TRƯỜNG ĐẠI HỌC BÁCH KHOA - ĐẠI HỌC ĐÀ NẴNG

KHOA ĐIỆN TỬ - VIỄN THÔNG

o0o

CHUYÊN ĐỀ 2
BÁO CÁO CUỐI KỲ

Đề tài:

Hệ Thống Phát Hiện Làn Đường Và Phương Tiện Từ Video Camera Hành Trình Sử Dụng Xử Lý Ảnh Và Học Sâu

Giảng viên hướng dẫn	: TS. Nguyễn Văn Hiếu	
Sinh viên thực hiện	: Nguyễn Ngọc Trung	21KTMT2
	Lê Dương Khang	21KTMT2
	Hoàng Bảo Long	21KTMT1
Lớp học phần	: 21.44	
Nhóm	: 2	

Đà Nẵng, tháng 12 năm 2025

LỜI NÓI ĐẦU

Với sự đồng ý và hướng dẫn tận tình của *Thầy Nguyễn Văn Hiếu*, nhóm đã có cơ hội thực hiện đề tài *"Hệ thống phát hiện làn đường và phương tiện từ video camera hành trình sử dụng xử lý ảnh và học sâu"*. Đây là một chủ đề có tính ứng dụng cao và cũng mang nhiều thách thức, đòi hỏi nhóm phải kết hợp kiến thức lý thuyết và khả năng nghiên cứu độc lập.

Nhóm xin bày tỏ lòng biết ơn sâu sắc đến thầy, người đã tận tình giảng dạy và hỗ trợ nhóm trong suốt quá trình thực hiện học phần. Thầy đã không chỉ cung cấp những định hướng chuyên môn quan trọng, mà còn luôn tận tâm giải đáp thắc mắc, đưa ra những góp ý chi tiết giúp nhóm có những ý tưởng để hoàn thiện nội dung của báo cáo.

Chúng em trân trọng cảm ơn thầy vì sự tận tụy, trách nhiệm và sự nghiêm túc trong giảng dạy. Những kiến thức, kinh nghiệm và tinh thần làm việc mà thầy truyền đạt sẽ là hành trang quý báu cho chúng em trong những học phần tiếp theo cũng như trong con đường nghiên cứu và phát triển sau này.

Mặc dù thời gian thực hiện dự án có hạn, đề tài đã giúp chúng em hiểu rõ hơn cách tiếp cận một bài toán thực tế trong lĩnh vực thị giác máy tính và hệ thống hỗ trợ lái xe. Những gì nhóm đạt được không chỉ là kết quả của một học phần, mà còn là một trải nghiệm giá trị, giúp chúng em trưởng thành hơn trong tư duy và kỹ năng làm việc.

Một lần nữa, nhóm xin gửi lời cảm ơn chân thành nhất đến thầy!

Đà Nẵng, ngày 15 tháng 12 năm 2025

Các thành viên trong nhóm

Nguyễn Ngọc Trung Lê Dương Khang Hoàng Bảo Long

PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

STT	Họ và tên	Nội dung chính phụ trách	Tỉ lệ đóng góp
1	Nguyễn Ngọc Trung	UFLDv2, tích hợp hệ thống	34%
2	Lê Dương Khang	YOLOv8, cảnh báo phương tiện	33%
3	Hoàng Bảo Long	Xử lý ảnh truyền thống, đánh giá kết quả	33%

Nhóm đã thống nhất tỉ lệ phân công công việc như trên và cam kết không thay đổi sau khi nộp báo cáo.

MỤC LỤC

LỜI NÓI ĐẦU 1
PHÂN CÔNG CÔNG VIỆC TRONG NHÓM 2
MỤC LỤC 3
1.1. Đặt vấn đề 4
1.2. Mục tiêu và phạm vi nghiên cứu 4
1.3. Phương pháp tiếp cận 5
1.4. Điểm mới và đóng góp của đề tài 5
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 6
2.1. Cơ sở xử lý ảnh truyền thống 6
2.2. Cơ sở lý thuyết UFLDv2 10
2.3. Cơ sở lý thuyết YOLO 13
CHƯƠNG 3. THIẾT KẾ HỆ THỐNG 15
3.1. Tổng quan hệ thống 15
3.2. Mô hình hóa bài toán ở mức hệ thống 16
3.3. Nhận dạng làn đường bằng xử lý ảnh truyền thống 17
3.4. Nhận dạng làn đường bằng UFLDv2 27
3.5. Phát hiện và cảnh báo phương tiện phía trước 31
3.6. Đánh giá độ phức tạp và tốc độ hội tụ 36
CHƯƠNG 4. KẾT QUẢ VÀ ĐÁNH GIÁ 37
4.1. Kết quả nhận dạng làn đường bằng xử lý ảnh truyền thống 37
4.2. Kết quả nhận dạng làn đường bằng phương pháp học sâu UFLDv2 40
4.3. Kết quả phát hiện và cảnh báo phương tiện phía trước 46
4.4. Đánh giá tốc độ xử lý của các mô-đun trong hệ thống 49
4.5. Mô tả các file kết quả và dữ liệu lưu trữ 50
KẾT LUẬN 51
TÀI LIỆU THAM KHẢO 52
PHỤ LỤC 54

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1. Đặt vấn đề

Tai nạn giao thông là một trong những vấn đề nghiêm trọng trên toàn cầu, trong đó phần lớn nguyên nhân xuất phát từ yếu tố con người như mất tập trung, mệt mỏi, chạy quá tốc độ hoặc phản ứng chậm trong các tình huống khẩn cấp. Theo thống kê của NHTSA, khoảng 94% các vụ tai nạn giao thông có liên quan trực tiếp đến lỗi của người lái **[1]**. Riêng tại Hoa Kỳ, mỗi năm ghi nhận khoảng 6400 ca tử vong do lái xe mệt mỏi, chiếm gần 17,6% tổng số vụ tai nạn gây chết người **[2]**. Những số liệu này cho thấy việc chỉ dựa vào kỹ năng và sự tập trung của người lái là chưa đủ trong bối cảnh giao thông ngày càng phức tạp.

Trước thực trạng đó, các hệ thống Hỗ trợ lái xe tiên tiến (Advanced Driver Assistance Systems - ADAS) đã được nghiên cứu và phát triển nhằm giảm thiểu rủi ro tai nạn thông qua việc hỗ trợ người lái trong quá trình điều khiển phương tiện. Nhiều nghiên cứu cho thấy các chức năng như phanh tự động khẩn cấp (AEB) và cảnh báo, giữ làn đường (LDW/LKA) có khả năng giảm đáng kể số vụ va chạm và tai nạn gây thương tích **[3][4]**. Điều này khẳng định vai trò quan trọng của các hệ thống hỗ trợ lái trong việc nâng cao an toàn giao thông và hướng tới giao thông thông minh.

Xuất phát từ nhu cầu thực tiễn đó, đề tài tập trung nghiên cứu và xây dựng một hệ thống hỗ trợ lái xe dựa trên camera hành trình, với hai chức năng cốt lõi là nhận dạng làn đường và cảnh báo phương tiện phía trước, nhằm hỗ trợ người lái duy trì làn đường an toàn và giảm nguy cơ va chạm trong điều kiện giao thông thực tế.

1.2. Mục tiêu và phạm vi nghiên cứu

Mục tiêu chính của đề tài là xây dựng và đánh giá một hệ thống hỗ trợ lái xe dựa trên camera hành trình, có khả năng nhận dạng làn đường và cảnh báo phương tiện phía trước theo thời gian thực. Hệ thống nhằm hỗ trợ người lái quan sát và đánh giá tình huống giao thông, góp phần giảm rủi ro do mất tập trung hoặc phản ứng chậm.

Cụ thể, đề tài hướng đến các mục tiêu sau:

- Nghiên cứu và triển khai phương pháp nhận dạng làn đường bằng xử lý ảnh truyền thống làm baseline.
- Ứng dụng và fine-tune mô hình học sâu UFLDv2 để nâng cao độ chính xác và độ ổn định của việc phát hiện làn đường trong điều kiện giao thông thực tế tại Việt Nam.
- Sử dụng mô hình YOLOv8 để phát hiện phương tiện phía trước và kết hợp với thông tin làn đường nhằm xây dựng chức năng cảnh báo an toàn.

- Đánh giá và so sánh hiệu quả của các phương pháp dựa trên các tiêu chí như độ chính xác, tốc độ xử lý và khả năng đáp ứng thời gian thực.

Trong phạm vi nghiên cứu, hệ thống chỉ tập trung vào nhận dạng làn đường và cảnh báo phương tiện phía trước, không xét đến các chức năng nâng cao như nhận dạng biển báo giao thông hay điều khiển phương tiện tự động. Dữ liệu sử dụng chủ yếu là video và hình ảnh thu thập từ camera hành trình.

1.3. Phương pháp tiếp cận

Đề tài tiếp cận bài toán theo hướng kết hợp giữa xử lý ảnh truyền thống và học sâu nhằm khai thác ưu điểm của từng phương pháp. Các kỹ thuật tiền xử lý, phát hiện biên Canny và biến đổi Hough được sử dụng để xây dựng mô hình nhận dạng làn đường cơ bản làm baseline so sánh.

Mô hình học sâu UFLDv2 được áp dụng và fine-tune trên tập dữ liệu làn đường thực tế tại Việt Nam do nhóm tự thu thập, giúp cải thiện độ chính xác trong các điều kiện như vạch kẻ đường mờ, ánh sáng thay đổi hoặc bị che khuất.

Song song đó, YOLOv8 được sử dụng để phát hiện phương tiện phía trước, kết hợp với thông tin hình học từ kết quả nhận dạng làn đường để ước lượng khoảng cách và đưa ra cảnh báo an toàn. Toàn bộ hệ thống được xây dựng theo pipeline xử lý thời gian thực, cho phép đánh giá hiệu quả hoạt động trong nhiều điều kiện khác nhau.

