

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
ĐẠI HỌC QUỐC GIA HÀ NỘI



**THEO DÕI VÀ MÔ PHỎNG ĐƯỜNG ĐI
CỦA THIẾT BỊ BAY KHÔNG NGƯỜI LÁI
SỬ DỤNG YOLOV9 VÀ DEEPSORT**

BÀI TẬP LỚN
Học phần : Quang Điện Tử

Hà Nội, 2024

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
ĐẠI HỌC QUỐC GIA HÀ NỘI

**THEO DÕI VÀ MÔ PHỎNG ĐƯỜNG ĐI
CỦA THIẾT BỊ BAY KHÔNG NGƯỜI LÁI
SỬ DỤNG YOLOV9 VÀ DEEPSORT**

BÀI TẬP LỚN
Học phần : Quang Điện Tử

Sinh viên:

Bùi Khương Duy (NT) - MSSV: 21020253

Giảng viên:

TS. Phạm Đức Quang

Hà Nội, 2024

Theo dõi và mô phỏng đường đi của thiết bị bay không người lái sử dụng Yolov9 và deepSORT

Tóm tắt

Bài tập lớn của nhóm trình bày một hệ thống phát hiện và theo dõi drone thời gian thực sử dụng YOLOv9 (You Only Look Once 9) kết hợp với thuật toán DeepSORT (Simple Online and Realtime Tracking with a Deep Association Metric). Hệ thống nhằm giải quyết những thách thức ngày càng tăng trong việc giám sát và theo dõi drone, bằng cách cung cấp một giải pháp hiệu quả để phát hiện, theo dõi và mô phỏng đường đi drone trong các điều kiện môi trường khác nhau. Phương pháp của nhóm áp dụng khả năng phát hiện đối tượng của YOLOv9 để phát hiện drone, sau đó sử dụng DeepSORT để duy trì việc theo dõi nhất quán qua các khung hình video. Chương trình có thể xử lý luồng video theo thời gian thực và cung cấp tọa độ không gian chính xác so với tâm khung hình, sau đó sử dụng các phép tính toán, ước lượng tọa độ và quãng đường di chuyển thực tế, cho phép giám sát vị trí drone một cách chính xác.

Mục lục

ABSTRACT	i
1 PHƯƠNG PHÁP THỰC HIỆN	1
1.1 YOLOv9	1
1.2 DeepSORT	2
1.3 Phương pháp thực hiện trong bài tập lớn	3
1.3.1 Kiến trúc hệ thống	3
1.3.2 Quy trình thực hiện	5
1.3.2.1 Phát hiện drone:	5
1.3.2.2 Theo dõi drone bằng deepSORT:	5
1.3.2.3 Xử lý tọa độ	7
2 KẾT QUẢ VÀ THẢO LUẬN	10
2.1 Tài nguyên và môi trường	10
2.1.1 Dataset và huấn luyện	10
2.1.1.1 Phân bố dữ liệu:	
11	
2.1.1.2 Confusion matrix:	
12	
2.1.1.3 Kết quả	
12	
2.1.2 Kết quả huấn luyện	13
2.2 Kết quả thực hiện phát hiện và theo dõi	14
2.2.1 Xử lý trực tiếp trên khung hình	14
2.2.2 Biểu diễn tọa độ và đường đi	16
3 KẾT LUẬN	18

Chương 1

Phương pháp thực hiện

1.1 YOLOv9

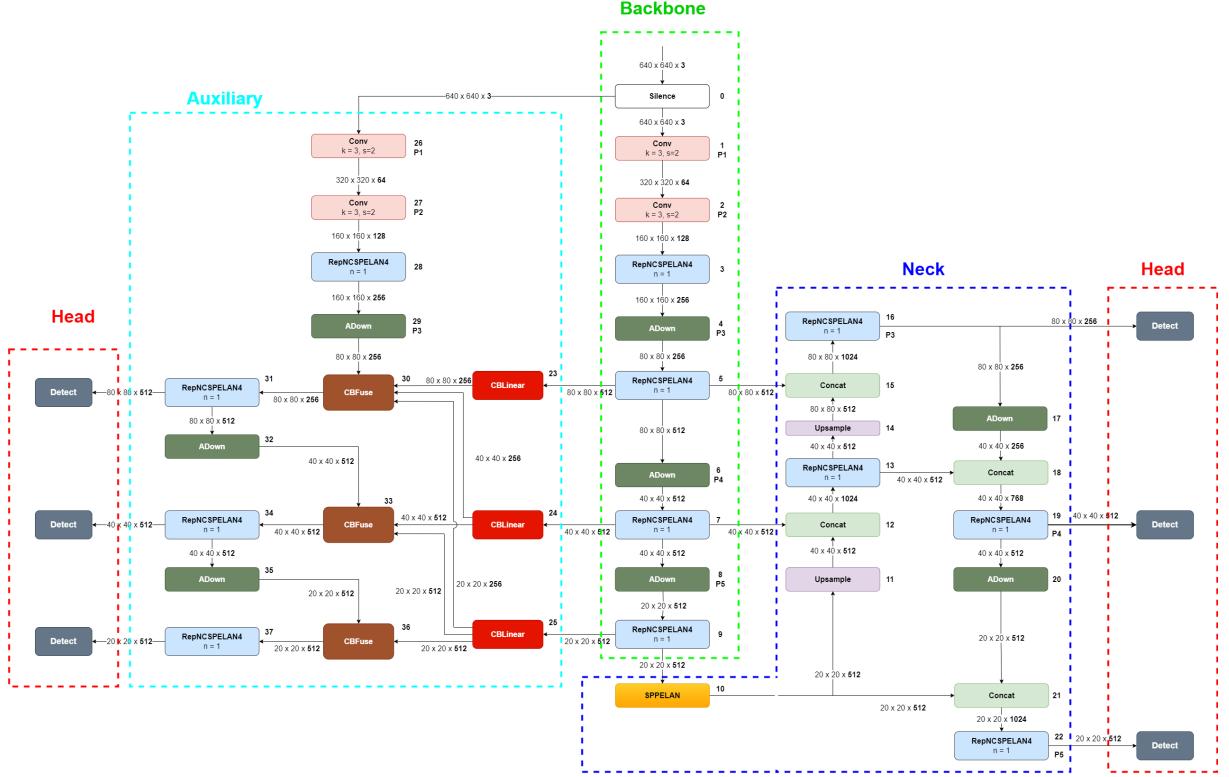
YOLOv9 kế thừa và phát triển từ kiến trúc của YOLOv8, với ba thành phần chính: backbone (xương sống), neck (cổ) và head (đầu) cùng với một thành phần phụ trợ (auxiliary), như minh họa trong Hình 1[1]. Về cấu trúc backbone, YOLOv9 sử dụng mạng RepNCSPELAN4 làm nền tảng, với các lớp tích chập (Conv) và cơ chế Silence để trích xuất đặc trưng. Backbone xử lý ảnh đầu vào kích thước $640 \times 640 \times 3$ và tạo ra các feature map ở các tỷ lệ khác nhau ($320 \times 120 \times 64$, $160 \times 160 \times 128$, và $80 \times 80 \times 256$) thông qua các khối ADown.

