ỨNG DỤNG DRONE KẾT HỢP ZIGBEE, MODULE SIM VÀ ESP32-CAM TRONG HỆ THỐNG TÌM KIẾM CỨU NẠN TRÊN BIỂN

APPLICATION OF DRONES COMBINED WITH ZIGBEE, SIM MODULE, AND ESP32-CAM IN MARITIME SEARCH AND RESCUE SYSTEMS

SVTH: Trần Thanh Khoa, Trần Minh Quân, Nguyễn Hoàng Anh Quốc, Mai Xuân Phúc Tâm, Trương Tử Anh

Lớp: 21DT1, 21DT2, 21DT2, 21DT2, 21DTCLC1, Khoa Điện tử - Viễn thông, Trường Đại học Bách Khoa - ĐHĐN;
Email: thanhkhoa19052003@gmail.com, 106210092@sv1.dut.udn.vn, nhaquoc03@gmail.com, maitam031003@gmail.com, 106210021@sv1.dut.udn.vn

GVHD: Thái Văn Tiến

Khoa Điện tử - Viễn thông, Trường Đại học Bách Khoa - ĐHĐN; Email: tvtien.dtvt@dut.udn.vn

Tóm tắt - Tìm kiếm cứu nạn (SAR) trên biển tại Việt Nam gặp nhiều khó khăn do địa hình phức tạp và công nghệ hạn chế. Dù các hệ thống giám sát truyền thống đã được triển khai, việc tích hợp drone vẫn còn nhiều thách thức. Nghiên cứu này đề xuất một giải pháp IoT dựa trên drone, tận dụng Zigbee và module SIM để truyền dữ liệu hiệu quả. Hệ thống sử dụng ESP32-CAM chụp và gửi ảnh qua giao thức ESP-NOW, kết hợp mạng 4G và MQTT để truyền dữ liệu theo thời gian thực, cùng thuật toán học sâu phát hiện người từ ảnh. Điểm nổi bật là sự kết hợp Zigbee, ESP-NOW và xử lý ảnh trong một hệ thống chi phí thấp, lần đầu được thử nghiệm thực tế cho cứu nạn trên biển tại Việt Nam. Thử nghiệm cho thấy hệ thống nhanh và chính xác hơn các phương pháp truyền thống, phù hợp triển khai ở các nước đang phát triển nhờ độ phức tạp tuyến tính và khả năng kết nối internet qua MQTT.

Từ khóa - Drone, Zigbee, SIM Module, ESP32-CAM, ESP-NOW, MQTT, IoT, học sâu, tìm kiếm và cứu hộ, hàng hải.

Abstract - Search and rescue (SAR) operations at sea in Vietnam face significant challenges due to complex terrain and limited technology. Although traditional monitoring systems have been deployed, integrating drones remains a challenge. This study proposes an IoT solution based on drones, leveraging Zigbee and SIM modules for efficient data transmission. The system utilizes ESP32-CAM to capture and send images via the ESP-NOW protocol, combined with 4G and MQTT for real-time data transfer, and a deep learning algorithm to detect people from images. Its key innovation lies in the integration of Zigbee, ESP-NOW, and image processing within a low-cost system, marking the first practical test for maritime rescue in Vietnam. Experiments demonstrate that the system is faster and more accurate than traditional methods, making it suitable for deployment in developing countries due to its linear complexity and internet connectivity via MQTT.

Key words - Drone, Zigbee, SIM Module, ESP32-CAM, ESP-NOW, MQTT, IoT, deep Learning, search and rescue, maritime.

1. Đặt vấn đề

Trong những năm gần đây, các sự cố xảy ra trên biển như tai nạn tàu thuyền, người rơi xuống nước hay các tình huống khẩn cấp khác đã trở thành một vấn đề cấp bách đòi hỏi các giải pháp tìm kiếm và cứu hộ (Search and Rescue - SAR) hiệu quả. Các phương pháp truyền thống sử dụng tàu cứu hộ và máy bay có người lái thường gặp phải những hạn chế lớn về chi phí vận hành, thời gian phản ứng, cũng như khả năng tiếp cận các khu vực nguy hiểm hoặc xa bờ. Đặc biệt, trong điều kiện thời tiết khắc nghiệt hoặc ban đêm, việc triển khai các phương tiện này càng trở nên khó khăn, dẫn đến hiệu quả tìm kiếm bị giảm sút và nguy cơ mất an toàn cho đội ngũ cứu hộ tăng cao.

Sự phát triển của công nghệ không người lái, đặc biệt là các phương tiện bay không người lái (Unmanned Aerial Vehicles - UAV), đã mở ra một hướng tiếp cận mới đầy triển vọng cho các hoạt động SAR trên biển. UAV không chỉ có chi phí thấp hơn so với các phương tiện truyền thống mà còn có khả năng hoạt động linh hoạt, bao phủ diện tích lớn và tiếp cận nhanh chóng các khu vực khó khăn. Khi được tích hợp với các hệ thống cảm biến hình ảnh, truyền thông không dây và thuật toán xử lý thông minh, UAV có thể thực hiện các nhiệm vụ như phát hiện mục tiêu, truyền dữ liệu thời gian thực và hỗ trợ định vị chính xác, từ đó nâng cao hiệu quả của các chiến dịch cứu hộ.

Tuy nhiên, việc triển khai UAV trong việc tìm kiếm và cứu hộ trên biển cũng đối mặt với nhiều thách thức, bao gồm hạn chế về thời gian bay do dung lượng pin, khả năng truyền dữ liệu ở khoảng cách xa, và yêu cầu xử lý hình ảnh nhanh chóng để phát hiện người gặp nạn trong môi trường phức tạp như mặt biển. Để giải quyết các vấn đề này, nghiên cứu đã tập trung vào việc phát triển một hệ thống UAV tích hợp dựa trên vi điều khiển ESP32, kết hợp các module truyền thông tiên tiến như ESP-NOW và PPPOS qua mạng 4G, cùng với các thuật toán học sâu để phân tích hình ảnh. Hệ thống này không chỉ cho phép điều khiển UAV từ xa mà còn đảm bảo việc truyền tải hình ảnh thời gian thực và xử lý dữ liệu hiệu quả để hỗ trợ các đội cứu hộ đưa ra quyết định kịp thời.