1.4. Điểm mới và đóng góp của đề tài

Điểm mới của đề tài nằm ở việc xây dựng một pipeline hoàn chỉnh cho hệ thống hỗ trợ lái xe dựa trên camera hành trình, kết hợp giữa xử lý ảnh truyền thống và các mô hình học sâu hiện đại. Thay vì chỉ tập trung vào một phương pháp riêng lẻ, đề tài tiến hành so sánh và đánh giá toàn diện hiệu quả của từng hướng tiếp cận trong cùng một điều kiện dữ liệu và môi trường thực nghiệm.

Bên cạnh đó, mô hình UFLDv2 được fine-tune trên tập dữ liệu làn đường thực tế tại Việt Nam (Đà Nẵng) do nhóm tự thu thập và gán nhãn, giúp mô hình thích nghi tốt hơn với các đặc thù như vạch kẻ đường mờ, xuống cấp, điều kiện ánh sáng phức tạp – những yếu tố chưa được phản ánh đầy đủ trong các bộ dữ liệu chuẩn quốc tế.

Ngoài ra, đề tài xây dựng hệ thống cảnh báo phương tiện phía trước chỉ sử dụng camera, không phụ thuộc vào các cảm biến đắt tiền như radar hoặc LiDAR, hướng đến khả năng triển khai thực tế với chi phí thấp. Các kết quả thực nghiệm và đánh giá hiệu suất cho thấy hệ thống đáp ứng yêu cầu xử lý thời gian thực và có tiềm năng mở rộng cho các nghiên cứu tiếp theo trong lĩnh vực giao thông thông minh.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Chương 2 trình bày các kiến thức nền tảng liên quan đến các phương pháp được áp dụng trong hệ thống, bao gồm: xử lý ảnh truyền thống, mô hình học sâu UFLDv2 cho phát hiện làn đường, và mô hình YOLOv8 dùng để nhận dạng phương tiện phục vụ chức năng cảnh báo khoảng cách phía trước. Những cơ sở lý thuyết này cung cấp nền tảng vững chắc cho việc thiết kế hệ thống được trình bày trong Chương 3, đồng thời làm cơ sở để phân tích, đánh giá kết quả trong Chương 4.

2.1. Cơ sở xử lý ảnh truyền thống

Phương pháp xử lý ảnh truyền thống là nhóm thuật toán trực quan, dễ triển khai, được sử dụng như baseline để đánh giá hiệu quả so với phương pháp học sâu. Các bước chính bao gồm tiền xử lý ảnh, phát hiện biên bằng phương pháp Canny, xây dựng vùng quan tâm ROI, và trích xuất làn đường bằng Hough Transform.

2.1.1. Tiền xử lý ảnh

Bước đầu tiên của mô hình là chuyển ảnh màu sang ảnh mức xám. Ảnh RGB gồm 3 kênh màu R, G, B, trong khi phần lớn các thuật toán phát hiện biên chỉ quan tâm đến sự thay đổi cường độ sáng. Việc chuẩn hóa ảnh về một kênh grayscale giúp giảm nhiễu màu, giảm khối lượng dữ liệu cần xử lý và tăng tốc độ tính toán. Đồng thời, ảnh xám làm cho các thao tác trích xuất biên trở nên ổn định hơn, tốt hơn do hệ thống chỉ làm việc với 1 giá trị là cường độ tại mỗi điểm ảnh thay vì 3 giá trị như ban đầu.

2.1.2. Phát hiện biên bằng phương pháp Canny

Sau khi chuyển sang ảnh xám, mô hình áp dụng thuật toán Canny để phát hiện các biên quan trọng trong khung hình. Canny là một trong những phương pháp phát hiện biên được sử dụng rộng rãi nhờ độ chính xác cao và khả năng loại nhiễu tốt **[5]**.

Đầu tiên thuật toán này sẽ làm trơn ảnh bằng bộ lọc *Gaussian (1)*. Trong đó, bộ lọc Gaussian được tính theo công thức sau:

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Trong đó:

- (x, y): là tọa độ tương đối so với điểm ảnh trung tâm.
- σ $(sigma)$: là độ lệch chuẩn, xác định độ rộng của kernel.

Thuật toán tiếp tục tính toán gradient bằng cách lấy đạo hàm của hàm phân phối Gaussian theo hai hướng: chiều ngang (G_x) và chiều dọc (G_y).

Từ đó, cường độ gradient được tính theo công thức (2):

$$Edge_{Gradient(G)} = \sqrt{{G_x}^2 + {G_y}^2} \quad (2)$$

Đồng thời, hướng của gradient được xác định bằng công thức (3) sau:

$$Angle(\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad (3)$$

Trong đó:

- G_x: Gradient theo hướng ngang
- G_y: Gradient theo hướng dọc

Tiếp tục thuật toán thực hiện *non-maximum suppression* nhằm loại bỏ những điểm không phải cực đại và giữ lại những điểm biên sắc nét, loại bỏ, và cuối cùng là sử dụng ngưỡng kép (Double threshold) kết hợp truy vết biên (Edge tracking Hysteresis) để xác định các biên thực sự.

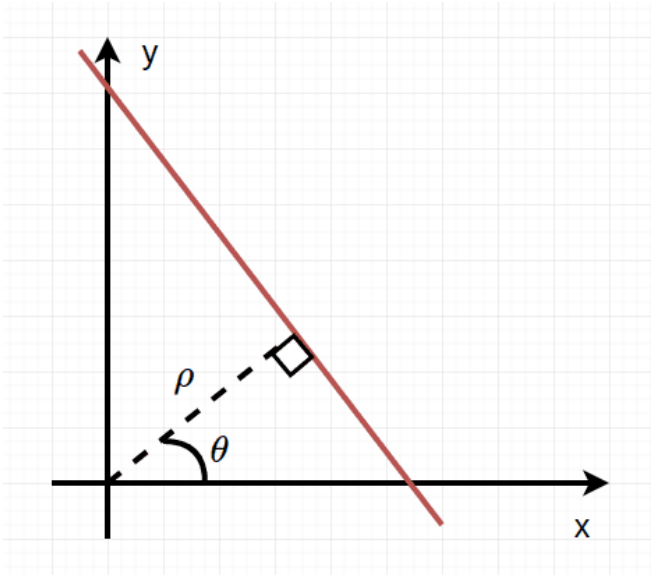
2.1.3. Xây dựng vùng quan tâm

Trong bài toán phát hiện làn đường, khung hình camera thường chứa nhiều chi tiết không liên quan (cây cối, nhà cửa, bầu trời, xe ở làn khác) gây nhiễu cho thuật toán Hough Transform. Để giảm nhiễu, mô hình chỉ phân tích vùng quan tâm (ROI) dạng hình thang: đáy rộng ở phía dưới nơi đường rõ ràng, đỉnh hẹp ở phía trên tương ứng với khu vực xa.

2.1.4. Trích xuất làn đường bằng phép biến đổi Hough

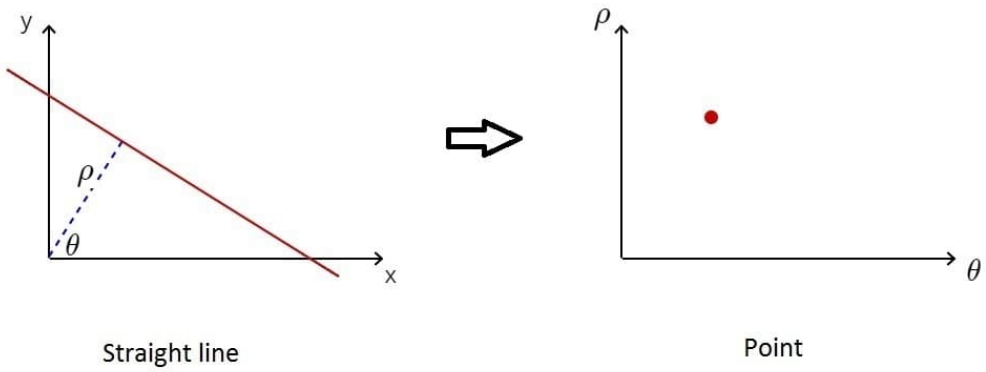
Sau khi xây dựng vùng quan tâm (ROI), ta có các điểm biên trong vùng ROI, từ đó hệ thống áp dụng phép biến đổi Hough để phát hiện các đường thẳng, vốn là đặc trưng của mép làn đường.

Biến đổi Hough là một kỹ thuật trích xuất tính năng được sử dụng để phát hiện các hình dạng đơn giản như đường thẳng và vòng tròn trong ảnh **[6]**.



Hình 2.1. Hệ trục không gian ảnh ***[7]***

Biến đổi Hough hoạt động dựa trên nguyên lý ánh xạ các điểm biên từ không gian ảnh (*Hình 2.1*) sang không gian tham số (ρ, θ) (*Hình 2.2*).



