



TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO

PBL 4 - DỰ ÁN HỆ THỐNG THÔNG MINH

ĐỀ TÀI : DRONE BÁO CHÁY RỪNG

GIẢNG VIÊN HƯỚNG DẪN: TS. Ninh Khánh Duy.

SINH VIÊN THỰC HIỆN:

Lưu Võ Hoàng	Lớp: 22T_Nhat1	MSV: 102220275
Nguyễn Đình Huy	Lớp: 22T_Nhat1	MSV: 102220278
Phan Thanh Kiệt	Lớp: 22T_Nhat1	MSV: 102220294
Nguyễn Văn Hòa	Lớp : 22T_Nhat1	MSV: 102220273

Đà Nẵng, tháng 12 năm 2024.

TÓM TẮT ĐỒ ÁN

Drone báo cháy rừng là một hệ thống sử dụng công nghệ drone để phát hiện và cảnh báo sớm các vụ cháy rừng. Với khả năng bay cao và quan sát từ xa, drone có thể giám sát diện tích rừng rộng lớn mà không gặp phải những hạn chế về địa hình hoặc môi trường. Drone được trang bị công nghệ AI giúp phát hiện sớm các dấu hiệu cháy, như lửa hoặc khói. Khi phát hiện cháy, drone sẽ gửi tín hiệu cảnh báo, giúp họ nhanh chóng xác định vị trí và mức độ nguy hiểm của đám cháy.

Nhờ vào việc cung cấp dữ liệu hình ảnh, video chất lượng cao và phân tích chính xác tình hình, drone hỗ trợ công tác ra quyết định kịp thời, từ đó giảm thiểu thiệt hại và tăng cường hiệu quả trong việc ứng phó với cháy rừng. Ngoài ra, việc sử dụng drone còn giúp giảm rủi ro cho nhân viên cứu hỏa, bởi drone có thể tiếp cận những khu vực nguy hiểm mà con người khó có thể đến gần.

Tóm lại, việc sử dụng drone trong công tác báo cháy rừng không chỉ giúp phát hiện cháy nhanh chóng mà còn tối ưu hóa các biện pháp ứng phó, bảo vệ môi trường và cộng đồng một cách hiệu quả hơn.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tự đánh giá
Nguyễn Đình Huy	Lắp ráp và thiết lập các module drone Test và tuning PID Lập trình PS2	Đã hoàn thành
Lưu Võ Hoàng	Lắp ráp và xây dựng Controller PID Test và tuning PID Lập trình PS2	Đã hoàn thành
Phan Thanh Kiệt	Thực hiện model AI và xây dựng Search API Thiết kế giao diện, xây dựng website	Đã hoàn thành
Nguyễn Văn Hòa	Thực hiện model AI và xây dựng Search API Thiết kế giao diện, xây dựng website	Đã hoàn thành

MỤC LỤC

1. Giới Thiệu	6
1.1 Hiện trạng và tính cấp thiết của đề tài	6
1.2 Vấn đề cần giải quyết	6
2. Giải Pháp	7
2.1. Tổng quan	7
2.1.1. Đề xuất giải pháp tổng quan	7
2.1.2. Sơ đồ hoạt động tổng quan	7
2.2 Giải Pháp thuật toán :	8
2.2.1. Bài toán nhận diện đối tượng (Object Detection)	8
2.2.2. Các thuật toán chính trong YOLOv8 và Các thuật toán hỗ trợ	9
2.2.3. Các thuật toán hỗ trợ trong YOLOv8	10
2.2.4. Lựa chọn mô hình cho giải pháp bài toán	11
2.3. Giải pháp về phần cứng	12
2.3.1. Các linh kiện phần cứng	12
2.3.2. Sơ đồ lắp mạch	17
2.3.3. Lý thuyết điều khiển Quadcopter	17
2.3.4. Lắp đặt mạch in	19
2.3.5. Thuật toán PID	20
2.4. Giải pháp truyền thông và hệ thống	20
2.4.1. Giao thức NRF24L01 trên tay cầm PS2	20
2.4.2. Giao thức NRF24L01 trên drone	21
2.4.3. Giao thức UART của ESP32-CAM sử dụng:	22
2.4.4 Sơ đồ IoT	23
2.4.5 Giao thức truyền từ AI server đến Database	23
2.4.6 Các giao thức truyền từ Database lên Web	24
2.5. Giới thiệu phần mềm	25
2.5.1 AI server	25
2.5.2 Web server	25
2.5.3 Sơ đồ minh họa luồng dữ liệu	26
3. Kết quả	28
3.1 Thu thập và xử lý dữ liệu:	28
3.1.1 Bài toán thu thập dữ liệu:	28
3.1.2. Xử lý dữ liệu:	29
3.2. Công cụ và framework	29

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

3.2.1 Môi trường huấn luyện	29
3.2.2 Framework	30
3.2.3. Tham số huấn luyện	30
3.3. Kết quả quá trình huấn luyện	31
3.4. Quy trình thực nghiệm và kiểm thử	31
3.5. Kết quả trên tập kiểm thử	32
3.6. Giao diện ứng dụng	34
3.6.1. Giao diện chính	34
3.6.2. Giao diện xem lại	35
3.6.3. Giao diện xem kết quả	36
4. Kết luận và hướng phát triển	37
4.1 Kết luận	37
4.2 Hướng phát triển	37
5. Tài liệu tham khảo	39

Danh mục hình ảnh và bảng biểu

Hình 1. Sơ đồ hoạt động tổng quan	8
Hình 2. Các Linh kiện phần cứng	17
Hình 3. Sơ đồ lắp mạch	17
Hình 4. Cách hoạt động của Quadcopter	18
Hình 5. Cách hoạt động của Quadcopter	18
Hình 6. Cách điều khiển Quadcopter	19
Hình 7. Sơ đồ Lắp đặt mạch in	19
Hình 8. Sơ đồ IoT	23
Hình 9 . Minh họa luồng dữ liệu	26
Hình 10 . Biểu đồ thể hiện tỉ lệ tỷ lệ phần trăm của các loại dữ liệu	28
Hình 11. Training Box loss theo epoch	31
Hình 12. Validate Box Loss theo epoch	31
Hình 13. Các thông số chung của mô hình	33
Hình 14. Ảnh không có lửa, khói trước khi kiểm thử	33
Hình 15. Ảnh không có lửa, khói sau khi kiểm thử	33
Hình 16. Ảnh chỉ có lửa trước khi kiểm thử	34
Hình 17. Ảnh chỉ có lửa sau khi kiểm thử	34
Hình 18. Ảnh chỉ có khói trước khi kiểm thử	34
Hình 19. Ảnh chỉ có khói sau khi kiểm thử	34
Hình 20. Ảnh có cả lửa và khói trước khi kiểm thử	34
Hình 21. Ảnh có cả lửa và khói sau khi kiểm thử	34
Hình 22. Giao diện chính	35
Hình 23. Giao diện xem lại	35
Hình 24. Giao diện kết quả	36

1. Giới Thiệu

1.1 Hiện trạng và tính cấp thiết của đề tài

Cháy rừng hiện đang là một trong những thảm họa môi trường lớn trên thế giới, đặc biệt là tại các quốc gia có diện tích rừng lớn như Úc, Mỹ, Brazil và Việt Nam. Theo báo cáo của Tổ chức Lương thực và Nông nghiệp Liên Hợp Quốc (FAO), mỗi năm trên toàn cầu có hàng triệu hecta rừng bị thiêu rụi do cháy rừng, gây thiệt hại nặng nề về môi trường, kinh tế và đời sống con người. Tại Việt Nam, do ảnh hưởng của biến đổi khí hậu và thời tiết khắc nghiệt, nguy cơ cháy rừng đang ngày càng gia tăng, đặc biệt là vào mùa khô.

Hiện nay, công tác phòng chống cháy rừng vẫn còn gặp nhiều thách thức, đặc biệt là ở các khu vực đồi núi hiểm trở, khó tiếp cận. Các phương pháp truyền thống như quan sát thủ công từ các chòi canh hoặc sử dụng máy bay trực thăng giám sát có chi phí cao và hiệu quả không ổn định. Vì vậy, việc nghiên cứu và phát triển hệ thống tự động sử dụng drone để phát hiện cháy rừng sớm đã trở thành xu hướng mới và cần thiết.

1.2 Vấn đề cần giải quyết

Với thực trạng và nhu cầu cấp thiết nêu trên, đề tài này tập trung giải quyết các vấn đề sau:

- + Làm thế nào để phát hiện sớm các đám cháy trong rừng một cách chính xác và hiệu quả, đặc biệt ở những khu vực địa hình phức tạp?
- + Làm sao để hệ thống giám sát có thể hoạt động liên tục và cảnh báo kịp thời đến các cơ quan chức năng?
- + Tối ưu hóa chi phí vận hành và bảo trì so với các phương pháp truyền thống.

2. Giải Pháp

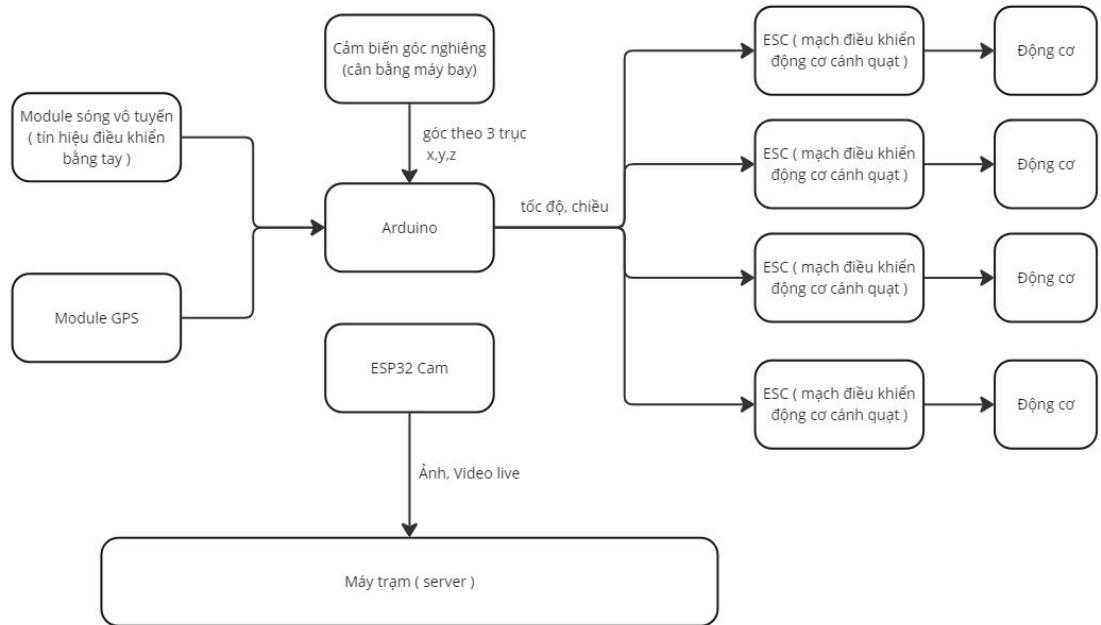
2.1. Tổng quan

2.1.1. Đề xuất giải pháp tổng quan

Vấn đề	Đề xuất giải pháp
Phần Cứng	<ul style="list-style-type: none">-Arduino Nano-Cảm biến : GPS, MPU6050-Giao thức : NRF24l01-Động cơ : 4 ESC - 4 Brushless Motor-Camera: ESP32-Pin lipo 11.1V
Nhận Diện	<ul style="list-style-type: none">-Khảo sát và lựa chọn mô hình thích hợp-Tận dụng mô hình trên Internet và tạo bộ nhớ cho dữ liệu-Huấn luyện trên Google Colab
Truyền Thông	<ul style="list-style-type: none">-Các giao thức giữa drone, tay cầm và ESP32-CAM-Cách hoạt động tổng quát
Ứng dụng web	<ul style="list-style-type: none">-Xây dựng website sử dụng Flask-Người dùng xem kết quả và vị trí đồng thời xem lại các ảnh đã nhận dạng trong quá khứ