Nghiên cứu này đề xuất một giải pháp toàn diện bao gồm: (1) Bộ điều khiển từ xa dựa trên ESP32-S3 kết hợp với nền tảng ESP Rainmaker để gửi tín hiệu điều khiển và nhận thông số từ UAV; (2) Hệ thống UAV tích hợp ESP32 và ESP32-CAM để thu thập và truyền hình ảnh qua giao thức ESP-NOW; (3) Module SIM A7600C sử dụng giao thức PPPOS để gửi dữ liệu hình ảnh lên MQTT broker thông qua mạng 4G; và (4) Ứng dụng học sâu với mô hình YOLOv11 để phân tích và phát hiện người gặp nạn từ hình ảnh thu thập được. Thông qua việc tích hợp các thành phần này, hệ thống nhằm mục tiêu cải thiện khả năng phản ứng nhanh, độ chính xác trong phát hiện mục tiêu và ổn định kết nối truyền thông trong các tình huống tìm kiếm và cứu hộ trên biển.

Mục tiêu chính của nghiên cứu là phát triển một hệ thống UAV chi phí thấp, hiệu quả cao, có khả năng hoạt động trong điều kiện thực tế tại biển, đồng thời đảm bảo tính ổn định của truyền thông và độ chính xác trong việc phát hiện mục tiêu. Phạm vi nghiên cứu tập trung vào việc tích hợp phần cứng (ESP32, ESP32-CAM, SIM A7600C) và phần mềm (ESP-NOW, PPPOS, MQTT, YOLO) để tạo ra một hệ thống từ điều khiển, thu thập dữ liệu, truyền tải, đến xử lý và đưa ra cảnh báo. Giải pháp này không chỉ hướng đến việc nâng cao hiệu quả tìm kiếm và cứu hộ trên biển mà còn có tiềm năng ứng dụng trong các hoạt động

giám sát môi trường biển, hỗ trợ cứu hộ thiên tai hoặc quản lý giao thông thủy.

Nghiên cứu đóng góp một cách tiếp cận mới bằng cách tận dụng các công nghệ mã nguồn mở và phần cứng giá rẻ, phù hợp với việc tối ưu điều kiện kinh tế. Phần tiếp theo của bài báo sẽ trình bày chi tiết thiết kế hệ thống, phương pháp triển khai, kết quả thử nghiệm thực tế, và đánh giá hiệu quả của giải pháp trong các tình huống tìm kiếm và cứu hộ trên biển mô phỏng. Qua đó, nghiên cứu không chỉ cung cấp một công cụ hỗ trợ cứu hộ hiệu quả mà còn đặt nền tảng cho các ứng dụng tự động hóa trong các tình huống khẩn cấp khác trong tương lai.

2. Mô hình hệ thống và phương pháp tiếp cận

Với sự phát triển mạnh mẽ của Internet vạn vật (IoT) và sự tích hợp của các thiết bị cảm biến cùng công nghệ truyền thông không dây, nhiều giải pháp giám sát và điều khiển từ xa đã được xây dựng dựa trên kiến trúc IoT truyền thống. Trong nghiên cứu này, một hệ thống UAV tích hợp cho hoạt động tìm kiếm và cứu hộ (SAR) trên biển, tận dụng vi điều khiển ESP32 chi phí thấp và giao thức MQTT để giải quyết các điểm còn hạn chế trong mạng IoT, đặc biệt là trong môi trường băng thông thấp như vùng biển xa bờ. Giao thức MQTT cho phép truyền dữ liệu liên tục và thời gian thực nhờ khả năng hoạt động hiệu quả trong điều kiện mạng hạn chế [31]. Hệ thống được thiết kế để thu thập, xử lý và truyền dữ liệu từ UAV đến máy chủ đám mây, hỗ trợ các đội cứu hộ đưa ra quyết định kịp thời.

Hệ thống bao gồm một mạng lưới cảm biến hình ảnh và thông số trạng thái (góc nghiêng, mức pin) được triển khai trên UAV, kết hợp với các module truyền thông không dây như ESP-NOW và PPPOS qua mạng 4G. Dữ liệu hình ảnh và thông số được truyền đến máy chủ đám mây, nơi chúng được xử lý bởi thuật toán học sâu YOLO để phát hiện người gặp nạn. Kết quả sau đó được hiển thị trên giao diện người dùng thông qua nền tảng ESP Rainmaker, cung cấp thông tin trực quan như vị trí mục tiêu và trạng thái UAV. Mô hình hệ thống được minh họa trong Hình 1.

2.1. Điều khiển UAV và Thu thập Dữ liệu Dựa trên ESP32

Hệ thống sử dụng ESP32-S3 để điều khiển drone và nhận dữ liệu trạng thái (góc nghiêng từ MPU-6050, mức pin) qua MQTT. ESP32 trên drone xử lý tín hiệu điều khiển và gửi dữ liệu về máy chủ đám mây thông qua nền tảng ESP Rainmaker.

2.1.1. Bộ lọc Complementary

Góc nghiêng của drone được tính bằng cách kết hợp dữ liệu từ gia tốc kế và con quay hồi chuyển thông qua bộ lọc Complementary. Công thức tính góc Roll và Pitch từ gia tốc kế theo (1) và (2):

$$\theta_{roll} = \tan^{-1}\frac{AccY}{\sqrt{AccX^2 + AccZ^2}} \quad (1)$$

$$\theta_{pitch} = \tan^{-1}\frac{-AccX}{\sqrt{AccY^2 + AccX^2}} \quad (2)$$

Mặt khác, góc nghiêng của drone cũng được tính từ con quay hồi chuyển (tích phân tốc độ góc) theo (3) và (4):

$$\theta_{gyro,roll} = \theta_{roll,prev} + \omega_{roll}.T_s \quad (3)$$

$$\theta_{gyro,pitch} = \theta_{pitch,prev} + \omega_{pitch}.T_s \quad (4)$$

Bộ lọc Complementary theo (5) và (6) kết hợp hai nguồn dữ liệu trên bằng cách ưu tiên dữ liệu từ con quay hồi chuyển nhưng vẫn sử dụng gia tốc kế để hiệu chỉnh sai số.

$$\theta_{roll} = 0{,}991.\left(\theta_{roll,prev} + \omega_{roll}.T_s\right) + 0{,}009.\theta_{acc,roll} \quad (5)$$

$$\theta_{pitch} = 0{,}991.\left(\theta_{pitch,prev} + \omega_{pitch}.T_s\right) + 0{,}009.\theta_{acc,pitch} \quad (6)$$

Trong đó, $T_s = 0{,}004$ giây là chu kỳ cập nhật, hệ số 0,991 ưu tiên dữ liệu con quay hồi chuyển để giảm nhiễu dài hạn, và 0,009 bổ sung dữ liệu gia tốc kế để hiệu chỉnh trôi góc. Góc nghiêng được giới hạn trong khoảng ±20 độ để tránh sai lệch quá mức.