Phần neck của YOLOv9 được thiết kế với cấu trúc phức tạp hơn so với các mô hình trước, bao gồm nhiều khối RepNCSPELAN4 kết hợp với các khối nối (Concat) và up-sample. Trong đó neck sử dụng khối SPPELAN để tăng cường khả năng trích xuất đặc trưng đa tỷ lệ. Cấu trúc này cho phép mô hình xử lý hiệu quả các đối tượng ở nhiều kích thước khác nhau, từ 80×80 đến 20×20 pixels.

Về phần head, YOLOv9 sử dụng ba nhánh Detect riêng biệt cho các tỷ lệ khác nhau (80×80 , 40×40 , và 20×20), mỗi nhánh được kết nối với một feature map tương ứng. Mô hình còn có thêm thành phần auxiliary với các khối CBFuse và CBLinear, đóng vai trò hỗ trợ quá trình học và cải thiện độ chính xác của mô hình.

Kiến trúc này cho phép YOLOv9 cân bằng được giữa độ chính xác và tốc độ

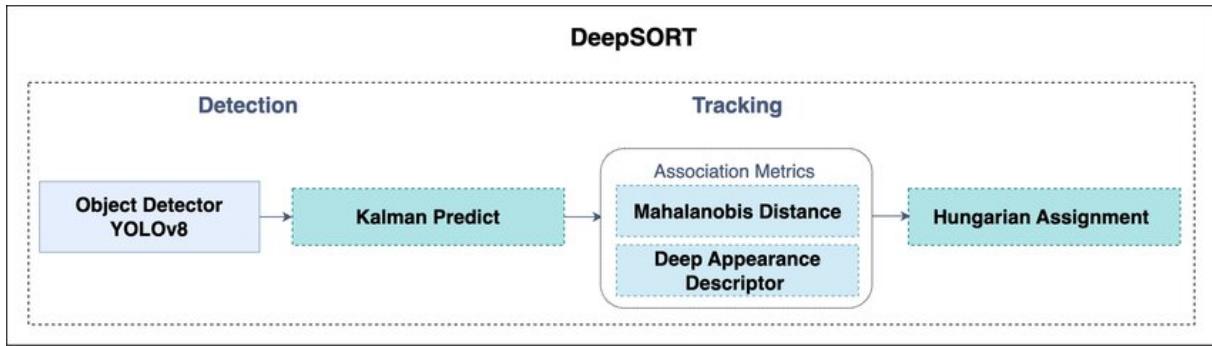
Hình 1.1.1: Sơ đồ khôi kiến trúc YOLOv9, thể hiện các thành phần backbone, neck, head và auxiliary[1]



xử lý, đặc biệt hiệu quả trong việc phát hiện các đối tượng nhỏ như drone. Việc sử dụng các khôi RepNCSPELAN4 và cơ chế auxiliary giúp mô hình học được các đặc trưng phong phú hơn, trong khi vẫn duy trì khả năng xử lý thời gian thực.

1.2 DeepSORT

DeepSORT là một bước tiến quan trọng trong lĩnh vực thị giác máy tính, đặc biệt trong bài toán theo dõi đa đối tượng (Multiple Object Tracking). Được phát triển như một phiên bản nâng cấp của thuật toán SORT (Simple Online Realtime Tracking), DeepSORT không chỉ kế thừa khả năng xử lý thời gian thực mà còn tích hợp thêm sức mạnh của học sâu để nâng cao độ chính xác trong việc theo dõi đối tượng.[2]



Hình 1.2.1: Cấu trúc của deepSORT

Điểm đột phá của DeepSORT nằm ở việc tích hợp mạng neural học sâu để trích xuất các đặc trưng ngoại hình (appearance features) của đối tượng. Cụ thể, thuật toán sử dụng một mạng neural tích chập được huấn luyện trước để tạo ra các vector đặc trưng có khả năng phân biệt cao giữa các đối tượng khác nhau. Kết hợp với bộ lọc Kalman để dự đoán chuyển động và thuật toán Hungarian để kết hợp các detection với tracks, DeepSORT tạo nên một hệ thống theo dõi mạnh mẽ và ổn định.

So với SORT nguyên bản, DeepSORT giải quyết được một trong những hạn chế lớn nhất là hiện tượng hoán đổi định danh (identity switches) - tình huống thuật toán nhầm lẫn giữa các đối tượng khi chúng di chuyển gần nhau hoặc có quỹ đạo chồng chéo. Điều này đặc biệt quan trọng trong ứng dụng theo dõi drone, nơi các đối tượng thường xuyên thay đổi hướng di chuyển và có thể bay gần nhau.