2.1.2. Sơ đồ hoạt động tổng quan

- Sơ đồ hoạt động tổng quan của hệ thống được thể hiện ở hình 1.
-



Hình 1. Sơ đồ hoạt động tổng quan

2.2 Giải Pháp thuật toán :

2.2.1. Bài toán nhận diện đối tượng (Object Detection)

Trong lĩnh vực thị giác máy tính, nhận diện đối tượng là một trong những bài toán quan trọng và cơ bản. Bài toán này kết hợp nhiều nhiệm vụ như định vị, phân loại, và phát hiện đối tượng, cụ thể như sau:

a. Định vị vật thể (Object Localization)

Định vị vật thể là nhiệm vụ xác định vị trí của một hoặc nhiều đối tượng trong hình ảnh thông qua việc tạo ra các bounding box (khung giới hạn). Các khung này khoanh vùng chính xác các vật thể để phục vụ các tác vụ xử lý tiếp theo. Đây là bước nền tảng trong nhận diện đối tượng.

b. Phân loại hình ảnh (Image Classification)

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Phân loại hình ảnh là nhiệm vụ gán nhãn cho hình ảnh hoặc các vật thể trong hình ảnh. Trong bối cảnh nhận diện đối tượng, bài toán phân loại được mở rộng để gán nhãn cho từng đối tượng riêng lẻ. Điều này cho phép hệ thống nhận biết và phân biệt nhiều loại đối tượng khác nhau trong cùng một khung hình.

c. Nhận diện đối tượng (Object Detection)

Nhận diện đối tượng là một bài toán phức tạp, kết hợp cả hai nhiệm vụ định vị và phân loại. Kết quả đầu ra bao gồm:

Các bounding box xung quanh mỗi đối tượng được phát hiện.

Nhãn phân loại (class label) xác định loại đối tượng.

Chỉ số confidence score, thể hiện mức độ chắc chắn của mô hình về nhãn được dự đoán.

2.2.2. Các thuật toán chính trong YOLOv8 và Các thuật toán hỗ trợ

1. Thuật toán chính trong YOLOv8

a. Backbone

Vai trò: Backbone là mạng trích xuất đặc trưng từ ảnh đầu vào. Trong YOLOv8, backbone sử dụng cấu trúc CSPNet cải tiến, giúp tăng khả năng tái sử dụng thông tin và giảm số lượng tham số.

b. Neck

Vai trò: Neck là thành phần kết nối backbone với head, tạo ra các feature map ở nhiều cấp độ khác nhau để phát hiện đối tượng với kích thước và hình dạng đa dạng.

c. Head

Vai trò: Head là phần đảm nhiệm việc dự đoán bounding box (khung chứa đối tượng), confidence score (độ tin cậy), và nhãn lớp (class labels).

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

2. Các thuật toán hỗ trợ trong YOLOv8

a. Loss Function

Loss function trong YOLOv8 được cải tiến để cân bằng giữa các nhiệm vụ phát hiện đối tượng:

Box Loss (CIoU Loss): Đo lường khoảng cách giữa dự đoán và thực tế, bao gồm vị trí, kích thước và góc nghiêng.

Class Loss (Cross-Entropy Loss): Đánh giá sai số trong phân loại nhãn.

Objectness Loss (Binary Cross-Entropy): Xác định liệu một khu vực có chứa đối tượng hay không.

Lợi ích:

Giảm thiểu hiện tượng mất cân bằng dữ liệu (imbalanced classes).

Tăng độ chính xác khi phát hiện các đối tượng phức tạp.

b. Post-processing

Thuật toán NMS (Non-Maximum Suppression):

Loại bỏ các khung dự đoán trùng lặp bằng cách giữ lại khung có confidence score cao nhất.

Improved NMS in YOLOv8: Sử dụng thuật toán Soft-NMS để giảm thiểu việc loại bỏ sai các đối tượng gần nhau.

Lợi ích:

Tăng độ chính xác trong môi trường có mật độ đối tượng cao.

Giảm tỷ lệ phát hiện trùng lặp.

c. Tối ưu hóa huấn luyện

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Data Augmentation: Sử dụng Mosaic Augmentation và CutMix để tăng tính đa dạng của dữ liệu huấn luyện.

Tự động điều chỉnh siêu tham số (AutoML): Hỗ trợ tìm kiếm hyperparameter tối ưu mà không cần thử nghiệm thủ công.

Batch Normalization: Tăng độ ổn định của quá trình huấn luyện.

2.2.3. Lựa chọn mô hình cho giải pháp bài toán

Để xây dựng giải pháp nhận diện lửa và khói theo thời gian thực cần khảo sát và lựa chọn mô hình phù hợp theo các tiêu chí sau:

a. Tiêu chí lựa chọn mô hình

Kích thước nhỏ gọn:

Dễ dàng triển khai trên các thiết bị với tài nguyên giới hạn, chẳng hạn như server local..

Độ chính xác cao:

Phù hợp với các bài toán yêu cầu hiệu suất cao trong nhận diện đối tượng.

Thời gian suy luận nhanh:

Đảm bảo khả năng phản hồi gần như thời gian thực, giúp nâng cao trải nghiệm người dùng

b. Các mô hình được khảo sát

Tiny-YOLO:

Phiên bản rút gọn của YOLO, phù hợp với các thiết bị có cấu hình thấp.

Tốc độ nhanh, nhưng thường hy sinh độ chính xác trong các bài toán yêu cầu chi tiết.

YOLOv8:

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

YOLOv8 (phiên bản mới nhất) vượt trội hơn nhờ các cải tiến:

Kiến trúc tối ưu hơn (backbone và head cải tiến).

Hỗ trợ tốt cho các thiết bị nhúng và tích hợp dễ dàng với framework như Flask.

Cải thiện độ chính xác mà vẫn duy trì tốc độ nhanh.

c. Kết luận

Dựa trên các tiêu chí trên, YOLOv8 được lựa chọn làm mô hình chính cho giải pháp vì các lý do:

Hiệu suất vượt trội: Độ chính xác cao hơn và tốc độ suy luận tốt hơn.

Dễ triển khai trên server local: YOLOv8 giúp tích hợp nhanh với Flask.

Mô hình có thể được tối ưu để chạy trên các thiết bị có tài nguyên hạn chế.

2.3. Giải pháp về phần cứng

2.3.1. Các linh kiện phần cứng

Tên linh kiện	Hình ảnh	Tham số kỹ thuật
ESP32-Cam		<p>Nguồn điện: 2.2-3.6V</p> <p>Cảm biến hình ảnh: OV2640</p> <p>Độ phân giải 2MP</p> <p>Tốc độ khung hình: 30fps</p> <p>Giao tiếp: UART</p> <p>Kích thước: 40x27mm</p> <p>GPIO:9</p> <p>Kết nối: Wi-Fi 802.11 b/g/n và Bluetooth 4.2 BR/EDR + BLE.</p>

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Đế nạp ESP32-Cam		Chip giao tiếp: CH340 Cổng giao tiếp: Micro USB Nguồn điện: 5V Chân kết nối: TX, RX, 5V, GND
4 Động cơ không chổi than brushless A2212 2700KV		Loại động cơ: Không chổi than (Brushless DC Motor). KV (Vòng/phút/V): 2700KV Điện áp hoạt động: 7.4V - 11.1V (2-3S LiPo). Dòng điện tối đa: 16A. Công suất tối đa: 150W. Đường kính trục: 3.17mm. Kích thước động cơ: 28mm x 22mm. Trọng lượng: 48g. Khuyến nghị cánh quạt: 4x4, 5x5, hoặc 6x4 inch.
4 Mạch điều khiển tốc độ động cơ không chổi than ESC 30A 4V-16V (Pin 2S-4S) BEC 5V		Loại ESC: Không chổi than (Brushless ESC). Dòng điện tối đa: 40A. Nguồn điện đầu vào: 4V - 16V (Pin LiPo 2S-4S). BEC (Output): 5V, 3A. Loại tín hiệu điều khiển: PWM. Tần số tín hiệu: 20Hz - 500Hz. Chế độ bảo vệ: Bảo vệ quá nhiệt, bảo

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

		vệ quá dòng, và bảo vệ điện áp thấp. Kích thước: 45mm x 25mm x 8mm. Trọng lượng: 25g (không tính dây).
Cảm biến gia tốc 3 trục MPU6050 6DOF GY-521		<p>Loại cảm biến: Gia tốc 3 trục + Con quay hồi chuyển 3 trục (6DOF).</p> <p>Gia tốc ké: + Phạm vi đo: $\pm 2g$, $\pm 4g$, $\pm 8g$, $\pm 16g$.</p> <p>Độ phân giải: 16-bit.</p> <p>Con quay hồi chuyển: + Phạm vi đo: $\pm 250^\circ/s$, $\pm 500^\circ/s$, $\pm 1000^\circ/s$, $\pm 2000^\circ/s$.</p> <p>+ Độ phân giải: 16-bit.</p> <p>Giao tiếp: I2C (Tối đa 400kHz) hoặc SPI.</p> <p>Nguồn cấp: 3.3V - 5V.</p> <p>Dòng tiêu thụ: $\sim 3.6mA$ (chế độ hoạt động).</p> <p>Tích hợp bộ lọc: Digital Motion Processing (DMP) giám nhiễu.</p> <p>Tần số lấy mẫu: Tối đa 1kHz.</p> <p>Kích thước module: 20mm x 15mm.</p>
Chip Arduino Nano		<p>Vì điều khiển: ATmega328P(oldboard loader).</p> <p>Điện áp hoạt động: 5V.</p> <p>Điện áp đầu vào (khuyến nghị): 7V - 12V.</p> <p>Điện áp đầu vào (giới hạn): 6V - 20V.</p> <p>Số chân digital I/O: 14 chân (trong đó 6 chân hỗ trợ PWM).</p>