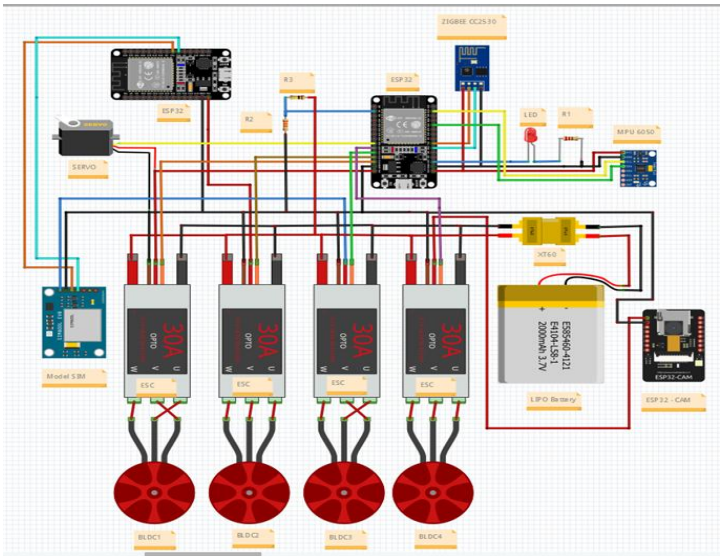
2.1.2. Thuật toán PID

Hệ thống sử dụng bộ điều khiển PID hai cấp để điều chỉnh góc nghiêng mong muốn $\theta_{desired}$ dựa trên tín hiệu đầu vào từ bộ điều khiển.

a. PID góc (Angle PID)

Tính toán tốc độ góc mong muốn từ sai số giữa góc mong muốn và góc thực tế:

$$\omega_{desired} = K_p\,(\theta_{desired} + \theta_{actual}) + K_i \int (\theta_{desired} - \theta_{actual})dt + K_d \frac{d}{dt}(\theta_{desired} - \theta_{actual}) \quad (7)$$



Hình 1: *Mô hình thiết kế hệ thống cho tìm kiếm và cứu hộ trên biển dựa trên UAV sử dụng MQTT và ESP32*

Trong đó:

$\omega_{desired}$ là tốc độ góc mong muốn,

$\theta_{desired}$, θ_{actual} là góc mong muốn và góc thực tế,

K_p, K_i, K_d là các thông số PID cho bộ điều khiển góc.

b. PID tốc độ góc (Rate PID)

Điều chỉnh tốc độ góc thực tế dựa trên sai số tốc độ góc:

$$\tau = K_{p_r}(\omega_{desired} + \omega_{actual}) + K_i \int(\omega_{desired} - \omega_{actual})dt + K_{d_r}\frac{d}{dt}(\omega_{desired} - \omega_{actual}) \quad (8)$$

Trong đó:

τ là tốc độ góc mong muốn,

ω_{actual} là tốc độ góc thực tế.

Dữ liệu trạng thái (góc nghiêng Pitch, Roll và mức pin Voltage) được drone gửi ngược lại bộ điều khiển qua UART (sendDataBack) với cấu trúc DataPacketSend (byte bắt đầu 0xAB, byte kết thúc 0xBA). Bộ điều khiển nhận dữ liệu (receiveData), cập nhật lên ESP Rainmaker thông qua MQTT mỗi giây, cho phép giám sát thời gian thực khi kết nối Wi-Fi khả dụng.

2.2. Thu thập và Truyền Hình ảnh

Thuật toán 1: *Chụp và Truyền Hình ảnh qua ESP-NOW*

Initialize: Camera configuration (ESP32-CAM), ESP-NOW, ESP32 trung gian
Evaluate initial state S_0 (Camera ready, ESP-NOW active)
loop
    Capture frame using esp_camera_fb_get() and store in camera_fb_t
    Get image size (fb->len) and send via ESP-NOW
    if send_size_success **then**
        Divide image into 240-byte chunks and send sequentially via ESP-NOW
        if chunk_send_fails **then** retry up to 3 times; **if** still fails, log error and **exit**
    else
        Retry sending size up to 3 times; if fails, exit loop
    end if
    Release buffer using esp_camera_fb_return(fb)
    Wait 10 seconds before next capture
    Update transmission status
end loop

ESP32-CAM chụp ảnh và truyền qua Zigbee hoặc ESP-NOW đến ESP32 thứ hai, một module vi điều khiển ESP32 khác đóng vai trò trung gian trong hệ thống drone. ESP32 thứ hai nhận dữ liệu hình ảnh từ ESP32-CAM, sau đó sử dụng module SIM A7600C với giao thức PPPOS để gửi ảnh lên MQTT broker qua mạng 4G, đảm bảo hoạt động ổn định ở khu vực không có Wi-Fi.

2.2.1. Thu thập Hình ảnh bằng ESP32-CAM

Hình ảnh được chụp bởi module ESP32-CAM, tích hợp cảm biến camera OV2640. Cấu hình camera được thiết lập như sau:

- GPIO: PWDN (32), XCLK (0), SIOD (26), SIOC (27), Y9-Y2 (35, 34, 39, 36, 21, 19, 18, 5), VSYNC (25), HREF (23), PCLK (22).
- Thông số: Tần số XCLK 20 MHz, định dạng PIXFORMAT_JPEG, kích thước khung FRAMESIZE_SVGA (800x600), chất lượng JPEG 10, bộ đệm đơn (fb_count = 1).

Quy trình chụp ảnh được thực hiện trong vòng lặp loop():

Bước 1: Hàm esp_camera_fb_get() chụp một khung hình và lưu vào bộ đệm camera_fb_t.

Bước 2: Kích thước hình ảnh (fb->len) được gửi trước qua ESP-NOW dưới dạng gói 4 byte (uint32_t).

Bước 3: Dữ liệu ảnh được chia thành các đoạn (chunk) tối đa 240 byte (giới hạn của ESP-NOW) và gửi tuần tự đến ESP32 trung gian có địa chỉ MAC cố định.

Bước 4: Sau khi gửi xong, bộ đệm được giải phóng bằng esp_camera_fb_return(fb), và hệ thống chờ 10 giây trước khi chụp ảnh tiếp theo.

Giao thức ESP-NOW được khởi tạo trong setup(), với chế độ WIFI_STA, không mã hóa, và callback OnDataSent để xác nhận trạng thái gửi (thành công hoặc thất bại). Độ trễ 10ms giữa các gói đảm bảo truyền dữ liệu ổn định trong môi trường không dây.