1.3 Phương pháp thực hiện trong bài tập lớn

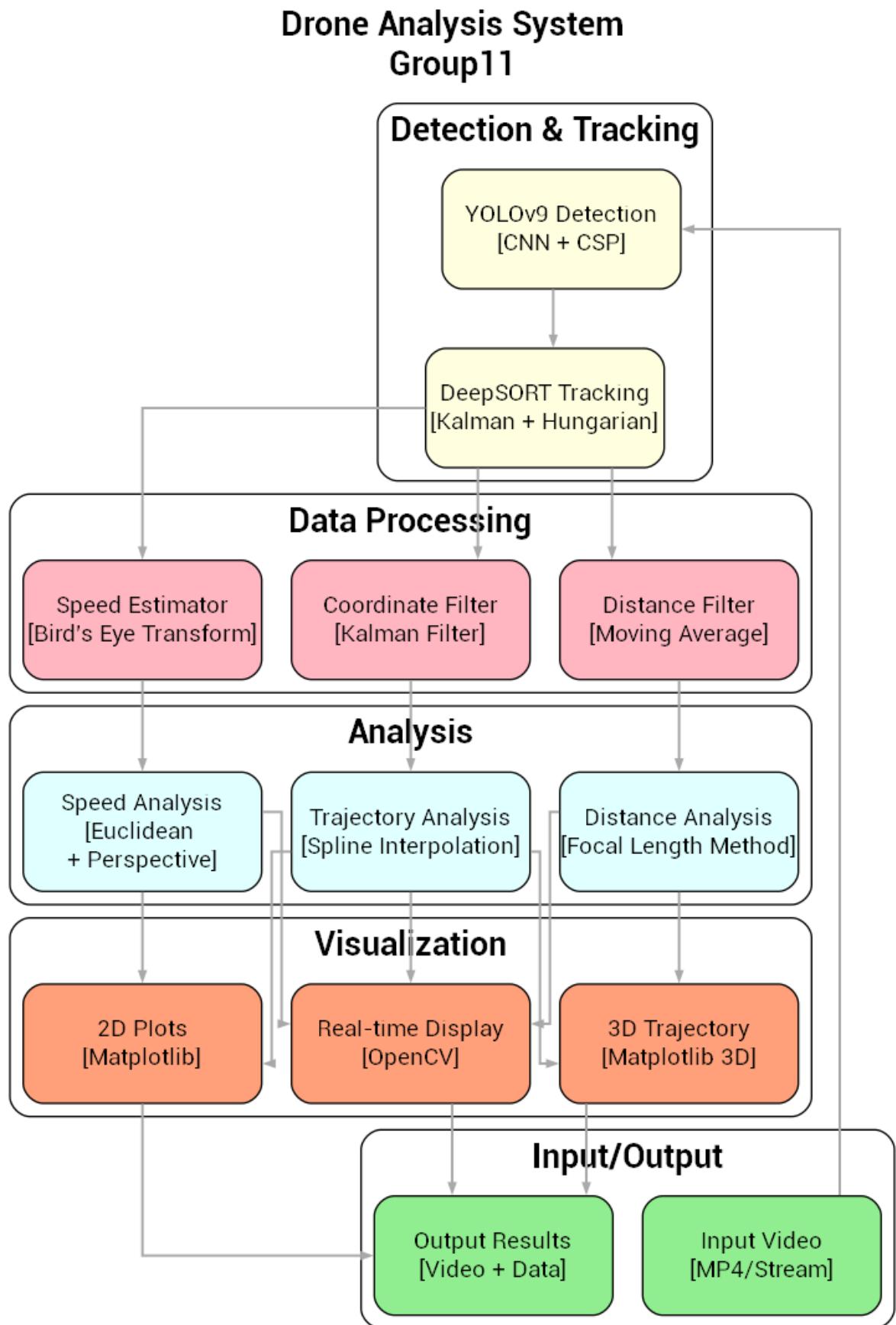
1.3.1 Kiến trúc hệ thống

Hệ thống được thiết kế với kiến trúc pipeline gồm ba thành phần chính:

1. Module phát hiện drone sử dụng YOLOv9
2. Module theo dõi đối tượng sử dụng DeepSORT
3. Module phân tích quỹ đạo và tính toán khoảng cách

Quy trình xử lý được thực hiện theo thời gian thực được biểu diễn trong hình sau.

Hình 1.3.1: Sơ đồ hệ thống được thực hiện trong bài



1.3.2 Quy trình thực hiện

1.3.2 Phát hiện drone: YOLOv9 được cấu hình với các tham số tối ưu cho bài toán phát hiện drone:

Kích thước đầu vào: 640×640 pixels

Nguồn tin cậy (confidence threshold): 0.35

Nguồn IoU: 0.45

Số lượng phát hiện tối đa: 1000

Những tham số này là các giá trị được tối ưu hóa thông qua quá trình thử nghiệm để cân bằng giữa độ chính xác và khả năng phát hiện. Nếu đặt cao sẽ tăng độ chính xác nhưng lại giảm khả năng phát hiện, ngược lại quá thấp sẽ giảm độ chính xác, dễ nhận diện nhầm vật thể khác.

YOLOv9 xử lý mỗi khung hình và tạo ra các bounding boxes kèm theo điểm tin cậy cho mỗi drone được phát hiện.

```
1 im = letterbox(im0s, imgsz, stride=stride, auto=True)[0]
2 im = im.transpose((2, 0, 1))[:, :, ::-1]
3 im = np.ascontiguousarray(im)
4 im = torch.from_numpy(im).to(device)
5 im = im.half() if model.fp16 else im.float()
6 im /= 255
```

Listing 1.1: Tiền xử lý ảnh cho YOLOv9

1.3.2 Theo dõi drone bằng deepSORT: DeepSORT được triển khai với các cải tiến:

Max age: 5 frames để duy trì ID khi mất dấu tạm thời

Sử dụng Kalman Filter để dự đoán vị trí

Tích hợp deep appearance descriptor để phân biệt các đối tượng

DeepSORT sử dụng thông tin từ các khung hình trước đó kết hợp với dữ liệu phát hiện mới để duy trì ID theo dõi nhất quán cho mỗi drone[2]. Quá trình này rất quan trọng trong việc duy trì tracking liên tục, ngay cả khi drone tạm thời bị che khuất hoặc YOLO không phát hiện được trong một số khung hình. Vì vậy trong project đã giới hạn duy trì tracking cùng 1 ID cho 5 khung hình liên tiếp, ví dụ trong frame 3, drone số 10 bị che mất, hoặc chương trình không nhận diện được con drone số 10 này, mà tới frame số 4 lại nhận diện được, thì bounding box

cho drone số 10 vẫn được duy trì, và trajectory của drone 10 vẫn được lưu.

Kalman Filter trong DeepSORT:

DeepSORT sử dụng Kalman Filter để dự đoán vị trí tiếp theo của đối tượng. Trạng thái của đối tượng được biểu diễn bởi vector:

$$\mathbf{x} = [x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}]^T$$

Trong đó:

(x, y) là tọa độ trung tâm bounding box

a là tỷ lệ khung hình

h là chiều cao

$(\dot{x}, \dot{y}, \dot{a}, \dot{h})$ là các thành phần vận tốc tương ứng

Kalman Filter update:

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$$

$$x_k = \hat{x}_k + K_k(z_k - H_k \hat{x}_k)$$

$$P_k = (I - K_k H_k) P_k$$

Phương trình cập nhật trạng thái:

$$\mathbf{x}_k = \mathbf{F} \mathbf{x}_{k-1} + \mathbf{w}_k$$

Phương trình đo lường:

$$\mathbf{z}_k = \mathbf{H} \mathbf{x}_k + \mathbf{v}_k$$

Trong đó:

\mathbf{F} là ma trận chuyển trạng thái

\mathbf{H} là ma trận đo lường

\mathbf{w}_k và \mathbf{v}_k là nhiễu Gaussian

Metric khoảng cách trong DeepSORT:

Khoảng cách Mahalanobis được sử dụng để đo lường sự tương đồng giữa track và detection:

$$d_{(i,j)}^{(1)} = (\mathbf{d}_j - \mathbf{y}_i)^T \mathbf{S}_i^{-1} (\mathbf{d}_j - \mathbf{y}_i)$$

Trong đó:

\mathbf{d}_j là measurement vector của detection thứ j

\mathbf{y}_i là predicted measurement của track thứ i

\mathbf{S}_i là covariance matrix

Khoảng cách cosine giữa các feature vectors:

$$d_{(i,j)}^{(2)} = \min\{1 - \mathbf{r}_j^T \mathbf{r}_i^k | \mathbf{r}_i^k \in \mathcal{R}_i\}$$

Trong đó \mathbf{r} là các feature vectors được trích xuất từ appearance model.

1.3.2 Xử lý tọa độ Việc chuyển đổi từ tọa độ tuyệt đối sang tọa độ tương đối được thực hiện thông qua các công thức sau:

Tọa độ tâm khung hình:

$$x_{center} = \frac{W}{2}, y_{center} = \frac{H}{2}$$

Trong đó W và H là chiều rộng và chiều cao của khung hình.

Tọa độ tương đối của drone:

$$x_{rel} = x_{obj} - x_{center}$$

$$y_{rel} = y_{center} - y_{obj}$$

Trong đó: (x_{obj}, y_{obj}) là tọa độ tâm của đối tượng drone

(x_{rel}, y_{rel}) là tọa độ tương đối sau khi chuyển đổi

Tọa độ tâm của đối tượng được tính từ bounding box:

$$x_{obj} = \frac{x_1 + x_2}{2}$$

$$y_{obj} = \frac{y_1 + y_2}{2}$$

Trong đó (x_1, y_1) và (x_2, y_2) là các góc của bounding box.

Lọc nhiễu: Tuy nhiên, do nhiễu và sự sai lệch trong quá trình vẽ bounding box của hệ thống phát hiện (do xử lý từng frame nên sẽ có sự sai lệch về kích thước của bbox, dẫn đến tọa độ tâm box sẽ có lúc không trùng với tâm drone, ngoài ra khi drone xoay cũng dẫn đến hiện tượng này), ta áp dụng bộ lọc moving average, để xử lý những trường hợp tọa độ bị tăng (giảm) quá đột ngột:

$$y[n] = \frac{1}{M} \sum_{i=0}^{M-1} x[n-i]$$

Xử lý đường đi và vẽ đồ thị: Với mỗi frame, tính khoảng cách Euclidean giữa vị trí hiện tại và vị trí trước đó:

$$d_{pixel} = \sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2 + (z_t - z_{t-1})^2}$$

Tổng quãng đường tích lũy theo pixel:

$$D_{pixel} = \sum_{t=1}^n d_{pixel}(t)$$

Chuyển đổi từ pixel sang mét:

Kích thước thực của đối tượng drone (theo mét): L_{real}

Kích thước đối tượng trong ảnh (theo pixel): L_{pixel}

Tỷ lệ chuyển đổi:

$$scale_{ratio} = \frac{L_{real}}{L_{pixel}} \text{ (mét/pixel)}$$

Quãng đường thực tế:

$$D_{meters} = D_{pixel} \times scale_{ratio}$$

Biểu diễn đường đi: Vì hệ thống trả về một mảng các tọa độ, nên khi nối các tọa độ này với nhau theo cách thông thường ta sẽ được những đoạn thẳng tuyến tính nối tiếp nhau. Đường đi này không phù hợp với thực tế, vì vậy project đã áp dụng nội suy Spline bậc 3, để mô phỏng đường đi một cách mượt mà:

Spline Interpolation:

$$S(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i$$

Tính toán hướng đi chuyển theo góc:

$$\theta = \arctan 2(y_2 - y_1, x_2 - x_1)$$

Gia tốc:

$$a = \frac{v_2 - v_1}{\Delta t}$$

Tính toán vận tốc:

$$speed = \frac{distance}{time}$$

Ước lượng tổng quãng đường thực tế:

$$distance = \frac{real_width \times focal_length}{bbox_width}$$

$$focal_length = \frac{bbox_width \times known_distance}{real_width}$$

Chương 2

Kết quả và thảo luận

2.1 Tài nguyên và môi trường

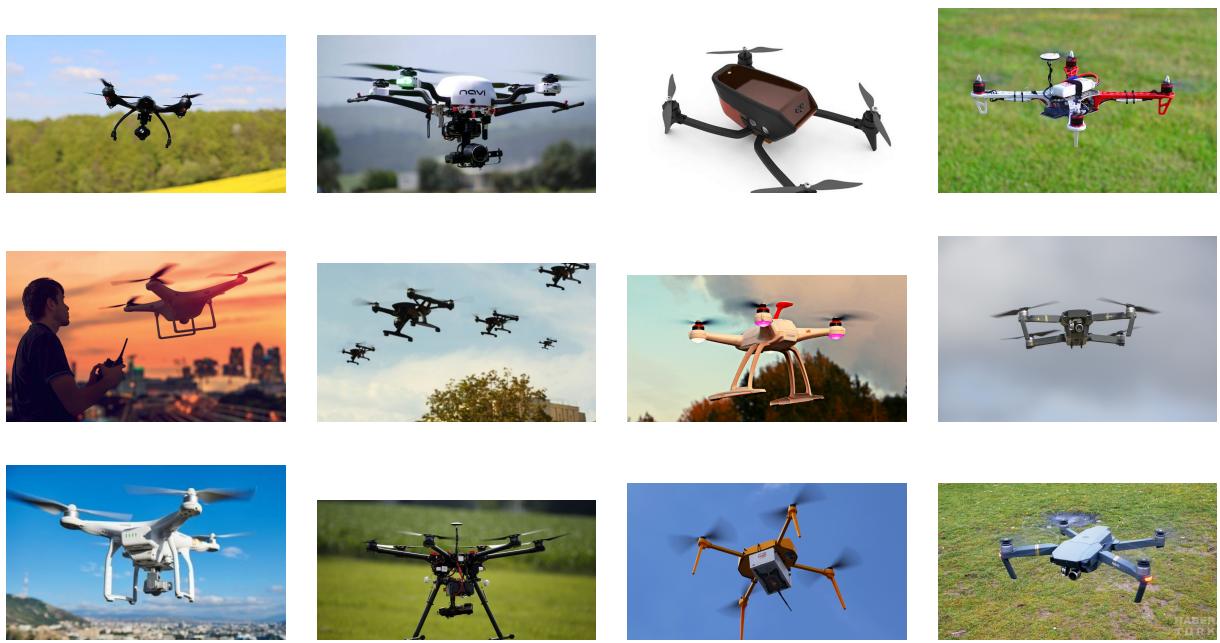
2.1.1 Dataset và huấn luyện

Bài tập lớn sử dụng dataset ảnh các drone với đa dạng mẫu mã, môi trường, cảnh quan. Việc lựa chọn dataset rất quan trọng trong việc huấn luyện mô hình nhận diện, vì vậy em đã sử dụng dataset gồm hơn 1000 ảnh khác nhau, bao gồm dataset từ kaggle [5] và data thu thập thêm.