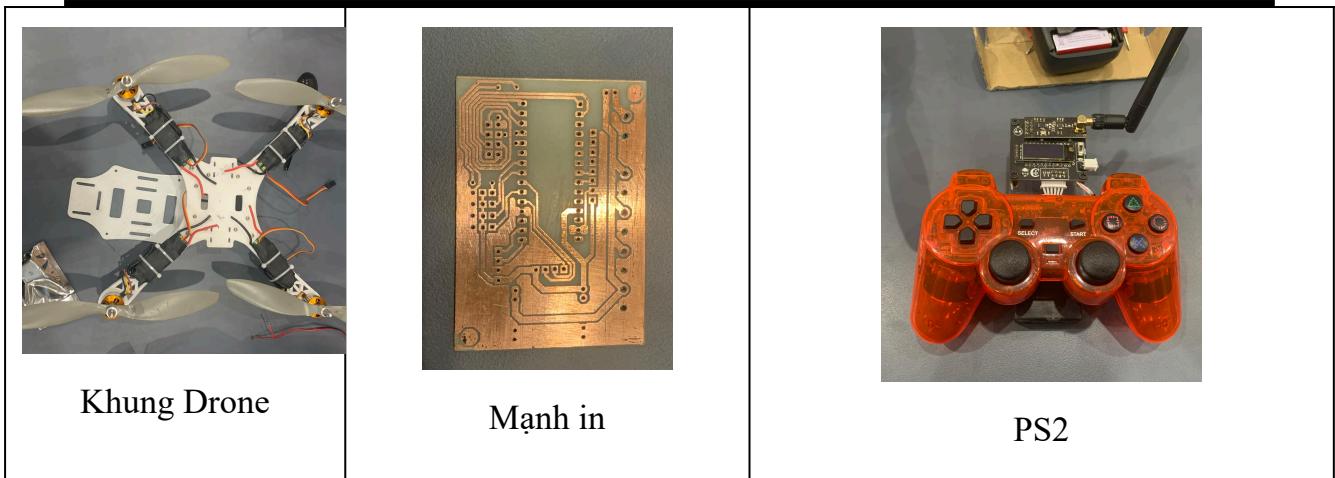
PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

		<p>Số chân analog input: 8 chân (độ phân giải 10-bit).</p> <p>Dòng tối đa qua mỗi chân I/O: 40mA.</p> <p>Flash Memory: 32KB (trong đó 2KB dùng cho bootloader).</p> <p>SRAM: 2KB.</p> <p>EEPROM: 1KB.</p> <p>Tốc độ xung nhịp: 16MHz.</p> <p>Cổng giao tiếp: UART, SPI, I2C.</p> <p>Kích thước: 45mm x 18mm.</p> <p>Trọng lượng: ~7g.</p> <p>Kết nối USB: Mini-USB.</p>
Module định vị GPS NEO-6M (Kèm anten)		<p>Loại module: GPS (Global Positioning System).</p> <p>Chip GPS: u-blox NEO-6M.</p> <p>Nguồn cấp: 3.3V - 5V.</p> <p>Dòng tiêu thụ: ~45mA.</p> <p>Giao tiếp: UART (TX, RX).</p> <p>Tốc độ baud mặc định: 9600bps (có thể cấu hình).</p> <p>Độ nhạy:</p> <ul style="list-style-type: none"> + Thu tín hiệu: -161 dBm. + Theo dõi tín hiệu: -157 dBm. <p>Độ chính xác vị trí:</p> <ul style="list-style-type: none"> + Tĩnh: ±2.5m. + Tốc độ: ±0.1m/s. <p>Thời gian khởi động: ~1 giây.</p> <p>Tần số cập nhật: Lên đến 5Hz (mặc định 1Hz).</p>

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

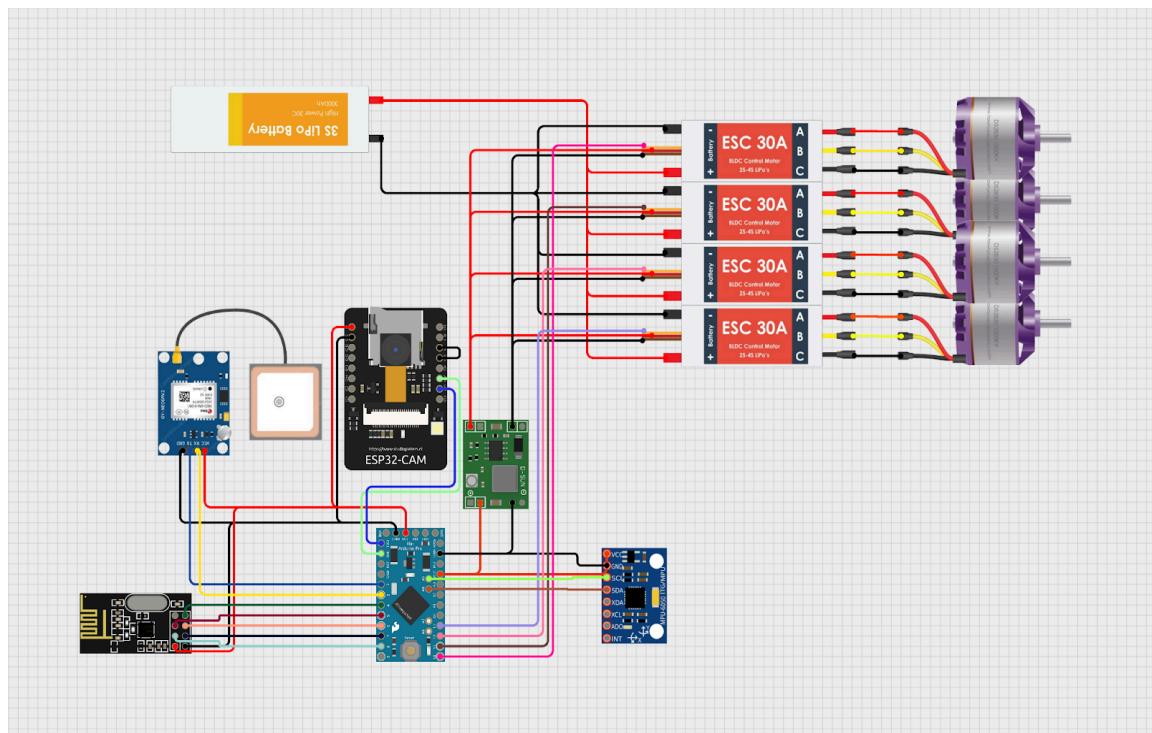
		<p>Hỗ trợ anten: Anten GPS tích hợp (đi kèm), loại Active với giao tiếp SMA.</p> <p>Nhiệt độ hoạt động: -40°C đến +85°C.</p>
Module Thu Phát Sóng RF Không Dây NRF24L01		<p>Tần số hoạt động: 2.4GHz - 2.525GHz (ISM Band).</p> <p>Công suất truyền: Tối đa +0dBm.</p> <p>Tốc độ truyền dữ liệu: 250kbps, 1Mbps, 2Mbps (có thể cấu hình).</p> <ul style="list-style-type: none"> + Khoảng cách truyền tín hiệu: Lên đến 100m (khi không có vật cản). + Điện áp hoạt động: 1.9V - 3.6V.. <p>Giao tiếp: SPI (tối đa 10Mbps).</p> <p>Số kênh: 126 kênh</p> <p>Chế độ truyền nhận: Full-duplex hoặc Half-duplex.</p>
Pin LiPo 11V 3S 2200Mah		<p>Loại pin: Lithium Polymer</p> <p>Điện áp danh định: 11.1V</p> <p>Điện áp tối đa: 12.6V</p> <p>Điện áp tối thiểu: 9V (3V mỗi cell, không nên giảm thấp hơn để tránh hỏng pin).</p> <p>Dung lượng: 1K mAh - 5K mAh</p> <p>Dòng sạc: Khuyến nghị 1C (ví dụ, pin 2200mAh sạc 2.2A).</p> <p>Kích thước: Tùy thuộc vào dung lượng (ví dụ: 105mm x 35mm x 25mm cho pin 2200mAh).</p> <p>Trọng lượng: Khoảng 200g - 300g (tùy model).</p>

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH



Hình 2. Các Linh kiện phần cứng

2.3.2. Sơ đồ lắp mạch



Hình 3. Sơ đồ lắp mạch

2.3.3. Lý thuyết điều khiển Quadcopter

Quadcopter (Quadrotor Helicopter) là thiết bị bay không người lái (UAV) với 4 động cơ và cánh quạt. Cánh trước và sau quay ngược chiều kim đồng hồ, trong khi cánh trái và phải quay cùng chiều kim đồng hồ để cân bằng momen xoắn.

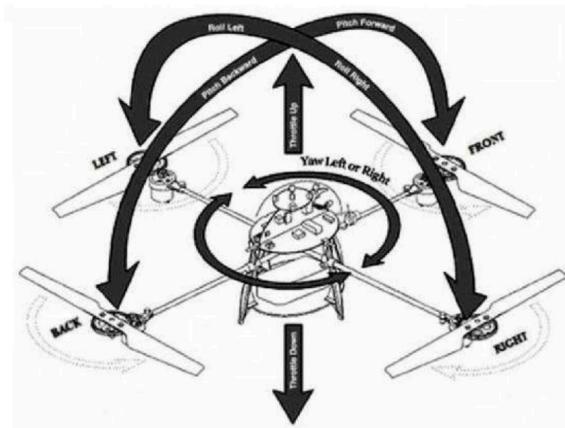
Roll chuyển động đưa quadcopter sang trái hoặc phải.

Pitch làm nghiêng quad dẫn đến chuyển động tiến hoặc lùi của quad.