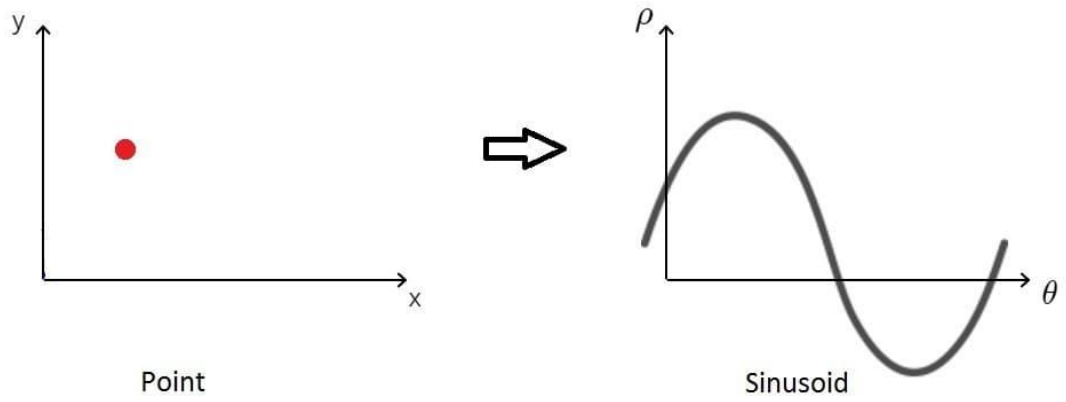
Hình 2.2. Chuyển đổi từ không gian ảnh sang hệ trục không gian tham số (ρ, θ) ***[7]***

Trong không gian này, mỗi điểm biên (x, y)trên ảnh sẽ tương ứng với một đường cong hình Sin (*Hình 2.3*) biểu diễn tất cả các đường thẳng có thể đi qua điểm đó, được xác định bởi công thức (4):

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \tag{4}$$

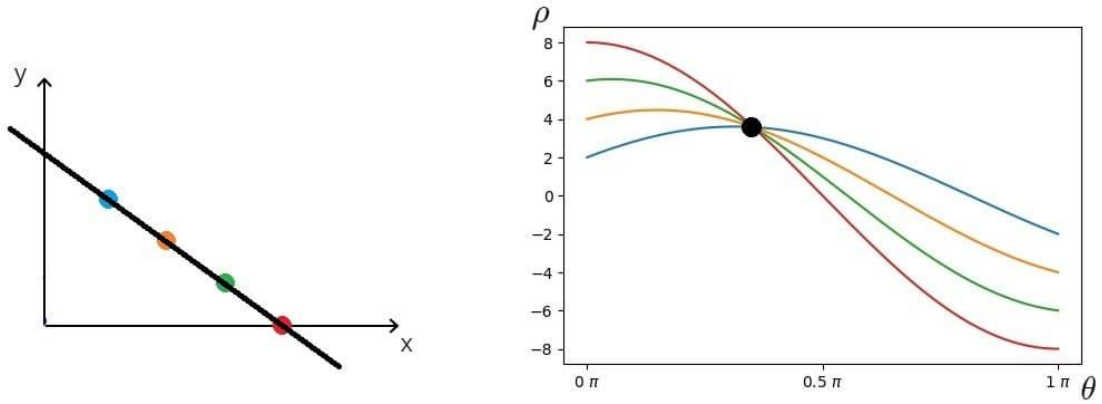
Trong đó:

- ρ: khoảng cách từ gốc tọa độ đến đường thẳng.
- θ: góc giữa đường thẳng và trục hoành.



Hình 2.3. Chuyển đổi từ điểm biên (x, y) sang đường cong hình Sin ***[7]***

Khi nhiều điểm biên cùng thuộc một đường thẳng trong ảnh, các đường cong hình Sin tương ứng của chúng sẽ giao nhau tại một điểm trong không gian tham số (*Hình 2.4*). Điểm giao này thể hiện sự tích tụ lớn và được nhận dạng là một đường thẳng thực sự trên ảnh gốc. Nhờ cơ chế này, Hough Transform có khả năng phát hiện các đường thẳng ngay cả khi chúng bị đứt đoạn hoặc nhiễu, bằng cách tìm cực đại trong không gian tham số.



Hình 2.4. Các đường cong hình Sin tương ứng với mỗi điểm thuộc cùng 1 đường thẳng ***[7]***

Mô hình phân tách thành hai đường riêng biệt:

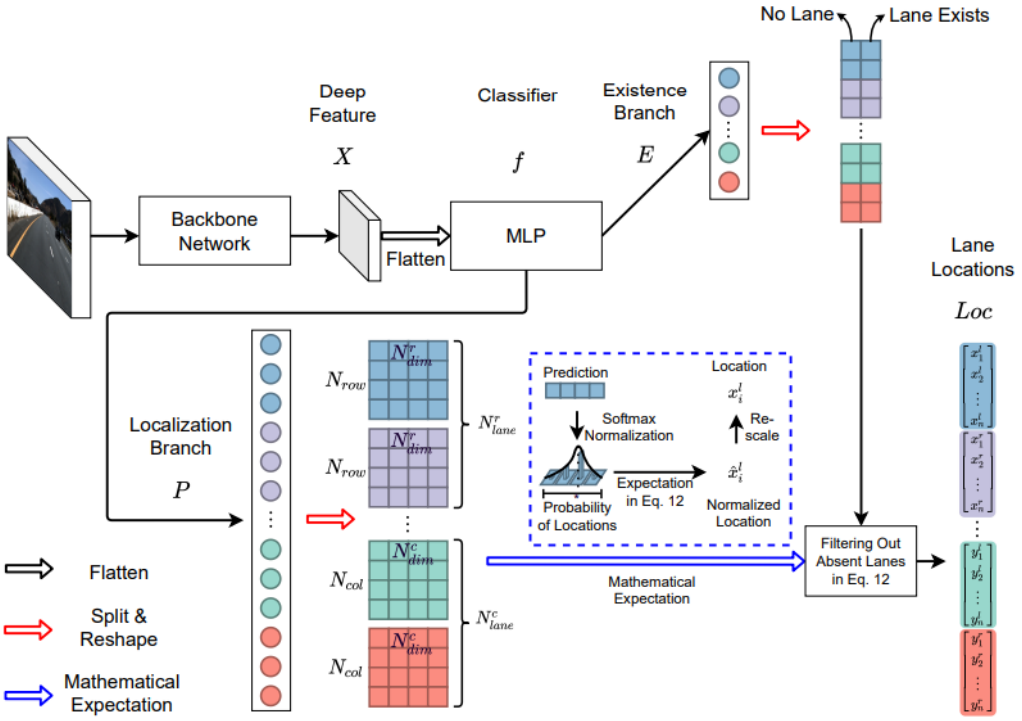
- Đường làn bên trái
- Đường làn bên phải

Sau khi xác định được làn đường, mô hình tiến hành ghép (overlay) các thông tin này lên khung hình gốc. Các đường làn được vẽ bằng màu nổi bật. Đây là bước quan trọng để tạo ra video kết quả phục vụ phân tích hoặc trình bày trong báo cáo.

2.2. Cơ sở lý thuyết UFLDv2

UFLDv2 (Ultra-Fast Lane Detection v2) là mô hình học sâu được đề xuất cho bài toán phát hiện làn đường theo thời gian thực từ camera hành trình. Mục tiêu chính của UFLDv2 là đạt được tốc độ suy luận cao trong khi vẫn đảm bảo độ chính xác và độ ổn định trong các điều kiện giao thông phức tạp như làn đường cong, vạch mờ hoặc bị che khuất.

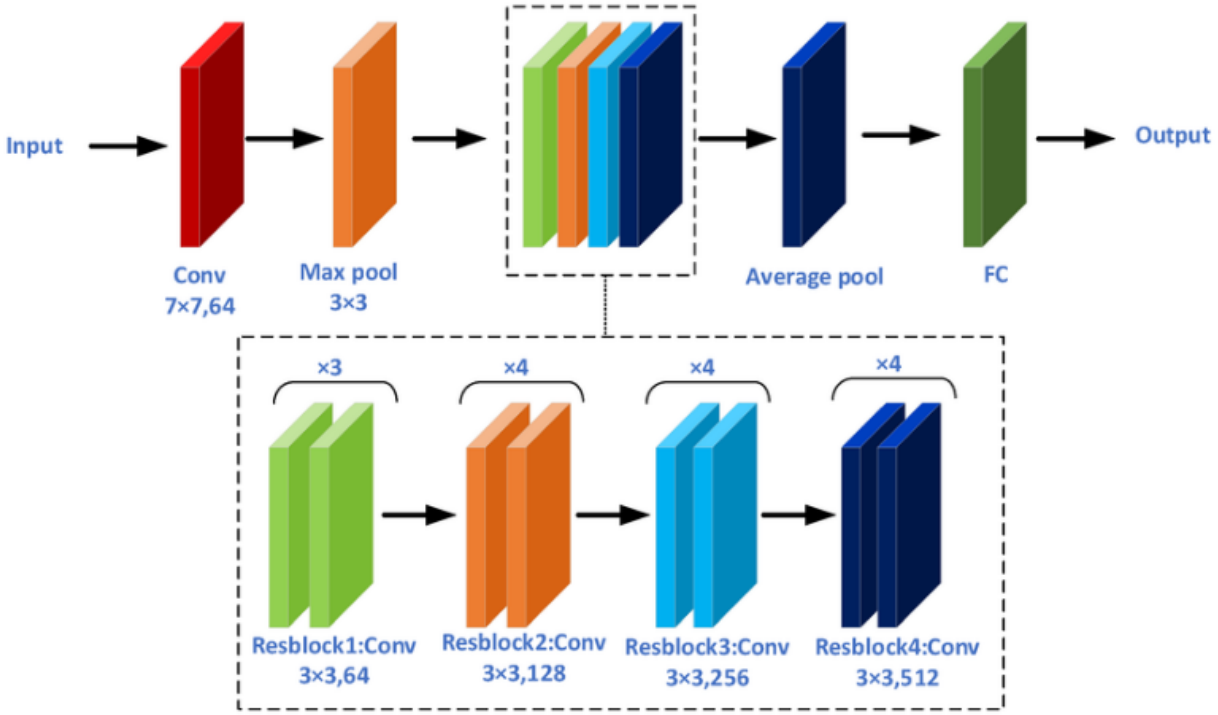
Kiến trúc tổng quan của mô hình UFLDv2 được minh họa trong *Hình 2.5*, bao gồm ba khối chính: Backbone, Feature Compression Module và Lane Prediction Head.



Hình 2.5. Sơ đồ tổng quan kiến trúc UFLDv2 ***[8]***

a) Backbone

Backbone là thành phần đầu tiên của mô hình, có chức năng trích xuất các đặc trưng quan trọng từ ảnh đầu vào. Trong UFLDv2, backbone được lựa chọn theo hướng nhẹ nhưng hiệu quả, trong đó ResNet-34 là cấu trúc được sử dụng phổ biến nhờ khả năng cân bằng tốt giữa tốc độ và độ sâu trích xuất đặc trưng. Nhờ hệ thống residual blocks, ResNet-34 cho phép mô hình học được các đặc trưng hình học đặc thù của làn đường trong nhiều điều kiện môi trường khác nhau mà không làm tăng đáng kể chi phí tính toán.

