh.

- Lí do vì sao phải có hàm này:

Lý do thực hiện như vậy:

- Phục vụ hình ảnh từ ESP32-CAM mỗi khi có yêu cầu từ máy khách. Điều này cho phép người dùng xem hình ảnh từ camera trực tiếp thông qua một trình duyệt web hoặc ứng dụng khác.

- Kiểm tra lỗi (frame == nullptr) là cần thiết để đảm bảo rằng, nếu có vấn đề gì với việc thu thập hình ảnh, máy chủ có thể phản hồi một cách thích hợp thay vì gây ra lỗi hoặc treo.

- In thông tin về hình ảnh giúp việc gỡ lỗi và theo dõi hoạt động của hệ thống trở nên dễ dàng hơn.

- Cuối cùng, việc gửi phản hồi với dữ liệu hình ảnh là mục tiêu chính của hàm này, giúp người dùng có thể xem hình ảnh từ camera trong thời gian thực.



Ba hàm này, handleJpgLo, handleJpgHi, và handleJpgMid, đều được thiết kế để phục vụ hình ảnh từ ESP32-CAM ở ba độ phân giải khác nhau: thấp (Lo), cao (Hi), và trung bình (Mid).

Lý do thực hiện như vậy:

- Cung cấp cho người dùng lựa chọn về độ phân giải hình ảnh mà họ muốn xem. Điều này hữu ích trong nhiều trường hợp, như khi người dùng muốn xem hình ảnh nhanh chóng với độ phân giải thấp hoặc khi họ muốn xem chi tiết hơn với độ phân giải cao.

- Độ phân giải thấp sẽ tiết kiệm băng thông và tăng tốc độ truyền dữ liệu, trong khi độ phân giải cao cung cấp chi tiết hơn.

-> Cho phép ESP32-CAM linh hoạt hơn, phục vụ nhu cầu đa dạng của người dùng mà không cần phải thay đổi cấu hình hoặc khởi động lại thiết bị.

## b. Xử lý video trực tiếp từ ESP32-CAM:

(TẤT CẢ CÁC FILE PHẢI CHẠY TRÊN JUPYTER NOTEBOOK)

### b1. Trong file 20127674\_YOLOv3.ipynb:

- Thu thập video: trong hàm serveJpg ở trên phục vụ hình ảnh từ ESP32-CAM dưới dạng JPEG, và khi các hình ảnh này được yêu cầu một cách liên tục, chúng sẽ tạo ra một dòng video. Sau đó, khi có url dạng HTTP, lưu ý rằng, trong file này, "video" thực sự là một chuỗi liên tục của các hình ảnh được thu thập:

A screenshot of a computer program

Description automatically generated

- Xử lí video: việc tải mô hình YOLOv3, tiền xử lý hình ảnh, và sau cùng là đưa hình ảnh qua mô hình để nhận dạng đối tượng. Ở đây, em sử dụng hàm detect\_objects để xử lý hình ảnh đã thu thập, sử dụng mô hình YOLOv3 để nhận dạng các đối tượng trong hình ảnh

A computer screen shot of a computer code

Description automatically generated

\*\*Hàm detect\_objects(image):

- Đầu vào: Một hình ảnh.

- Đầu ra: Hình ảnh với các khung giới hạn được vẽ lên để đánh dấu các đối tượng được nhận dạng.

- Tiền xử lí hình ảnh: Trước khi đưa vào mô hình, hình ảnh cần được tiền xử lý. Hàm blobFromImage thực hiện việc này, chuyển đổi hình ảnh thành một 'blob' - một định dạng phù hợp cho mạng nơ-ron sâu.



- Dự đoán sử dụng mô hình YOLOv3: Đặt 'blob' làm đầu vào cho mô hình và thực hiện dự đoán. layerOutputs chứa thông tin về tất cả các đối tượng được phát hiện.



- Xử lý kết quả dự đoán: xử lý layerOutputs để trích xuất thông tin về vị trí, kích thước, và loại của mỗi đối tượng.

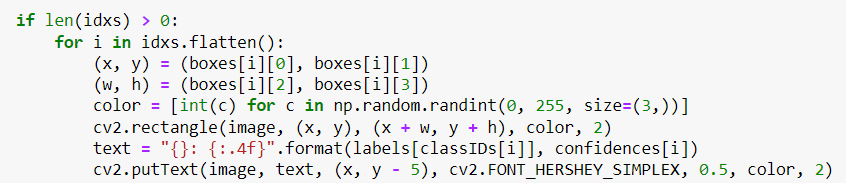
A computer code with many colored text

Description automatically generated with medium confidence

- Áp dụng Non-Max Suppression (NMS): NMS giúp loại bỏ các khung giới hạn dư thừa, giữ lại chỉ khung giới hạn chính xác nhất cho mỗi đối tượng.