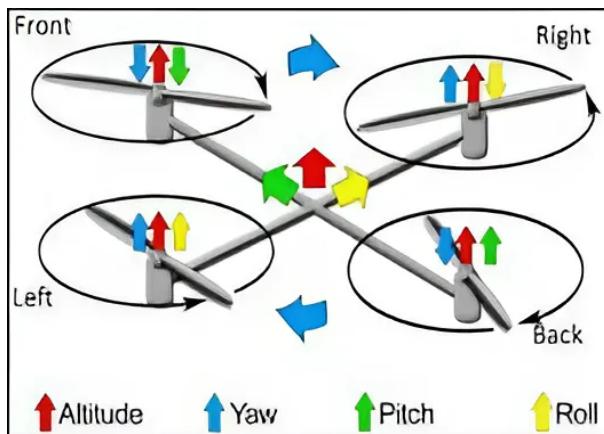
Yaw xoay quad theo chiều kim đồng hồ hoặc ngược chiều kim đồng hồ.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Altitude cung cấp cho cánh quạt trên quadcopter đủ công suất để bay lên hoặc hạ thấp trên không.

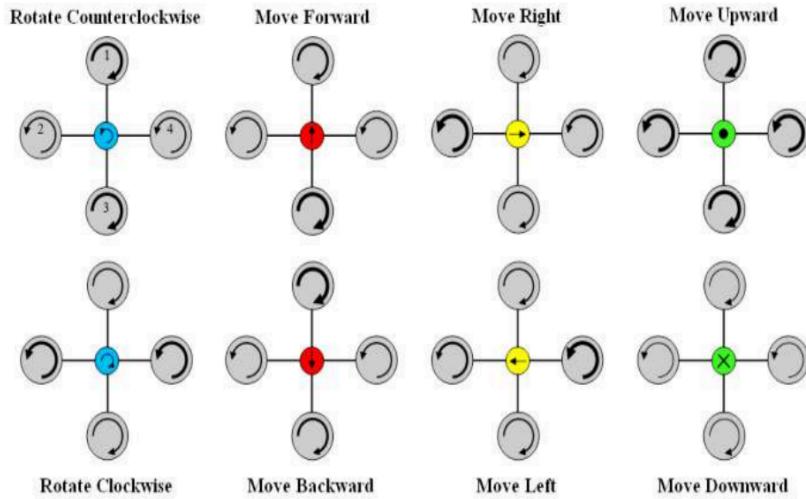


Hình 4. Cách hoạt động của Quadcopter



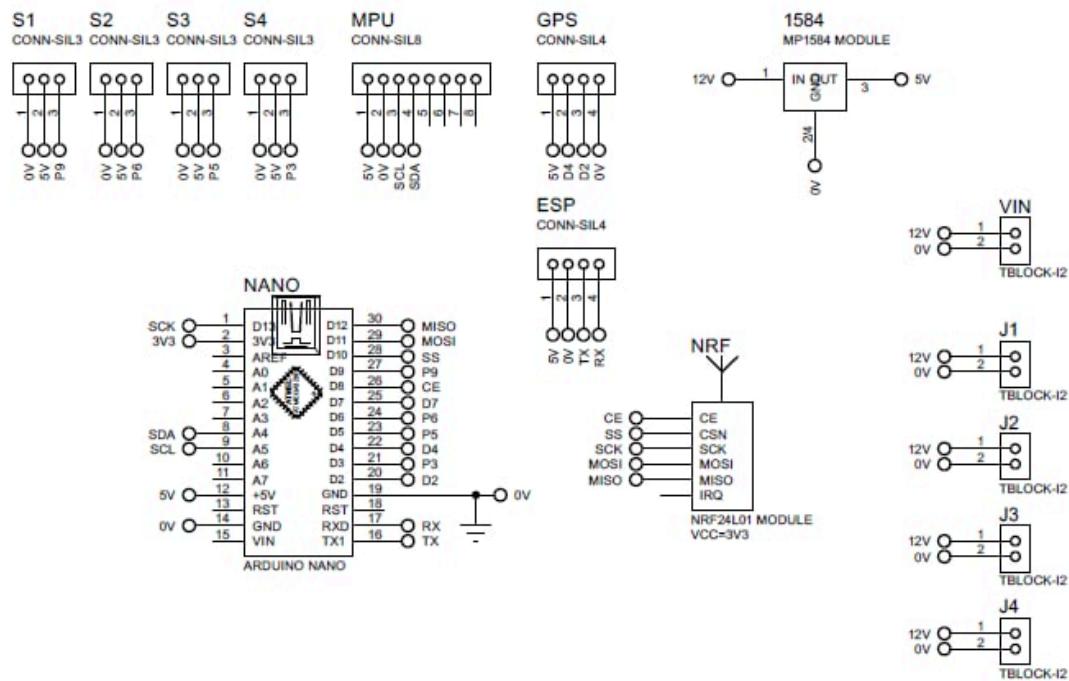
Hình 5. Cách hoạt động của Quadcopter

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH



Hình 6. Cách điều khiển Quadcopter

2.3.4. Lắp đặt mạch in



Hình 7. Sơ đồ Lắp đặt mạch in

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

2.3.5. Thuật toán PID

PID (Proportional-Integral-Derivative) là bộ điều khiển quan trọng trong việc ổn định quadcopter.

Nó giúp điều chỉnh ba trục chính: Roll, Pitch, và Yaw, bằng cách tính toán sự chênh lệch giữa giá trị mong muốn và giá trị thực tế.

Proportional (P): Điều chỉnh lực dựa trên sai số hiện tại. Nếu sai số lớn, lực điều chỉnh lớn, giúp nhanh chóng đưa quadcopter về vị trí mong muốn.

Integral (I): Tính tổng sai số qua thời gian để điều chỉnh, nhằm loại bỏ sai số tích lũy do các yếu tố môi trường.

Derivative (D): Điều chỉnh dựa trên tốc độ thay đổi của sai số, giúp giảm rung lắc và làm mượt các phản ứng. PID kết hợp ba yếu tố này để giúp quadcopter bay ổn định và chính xác.

Nhiệm vụ: Tìm các thông số thích hợp để cân bằng phương trình PID.

2.4. Giải pháp truyền thông và hệ thống

2.4.1. Giao thức NRF24L01 trên tay cầm PS2

Nguyên lý hoạt động của NRF24L01 trên PS2:

NRF24L01 là một module RF 2.4GHz hoạt động như một kênh giao tiếp không dây giữa tay cầm PS2 và thiết bị điều khiển (ví dụ, drone hoặc robot). Giao thức này được sử dụng để truyền dữ liệu điều khiển từ tay cầm đến thiết bị mục tiêu.

Truyền và nhận dữ liệu:

Module NRF24L01 trên tay cầm PS2 truyền dữ liệu điều khiển (tín hiệu từ joystick, nút bấm) qua sóng RF đến NRF24L01 trên thiết bị đích.

NRF24L01 hỗ trợ truyền dữ liệu song công (Full-Duplex), nhưng thông thường chỉ truyền đơn công (Half-Duplex) trong các ứng dụng này để giảm độ phức tạp.

Cấu trúc dữ liệu:

Dữ liệu được đóng gói trong các gói nhỏ, mỗi gói tối đa 32 byte.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Gói dữ liệu chứa thông tin về trạng thái của các nút bấm, giá trị tọa độ joystick, và các lệnh khác.

Tần số và kênh truyền:

NRF24L01 sử dụng băng tầnISM 2.4GHz và có 126 kênh. Tay cầm PS2 và thiết bị đích được cấu hình trên cùng một kênh để đảm bảo giao tiếp chính xác.

Tốc độ truyền dữ liệu:

NRF24L01 hỗ trợ tốc độ truyền dữ liệu 250kbps, 1Mbps, hoặc 2Mbps. Trong ứng dụng PS2, tốc độ 1Mbps thường được chọn để cân bằng giữa tốc độ và độ tin cậy.

Xử lý lỗi:

NRF24L01 tích hợp cơ chế ACK (Acknowledgment) để xác nhận gói dữ liệu đã được nhận thành công. Nếu gói dữ liệu không được nhận, tay cầm PS2 sẽ tự động gửi lại.

Ứng dụng:

Sử dụng trong các dự án điều khiển từ xa như drone, xe robot, hoặc thiết bị tự hành.

2.4.2. Giao thức NRF24L01 trên drone

Nguyên lý hoạt động của NRF24L01 trên drone:

NRF24L01 trên drone nhận tín hiệu điều khiển từ tay cầm hoặc các thiết bị phát khác và gửi lại thông tin trạng thái (nếu cần) qua giao tiếp RF không dây.

Nhận tín hiệu điều khiển:

Module NRF24L01 trên drone hoạt động ở chế độ Slave, liên tục lắng nghe dữ liệu từ tay cầm PS2.

Dữ liệu nhận được được giải mã và chuyển tới bộ vi điều khiển (thường là STM32, ESP32, hoặc Arduino) để điều khiển động cơ và các chức năng khác của drone.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Phản hồi dữ liệu:

NRF24L01 cũng có thể gửi phản hồi trạng thái (như mức pin, tốc độ quay động cơ) về tay cầm.

Dữ liệu phản hồi thường ít hơn dữ liệu điều khiển để giảm tải băng thông.

Tần số và kênh truyền:

Drone và tay cầm PS2 được cấu hình trên cùng kênh tần số 2.4GHz.

Trong trường hợp sử dụng nhiều drone, các kênh riêng biệt sẽ được chỉ định để tránh nhiễu tín hiệu.

Độ trễ thấp:

Giao thức NRF24L01 được tối ưu hóa để truyền dữ liệu với độ trễ rất thấp (thường dưới 10ms), phù hợp với các ứng dụng thời gian thực như điều khiển drone.

Chống nhiễu:

NRF24L01 sử dụng công nghệ GFSK (Gaussian Frequency-Shift Keying) và có khả năng nhảy kênh (Frequency Hopping) để giảm thiểu nhiễu từ các thiết bị RF khác.

Ứng dụng:

Giao tiếp không dây trong hệ thống điều khiển drone, đảm bảo khả năng phản ứng nhanh và ổn định trong điều kiện hoạt động đa kênh.