Quá trình huấn luyện được thực hiện trên nền tảng Google Colab Pro với GPU Tesla T4 16GB, tận dụng sức mạnh tính toán đám mây để tối ưu thời gian huấn luyện. Dữ liệu được gán nhãn thủ công với hai lớp: "drone" và "background" sử dụng công cụ LabelImg, sau đó được chuyển đổi sang định dạng YOLO để phù hợp với kiến trúc YOLOv9.

YOLOv9 được cấu hình với kích thước đầu vào 640×640 pixels[3], phù hợp với việc phát hiện đối tượng drone có kích thước nhỏ trong khung hình.

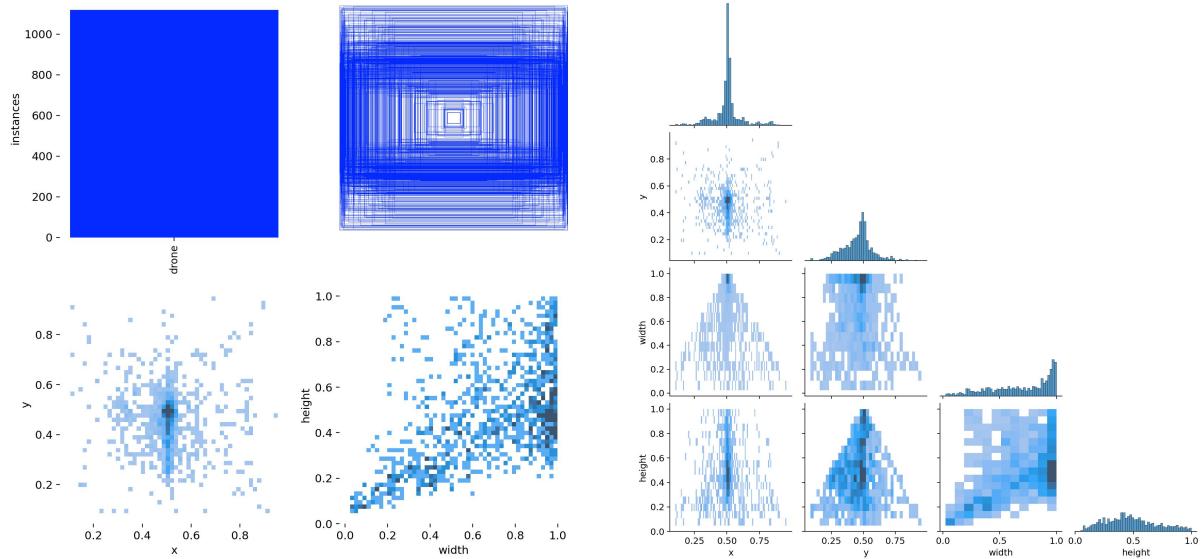
Bộ dữ liệu được chia theo tỷ lệ 80:20 cho tập huấn luyện và tập kiểm thử, với hơn 800 mẫu dùng cho huấn luyện (train) và 200 mẫu dùng cho đánh giá (valid). Quá trình huấn luyện được thực hiện trong 30 epochs với batch size 16, sử dụng bộ tối ưu Adam và learning rate scheduler cosine.



Hình 2.1.1: Dataset

2.1.1 Phân bố dữ liệu:

Tập dữ liệu có phân bố tọa độ tương đối tập trung ở khu vực trung tâm ($x = 0.4-0.6$, $y = 0.2-0.5$). Biểu đồ bounding box cho thấy mỗi tọa độ quan sát có một khung hình tương đối ổn định của đối tượng trong dataset.



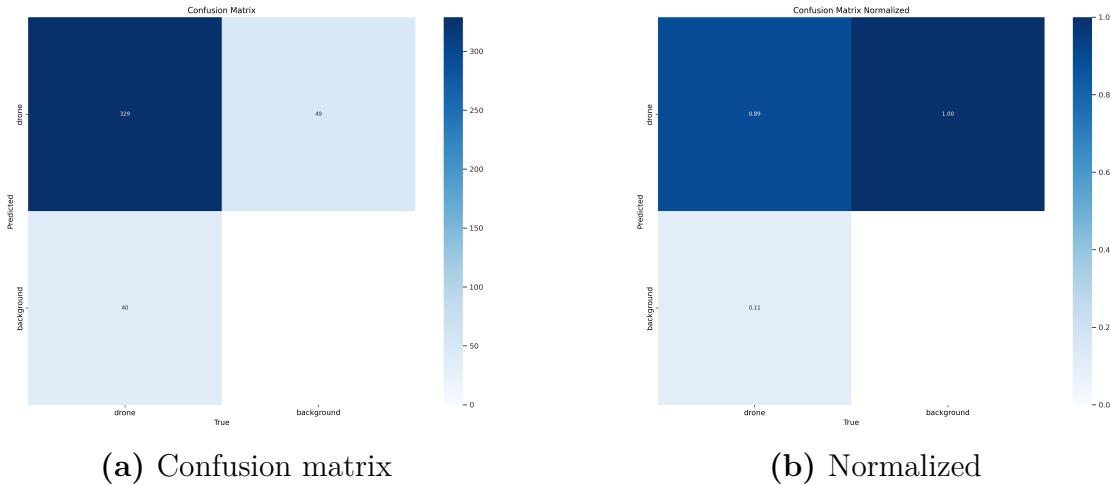
(a) Phân bố dữ liệu huấn luyện

(b) Tương quan dữ liệu

Hình 2.1.2: Phân bố dữ liệu

2.1.1 Confusion matrix:

Mà trận cho thấy hiệu suất phân loại tốt với 329 mẫu drone được phân loại chính xác (True Positive) và chỉ 49 trường hợp nhầm lẫn với background (False Positive). Tỷ lệ False Negative khá thấp (40 trường hợp), cho thấy mô hình có độ nhạy (recall) cao trong việc phát hiện drone.