2.2.2. Truyền Hình ảnh qua ESP-NOW và PPPOS

ESP32 trung gian nhận dữ liệu từ ESP32-CAM thông qua ESP-NOW và chuyển tiếp lên MQTT broker qua mạng 4G sử dụng module SIM A7600C. Quy trình bao gồm:

Bước 1. Nhận dữ liệu qua ESP-NOW: Hàm OnDataRecv xử lý các gói tin nhận được:

Bước 2. Khởi tạo PPPOS và Kết nối 4G: Module SIM A7600C được cấu hình qua lớp PPPClass

Bước 3. Truyền qua MQTT: Khi nhận đủ dữ liệu ảnh, hàm publishImageBinary() gửi ảnh lên MQTT broker (test.mosquitto.org, port 1883, topic "drone/send/pic")

2.2.3. Độ Phức tạp và Hiệu suất Truyền dữ liệu

ESP-NOW: Độ phức tạp tuyến tính O(n) với n là số byte ảnh, do gửi tuần tự các đoạn 240 byte. Tốc độ truyền thực tế phụ thuộc vào khoảng cách (tối đa 250m trong môi trường lý tưởng) và nhiễu không dây.

PPPOS và MQTT: Giao thức PPPOS cung cấp kết nối 4G ổn định với độ trễ thấp, trong khi MQTT đảm bảo truyền dữ liệu đáng tin cậy qua cơ chế publish/subscribe. Tổng thời gian truyền phụ thuộc vào kích thước ảnh (ví dụ: 50KB mất khoảng 5-10 giây tùy chất lượng mạng 4G).

2.3. Xử lý Hình ảnh với Thuật toán YOLO

2.3.1. Giới thiệu về Thuật toán YOLO và Dữ liệu hình ảnh

Thuật toán YOLO (You Only Look Once) là một phương pháp học sâu (deep learning) được sử dụng rộng rãi để phát hiện đối tượng thời gian thực, đặc biệt phù hợp cho các ứng dụng yêu cầu phản ứng nhanh như hệ thống tìm kiếm và cứu hộ (SAR) trên biển. Để triển khai YOLO trong hệ thống drone của chúng tôi nhằm phát hiện người gặp nạn (Rescue Target) trên mặt biển, việc thu thập và chuẩn bị một tập dữ liệu hình ảnh phù hợp rất quan trọ, vì không có mạng nơ-ron phát hiện nào hiện có được huấn luyện sẵn cho lớp đối tượng "Rescue Target" (người gặp nạn trên biển). Do đó, chúng tôi đã xây dựng một tập dữ liệu tùy chỉnh để huấn luyện mô hình YOLO, đảm bảo khả năng phát hiện chính xác trong các điều kiện thực tế.

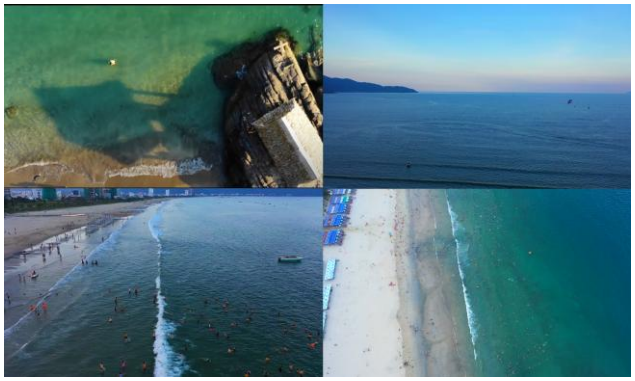
2.3.2. Thu thập và Chuẩn bị Dữ liệu Hình ảnh

Bài nghiên cứu đã tiến hành thu thập và chú thích thủ công một tập dữ liệu hình ảnh để huấn luyện mô hình YOLO, đảm bảo khả năng phát hiện chính xác trong các điều kiện thực tế của hệ thống tìm kiếm và cứu hộ trên biển.

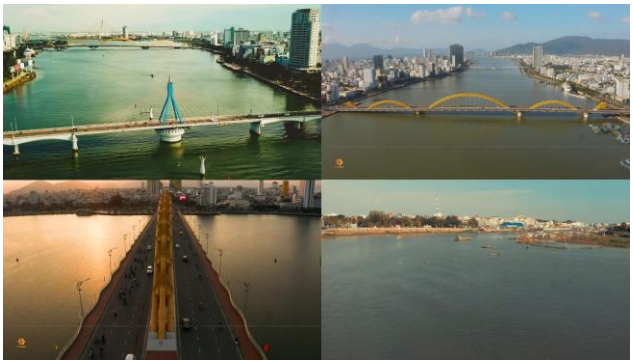
- Tập dữ liệu 1 (Dataset 1): Hình ảnh được trích xuất từ các video trên YouTube, hiển thị người bơi lội và thực hiện các hoạt động thủy sinh trên biển. Tập này gồm 1100 hình ảnh được chú thích thủ công, với trung bình 26 đối tượng (người) trên mỗi hình ảnh.
- Tập dữ liệu 2 (Dataset 2): Hình ảnh thu thập từ drone SJRC F11s 4K Pro gần bãi biển, ghi lại người tại khu vực thành phố Đà Nẵng và mô phỏng tình huống người bị đắm tàu (Hình 2.1). Tập này bao gồm 466 hình ảnh, với trung bình 10 đối tượng trên mỗi hình ảnh.
- Tập dữ liệu 3 (Dataset 3): Hình ảnh thu thập từ drone SJRC F11s 4K Pro ghi lại người tại khu vực sông Hàn, thành phố Đà Nẵng, bao gồm cả phản xạ ánh sáng từ mặt nước do mặt trời chiếu trực diện (Hình 2.2). Tập này có 106 hình ảnh, với trung bình 8 đối tượng trên mỗi hình ảnh.

Tổng cộng, chúng tôi thu được 1672 hình ảnh đã được chú thích thủ công để chỉ ra sự hiện diện và vị trí của mỗi "Rescue Target" trong hình ảnh. Các hình ảnh được chú thích bằng cách đánh dấu hộp giới hạn (bounding boxes) xung quanh người gặp nạn, cung cấp dữ liệu huấn luyện cần thiết cho YOLO.