2.4.3. Giao thức UART của ESP32-CAM sử dụng:

UART (Universal Asynchronous Receiver Transmitter) là một giao thức truyền thông tuần tự (serial communication protocol), cho phép truyền và nhận dữ liệu giữa ESP32-CAM và các thiết bị bên ngoài như máy tính, cảm biến, hoặc các module khác. Giao thức này rất phổ biến trong các ứng dụng vi điều khiển do tính đơn giản và dễ triển khai.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

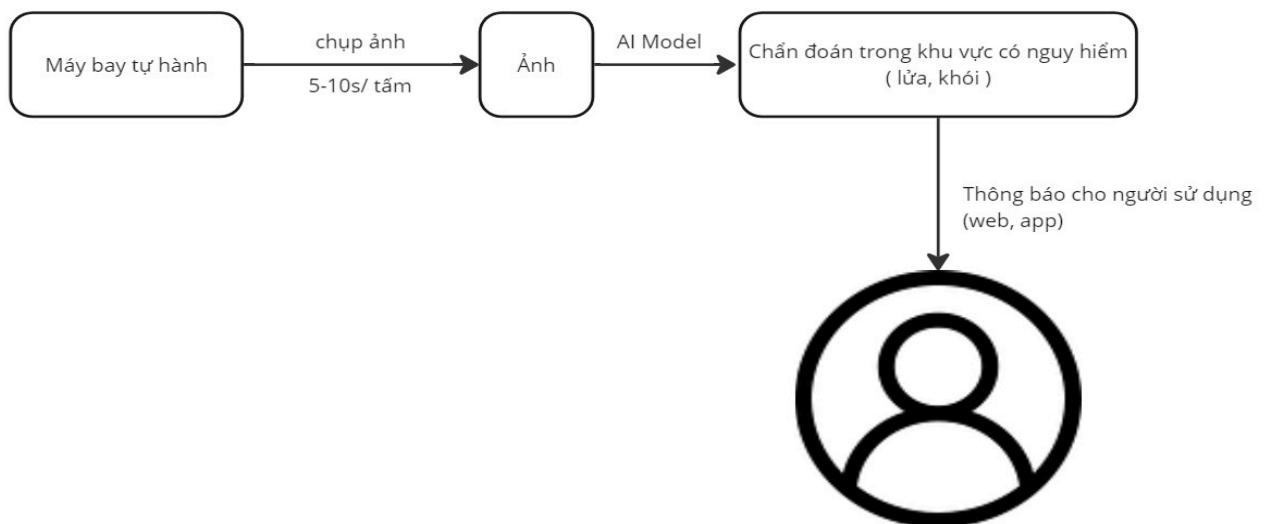
Nguyên lý hoạt động của UART trên ESP32-CAM:

Truyền và nhận dữ liệu: ESP32-CAM sử dụng UART để truyền và nhận dữ liệu dạng bit (chuỗi nhị phân) giữa các thiết bị. Giao tiếp UART gồm hai chân chính: TX (Transmit) để truyền dữ liệu và RX (Receive) để nhận dữ liệu.

Cấu trúc dữ liệu: Dữ liệu được truyền qua UART dưới dạng khung dữ liệu, mỗi khung gồm một bit start, các bit dữ liệu, và một hoặc hai bit stop. Dữ liệu được truyền một cách tuần tự, không đồng bộ, các thiết bị không cần phải đồng bộ thời gian của chúng khi truyền tải thông tin.

Tốc độ truyền dữ liệu: ESP32-CAM hỗ trợ các tốc độ truyền thông UART đa dạng, từ 9600 baud đến hàng triệu baud, tùy thuộc vào ứng dụng và yêu cầu tốc độ của hệ thống. Việc lựa chọn tốc độ baud phù hợp rất quan trọng để đảm bảo tính chính xác trong việc truyền và nhận dữ liệu.

2.4.4 Sơ đồ IoT



Hình 8. Sơ đồ IoT

2.4.5 Giao thức truyền từ AI server đến Database

Giao thức truyền từ AI server đến database được thực hiện bằng cách sử dụng MySQL connector for Python

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

MySQL Connector sử dụng giao thức TCP/IP để kết nối và truyền thông tin giữa server AI (chạy Python) và database MySQL.

Giao tiếp giữa AI server và database được thực hiện thông qua các câu lệnh SQL (INSERT, SELECT).

2.4.6 Các giao thức truyền từ Database lên Web

1. Giao thức truyền giữa Database và Flask server :

Giao thức: TCP/IP

Sử dụng thư viện Flask-MYSQLdb để giao tiếp với MySQL database.

Dữ liệu được truy vấn từ MySQL qua các lệnh SQL (SELECT), sau đó được chuyển về server Flask qua giao thức TCP/IP.

2. Giao thức truyền giữa Flask server và Web client :

Giao thức: HTTP/HTTPS (Hypertext Transfer Protocol).

Flask server trả dữ liệu dưới dạng HTML, được kết hợp với dữ liệu truy vấn từ MySQL database.

Template engine Jinja2 trong Flask được sử dụng để chèn dữ liệu vào các tệp HTML (template).

Ví dụ luồng truyền dữ liệu

Giả sử một người dùng truy cập /storage:

1. Flask server nhận yêu cầu HTTP GET từ trình duyệt.
2. Flask gửi truy vấn SQL qua TCP/IP tới MySQL để lấy danh sách hình ảnh (images_data).
3. MySQL trả về kết quả truy vấn qua TCP/IP.
4. Flask sử dụng Jinja2 để chèn dữ liệu vào template storage.html.
5. Flask trả tệp HTML hoàn chỉnh về trình duyệt qua HTTP.
6. Trình duyệt hiển thị danh sách hình ảnh.

2.5. Giới thiệu phần mềm

2.5.1 AI server

Chức năng chính: AI Server là nơi thực hiện quá trình phân tích dữ liệu hình ảnh từ ESP32-CAM để nhận diện lửa và khói. Phần này sử dụng các thuật toán Machine Learning và các thư viện liên quan.

Thành phần chi tiết :

Nhận hình ảnh từ ESP32-CAM:

Hình ảnh từ ESP32-CAM được gửi tới AI Server qua địa chỉ Wifi IP. AI Server sẽ xử lý dữ liệu này để phân tích hình ảnh và xác gửi kết quả đến Web server.

Xử lý hình ảnh:

Sử dụng OpenCV để chuẩn hóa hình ảnh thu thập được.

Trích xuất các đặc trưng quan trọng.

Dự đoán nhãn (Predict Label):

Mô hình AI dự đoán nhãn: tương ứng với các class trong dataset với 0 là lửa và 1 là khói.

Thuật toán như mạng nơ-ron CNN được áp dụng để phân tích dữ liệu.

Lưu trữ dữ liệu:

Lưu kết quả phân tích từ AI Server vào cơ sở dữ liệu.

Các dữ liệu đã được sử dụng sẽ lưu vào Database.

Gửi kết quả về Web Server:

Sau khi phân tích, lưu kết quả vào database, ảnh được gửi ngược về Web Server để hiển thị cho người dùng

2.5.2 Web server

Chức năng chính:

Web Server là thể hiện kết quả phân tích từ AI Server. Đảm bảo rằng dữ liệu được thể

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

hiện một cách trực quan và cập nhật nhanh chóng.

Thành phần chi tiết:

Giao tiếp với AI Server:

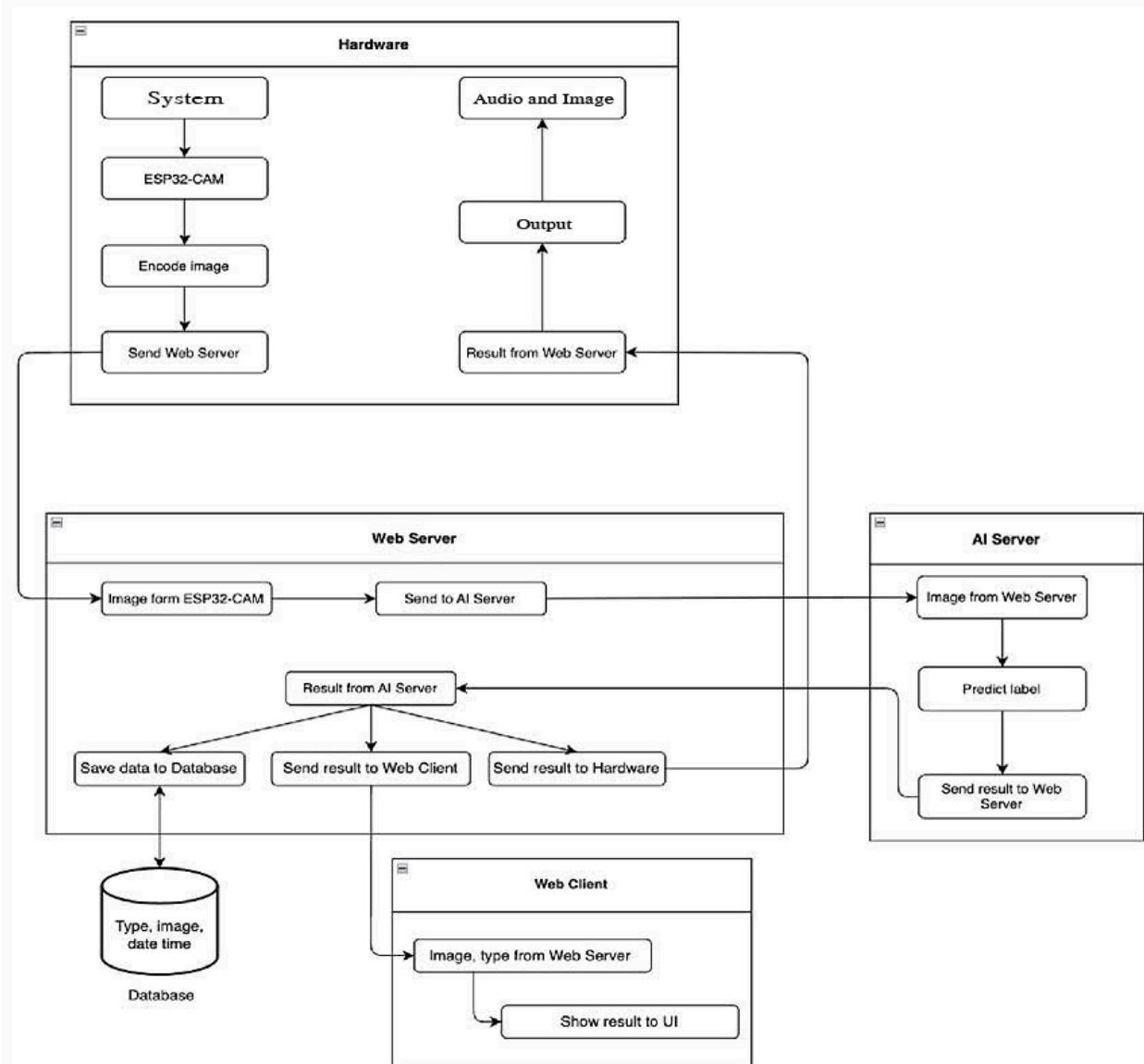
chờ nhận kết quả phân tích từ AI Server.

Đảm bảo tốc độ xử lý nhanh bằng cách tối ưu các giao thức truyền tải.

Truyền kết quả:

Web Server hiển thị kết quả phân tích từ AI server và trực quan hóa tới người dùng.