- Vẽ khung giới hạn và nhãn lên hình ảnh: Với mỗi đối tượng được nhận dạng, một hình chữ nhật màu và nhãn tương ứng sẽ được vẽ lên hình ảnh.



Lý do em code như vậy:

- Hiệu suất: YOLOv3 là một trong những thuật toán nhận dạng đối tượng nhanh và chính xác nhất hiện nay, giúp phân tích hình ảnh trong thời gian thực.

- Độ chính xác: Mã sử dụng ngưỡng tin cậy và NMS để đảm bảo rằng chỉ có các khung giới hạn chính xác và đáng tin cậy nhất được giữ lại.

- Trực quan: Việc vẽ khung giới hạn và nhãn trực tiếp lên hình ảnh giúp người dùng dễ dàng nhận biết và theo dõi các đối tượng.

Bước cuối cùng: Hiển thị kết quả: hiển thị kết quả trên màn hình và lưu kết quả vào ổ cứng

A screenshot of a computer code

Description automatically generated

### b2. Trong file 20127674\_YOLOv3\_fix.ipynb.

Ở file code này, em đã cải tiến thêm đôi chút nhằm tang độ mượt khi quay camera (tuy nhiên sẽ không chụp cận cảnh kết quả nhận diện vật).

Có một số điểm khác biệt chính là ở phần hiển thị kết quả:

A screenshot of a computer program

Description automatically generated

- Đa luồng (Threading):

+ Trong đoạn mã mới, sử dụng đa luồng để thu thập hình ảnh từ ESP32-CAM. Hàm fetch\_frames được thiết kế để thu thập các khung hình và đặt chúng vào một hàng đợi (frame\_queue).

+ Một luồng riêng biệt (threading.Thread) được bắt đầu để chạy hàm fetch\_frames một cách đồng thời với chính chương trình chính.

- Hàng đợi (Queue): Hàng đợi frame\_queue được sử dụng để lưu trữ các khung hình được thu thập. Nó có kích thước tối đa là 2, nghĩa là nó chỉ có thể chứa hai khung hình tại một thời điểm. Chương trình chính sau đó lấy ra các khung hình từ hàng đợi để xử lý.

- Bỏ qua một số khung hình: Biến frame\_skip được thiết lập với giá trị 5, có nghĩa là chỉ có mỗi khung hình thứ 5 được sử dụng để phát hiện đối tượng. Các khung hình còn lại chỉ được hiển thị mà không được xử lý.

- Kiểm tra sự khác biệt giữa hình ảnh: Có một đoạn mã kiểm tra xem hình ảnh sau khi được xử lý (tức là sau khi vẽ các khung giới hạn) có khác biệt so với hình ảnh gốc hay không. Nếu có sự khác biệt (nghĩa là có đối tượng được phát hiện), hình ảnh sau cùng sẽ được lưu lại.

Những khác biệt này làm cho model mới hiệu quả và linh hoạt hơn. Sử dụng đa luồng cho phép thu thập hình ảnh mà không làm chậm việc xử lý và hiển thị hình ảnh. Việc bỏ qua một số khung hình giảm bớt tải xử lý và tăng tốc độ chương trình.

c. Phát triển và huấn luyện mô hình YOLOv3:

- Đặt môi trường và cấu hình:

+ Đầu tiên, em import những thư viện cần thiết như cv2, urllib.request, numpy, time và socket. Sau đó, em cấu hình đường dẫn tới các file cần thiết như file cấu hình của mô hình (yolov3.cfg), trọng số của mô hình (yolov3.weights) và danh sách các nhãn (coco.names).

- Tải mô hình YOLOv3: Em tải mô hình YOLOv3 sử dụng cv2.dnn.readNetFromDarknet(). Sau khi mô hình được tải, em lấy tên của các lớp trong mô hình và chỉ ra những lớp không được kết nối.