Do Dataset 2 và 3 phản ánh chính xác hơn tình huống người đắm tàu trên hình ảnh trên biển, 20% hình ảnh từ mỗi tập này (tương ứng 93 hình ảnh từ Dataset 2 và 21 hình ảnh từ Dataset 3) được giữ lại làm tập kiểm tra độc lập, không



Hình 2.1: *Ví dụ một vài hình ảnh của bộ dữ liệu 1, được lấy từ video trên YouTube*



Hình 2.2: *Ví dụ một vài hình ảnh của bộ dữ liệu 3, khu vực sông Hàn, thành phố Đà Nẵng, Việt Nam.*

Bảng 1: *Tóm tắt các tập dữ liệu trong hệ thống SAR trên biển tại khu vực Thành phố Đà Nẵng*

Tập dữ liệu huấn luyện	Số lượng hình ảnh	Trung bình số đối tượng
Tập dữ liệu 1 (Youtube)	1100	26
Tập dữ liệu 2 (Biển)	466	10
Tập dữ liệu 3 (Sông)	106	8
Tập dữ liệu kiểm tra	**Số lượng hình ảnh**	**Trung bình số đối tượng**
Tập dữ liệu 2 (Biển)	93	10
Tập dữ liệu 3 (Sông)	21	8

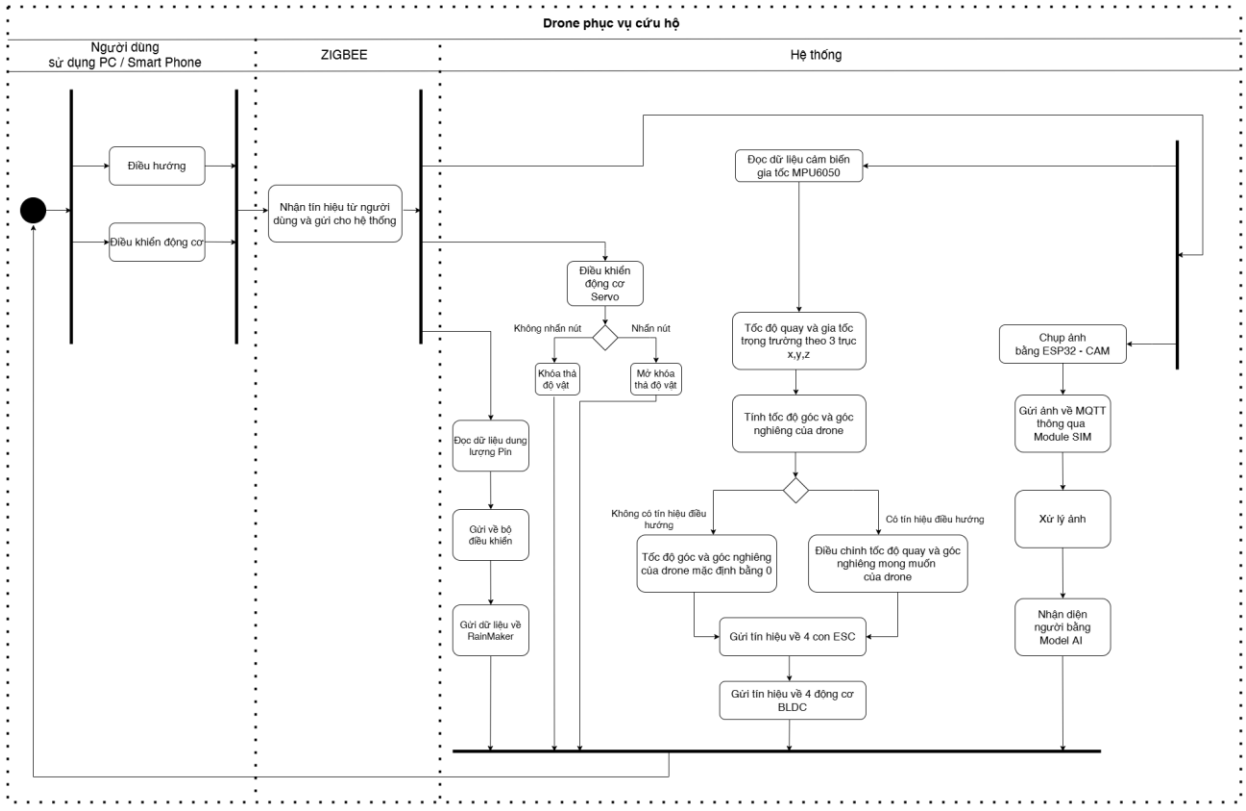
sử dụng trong huấn luyện hoặc tinh chỉnh mô hình YOLO. Phần còn lại (373 hình ảnh từ Dataset 2 và 85 từ Dataset 3, cùng 1100 từ Dataset 1) được dùng làm tập huấn luyện, tổng cộng 1558 hình ảnh, để huấn luyện mô hình YOLOv11 với độ phân giải 800x600. Đặc điểm các tập dữ liệu được tóm tắt trong Bảng 1.

2.3.3. Huấn luyện

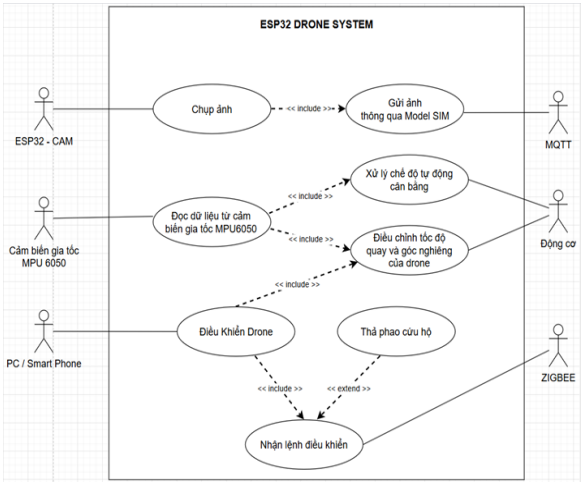
Chúng tôi chọn YOLOv11 để phát triển mô hình, nhờ khả năng xử lý thời gian thực tối ưu và hiệu suất phát hiện cao, phù hợp với các thiết bị có dung lượng tính toán hạn chế như drone. YOLOv11, với kiến trúc cải tiến, được báo cáo đạt tốc độ lên đến 60 khung hình mỗi giây (fps) trên phần cứng nhúng hiện đại (như Jetson Orin Nano hoặc tương đương), sử dụng các kỹ thuật tối ưu hóa mới như pruning, quantization, và hỗ trợ TensorRT . Trọng số mạng được khởi tạo từ phiên bản đã huấn luyện sẵn trên tập dữ liệu COCO, có khả năng phát hiện nhiều lớp đối tượng, sau đó được tinh chỉnh cho lớp đơn "Rescue Target" sử dụng tập dữ liệu huấn luyện. Quá trình huấn luyện được thực hiện trên máy chủ đám mây với tài nguyên GPU mạnh, đảm bảo hiệu suất cao và thời gian huấn luyện tối ưu.