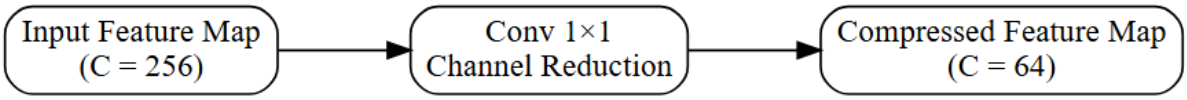


Hình 2.6. Kiến trúc Backbone ResNet-34 được sử dụng trong mô hình UFLDv2 ***[9]***

Kiến trúc tổng quát của backbone ResNet-34 được minh họa trong *Hình 2.6*, thể hiện các lớp convolution ban đầu, tầng pooling và bốn nhóm residual block với số lớp lần lượt là 3–4–4–4, giúp cải thiện khả năng lan truyền gradient và hiệu quả học của mô hình.

b) Feature Compression Module

Feature Compression Module được sử dụng sau Backbone nhằm giảm số lượng kênh của feature maps và giữ lại những đặc trưng quan trọng nhất liên quan đến làn đường. Việc nén đặc trưng giúp mô hình giảm chi phí tính toán, tăng tốc độ suy luận và vẫn duy trì độ chính xác cần thiết cho bài toán phát hiện làn đường thời gian thực. Module này thường được triển khai bằng các lớp convolution kích thước nhỏ (1×1) với số lượng kênh giảm dần so với đầu ra của Backbone.

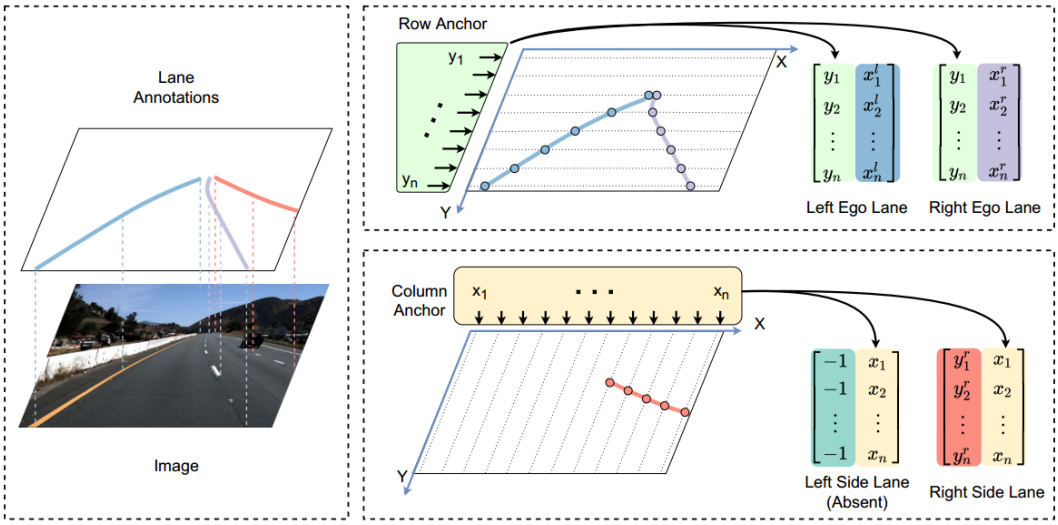


Hình 2.7. Sơ đồ khối Feature Compression Module được sử dụng trong UFLDv2

Cấu trúc cơ bản của Feature Compression Module được minh họa trong *Hình 2.7*, thể hiện quá trình nén từ feature maps có số kênh lớn sang feature maps gọn nhẹ hơn phục vụ giai đoạn dự đoán làn đường.

c) Lane Prediction Head

Lane Prediction Head là thành phần quyết định của UFLDv2, chịu trách nhiệm dự đoán vị trí của các làn đường dựa trên các đặc trưng đã được nén từ các tầng trước đó. Điểm khác biệt quan trọng của UFLDv2 so với các mô hình phát hiện làn đường truyền thống nằm ở cơ chế dự đoán dựa trên anchor theo hàng (row-anchor) và anchor theo cột (column-anchor).



Hình 2.8. Cơ chế dự đoán theo row-anchor và column-anchor trong UFLDv2 ***[8]***

UFLDv2 sử dụng kiến trúc hai nhánh song song (*Hình 2.8*) để phát hiện các loại lane khác nhau:

- Nhánh row-anchor: Chia ảnh thành n hàng theo chiều dọc, dự đoán tọa độ x tại mỗi hàng. Nhánh này chủ yếu phát hiện hai làn đường chính (Left Ego Lane và Right Ego Lane).
- Nhánh column-anchor: Chia ảnh thành n cột theo chiều ngang, dự đoán tọa độ y tại mỗi cột. Nhánh này phát hiện các làn phụ (Left Side Lane, Right Side Lane), gán giá trị -1 khi lane không tồn tại.

Cơ chế kết hợp này giúp mô hình xử lý tốt các tình huống phức tạp như đường cong, làn bị đứt hoặc che khuất, đồng thời giảm số lượng tham số cần ước lượng và cải thiện tốc độ suy luận.

2.3. Cơ sở lý thuyết YOLO

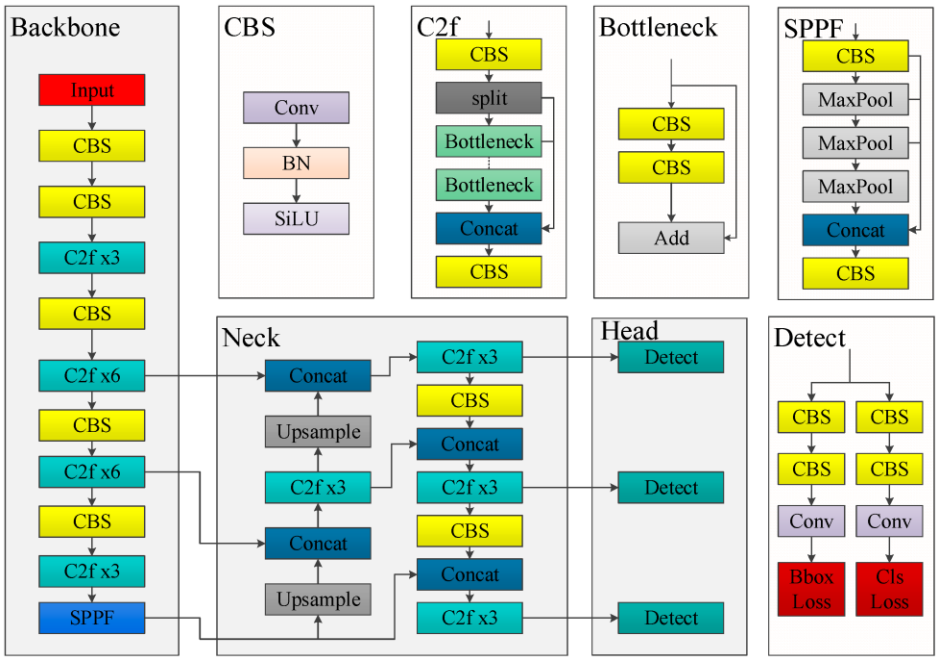
Trong hệ thống của nhóm, YOLO (You Only Look Once) được sử dụng làm mô-đun phát hiện phương tiện phía trước, phục vụ cho bài toán ước lượng khoảng cách và cảnh báo nguy cơ va chạm. Đây là một trong những bài toán quan trọng trong lĩnh vực thị giác máy tính cho giao thông thông minh, hệ thống hỗ trợ lái và robot di động.

Trong đề tài, nhóm sử dụng phiên bản YOLOv8 (You Only Look Once version 8) được huấn luyện trước trên tập dữ liệu COCO **[10][11]**, có khả năng nhận dạng các loại phương tiện phổ biến như car, motorbike, bus và truck, sau đó kết hợp với mô-đun ước lượng khoảng cách để xây dựng chức năng cảnh báo phương tiện phía trước.

2.3.1. Kiến trúc tổng quan của mô hình YOLO

Các phiên bản YOLO hiện đại đều tuân theo kiến trúc ba khối tương tự nhau **[12]**, bao gồm:

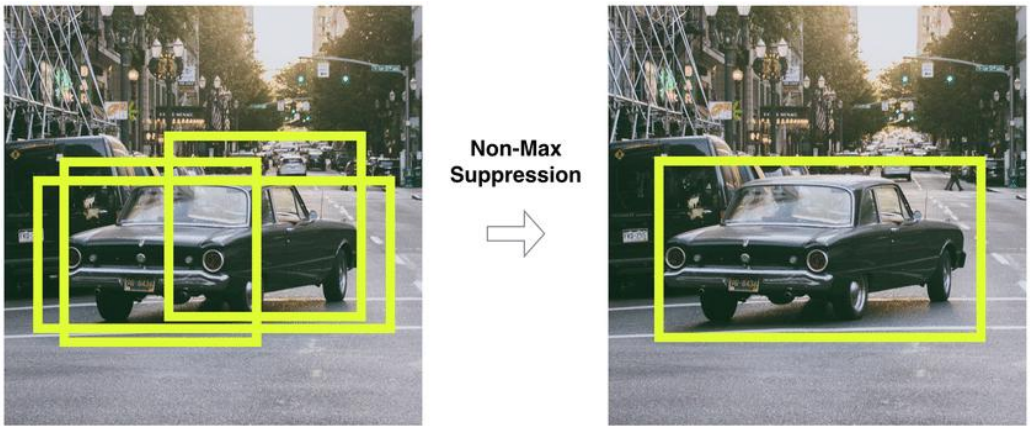
- Backbone: Trích xuất đặc trưng từ ảnh đầu vào
- Neck: Hợp nhất đặc trưng đa tỉ lệ
- Head: Dự đoán bounding box, độ tin cậy và nhãn lớp



Hình 2.9. Sơ đồ tổng quan kiến trúc YOLOv8 ***[12]***

Backbone chịu trách nhiệm học các đặc trưng không gian và ngữ nghĩa của phương tiện, từ cạnh, hình dạng đến cấu trúc tổng thể. Neck sử dụng các cơ chế hợp nhất đặc trưng

đa tỉ lệ (như FPN/PAN) nhằm đảm bảo mô hình có thể phát hiện cả phương tiện ở xa (nhỏ) và ở gần (lớn). Cuối cùng, Head sinh ra các dự đoán bounding box và phân loại đối tượng.