2.5.3 Sơ đồ minh họa luồng dữ liệu



Hình 9 . Minh họa luồng dữ liệu

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

- Nền tảng vật lý cho hệ thống: Bao gồm camera, các module, băng chuyền,... có nhiệm vụ thu thập dữ liệu hình ảnh, gửi dữ liệu hình ảnh lên AI Server, nhận dữ liệu trả về và hiển thị kết quả lên Web Server.
- Web Server: Nơi biểu hiện dữ liệu phân tích: là kết quả các quá trình từ phần cứng, AI Server, cho đến Web Client. ESP32-CAM Gửi hình ảnh tới AI Server để nhận diện sau đó thể hiện kết quả lên Web Server.

AI Server:

- Nhận hình ảnh từ ESP32-CAM.
- Thực hiện các thuật toán nhận diện đối tượng. Dùng thuật toán trí tuệ nhân tạo (AI) để dự đoán nhãn (Predict label) cho đối tượng trong hình ảnh.
- Gửi kết quả (hình ảnh) trở lại Web Server.
- Lưu trữ hình ảnh vào database khi phát hiện được lửa và khói.

Database: Lưu trữ dữ liệu và các loại kết quả trả về. Phần này có nhiệm vụ lưu trữ và truy xuất dữ liệu một cách hiệu quả.

3. Kết quả

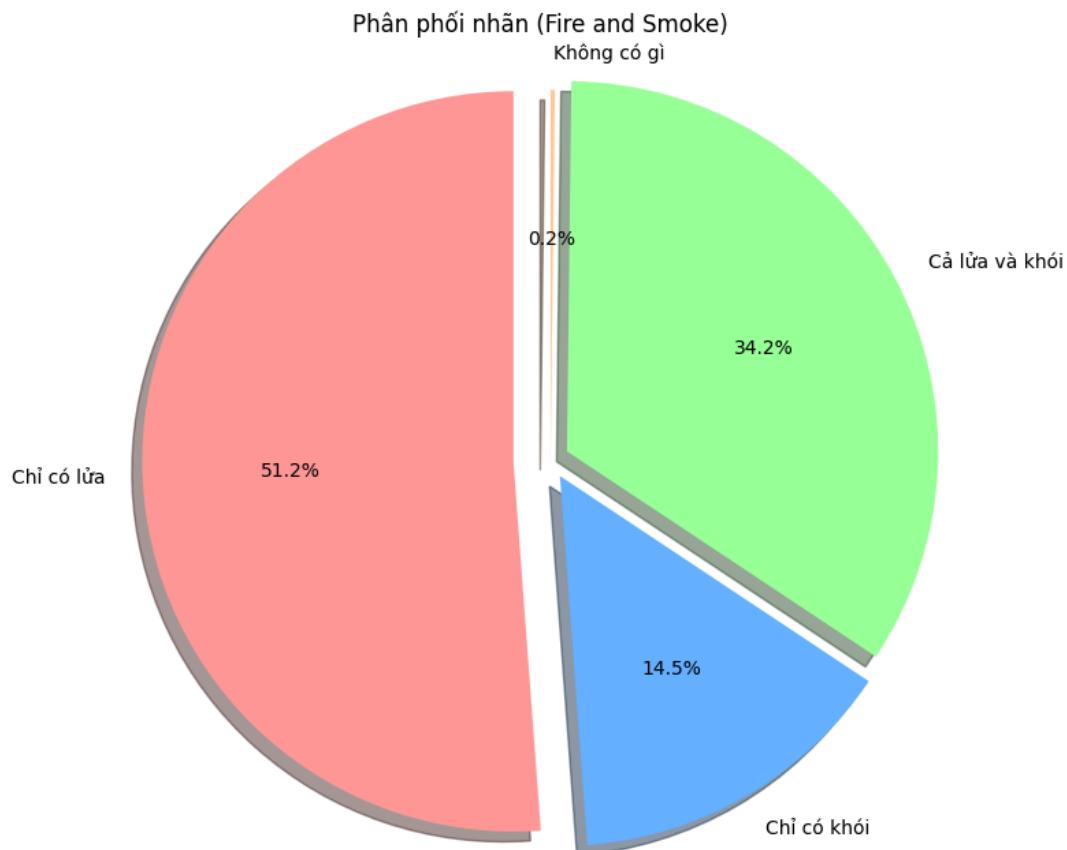
3.1 Thu thập và xử lý dữ liệu:

3.1.1 Bài toán thu thập dữ liệu:

Dữ liệu được thu thập từ các nguồn: Google

Phân loại: Dữ liệu cần training gồm 4 loại :

- ảnh không có cả lửa và khói
- ảnh chỉ có lửa
- ảnh chỉ có khói
- ảnh có cả lửa và khói



Hình 10 . Biểu đồ thể hiện tỉ lệ % của các loại dữ liệu

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Số Lượng ảnh	Mô tả chi tiết
1634	Chỉ có lửa
462	Chỉ có khói
1091	Cả lửa và khói
6	Không có gì

3.1.2. Xử lý dữ liệu:

Bước 1: Duyệt qua và đọc ảnh từ thư mục

Đầu tiên, chúng ta sẽ đọc tất cả hình ảnh và nhãn từ các thư mục dữ liệu. Dữ liệu ảnh thường được lưu trữ dưới dạng các định dạng hình ảnh thông dụng như .jpg hoặc .png. Trong khi đó, nhãn sẽ được lưu trữ dưới dạng file .txt tương ứng.

Bước 2: Chọn và làm sạch dữ liệu

Tiến hành làm sạch và loại bỏ các dữ liệu bị lỗi hoặc không hợp lệ:

Kiểm tra xem các file ảnh và file nhãn có tồn tại song song không.

Đảm bảo dữ liệu được lưu trữ ở các định dạng phù hợp.

Bước 3: Chuyển đổi dữ liệu

Chúng ta cần chuyển đổi dữ liệu (ảnh và nhãn) thành thông tin mà YOLOv8 có thể hiểu:

Chuyển đổi thông tin nhãn: Dữ liệu nhãn (lửa, khói, lửa và khói) sẽ được chuyển đổi thành các danh sách tọa độ hợp lệ theo chuẩn YOLO.

Chọn kích thước ảnh đồng nhất: YOLO yêu cầu ảnh có kích thước cố định, ví dụ như 640x640.

3.2. Công cụ và framework

3.2.1 Môi trường huấn luyện

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Mô hình được huấn luyện trên Google Colab với cấu hình GPU 12GB và RAM 12GB. Google Colab là một dịch vụ miễn phí hoặc có thể nâng cấp lên bản trả phí của Google, cho phép sử dụng các máy ảo có cấu hình cao. Tuy nhiên, thời gian sử dụng liên tục của Google Colab miễn phí thường bị giới hạn trong khoảng 12 giờ.

3.2.2 Framework

YOLO (You Only Look Once) là một hệ thống phát hiện đối tượng thời gian thực được phát triển trên nền tảng Pytorch - một framework mã nguồn mở do Meta phát triển, chuyên phục vụ nghiên cứu và phát triển các mô hình học sâu (Deep Learning).

3.2.3. Tham số huấn luyện

Số epoch: 500

Kích thước ảnh: 640px

Batch size: 16

Optimizer: SGD.

Pretrained weight (pretrained=True).

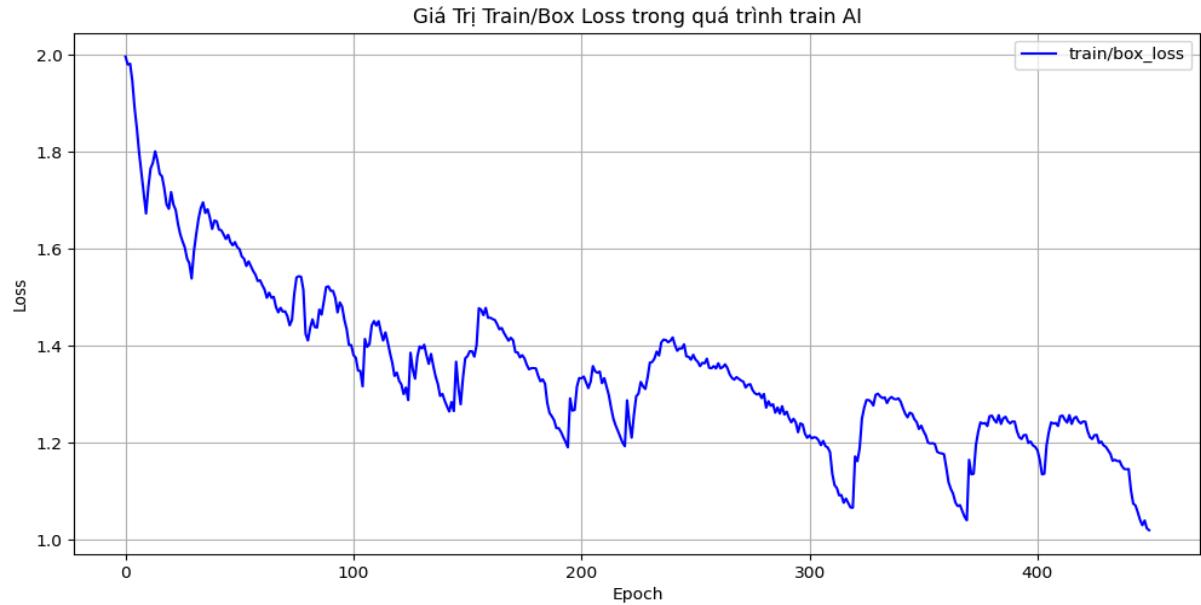
Learning rate: 0.001

Momentum: 0.937

(Các tham số này được lựa chọn dựa trên các giá trị tối ưu đã được thử nghiệm hoặc đề xuất bởi tác giả mô hình.)