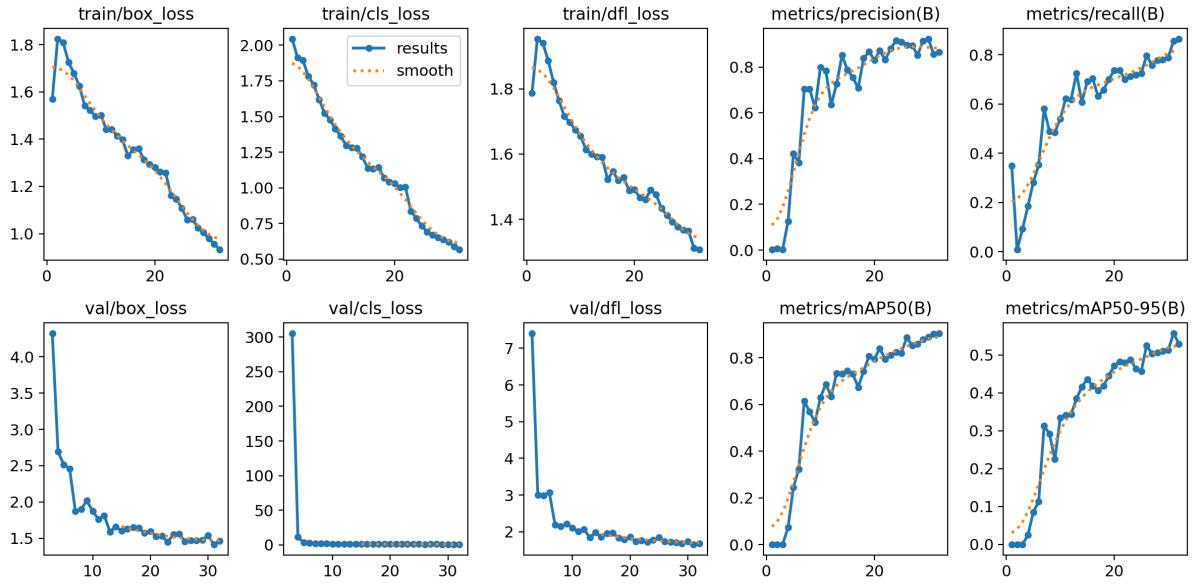
Hình 2.1.3: Confusion matrix

2.1.1 Kết quả

Loss curves (train/val) cho cả box, classification và DFL loss đều cho thấy xu hướng giảm ổn định, không có dấu hiệu overfitting rõ rệt.

Metrics về precision và recall tăng dần và ổn định ở mức cao (trên 0.8), cho thấy mô hình học hiệu quả.

mAP50 và mAP50-95 cũng có xu hướng tăng và ổn định, đạt khoảng 0.8 và 0.5 tương ứng, thể hiện khả năng phát hiện tốt ở nhiều ngưỡng IoU khác nhau.



Hình 2.1.4: Đồ thị quá trình huấn luyện

2.1.2 Kết quả huấn luyện

Kết quả huấn luyện mô hình được thể hiện qua ba nhóm hình ảnh minh họa. Hình 2.1.5 cho thấy quá trình gán class (class assignment) cho các mẫu drone trong tập dữ liệu, với sự đa dạng về góc nhìn, điều kiện ánh sáng và khoảng cách. Các hình ảnh bao gồm cả drone được chụp từ mặt đất, trên không, trong điều kiện ánh sáng tự nhiên và nhân tạo, đảm bảo tính đại diện của dữ liệu huấn luyện.

Hình 2.1.6 minh họa quá trình gán nhãn tên (labeling) cho các đối tượng drone. Mỗi drone được đánh dấu bằng bounding box với tọa độ chuẩn hóa, đảm bảo tính nhất quán trong việc train model.

Hình 2.1.7 thể hiện kết quả huấn luyện với độ tin cậy (confidence score) của mô hình. Các bounding box được hiển thị kèm theo độ tin cậy, cho thấy khả năng phát hiện chính xác của mô hình trong các điều kiện khác nhau. Ta thấy mô hình thể hiện độ tin cậy cao trong việc phát hiện drone ngay cả trong các trường hợp khó như điều kiện ánh sáng yếu, góc chụp phức tạp hoặc có nhiều đối tượng trong cùng một khung hình.