Hình 2.10. Hình ảnh minh hoạ bước hậu xử lý NMS

Sau khi mô hình sinh ra nhiều bounding box chồng lấp cho cùng một đối tượng, YOLO áp dụng bước Non-Maximum Suppression (NMS) để loại bỏ các hộp dư thừa. Nguyên lý của NMS là:

- Giữ lại bounding box có độ tin cậy cao nhất
- Loại bỏ các bounding box còn lại có IoU vượt quá một ngưỡng xác định

Nhờ NMS, kết quả phát hiện trở nên gọn gàng và bám sát vị trí thực của phương tiện trong ảnh.

2.3.2. Nguyên lý cảnh báo phương tiện phía trước

Trong phạm vi đề tài, YOLO không chỉ được sử dụng để phát hiện phương tiện mà còn đóng vai trò nền tảng cho chức năng cảnh báo an toàn. Cụ thể:

- YOLO phát hiện các phương tiện xuất hiện trong khung hình
- Hệ thống lọc phương tiện theo vùng quan tâm (ROI) phía trước
- Bounding box của phương tiện được dùng để ước lượng khoảng cách tương đối
- Khoảng cách này là cơ sở để phân loại trạng thái An toàn – Cảnh báo – Nguy hiểm

Cách tiếp cận này cho phép hệ thống cảnh báo va chạm hoạt động chỉ với camera hành trình, không cần sử dụng các cảm biến đắt tiền như radar hoặc LiDAR.

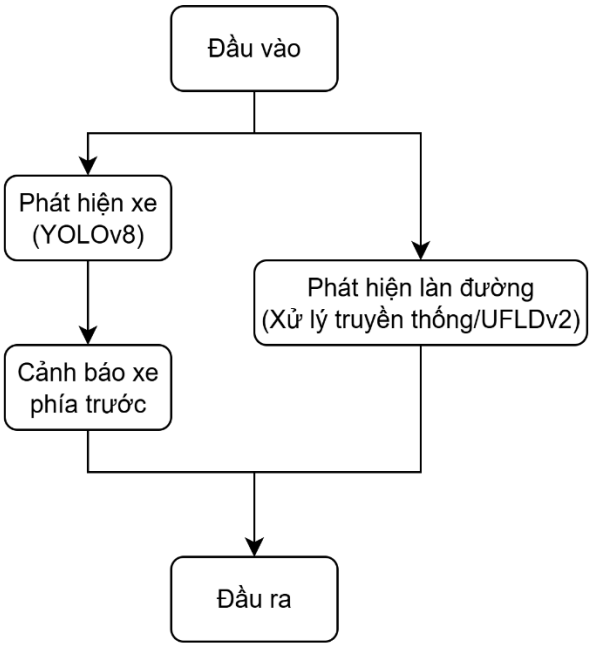
CHƯƠNG 3. THIẾT KẾ HỆ THỐNG

Chương 3 trình bày thiết kế tổng thể của hệ thống, bao gồm kiến trúc pipeline, chức năng của từng mô-đun và cách thức các mô-đun phối hợp để xử lý video từ camera hành trình. Trọng tâm của chương là mô tả chi tiết ba thành phần chính của hệ thống: mô-đun phát hiện làn đường (sử dụng phương pháp xử lý ảnh truyền thống hoặc mô hình UFLDv2), mô-đun nhận dạng phương tiện bằng YOLO, và mô-đun phát hiện và cảnh báo phương tiện phía trước.

Ngoài việc mô tả cấu trúc từng mô-đun, chương này cũng trình bày rõ luồng xử lý dữ liệu, cách tích hợp các mô hình học sâu vào pipeline, cũng như cơ chế ra quyết định cảnh báo. Nhờ đó, chương 3 đóng vai trò cầu nối giữa phần cơ sở lý thuyết ở Chương 2 và phần đánh giá - thực nghiệm ở Chương 4, đảm bảo người đọc nắm được toàn bộ quy trình xây dựng hệ thống hỗ trợ lái xe của đề tài.

3.1. Tổng quan hệ thống

Hệ thống được xây dựng dựa trên sự phối hợp giữa nhiều mô-đun xử lý khác nhau, bao gồm phát hiện làn đường, phát hiện phương tiện và cảnh báo phương tiện phía trước (*Hình 3.1*). Mỗi mô-đun đảm nhận một nhiệm vụ riêng, nhưng được thiết kế để hoạt động đồng bộ nhằm cung cấp đánh giá toàn diện về tình huống giao thông phía trước phương tiện.



Hình 3.1. Sơ đồ tổng thể của hệ thống

Trước hết, mô-đun nhận dạng làn đường (sử dụng xử lý ảnh truyền thống hoặc mô hình UFLDv2) có vai trò xác định cấu trúc làn đường, hỗ trợ định hướng và cung cấp thông tin ngữ cảnh cho toàn hệ thống. Tiếp theo, mô-đun nhận dạng phương tiện (ứng dụng YOLOv8) đảm nhiệm việc nhận dạng và xác định vị trí các phương tiện đang xuất hiện trong khung hình. Trên cơ sở đó, mô-đun cảnh báo phương tiện phía trước ước lượng khoảng cách tương đối tới phương tiện phía trước và phân loại mức độ an toàn theo thời gian thực nhằm đưa ra cảnh báo phù hợp.

Sự kết hợp giữa mô hình học sâu (YOLOv8, UFLDv2) và các kỹ thuật xử lý ảnh truyền thống giúp hệ thống khai thác được ưu điểm của cả hai hướng tiếp cận: tính chính xác và khả năng khái quát của mạng học sâu, cùng với tốc độ và sự linh hoạt của xử lý ảnh truyền thống. Nhờ đó, hệ thống vừa đạt độ tin cậy cao trong thực nghiệm, vừa đảm bảo thuận lợi cho việc triển khai, tinh chỉnh và đánh giá.

3.2. Mô hình hóa bài toán ở mức hệ thống

Bài toán của hệ thống hỗ trợ lái xe trong đề tài được mô hình hóa như một ánh xạ từ chuỗi khung hình video đầu vào sang các thông tin làn đường và trạng thái cảnh báo an toàn. Cụ thể, hệ thống có thể được biểu diễn dưới dạng hàm sau:

$$F : V \rightarrow \{L_{left}, L_{right}, S\} \tag{5}$$

<u>Trong đó:</u>

- F — *Hàm ánh xạ của hệ thống hỗ trợ lái xe, biểu diễn mối quan hệ từ dữ liệu video đầu vào sang các thông tin nhận dạng và trạng thái cảnh báo an toàn.*
- V — *Chuỗi các khung hình video thu được từ camera hành trình, trong đó mỗi khung hình chứa thông tin hình ảnh của làn đường và các phương tiện tham gia giao thông phía trước.*
- L_{left} — *Tập hợp các đặc trưng hình học hoặc các điểm biểu diễn làn đường bên trái phương tiện, được trích xuất từ dữ liệu video đầu vào.*
- L_{right} — *Tập hợp các đặc trưng hình học hoặc các điểm biểu diễn làn đường bên phải phương tiện, được trích xuất từ dữ liệu video đầu vào.*
- S — *Trạng thái cảnh báo an toàn của hệ thống, được xác định dựa trên khoảng cách tương đối và mức độ nguy hiểm giữa phương tiện và các đối tượng phía trước.*

3.3. Nhận dạng làn đường bằng xử lý ảnh truyền thống

3.3.1. Mục tiêu thiết kế hệ thống

Hệ thống được thiết kế để phát hiện và xác định chính xác vị trí làn đường trong từng khung hình video bằng các kỹ thuật xử lý ảnh truyền thống, đảm bảo hoạt động ổn định dưới nhiều điều kiện môi trường (thay đổi ánh sáng, nhiễu, rung lắc). Đồng thời, hệ thống đáp ứng yêu cầu xử lý gần thời gian thực cho ứng dụng hỗ trợ lái hoặc mô phỏng. Kết quả đầu ra hiển thị rõ ràng hai vạch làn trái/phải và vùng làn đường được làm nổi bật, giúp người dùng dễ quan sát và khai thác thông tin.

3.3.2. Chuẩn bị dữ liệu

Nguồn dữ liệu là video từ camera hành trình với góc nhìn cố định, bao quát ít nhất hai làn đường phía trước. Video có độ phân giải 1280×720 đến 1920×1080, tốc độ 25–30 FPS, lưu dưới dạng RGB trước khi chuẩn hóa. Tập dữ liệu gồm nhiều bối cảnh thực tế (cao tốc, thành phố, quốc lộ) với đường thẳng, cong, giao lộ; vạch kẻ liền, đứt hoặc mòn nhẹ, kèm yếu tố nhiễu như xe phía trước, bóng đổ, vật cản để đánh giá khả năng thích ứng của thuật toán.

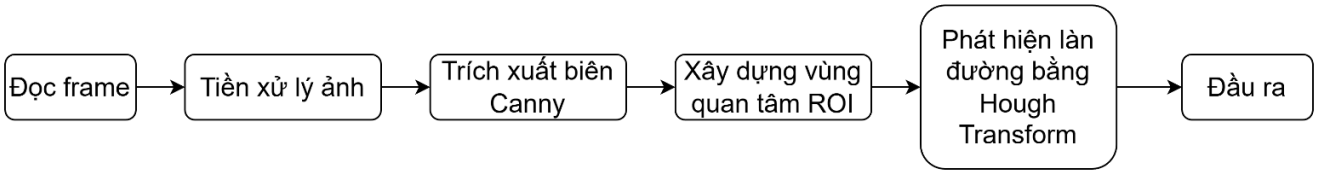
3.3.3. Quy trình thực hiện

Gọi video đầu vào là một chuỗi các khung hình rời rạc theo thời gian:

$$\mathcal{V} = \{ I_t \mid t = 1,2, \dots, T \} \tag{6}$$

Trong đó:

- $I_t \in \mathbb{R}^{H \times W \times 3}$: frame màu RGB tại thời điểm t
- H, W: chiều cao và chiều rộng ảnh



Hình 3.2. Sơ đồ khối của phương pháp xử lý ảnh truyền thống trong bài toán nhận dạng làn đường.