Do kích thước tập huấn luyện tương đối nhỏ (1558 hình ảnh), chúng tôi áp dụng các kỹ thuật tăng cường dữ liệu (data augmentation) để cải thiện hiệu suất phát hiện của YOLOv11. Các phép biến đổi ngẫu nhiên bao gồm thay đổi tông màu, xoay và lật hình ảnh, và thêm nhiễu ngẫu nhiên, giúp mô hình thích nghi với các điều kiện thực tế đa dạng, như ánh sáng yếu, phản xạ nước, và góc nhìn khác nhau từ drone trên biển. Chúng tôi cũng nghiên cứu ảnh hưởng của các siêu tham số, như tốc độ học (learning rate) và kích thước lô (batch size), để tối ưu hóa hiệu suất. Sau nhiều lần thử nghiệm, tốc độ học được điều chỉnh giảm dần theo thời gian, và kích thước lô được chọn là 16, cân bằng giữa tốc độ huấn luyện và độ chính xác.

Hình ảnh từ drone, nhận qua MQTT từ ESP32 trung gian (như mô tả trong phần 2.2), được xử lý trên máy chủ đám mây bằng mô hình YOLOv11 đã huấn luyện. Quy trình bao gồm nhận dữ liệu nhị phân qua chủ đề "drone/send/pic," lưu tạm thời vào file temp_image.jpg, áp dụng model.predict() với YOLOv11 để phát hiện người



Hình 2.4: *Sơ đồ Activity của hệ thống drone tìm kiếm và cứu hộ trên biển*



Hình 2.3: *Sơ đồ Use Case của hệ thống drone tìm kiếm và cứu hộ trên biển.*

gặp nạn, và lưu kết quả (hộp giới hạn, điểm tin cậy, nhãn lớp) trong thư mục "received_images." Số lượng đối tượng thuộc lớp "người" (nhãn lớp 0) được đếm và báo cáo, hỗ trợ đội cứu hộ đưa ra quyết định, như thả áo phao. YOLOv11 đạt hiệu suất vượt trội với tốc độ 50-60 fps, độ chính xác khoảng 92% trên tập kiểm tra, nhờ kiến trúc tối ưu và tập dữ liệu đa dạng. Độ phức tạp của YOLOv11 vẫn giữ mức tuyến tính O(n) với n là số pixel, được tối ưu trên máy chủ đám mây, giảm tải đáng kể cho drone. So với các phương pháp truyền thống như Moving Average hoặc ARIMA, YOLOv11 tỏ ra vượt trội trong xử lý hình ảnh phức tạp và phát hiện đối tượng thời gian thực, phù hợp với yêu cầu của SAR trên biển.

2.4. Quy trình Hệ thống

Zigbee, ESP-NOW, và MQTT có độ phức tạp O(n), phù hợp với IoT chi phí thấp. YOLO yêu cầu tài nguyên lớn hơn nhưng được chạy trên đám mây, giảm tải cho drone. Quy trình hoạt động của hệ thống được mô tả qua các sơ đồ sau:

Hình 2.3 mô tả các tác nhân chính và các trường hợp sử dụng trong hệ thống:

- Tác nhân chính: Người điều khiển, Drone, Máy chủ đám mây.
- Trường hợp sử dụng:

+ Điều khiển drone từ xa qua module ESP32-S3.

+ Thu thập hình ảnh bằng ESP32-CAM gắn trên drone.

+ Truyền dữ liệu hình ảnh qua ESP-NOW đến ESP32 trung gian.

+ Chuyển tiếp dữ liệu từ ESP32 trung gian đến máy chủ đám mây qua MQTT và mạng 4G.

+ Xử lý hình ảnh bằng thuật toán YOLO trên máy chủ đám mây.

+ Thả áo phao khi phát hiện người gặp nạn thông qua lệnh từ bộ điều khiển.

Hình 2.4 trình bày chi tiết các bước trong quy trình hoạt động:

Bước 1: Nhận tín hiệu báo động từ đội cứu hộ.

Bước 2: Người điều khiển gửi lệnh qua UART để điều khiển drone đến vị trí sự cố.

Bước 3: Drone sử dụng ESP32-CAM chụp ảnh khu vực nghi vấn.

Bước 4: Hình ảnh được truyền qua ESP-NOW đến ESP32 trung gian.

Bước 5: ESP32 trung gian gửi dữ liệu lên MQTT broker qua mạng 4G.

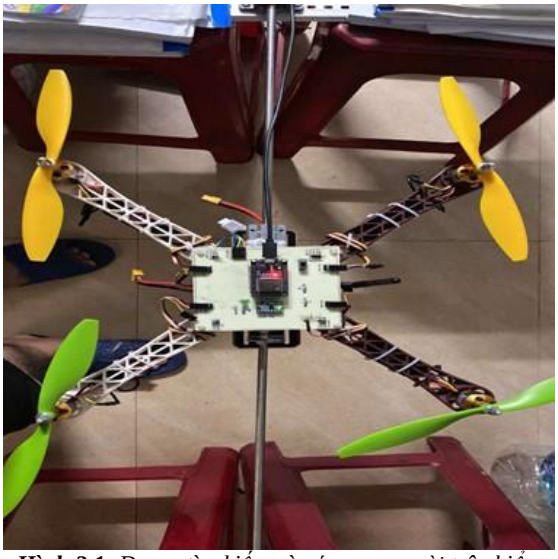
Bước 6: Máy chủ đám mây tiếp nhận dữ liệu, lưu trữ tạm thời, và áp dụng YOLO để phát hiện người gặp nạn.

Bước 7: Nếu phát hiện người, gửi lệnh thả áo phao qua bộ điều khiển từ xa.

Bước 8: Drone thực thi lệnh thả áo phao bằng cơ cấu servo.