Hình 2.1.5: Gắn class



Hình 2.1.6: Gắn nhãn tên

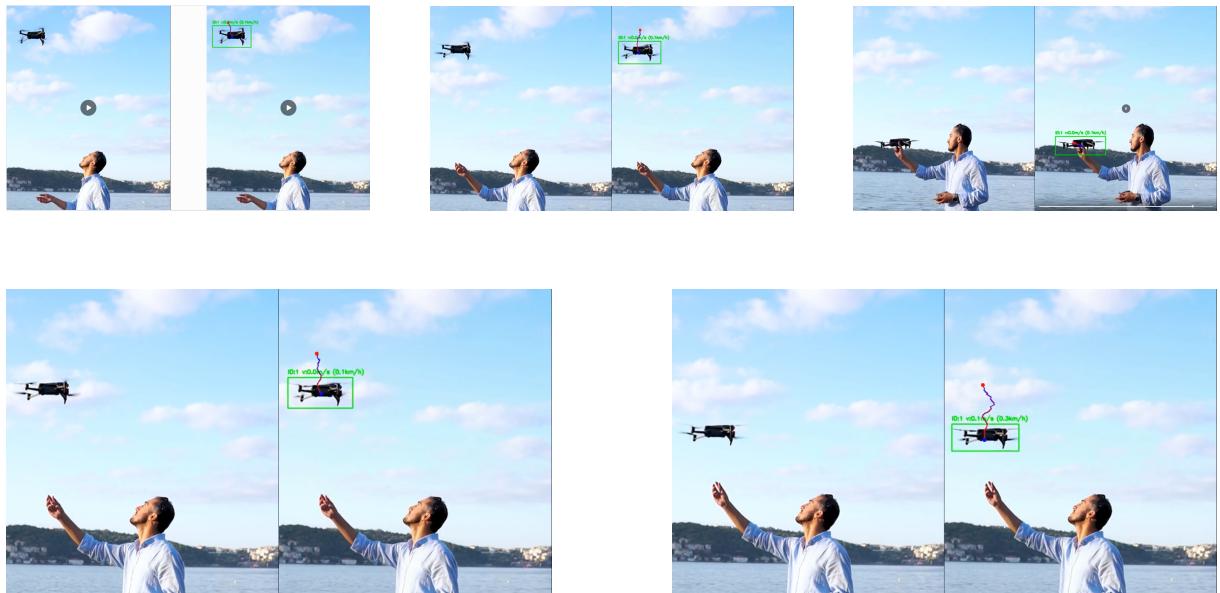


Hình 2.1.7: Kết quả train với độ tin cậy

2.2 Kết quả thực hiện phát hiện và theo dõi

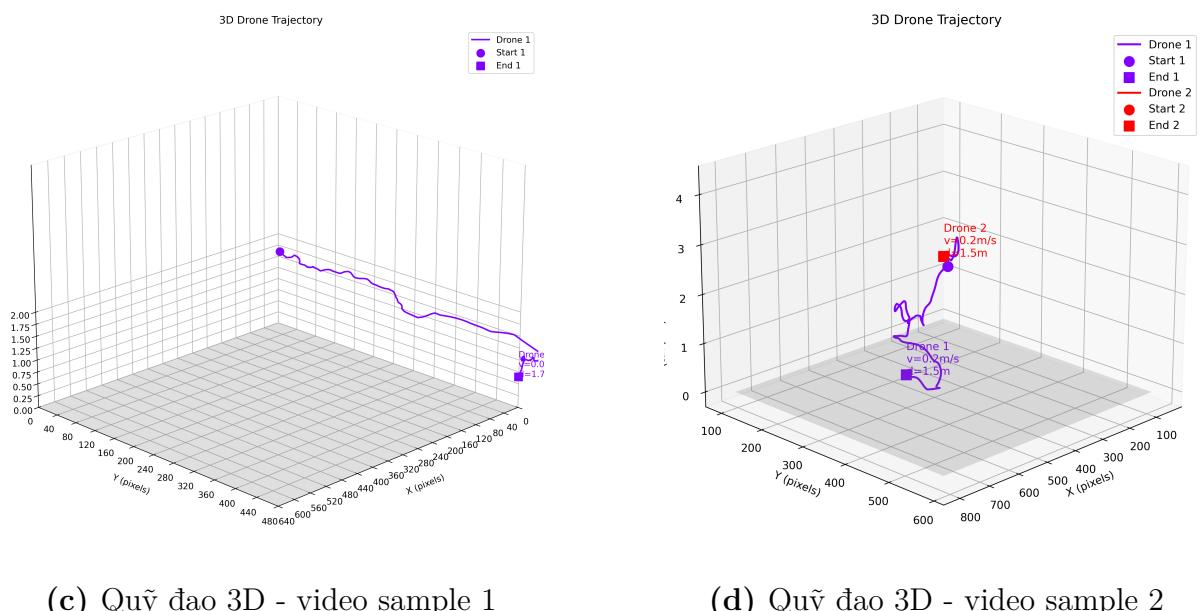
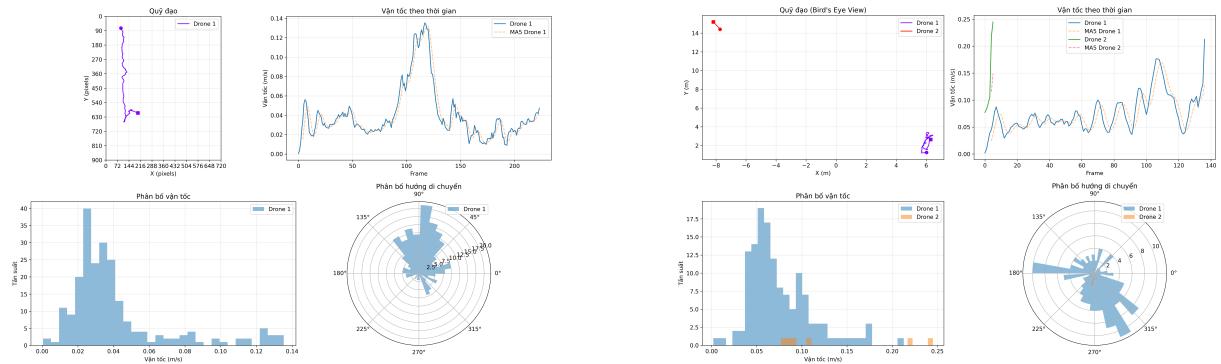
2.2.1 Xử lý trực tiếp trên khung hình

Hình 2.2.1 thể hiện kết quả của việc phát hiện và theo dõi quỹ đạo bay của drone trong một chuỗi khung hình video. Trong đó, drone được đánh dấu bằng khung hình chữ nhật màu xanh lá (bounding box) kèm theo nhãn định danh (ID). Quá trình tracking được thực hiện liên tục qua các frame, cho phép hệ thống duy trì nhận dạng đối tượng drone một cách nhất quán ngay cả khi nó di chuyển. Đặc biệt, hệ thống có khả năng theo dõi drone ở các góc độ và khoảng cách khác nhau, từ khi drone ở xa cho đến khi nó di chuyển gần hơn về phía camera. Điều này cho thấy độ ổn định và hiệu quả của thuật toán tracking được sử dụng trong hệ thống.