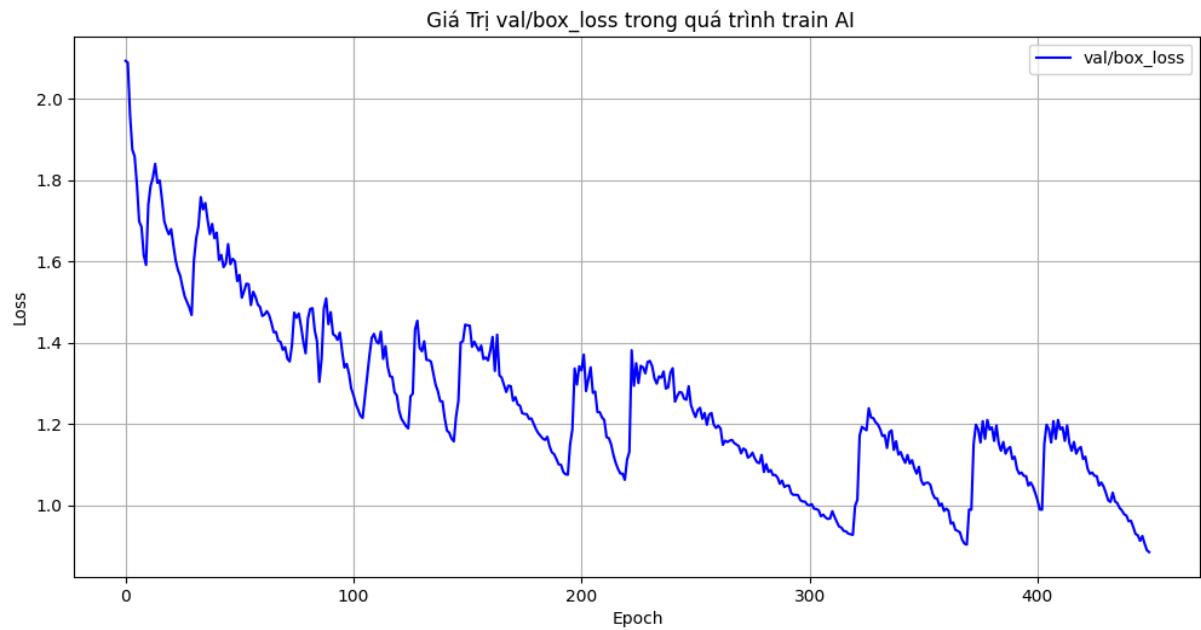
3.3. Kết quả quá trình huấn luyện

- Training Box Loss theo từng epoch được thể hiện ở hình



Hình 11. Training Box loss theo epoch

- Validate Box Loss theo từng epoch được thể hiện ở hình



Hình 12. Validate Box Loss theo epoch

3.4. Quy trình thực nghiệm và kiểm thử

Kết nối ESP32-CAM với AI server hoặc giả lập trên colab để kiểm thử hiệu suất mô hình hoạt động.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Thực hiện nhận diện và phân loại lửa và khói từ các hình ảnh và video mẫu. Các tình huống thử nghiệm được lấy từ các nguồn khác nhau, bao gồm cả dữ liệu thực tế và giả lập.

Kiểm tra khả năng lưu trữ và truy xuất dữ liệu từ máy chủ để phục vụ người dùng khi cần. Điều này đảm bảo hệ thống hoạt động mượt mà và đáp ứng nhanh chóng các yêu cầu.

3.5. Kết quả trên tập kiểm thử

a. Precision

Precision (Độ chính xác) là chỉ số đo lường tỷ lệ đối tượng được mô hình dự đoán là lửa hoặc khói mà thực tế đúng là như vậy. Precision càng cao, mô hình càng ít mắc phải lỗi loại I (false positives).

$$\text{Công thức: Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

b. Recall

Recall (Độ nhạy) là chỉ số đo lường tỷ lệ đối tượng thực sự có lửa hoặc khói mà mô hình phát hiện được. Recall càng cao, mô hình càng ít mắc phải lỗi loại II (false negatives).

$$\text{Công thức: Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

c. mAP (Mean Average Precision)

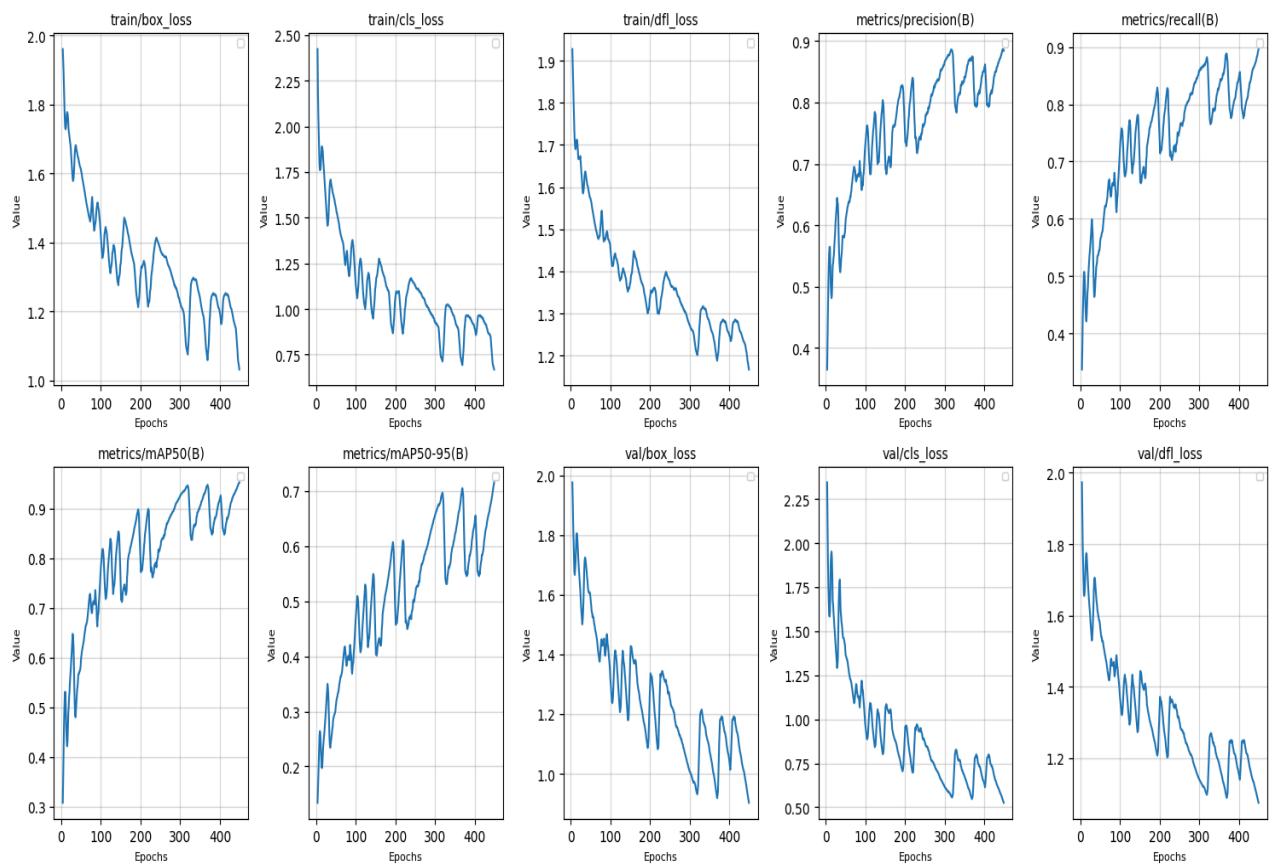
mAP (Mean Average Precision) là chỉ số tổng hợp đánh giá độ chính xác của mô hình đối với tất cả các lớp đối tượng. Trong trường hợp này, các lớp đối tượng là **lửa** và **khói**. mAP tính toán trung bình độ chính xác của mô hình ở các mức độ khác nhau của recall, cung cấp cái nhìn toàn diện về hiệu suất mô hình.

Các metrics thu được:

- Precision (P): 0.89291
- Recall (R): 0.91163
- mAP@.5: 0.95878
- mAP50-95(B): 0.73487

Các thông số chung của mô hình được biểu hiện ở hình 13.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH



Hình 13. Các thông số chung của mô hình

d. Kiểm thử kết quả giả lập

- Kết quả kiểm thử ảnh không có lửa và không có khói trước và sau khi kiểm thử được thể hiện ở hình 14 và 15



hình 14 và 15 hiển thị trước và sau khi kiểm thử

- Kết quả kiểm thử ảnh chỉ có lửa trước và sau khi kiểm thử được thể hiện ở hình 16 và 17

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH



hình 16 và 17 hiển thị trước và sau khi kiểm thử

- Kết quả kiểm thử ảnh chỉ có khói trước và sau khi kiểm thử được thể hiện ở hình 18 và 19



hình 18 và 19 hiển thị trước và sau khi kiểm thử

- Kết quả kiểm thử ảnh có lửa và có khói trước và sau khi kiểm thử được thể hiện ở hình 20 và 21



hình 20 và 21 hiển thị trước và sau khi kiểm thử

3.6. Giao diện ứng dụng

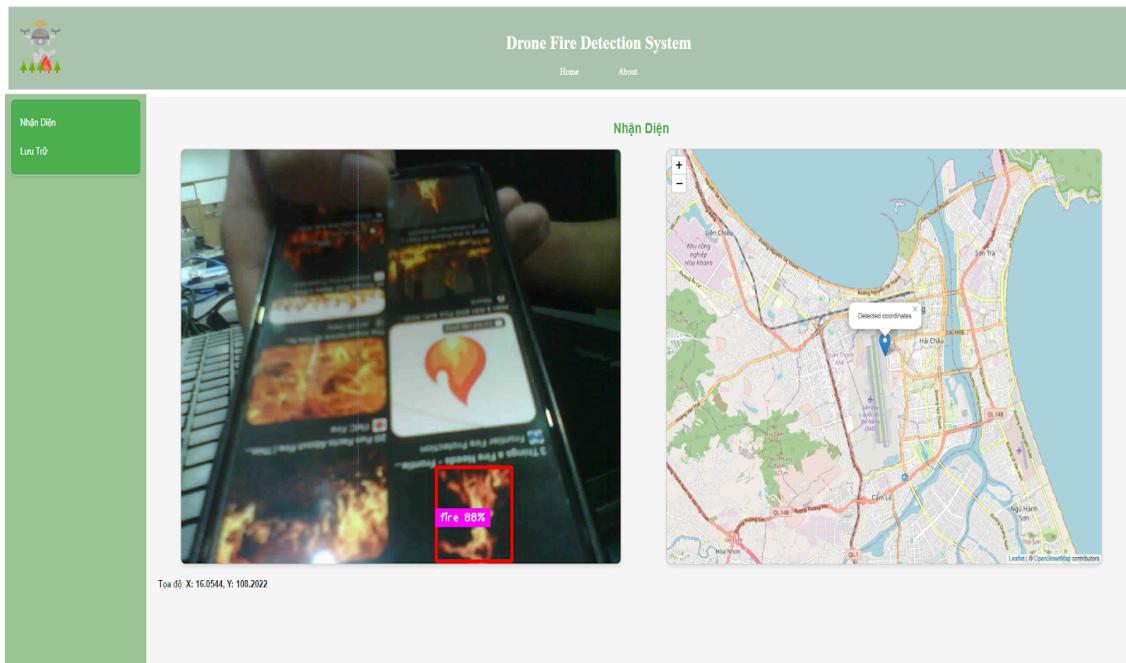
3.6.1. Giao diện chính

Web Client cung cấp các chức năng sau cho người dùng tương tác với hệ thống.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

Giao diện gồm khung hình ảnh trên ESP32_CAM chụp ảnh và nhận dạng mỗi 5 giây và header tên ứng dụng, Giao diện chính của chương trình được thể hiện ở hình 22.