Từ sơ đồ khối của phương pháp xử lý ảnh truyền thống, ta có pipeline xử lý là một ánh xạ hàm tuần tự như sau:

$$I_t \xrightarrow{f_1} G_t \xrightarrow{f_2} E_t \xrightarrow{f_3} E_t^{ROI} \xrightarrow{f_4} L_t \xrightarrow{f_5} \hat{I}_t$$

Trong đó:

- G_t: ảnh xám
- E_t: bản đồ biên
- E_t^{ROI}: biên trong vùng quan tâm
- L_t: tập các đường thẳng làn đường
- $\hat{I}_t$: frame đầu ra có vẽ làn đường

Bước 1: Đọc frame: Đọc từng khung hình liên tiếp từ video đầu vào, mỗi frame tại thời điểm t sẽ có dạng như sau:

$$I_t(x, y) = [R_t(x, y), G_t(x, y), B_t(x, y)] \tag{7}$$

với: $(x, y) \in \Omega \subset \mathbb{Z}^2$ là tọa độ điểm ảnh

Hình 3.3. Ảnh gốc ban đầu.

Bước 2: Tiền xử lý ảnh: Thực hiện chuyển hóa ảnh màu sang ảnh xám (RGB → Gray) để giảm nhiễu và giảm số chiều dữ liệu.

Ảnh xám được tính bằng tổ hợp tuyến tính của các kênh màu:

$$G_t(x, y) = 0.299R_t(x, y) + 0.587G_t(x, y) + 0.114B_t(x, y) \tag{8}$$

Trong đó: $G_t(x, y) \in [0,255]$

Hình 3.4. Ảnh sau bước tiền xử lý.

Từ *Hình 3.3* và *Hình 3.4* cho thấy ảnh màu đầu vào đã được chuyển đổi sang ảnh xám. Quá trình này vẫn giữ lại các đặc trưng quan trọng như vạch kẻ đường và hình dạng phương tiện, đồng thời loại bỏ thông tin màu không cần thiết. Nhờ đó, bước tiền xử lý giúp giảm nhiễu, giảm độ phức tạp tính toán và tạo điều kiện để thuật toán phát hiện biên Canny hoạt động hiệu quả hơn.

Bước 3: Trích xuất biên Canny: Sử dụng thuật toán Canny để trích xuất biên của vạch kẻ đường, loại bỏ nhiễu nền và giữ lại các cạnh dạng đường thẳng. Quy trình Canny gồm 5 bước chính như sau:

Làm mượt ảnh bằng bộ lọc Gaussian Blur

Đầu tiên, ảnh đầu vào thường chứa nhiễu, gây ảnh hưởng đến quá trình phát hiện biên. Do đó, ảnh được làm mượt bằng bộ lọc Gaussian:

$$G_t^{\sigma}(x,y) = G_t(x,y) * \mathcal{G}_{\sigma}(x,y) \tag{9}$$

Trong đó:

$$\mathcal{G}_{\sigma}(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{10}$$

Mục đích để giảm nhiễu mà vẫn giữ lại các đặc trưng quan trọng (*Hình 3.5*).

Hình 3.5. Ảnh sau khi lọc nhiễu Gaussian

i. Tính gradient theo Sobel

Sau khi làm mượt, thuật toán sử dụng toán tử Sobel để tính độ lớn và hướng của gradient tại mỗi điểm ảnh:

Gradient theo hai hướng:

$$\begin{aligned} G_x &= G_t^\sigma * S_x \\ G_y &= G_t^\sigma * S_y \end{aligned} \tag{11}$$

Độ lớn gradient:

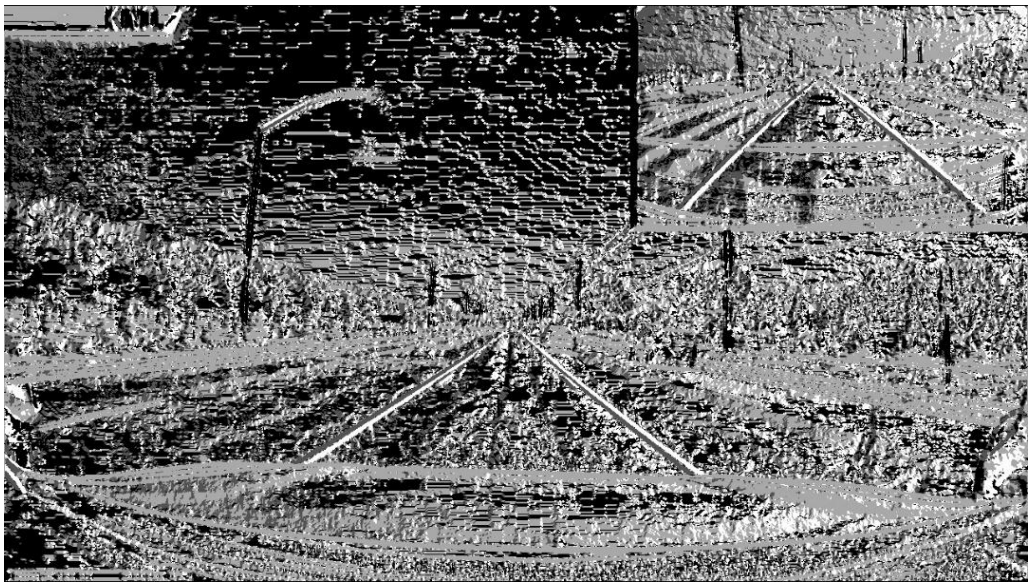
$$M(x,y) = \sqrt{G_x^2(x,y) + G_y^2(x,y)} \tag{12}$$

Hướng gradient:

$$\theta(x,y) = \arctan\left(\frac{G_y(x,y)}{G_x(x,y)}\right) \tag{13}$$

Đây là cơ sở để xác định vị trí và độ sắc nét của biên (*Hình 3.6* và *Hình 3.7*).

Hình 3.6. Ảnh mô tả cường độ Gradient



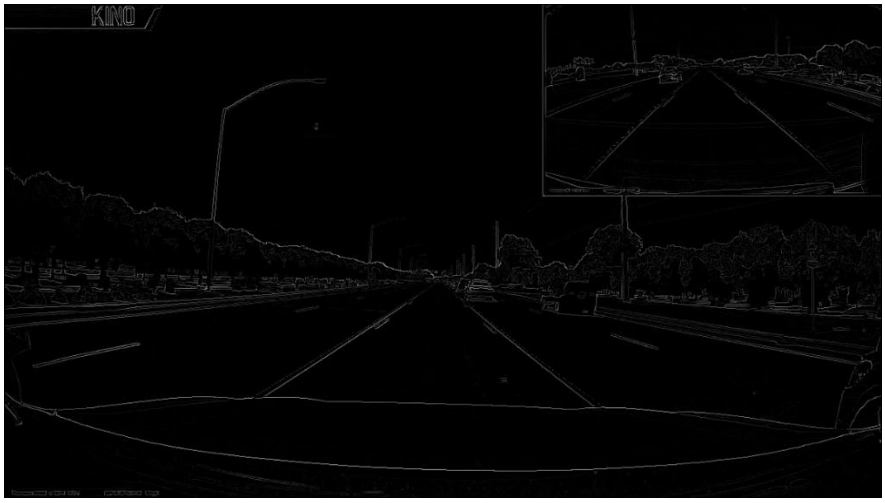
Hình 3.7. Ảnh mô tả hướng Gradient

ii. Non-Maximum Suppression (NMS)

Để làm mảnh biên và loại bỏ các điểm không phải là biên thực, thuật toán áp dụng Non-Maximum Suppression (NMS):

$$M_{NMS}(x,y) = \begin{cases} M(x,y), & \text{nếu } M(x,y) = \max(M_{dir}) \\ 0, & \text{ngược lại} \end{cases} \quad (14)$$

Mục đích nhằm giữ lại những điểm có giá trị gradient lớn nhất theo hướng gradient (*Hình 3.8*).



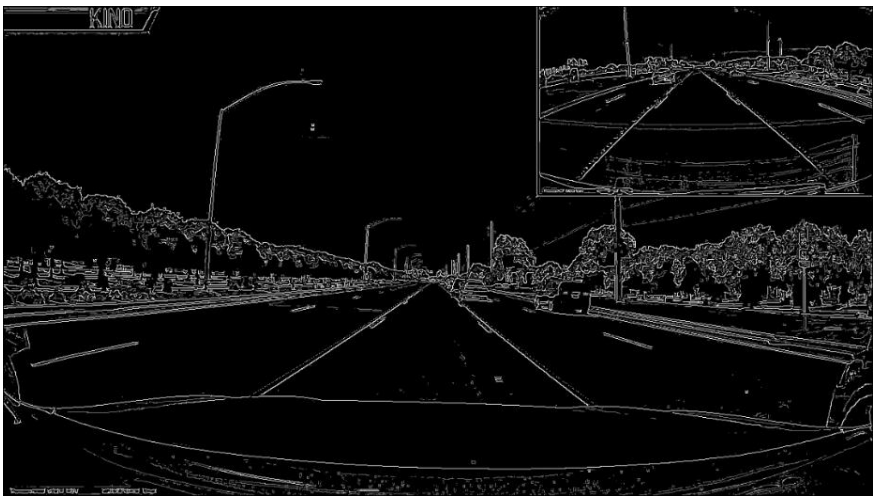
Hình 3.8. Ảnh sau khi áp dụng Non-Maximum Suppression (NMS)

iii. Double Threshold

Kết quả sau Non-Maximum Suppression (NMS) được phân loại dựa trên hai ngưỡng: ngưỡng thấp và ngưỡng cao. Các điểm có gradient lớn hơn ngưỡng cao được xem là biên mạnh, các điểm nằm giữa hai ngưỡng được xem là biên yếu, các điểm dưới ngưỡng thấp bị loại bỏ (*Hình 3.9*).