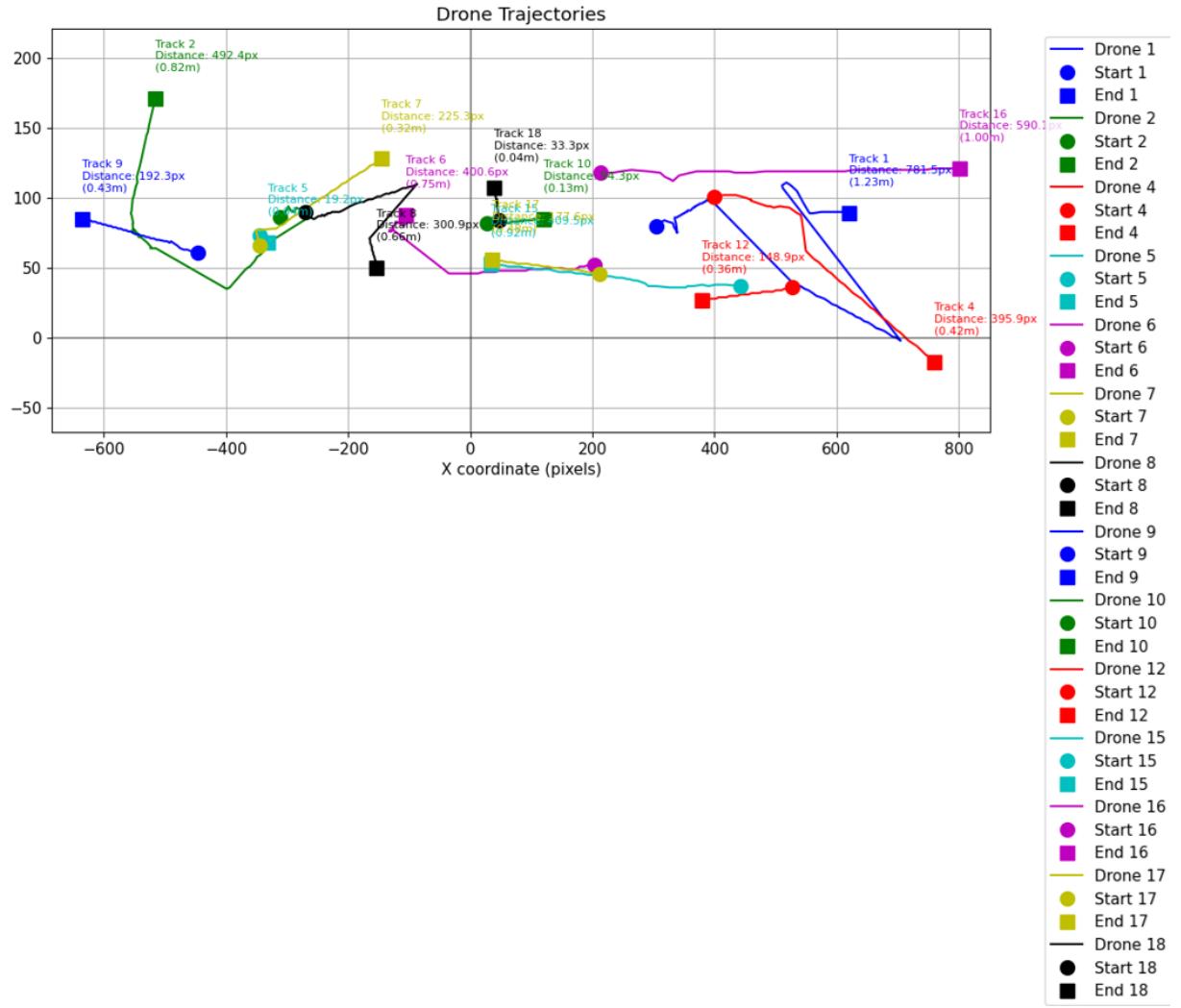
Hình 2.2.1: Kết quả tracking và vẽ trajectory lên khung hình

2.2.2 Biểu diễn tọa độ và đường đi



Hình 2.2.2: Các hình ảnh phân tích quỹ đạo trong ví dụ có 1 drone chính.

Phía trên là hình ảnh mô tả quá trình bay của drone, sự thay đổi vận tốc (nhanh - chậm) trong toàn bộ thời gian, cùng với đó là phân bố vận tốc và hướng di chuyển, cho thấy xu hướng di chuyển của drone. Đồ thị mô tả quãng đường đã khá mềm mại sau khi áp dụng các bộ lọc để giảm nhiễu và sai số trong quá trình theo dõi.



Hình 2.2.3: Hình ảnh quỹ đạo 2D trong video mẫu có nhiều drone

Kết quả cho thấy hệ thống có khả năng phát hiện và theo dõi drone một cách khá hiệu quả, với khả năng nhận diện và duy trì theo dõi đối tượng qua nhiều khung hình. Ta có thể nhìn trong hình trên quá trình di chuyển của drone trong không gian 3 chiều, cho phép đánh giá tốc độ, hướng di chuyển và khoảng cách đến camera.

Chương 3

Kết luận

Trong nghiên cứu này, nhóm đã phát triển và triển khai một hệ thống phát hiện và theo dõi drone sử dụng mô hình YOLOv9 kết hợp với thuật toán DeepSort. Hệ thống đã được thử nghiệm trên một tập dữ liệu đa dạng, bao gồm các video mẫu với nhiều điều kiện môi trường và góc nhìn khác nhau.

Việc sử dụng kết hợp đa nền tảng đã giúp tối ưu hóa quá trình huấn luyện mô hình, đảm bảo hiệu suất cao và thời gian xử lý nhanh chóng. Các kết quả phân tích cho thấy mô hình có độ nhạy và độ chính xác cao, với các chỉ số mAP50 và mAP50-95 đạt mức đáng kể, phản ánh khả năng phát hiện tốt ở nhiều ngưỡng khác nhau.

Tuy nhiên, vẫn còn một số thách thức cần được giải quyết, như việc cải thiện khả năng theo dõi trong các điều kiện ánh sáng quá yếu, hoặc khi vật thể ở khoảng cách rất xa cùng camera độ phân giải thấp hơn trên các thiết bị biên (edge device).

Tóm lại, hệ thống này đã chứng minh tiềm năng của việc sử dụng mô hình học sâu trong việc phát hiện và theo dõi drone, mở ra nhiều khả năng để ứng dụng trong các lĩnh vực như giám sát an ninh, quản lý không lưu và nghiên cứu hành vi drone.

Tài liệu tham khảo

- [1] P. Hidayatullah, R. Tubagus. YOLOv9 Architecture Explained. In 2024 *Stunning Vision AI Articles*. [Online] Available: <https://article.stunningvisionai.com/yolov9-architecture/>.
- [2] N. Wojke, A. Bewley, D. Paulus. Simple Online and Realtime Tracking with a Deep Association Metric. In 2017 *arXiv:1703.07402*. [Online] Available: <https://arxiv.org/abs/1703.07402>.
- [3] Ultralytics. YOLOv9: A breakthrough in real-time object detection technology. In 2024 *Ultralytics Documentation*. [Online] Available: <https://docs.ultralytics.com/vi/models/yolov9/>.
- [4] S. Tu, J. Du, Y. Liang, Y. Cao, W. Chen, D. Xiao, Q. Huang. Tracking and Behavior Analysis of Group-Housed Pigs Based on a Multi-Object Tracking Approach. In 2024 *Animals*, vol. 14, no. 19, pp. 2828. [Online] Available: <https://doi.org/10.3390/ani14192828>.
- [5] M. Dix. Drone Dataset UAV: A comprehensive dataset for drone detection and tracking. In 2023 *Kaggle Datasets*. [Online] Available: <https://www.kaggle.com/datasets/dasmehdixtr/drone-dataset-uav>.