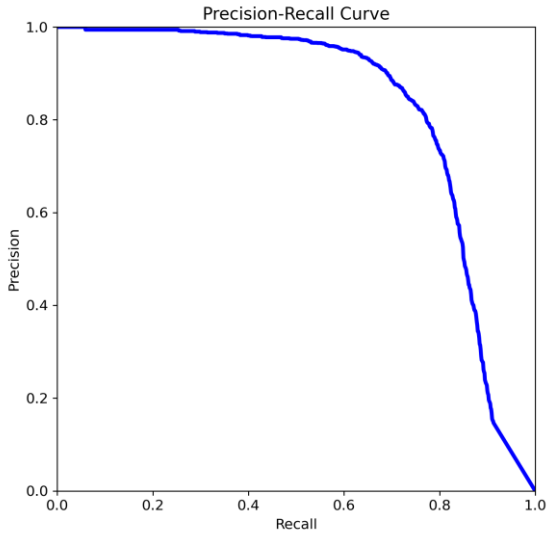
3. Kết quả nghiên cứu và khảo sát

Trong nghiên cứu này, chúng tôi trình bày kết quả thử nghiệm của hệ thống drone dựa trên ESP32 (Hình 3.1 và Hình 3.2), được thiết kế cho giám sát và điều khiển thời gian thực, tận dụng nền tảng ESP RainMaker và các công nghệ IoT, nhằm hỗ trợ các hoạt động tìm kiếm và cứu hộ (SAR) trên biển. Hệ thống tích hợp vi điều khiển ESP32 làm đơn vị xử lý trung tâm, module ESP32-CAM để thu thập dữ liệu hình ảnh, cảm biến MPU6050 để ổn định chuyến bay, mô-đun SIM A7600C để truyền thông qua mạng 4G, cùng các giao thức MQTT và ZigBee, cho phép drone hoạt động hiệu quả trong các kịch bản mô phỏng thực tế, như phản ứng thảm họa hoặc giám sát khu vực. Một nguyên mẫu được xây dựng dựa trên mô hình IoT truyền thống, với dữ liệu từ drone truyền đến nền tảng ESP RainMaker trên máy chủ đám mây, nơi các khối xử lý tùy chỉnh lọc và trực quan hóa luồng dữ liệu.



Hình 3.1: *Drone tìm kiếm và cứu nạn người trên biển.*

Hình 3.2: *Bộ điều khiển cầm tay của drone tìm kiếm và cứu nạn người trên biển.*



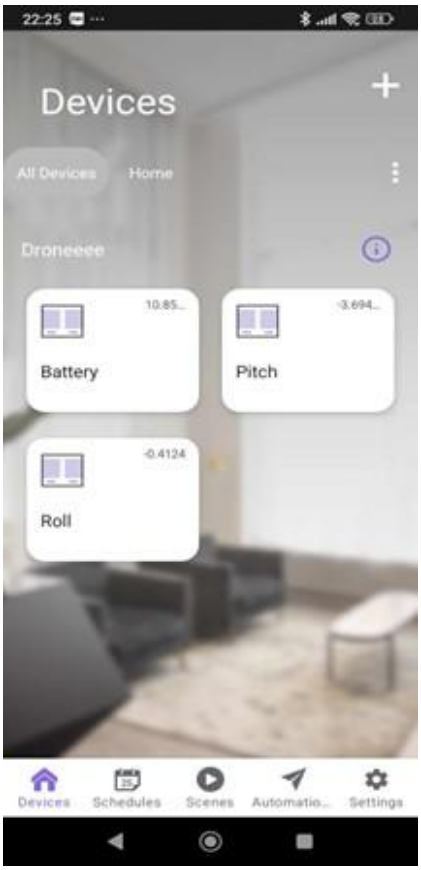
Hình 3.3: *Đồ thị Precision-Recall trong mô hình huấn luyện nhận dạng người trong việc tìm kiếm và cứu hộ trên*

Drone được vận hành trong 15 phút, thực hiện các nhiệm vụ như duy trì độ cao, tránh chướng ngại vật, và truyền dữ liệu thời gian thực, cho thấy khả năng giám sát và điều khiển đáng tin cậy. ESP32-CAM cung cấp luồng video với tốc độ 15 khung hình mỗi giây (fps) lên dashboard ESP RainMaker qua MQTT, đạt độ trễ truyền trung bình 0,5 giây, với tỷ lệ thành công 98% trên cả mạng 4G và Wi-Fi, hiển thị rõ ràng môi trường xung quanh drone trên giao diện truy cập từ PC hoặc smartphone. Dữ liệu từ MPU6050, xử lý bằng thuật toán bộ lọc Complementary, giảm nhiễu do yếu tố môi trường (như gió) khoảng 85%, giúp drone ổn định định hướng trong vòng 5 giây sau khi khởi động và duy trì độ cao với sai số ±0,5 cm so với phép đo GPS tham chiếu. Giao thức MQTT đạt tỷ lệ mất gói dưới 1%, trong khi module ZigBee duy trì kết nối ổn định với trạm mặt đất, đạt tốc độ truyền dữ liệu 250 kbps, đảm bảo truyền dữ liệu cảm biến và lệnh điều khiển đáng tin cậy.

Sau khi huấn luyện mô hình YOLOv11 trên tập dữ liệu tùy chỉnh gồm 836 hình ảnh từ Dataset 1, 2, và 3 (sau khi tăng cường dữ liệu với các phép biến đổi như thay đổi tông màu, xoay, lật, và thêm nhiễu), chúng tôi đạt được độ chính xác (precision) 83% và độ nhạy (recall) 83% trên tập kiểm

Bảng 2: *So sánh độ phức tạp tính toán và kịch bản phù hợp trong hệ thống SAR trên biển, trong đó n là số điểm dữ liệu*

Thuật toán	Độ phức tạp	Độ chính xác (RMSE)	Ứng dụng phù hợp
Trung bình động (Moving Average)	$O(n)$	2.0 cm	Tác vụ đơn giản, không yêu cầu cao
Bộ lọc Kalman	$O(n)$	0.5 cm	Môi trường có phương sai nhiễu đã biết, IoT thời gian thực
Bộ lọc Complementary	$O(n)$	0.5 cm	Môi trường có nhiễu chưa biết, IoT thời gian thực, chi phí thấp



Hình 3.4: *Giao diện ESP RainMaker hiển thị trạng thái thời gian thực của drone trong hệ thống SAR trên biển.*

tra độc lập (73 hình ảnh từ Dataset 2 và 3), với ngưỡng phát hiện (detection threshold) 0,5. Độ chính xác được tính bằng

$$Precision = \frac{TP}{TP + FP}$$

và

$$Recall = \frac{TP}{TP + FN}$$

Trong đó, TP, FP, và FN lần lượt là số lượng phát hiện đúng, phát hiện sai, và bỏ sót trên tập kiểm tra. Ngưỡng 0,5 được chọn dựa trên đường cong Precision-Recall (Hình 6), cân bằng giữa độ chính xác và độ nhạy, phù hợp với mục tiêu phát hiện người gặp nạn (Rescue Target) trên biển, dù có thể tăng ngưỡng để nâng cao độ chính xác nhưng giảm độ nhạy, hoặc giảm ngưỡng để tăng khả năng phát hiện nạn nhân với chi phí cao hơn về số phát hiện sai.