Công thức toán học cho Double Threshold:

$$E(x,y) = \begin{cases} \text{strong edge}, & M_{NMS}(x,y) \geq T_{high} \\ \text{weak edge}, & T_{low} \leq M_{NMS}(x,y) < T_{high} \\ 0, & M_{NMS}(x,y) < T_{low} \end{cases} \tag{15}$$



Hình 3.9. Ảnh sau khi phân loại ngưỡng.

iv. Edge Tracking by Hysteresis

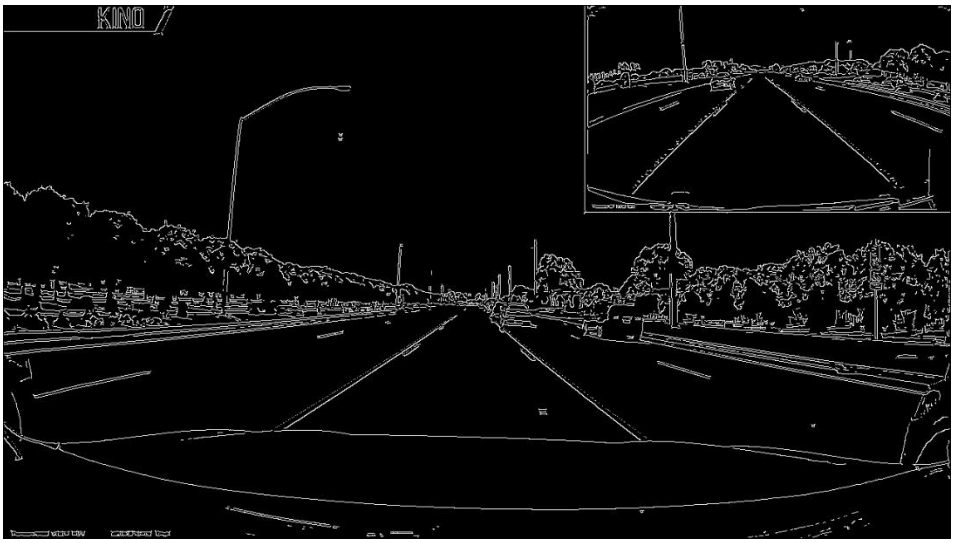
Cuối cùng thuật toán thực hiện liên kết biên. Các biên yếu chỉ được giữ lại nếu chúng nằm kề với biên mạnh, giúp loại bỏ nhiễu nhưng vẫn giữ được cấu trúc biên liên tục (*Hình 3.10*).

Biên yếu e_w chỉ được giữ nếu tồn tại biên mạnh e_s kề nó:

$$e_w \in E \iff \exists e_s \in \mathcal{N}(e_w) \tag{16}$$

Kết quả cuối:

$$E_t(x, y) \in \{0,1\}$$



Hình 3.10. Ảnh sau khi thực hiện liên kết biên bằng Hysteresis

Hình 3.11. Ảnh sau khi thực hiện trích xuất cạnh biên bằng phương pháp Canny.

Kết quả thu được sau khi thực hiện bước trích xuất biên Canny được thể hiện ở *Hình 3.11*, kết quả là một bản đồ biên rõ ràng, trong đó các đường kẻ làn đường xuất hiện nổi bật so với nền. Đây là tiền đề quan trọng để xác định chính xác vị trí làn đường ở bước tiếp theo.

Bước 4: Xây dựng vùng quan tâm:

Định nghĩa mặt nạ ROI:

$$R(x,y) = \begin{cases} 1, & (x,y) \in \Omega_{ROI} \\ 0, & \text{ngược lại} \end{cases} \tag{17}$$

Áp dụng ROI:

$$E_t^{ROI}(x,y) = E_t(x,y) \cdot R(x,y) \tag{18}$$

Trong đó:

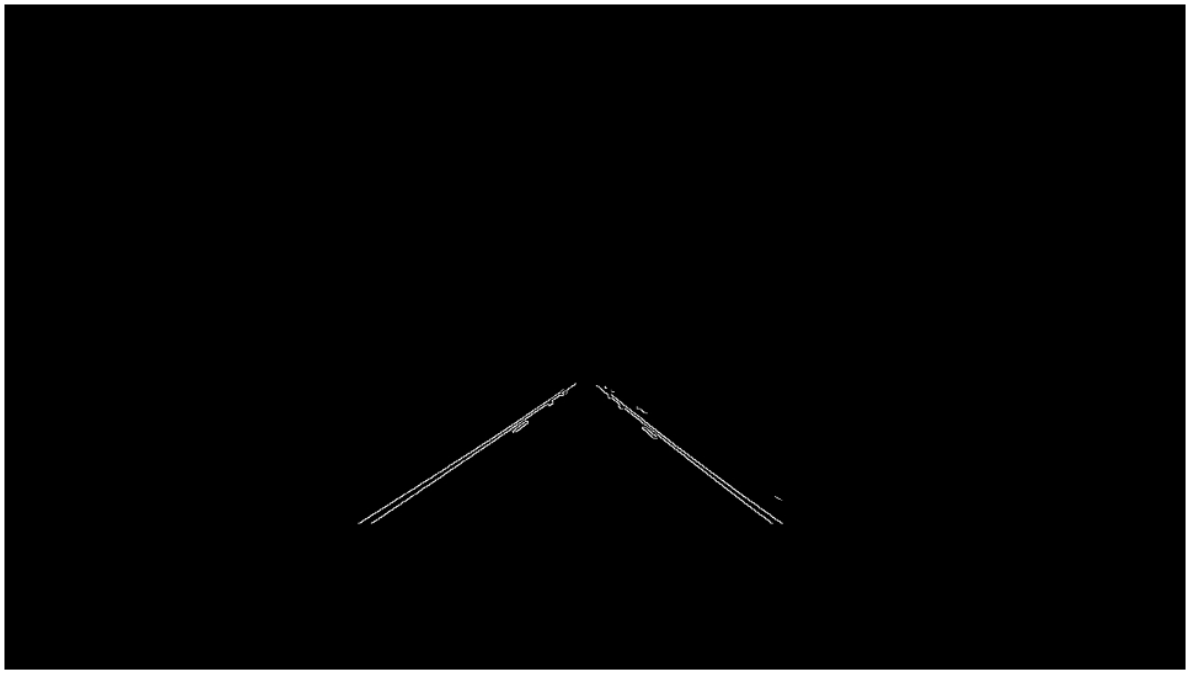
- Ω_{ROI}: vùng hình thang / tam giác xác định theo camera

Mục đích: Chỉ giữ lại vùng tam giác/trapezoid nơi làn đường xuất hiện. ROI giúp:

- Loại bỏ bầu trời, phương tiện phía trước, biển báo.
- Giảm độ phức tạp → tăng tốc xử lý.

Thường sử dụng mask hình thang cụ thể theo vị trí camera.

Hình 3.12. Hình ảnh cho vùng ROI.



Hình 3.13. Ảnh sau khi thực hiện xây dựng ROI.

So sánh *Hình 3.11.* và *Hình 3.13.* cho thấy rằng sau khi áp dụng vùng quan tâm (ROI), các vùng cạnh biên không cần thiết như cây cối, bầu trời hay nhà cửa đã được loại bỏ đáng kể. Nhờ chỉ tập trung xử lý trong ROI, hệ thống tránh được các đường cạnh không liên quan ở khu vực bên ngoài, từ đó giảm khối lượng tính toán và đồng thời nâng cao độ chính xác của mô hình.

Hình 3.12. cho thấy vùng quan tâm (ROI) được xác định nhằm giới hạn khu vực xử lý, giúp thuật toán tập trung nhận dạng chính xác các đường làn phương tiện trên mặt đường.

Bước 5: Phát hiện làn đường bằng phép biến đổi Hough: Sử dụng biến đổi Hough để tìm đoạn thẳng trong vùng biên đã được trích xuất như sau:

Mỗi đường thẳng trong không gian ảnh được biểu diễn bởi:

$$\rho = x\cos\theta + y\sin\theta \tag{19}$$

Mỗi điểm biên (x_i, y_i) bỏ phiếu vào không gian tham số:

$$H(\rho,\theta) = \sum_i \delta(\rho - x_i\cos\ \theta - y_i\sin\ \theta) \tag{20}$$

Các cực đại trong $H(\rho,\theta)$ tương ứng với các đường thẳng.

Tiếp theo thực hiện gom nhóm các đoạn thẳng thành hai nhóm là trái và phải:

Tính hệ số góc:

$$k = \frac{y_2 - y_1}{x_2 - x_1} \tag{21}$$

- $k < 0$ ⇒làn trái
- $k > 0$ ⇒làn phải

Tính toán trung bình và kéo dài 2 vạch kẻ đường chính biểu diễn làn đường:

$$\bar{L}_{left}, \bar{L}_{right}$$

và kéo dài đến biên ảnh để biểu diễn làn đường chính.

```
HoughLinesP output at frame 20: [[[382 561 624 410]]

 [[659 426 825 560]]

 [[395 561 606 424]]

 [[644 411 831 557]]

 [[660 427 804 544]]

 [[659 425 741 489]]

 [[569 447 605 424]]

 [[662 427 779 518]]

 [[394 561 560 458]]

 [[468 509 616 413]]]
```

Hình 3.14. Kết quả sau khi thực hiện Hough Transform

Từ kết quả *Hình 3.14* có thể thấy rằng mỗi phần tử trong mảng là một đoạn thẳng được phát hiện trong ảnh sau khi áp dụng thuật toán Canny Edge và giới hạn vùng quan tâm (ROI). Các đoạn thẳng này được biểu diễn dưới dạng cấu trúc [[x1 y1 x2 y2]], trong đó (x1, y1) là tọa độ điểm đầu và (x2, y2) là tọa độ điểm cuối của đoạn thẳng trong hệ tọa độ ảnh. Những giá trị này cung cấp thông tin hình học quan trọng để xác định vị trí và hướng của các đường biên, từ đó phục vụ cho việc nhận dạng làn đường trong quá trình xử lý ảnh.