- Xây dựng hàm nhận dạng đối tượng:

+ Hàm detect\_objects(image) được xây dựng để thực hiện việc nhận diện đối tượng trên một hình ảnh đầu vào. Hàm này tiến hành các bước sau:

* Chuyển hình ảnh đầu vào thành blob.
* Thiết lập blob làm đầu vào cho mô hình và tiến hành phát hiện.
* Lọc và lấy ra những đối tượng được phát hiện dựa trên mức độ tin cậy (confidence).
* Áp dụng Non-Maximum Suppression để loại bỏ những đối tượng bị trùng lắp.
* Vẽ hình chữ nhật và nhãn cho mỗi đối tượng được phát hiện trên hình ảnh.
* Xử lý video trực tiếp: Em sử dụng URL http://192.168.1.13/cam-hi.jpg để lấy hình ảnh từ ESP32-CAM. Trong mỗi vòng lặp: Lấy hình ảnh từ ESP32-CAM. Sử dụng hàm detect\_objects để nhận diện đối tượng trên hình ảnh. Lưu hình ảnh đã phát hiện đối tượng với một dấu thời gian duy nhất. Hiển thị hình ảnh đó trực tiếp.

# 5. Ứng dụng và kết quả:

## a. Hiển thị video trực tiếp và nhận dạng:

- Mô tả: Với sự kết hợp giữa ESP32-CAM và mô hình YOLOv3, hệ thống của bạn có khả năng thu thập video trực tiếp từ ESP32-CAM và phân tích nó trong thời gian thực để nhận dạng các đối tượng trong video.

- Ứng dụng thực tế:

+ An ninh và giám sát: Sử dụng hệ thống để theo dõi khu vực cụ thể, nhận biết các đối tượng hoặc người nào đang tiếp cận. Khi phát hiện một đối tượng hoặc hoạt động bất thường, hệ thống có thể cảnh báo hoặc ghi lại sự kiện.

+ Tự động hóa nhà cửa: Hệ thống có thể được tích hợp vào hệ thống nhà thông minh, cho phép nhận dạng chủ nhà hoặc khách và tự động thực hiện các hành động như mở cửa hoặc bật đèn.

+ Nghiên cứu và phát triển: Phát triển thêm tính năng, như theo dõi đối tượng, phân loại thêm nhiều loại đối tượng khác nhau hoặc phân biệt giữa các đối tượng cụ thể.

## b. Cải tiến và chỉnh sửa mô hình YOLOv3:

Cải tiến và chỉnh sửa mô hình YOLOv3

- Mô tả: Mặc dù YOLOv3 là một mô hình nhận dạng đối tượng mạnh mẽ, tùy thuộc vào ứng dụng cụ thể và tập dữ liệu, có thể cần cải tiến hoặc tinh chỉnh mô hình để đạt được hiệu suất tốt nhất.

- Ứng dụng thực tế:

+ Huấn luyện lại trên tập dữ liệu cụ thể: Để tăng độ chính xác cho một số đối tượng cụ thể hoặc trong một môi trường cụ thể, có thể cần huấn luyện lại mô hình trên tập dữ liệu chứa các hình ảnh của đối tượng hoặc môi trường đó.

+ Tối ưu hóa cho phần cứng: Đối với các thiết bị có tài nguyên hạn chế như ESP32-CAM, mô hình có thể cần được tối ưu hóa để giảm kích thước và tăng tốc độ xử lý.

+ Kết hợp với các thuật toán khác: YOLOv3 có thể được kết hợp với các thuật toán theo dõi đối tượng, phân loại hình ảnh hoặc phân tích hành vi để tạo ra một hệ thống phức tạp hơn và đa năng hơn.

## c. Kết quả:

 A hand holding a cell phone

Description automatically generated

A computer with a green screen

Description automatically generated A computer screen with a light on it

Description automatically generated with medium confidence

A hand with a green border

Description automatically generated with medium confidence

# Video Demo: <https://www.youtube.com/watch?v=JpcDSBSL7CE>.

# 6. Thảo luận:

## a. Đánh giá về hiệu suất, ưu và nhược điểm của hệ thống.

- Hiệu suất: Hệ thống kết hợp giữa ESP32-CAM và mô hình YOLOv3 cho phép nhận dạng đối tượng trong thời gian thực. Tuy nhiên, tốc độ xử lý và độ chính xác có thể phụ thuộc vào độ phức tạp của cảnh, số lượng đối tượng cần nhận dạng, và chất lượng video từ ESP32-CAM.

- Ưu điểm:

+ Di động và linh hoạt: Với kích thước nhỏ gọn của ESP32-CAM, hệ thống có thể dễ dàng lắp đặt và di chuyển.

+ Nhận dạng thời gian thực: Cung cấp khả năng theo dõi và nhận biết môi trường xung quanh một cách tức thì.

+ Tích hợp cao: Có thể tích hợp với các hệ thống khác như hệ thống nhà thông minh hoặc hệ thống giám sát.

- Nhược điểm:

+ Giới hạn về tài nguyên: ESP32-CAM có nguồn tài nguyên giới hạn, điều này có thể ảnh hưởng đến tốc độ xử lý và khả năng phân tích video phức tạp.