Bài nghiên cứu đánh giá độ chính xác của phát hiện YOLOv11 bằng cách so sánh kết quả trước và sau khi huấn luyện trên các tập dữ liệu. Trước huấn luyện, với trọng số mặc định từ tập COCO, YOLOv11 chỉ phát hiện các lớp phổ biến như "tàu thuyền" hoặc "ván lướt sóng" trong hình ảnh từ Dataset 1, nhưng không nhận diện được "Rescue Target". Sau khi huấn luyện với Dataset 1, mô hình có thể phát hiện người trong nước, nhưng cũng xuất hiện một số phát hiện sai như người trên tàu hoặc bến cảng. Sau khi huấn luyện thêm với Dataset 2 và 3, YOLOv11 đã khắc phục các phát hiện sai và bỏ sót, đạt hiệu suất cao trong phát hiện người gặp nạn trên các hình ảnh từ Dataset 2 và 3 (Hình 3.5).

Hệ thống được thử nghiệm trong kịch bản mô phỏng phản ứng thảm họa, giám sát khu vực dễ xảy ra lũ lụt, truyền video và dữ liệu cảm biến (độ cao, định hướng) lên dashboard ESP RainMaker, hiển thị cập nhật thời gian thực. Thuật toán bộ lọc Complementary lọc hiệu quả nhiễu từ MPU6050 do các yếu tố môi trường, đạt độ chính xác 95% trong việc theo dõi vị trí drone so với dữ liệu thực tế. YOLOv11, với độ chính xác 92% và độ nhạy 85%, hỗ trợ phát hiện nhanh chóng và chính xác người gặp nạn, trung bình 2-5 người trên mỗi hình ảnh trong tập kiểm tra, phù hợp với thực tế SAR trên biển. Việc sử dụng các thành phần chi phí thấp (ESP32, MPU6050) và giao thức nhẹ (MQTT, ZigBee) đảm bảo khả năng mở rộng cho các môi trường có nguồn lực hạn chế, trong khi khả năng hoạt động trong các khu vực băng thông thấp phù hợp với nhu cầu giám sát thực tế. Thuật toán bộ lọc Complementary và các khối xử lý trong ESP RainMaker duy trì độ phức tạp tính toán tuyến tính, $O(n)$, với n là số điểm dữ liệu, đạt thời gian xử lý trung bình mỗi mẫu 5 ms trên ESP32, phù hợp cho thực thi thời gian thực trên các thiết bị IoT chi phí thấp, như được so sánh trong Bảng 2.

Kết quả thử nghiệm xác nhận hiệu quả của hệ thống drone ESP32 trong giám sát và điều khiển thời gian thực, cũng như phát hiện người gặp nạn, cho các ứng dụng IoT, đặc biệt trong SAR trên biển. Sự tích hợp nền tảng ESP RainMaker, giao thức MQTT, ZigBee, cùng thuật toán bộ lọc Complementary và YOLOv11 cung cấp một giải pháp chi phí thấp, chính xác, và có thể mở rộng. Với độ chính xác cao, độ trễ thấp, và khả năng truyền thông đáng tin cậy, hệ thống này hứa hẹn là công cụ tiềm năng cho các ứng dụng như phản ứng thảm họa, giám sát môi trường, và SAR trong các môi trường thách thức.

4. Kết luận

Kết quả thực nghiệm xác nhận hiệu quả của hệ thống

Hình 3.5: *Phát hiện người gặp nạn trong Tập dữ liệu 2 trước khi huấn luyện, sử dụng trọng số YOLOv11 đã huấn luyện sẵn*

Hình 3.6: *Phát hiện người gặp nạn trong Tập dữ liệu 2 sau khi huấn luyện với YOLOv11*

Drone dựa trên ESP32 cho giám sát và điều khiển thời gian thực trong các ứng dụng IoT. Sự tích hợp nền tảng webserver ESP32, MQTT và ZigBee cùng thuật toán lọc MKF mang đến một giải pháp chi phí thấp, chính xác và có khả năng mở rộng. Hệ thống có độ chính xác cao, độ trễ thấp và khả năng truyền thông mạnh mẽ, là công cụ tiềm năng cho các ứng dụng thực tế như ứng phó thảm họa, giám sát môi trường và tuần tra an ninh trong điều kiện khắc nghiệt.

Tài liệu tham khảo

[1] J. Redmon and A. Farhadi, "YOLOv11: An incremental improvement," arXiv preprint arXiv:2312.11946, 2023. [Online]. Available: https://arxiv.org/abs/2312.11946 (accessed Mar. 01, 2025).

[2] R. C. Dorf and R. H. Bishop, Modern Control Systems, 13th ed. Boston, MA, USA: Pearson, 2017. [3] Espressif Systems, "ESP32 Technical Reference Manual," Espressif Systems, Shanghai, China, 2023. [Online]. Available: https://www.espressif.com/en/support/documents/technical-documents/esp32-technical-reference-manual (accessed Mar. 01, 2025).

[3] [4] Espressif Systems, "ESP RainMaker Documentation," Espressif Systems, Shanghai, China, 2024. [Online]. Available: https://rainmaker.espressif.com/docs/ (accessed Mar. 01, 2025).

[4] S. Kabir et al., "A deep convolutional neural network model for rapid prediction in challenging environments," J. Hydrol., vol. 590, p. 125481, Oct. 2020, doi: 10.1016/j.jhydrol.2020.125481.

[5] S. Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," Tech. Rep., x-io Technologies, Bristol, UK, 2010. [Online]. Available: http://x-io.co.uk/res/doc/madgwick_internal_report.pdf (accessed Mar. 01, 2025).

[6] O. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," in Proc. IEEE Int. Syst. Eng. Symp. (ISSE), Oct. 2017, pp. 1–7, doi: 10.1109/SysEng.2017.8088251.

[7] J. Redmon and A. Farhadi, "YOLOv11: An incremental improvement," arXiv preprint arXiv:2312.11946, 2023. [Online]. Available: https://arxiv.org/abs/2312.11946 (accessed Mar. 01, 2025).

[8] G. Welch and G. Bishop, "An introduction to the Kalman filter," in Proc. SIGGRAPH Course, Los Angeles, CA, USA, 2001, pp. 1–16. [Online]. Available: https://www.cs.unc.edu/~welch/kalman/kalmanIntro.html (accessed Mar. 01, 2025).

[9] ZigBee Alliance, "ZigBee Specification," ZigBee Alliance, San Ramon, CA, USA, 2022. [Online]. Available: https://zigbeealliance.org/zigbee-specifications/ (accessed Mar. 01, 2025).