- Hiển thị tin hình ảnh trong quá trình hệ thống xử lý



Hình 22. Giao diện chính

3.6.2. Giao diện xem lại

Hiển thị lịch sử thông kê các ảnh đã xử lý.

Giao diện nhận diện ảnh và độ chính xác được server trả về cùng với tọa độ.

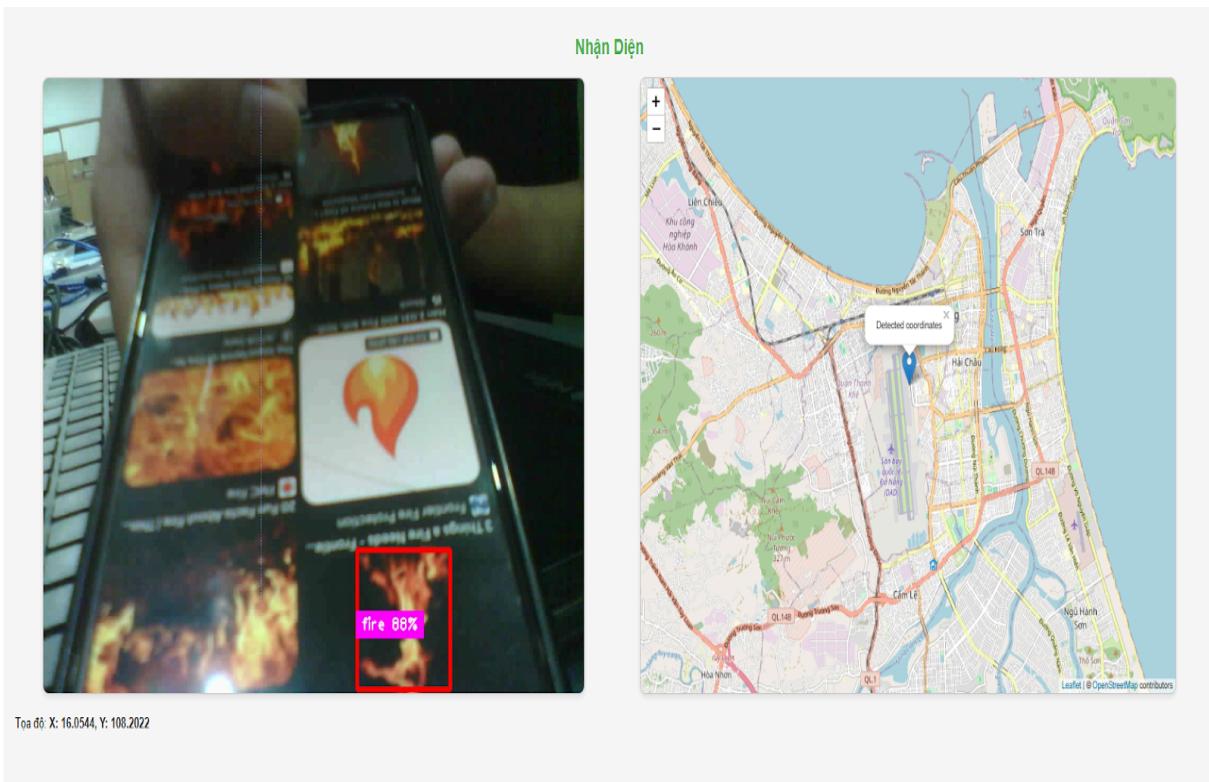
Giao diện xem kết quả nhận diện ảnh được thể hiện ở hình 23 .

Lưu Trữ Ảnh						
ID	Ngày giờ	Đường dẫn ảnh	Tọa độ: X	Tọa độ: Y		
9	2024-11-28 00:15:34	imagescanfire_20241128_001534.jpg	0	0		
8	2024-11-28 00:15:24	imagescanfire_20241128_001523.jpg	0	0		
7	2024-11-28 00:15:18	imagescanfire_20241128_001518.jpg	0	0		
6	2024-11-28 00:15:13	imagescanfire_20241128_001513.jpg	0	0		
5	2024-11-28 00:15:07	imagescanfire_20241128_001506.jpg	0	0		
4	2024-11-28 00:15:01	imagescanfire_20241128_001501.jpg	0	0		
3	2024-11-28 00:14:55	imagescanfire_20241128_001455.jpg	0	0		
2	2024-11-28 00:14:50	imagescanfire_20241128_001450.jpg	0	0		
1	2024-11-28 00:14:45	imagescanfire_20241128_001445.jpg	0	0		

Hình 23. Giao diện xem lại

3.6.3. Giao diện xem kết quả

- Giao diện xem lại chi tiết kết quả được thể hiện ở hình 24



Hình 24. Giao diện kết quả

4. Kết luận và hướng phát triển

4.1 Kết luận

Qua quá trình nghiên cứu và triển khai, nhóm đã đạt được những kết quả quan trọng như lắp ráp thành công drone dò thám và phát triển AI nhận diện lửa và khói. Thông qua đồ án này, nhóm không chỉ học hỏi được nhiều kiến thức chuyên môn về công nghệ AI, thiết kế hệ thống nhúng và lắp ráp thiết bị, mà còn rèn luyện kỹ năng làm việc nhóm, quản lý thời gian và giải quyết vấn đề.

Tuy nhiên, dự án vẫn còn một số hạn chế như thời lượng pin của drone chưa tối ưu, cần cải thiện độ chính xác của AI trong điều kiện ánh sáng yếu hoặc môi trường khói dày đặc, cũng như nâng cao thời gian bay của drone. Trong tương lai, chúng tôi kỳ vọng sẽ tiếp tục tối ưu hóa hệ thống, mở rộng phạm vi ứng dụng và đóng góp tích cực hơn vào việc bảo vệ môi trường.

Chúng tôi hy vọng rằng dự án này không chỉ mang lại ý nghĩa học thuật mà còn có tiềm năng đóng góp tích cực vào công tác bảo vệ môi trường và an toàn xã hội.

4.2 Hướng phát triển

Dự án lắp ráp drone dò thám tích hợp AI phát hiện lửa và khói trong rừng đã đạt được những kết quả quan trọng, nhưng vẫn còn nhiều tiềm năng để phát triển và cải tiến nhằm tăng cường hiệu quả và ứng dụng thực tế. Một số hướng phát triển cụ thể bao gồm:

1. Nâng cao độ chính xác của AI

- Tăng cường khả năng nhận diện trong các điều kiện phức tạp như ánh sáng yếu, khói dày đặc, hoặc môi trường rừng rậm rạp.
- Sử dụng các mô hình học sâu tiên tiến hơn, đồng thời mở rộng tập dữ liệu huấn luyện để bao quát nhiều trường hợp và điều kiện môi trường thực tế hơn.

2. Cải thiện hiệu năng của drone

- Nghiên cứu và áp dụng các loại pin hoặc nguồn năng lượng mới để kéo dài thời gian bay và tăng khả năng hoạt động liên tục.

PBL4: ĐỒ ÁN HỆ THỐNG THÔNG MINH

- Tích hợp thêm các cảm biến như nhiệt độ, độ ẩm, và khí gas để cải thiện khả năng thu thập dữ liệu và tăng độ chính xác khi phát hiện cháy.
- Nâng cấp khả năng chống chịu thời tiết khắc nghiệt để drone hoạt động ổn định hơn trong nhiều môi trường khác nhau.

3. Phát triển hệ thống cảnh báo thông minh

- Kết hợp công nghệ IoT để gửi cảnh báo nhanh chóng đến các trung tâm kiểm soát hoặc cơ quan chức năng khi phát hiện nguy cơ cháy.
- Xây dựng ứng dụng di động với giao diện thân thiện, cung cấp thông tin chi tiết về vị trí, mức độ nghiêm trọng và hướng di chuyển của đám cháy.

4. Tăng tính tự động hóa

- Phát triển các thuật toán điều khiển tự động để drone có thể hoạt động hoàn toàn độc lập, bao gồm tự động lên lịch bay, nhận diện các khu vực nguy cơ cao, và quay trở về điểm xuất phát khi hoàn thành nhiệm vụ.

5. Mở rộng ứng dụng

- Khả năng phát hiện khói và lửa có thể được mở rộng sang các lĩnh vực khác, như giám sát nhà kho, khu công nghiệp, hoặc phát hiện sớm nguy cơ cháy nổ ở các khu vực dân cư đông đúc.
- Kết hợp drone với các công nghệ chữa cháy để phản ứng nhanh hơn khi có sự cố, chẳng hạn như phun chất dập lửa ngay khi phát hiện nguy hiểm.

6. Phối hợp với cơ quan chức năng và cộng đồng

- Hợp tác với các cơ quan bảo vệ rừng, cứu hỏa, hoặc tổ chức môi trường để triển khai hệ thống trên quy mô lớn.
- Xây dựng các chương trình tuyên truyền và giáo dục cộng đồng về ứng dụng công nghệ này trong bảo vệ môi trường.

Những hướng phát triển này không chỉ giúp hoàn thiện dự án mà còn mở ra cơ hội ứng dụng rộng rãi, góp phần vào việc bảo vệ tài nguyên thiên nhiên và giảm thiểu thiệt hại do cháy rừng gây ra.

5. Tài liệu tham khảo

[1] Source code CAM :

https://github.com/Kangchua/PBL4_ESP32CAM_Webserver/tree/main/ESP32CAM_code

[2] Source code Web :

https://github.com/Kangchua/PBL4_ESP32CAM_Webserver/tree/main/Webserver

[3] Source code phần cứng :

https://github.com/Kangchua/PBL4_ESP32CAM_Webserver/tree/main/Drone_code

[4] THIẾT KẾ VÀ THI CÔNG MÔ HÌNH MÁY BAY KHÔNG NGƯỜI LÁI TỰ ĐỘNG ĐÁP CÁNH TRÊN MỤC TIÊU XÁC ĐỊNH

[5] THIẾT KẾ VÀ CHẾ TẠO MÔ HÌNH MÁY BAY-QUADROCOPTER, Lâm Ngọc Tâm, PGS.TS Trần Xuân Tùy

[6] Starlino, 12/2010, Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications.

[7] Yolov8 :[YOLOv8 - Ultralytics YOLO Docs](#)

[8] NGHIÊN CỨU THUẬT TOÁN THỊ GIÁC MÁY TÍNH ĐỂ THEO DÕI VIỆC THỰC HIỆN CÁC QUY ĐỊNH VỀ PHÒNG NGỪA COVID-19 SỬ DỤNG KỸ THUẬT HỌC SÂU, Nguyễn Đức Thiện, Lư Tất Thắng, Nguyễn Văn Thặng và Trương Quốc Bảo