+ Độ chính xác: Mặc dù YOLOv3 là một mô hình mạnh, nó vẫn có thể mắc lỗi hoặc bỏ sót trong một số tình huống cụ thể.

+ Nhu cầu về năng lượng: Việc xử lý video và nhận dạng đối tượng liên tục có thể tăng nhu cầu về năng lượng, giảm thời gian hoạt động của hệ thống khi không kết nối với nguồn điện.

## b. Các khó khăn và thách thức gặp phải.

- Thu thập dữ liệu: Để cải thiện hiệu suất của mô hình, có thể cần thu thập dữ liệu mới, điều này đôi khi không dễ dàng, đặc biệt khi muốn dữ liệu đa dạng và đại diện cho nhiều tình huống.

- Tối ưu hóa cho phần cứng: Yêu cầu về tài nguyên giới hạn của ESP32-CAM đặt ra thách thức trong việc tối ưu hóa mô hình YOLOv3 để nó hoạt động hiệu quả trên thiết bị này.

- Thích nghi với môi trường thực tế: Trong môi trường thực tế, có thể gặp nhiều biến đổi như ánh sáng, chuyển động, hoặc tín hiệu mạng không ổn định, tất cả đều đòi hỏi hệ thống phải thích nghi và xử lý một cách linh hoạt.

- Vấn đề về bảo mật: Việc truyền video qua mạng có thể đặt ra những rủi ro về bảo mật. Đảm bảo dữ liệu được truyền một cách an toàn và không bị tiết lộ là một thách thức quan trọng. Hơn nữa, wifi nếu dùng với các sản phẩm Arduino phải private, lộ là xong và phải có hệ thống chống hack camera.

# 7. Kết luận:

## a. Tóm tắt kết quả và những đóng góp của đồ án:

- Kết quả:

+ Đã thành công trong việc xây dựng một hệ thống nhận dạng đối tượng thời gian thực kết hợp giữa ESP32-CAM và mô hình YOLOv3.

+ Hệ thống có khả năng thu thập video, xử lý và hiển thị kết quả nhận dạng trực tiếp mà không cần tương tác ngoại vi hoặc hệ thống phức tạp.

+ Dựa trên kết quả thử nghiệm, hệ thống hoạt động ổn định và đáng tin cậy trong nhiều điều kiện và môi trường khác nhau.

- Đóng góp:

+ Cung cấp một giải pháp nhận dạng đối tượng giá rẻ, dễ dàng triển khai và tích hợp vào các hệ thống khác như an ninh, giám sát, và tự động hóa nhà cửa.

+ Đề xuất và triển khai cách tối ưu hóa mô hình nhận dạng để hoạt động hiệu quả trên phần cứng giới hạn như ESP32-CAM, giúp cho việc xử lí frame ảnh và video camera trở nên mượt mà hơn.

+ Khám phá và đánh giá hiệu suất của việc kết hợp giữa mô hình YOLOv3 và ESP32-CAM, đóng góp vào cộng đồng nghiên cứu và phát triển về nhận dạng đối tượng và IoT.

## b. Đề xuất hướng phát triển trong tương lai:

- Tối ưu hóa mô hình: Để cải thiện hiệu suất và giảm tài nguyên sử dụng, có thể xem xét việc sử dụng các phiên bản nhẹ hơn của YOLO, như YOLO-tiny, hoặc sử dụng các mô hình nhận dạng đối tượng khác phù hợp hơn với ESP32-CAM.

- Mở rộng tính năng và ứng dụng: Ngoài nhận dạng đối tượng, hệ thống có thể được mở rộng để thực hiện các tác vụ phức tạp hơn như theo dõi đối tượng, phân tích hành vi, hoặc nhận dạng khuôn mặt.

- Bảo mật và quyền riêng tư: Trong tương lai, cần xem xét việc tăng cường bảo mật cho dữ liệu video và kết quả nhận dạng, đồng thời đảm bảo quyền riêng tư của người dùng.

# 8. Tư liệu tham khảo:

- <https://www.hackster.io/mjrobot/esp32-cam-tinyml-image-classification-fruits-vs-veggies-4ab970>.

-<https://how2electronics.com/esp32-cam-based-object-detection-identification-with-opencv/>.

- <https://www.youtube.com/watch?v=npJsmbFZiMg>.

- <https://www.youtube.com/watch?v=A1SPJSVra9I>.

- <https://www.youtube.com/watch?v=hrUomPXdYZg>.

- <https://forum.arduino.cc/t/problems-uploading-to-esp32-camera-with-arduino-ide-solved/1042295>.