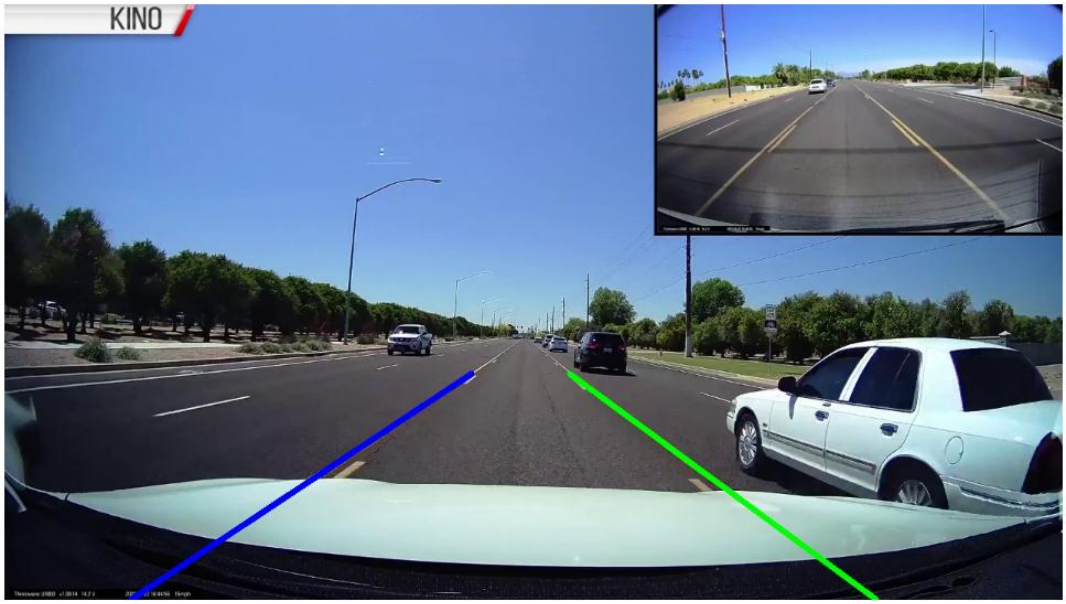
Bước 6: Xuất đầu ra:

Vẽ các đường làn lên frame gốc:

$$\hat{I}_t = I_t + \bar{L}_{left} + \bar{L}_{right} \tag{22}$$

Kết quả là chuỗi video:

$$\hat{\mathcal{V}} = \{\hat{I}_t \mid t = 1, \dots, T\} \tag{23}$$



Hình 3.15. Kết quả nhận dạng vạch kẻ làn đường.

Từ kết quả *Hình 3.15.* cho ta thấy thuật toán đã phát hiện và vẽ được hai đường làn chính trên mặt đường bằng các đường thẳng màu xanh lam và xanh lục. Các đường này cơ bản bám sát vị trí thực tế của vạch kẻ đường, chứng tỏ phương pháp Hough Transform kết hợp với Canny hoạt động hiệu quả trong việc nhận dạng biên.

3.4. Nhận dạng làn đường bằng UFLDv2

3.4.1. Mục tiêu của mô-đun UFLDv2

Mô-đun UFLDv2 có nhiệm vụ xác định vị trí và hình dạng làn đường từ video camera hành trình, đóng vai trò nền tảng cho chức năng cảnh báo an toàn trong hệ thống. Trong phạm vi đề tài, mô-đun chỉ thực hiện nhận dạng làn đường, không xét đến phân loại loại làn hay phát hiện hành vi lấn làn.

Để phù hợp với điều kiện giao thông thực tế tại Việt Nam, mô hình UFLDv2 được fine-tune trên hơn 2700 ảnh làn đường thu thập tại Đà Nẵng, giúp mô hình thích nghi tốt với các tình huống như vạch kẻ đường mờ, xuống cấp, nhiễu ánh sáng hoặc bị che khuất.

Bên cạnh độ chính xác, mô-đun được thiết kế để đáp ứng yêu cầu xử lý thời gian thực, cho phép nhận dạng làn đường liên tục theo từng khung hình và kết hợp với mô-đun phát hiện phương tiện nhằm ước lượng khoảng cách tương đối, phục vụ chức năng cảnh báo va chạm.

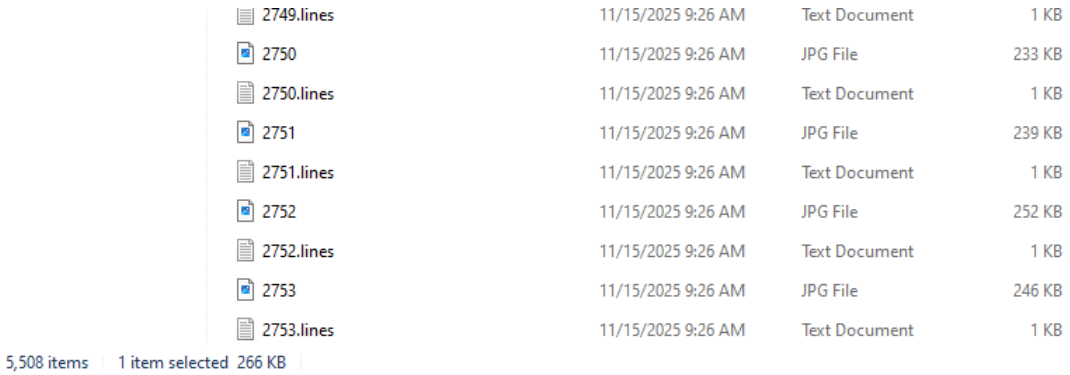
Đầu ra của mô-đun được chuẩn hóa dưới dạng các tập điểm lane-left và lane-right, thuận tiện cho việc tích hợp với các mô-đun khác và xây dựng thông tin vị trí tương đối giữa phương tiện và làn đường.

3.4.2. Chuẩn bị và tiền xử lý dữ liệu

Để tinh chỉnh mô hình UFLDv2 phù hợp với điều kiện đường sá tại Việt Nam, nhóm đã xây dựng một tập dữ liệu gồm 2754 ảnh thu thập từ camera hành trình trong nhiều điều kiện thời tiết và môi trường khác nhau. Tập dữ liệu được xây dựng theo chuẩn của UFLDv2, trong đó mỗi ảnh được gán nhãn thủ công dưới dạng các điểm mô tả hình dạng của từng làn đường.

Hình 3.16. Ví dụ ảnh làn đường sau khi được gán nhãn theo cơ chế anchor

Quá trình gán nhãn được thực hiện bằng công cụ CVAT (Computer Vision Annotation Tool) **[13]** thông qua việc đặt các điểm polyline theo cơ chế row-anchor và column-anchor. *Hình 3.16* minh họa một ví dụ gán nhãn, trong đó các điểm được phân bố dọc theo chiều cao khung hình để xác định hình dạng của lane-left và lane-right. Các điểm mốc này cung cấp thông tin hình học cần thiết để mô hình học được cấu trúc và sự thay đổi của làn đường trong các điều kiện môi trường khác nhau.



2749.lines	11/15/2025 9:26 AM	Text Document	1 KB
2750	11/15/2025 9:26 AM	JPG File	233 KB
2750.lines	11/15/2025 9:26 AM	Text Document	1 KB
2751	11/15/2025 9:26 AM	JPG File	239 KB
2751.lines	11/15/2025 9:26 AM	Text Document	1 KB
2752	11/15/2025 9:26 AM	JPG File	252 KB
2752.lines	11/15/2025 9:26 AM	Text Document	1 KB
2753	11/15/2025 9:26 AM	JPG File	246 KB
2753.lines	11/15/2025 9:26 AM	Text Document	1 KB

5,508 items 1 item selected 266 KB

Hình 3.17. Cấu trúc thư mục chứa ảnh và tệp nhãn (.lines) trong tập dữ liệu huấn luyện

Sau khi hoàn thành quá trình gán nhãn, dữ liệu được tổ chức theo cấu trúc chuẩn của UFLDv2. Mỗi ảnh (.jpg) đều có một tệp nhãn tương ứng (.lines) chứa danh sách điểm mô tả lane. Cấu trúc thư mục huấn luyện được minh họa trong *Hình 3.17*, trong đó các cặp tệp ảnh-nhãn nằm cùng thư mục để đảm bảo tính nhất quán khi load dữ liệu vào mô hình.

3.4.3. Quy trình fine-tune mô hình UFLDv2

Quá trình fine-tune được thực hiện trên mô hình UFLDv2 đã được huấn luyện trước, với mục tiêu điều chỉnh trọng số để mô hình thích nghi tốt hơn với điều kiện làn đường thực tế tại Việt Nam. Kiến trúc mạng được giữ nguyên hoàn toàn; chỉ các trọng số được cập nhật dựa trên tập dữ liệu 2754 ảnh đã gán nhãn. Việc huấn luyện được triển khai trên Google Colab với GPU để bảo đảm hiệu năng.

Tập dữ liệu được chia thành 80% cho huấn luyện (2203 ảnh) và 20% (551 ảnh) cho kiểm tra, bảo đảm sự cân bằng giữa khả năng học và khả năng đánh giá. Trong quá trình huấn luyện, mỗi lô dữ liệu (batch) gồm ảnh và tệp nhãn tương ứng sẽ được chuyển đổi về kích thước đầu vào chuẩn của UFLDv2 và mã hóa theo định dạng row-anchor và column-anchor. Một số phép biến đổi đơn giản như thay đổi độ sáng hoặc lật ngang được áp dụng nhằm tăng tính đa dạng của dữ liệu và hạn chế overfitting.

Mô hình được tối ưu hóa bằng thuật toán Stochastic Gradient Descent (SGD) **[14]** với momentum, vốn là bộ tối ưu mặc định của UFLDv2. SGD giúp quá trình cập nhật trọng số diễn ra ổn định hơn và phù hợp với kiến trúc backbone kiểu ResNet. Learning rate nhỏ được lựa chọn nhằm tránh làm mất thông tin đã học từ mô hình pretrained. Hàm mất mát của UFLDv2 gồm hai thành phần chính classification loss và regression loss:

$$L = L_{cls} + aL_{reg} \tag{24}$$

<u>Chú thích:</u>

L: tổng hàm mất mát của mô hình

L_{cls}: mất mát của hàm phân lớp

L_{reg}: mất mát hồi quy vị trí điểm lane

a: hệ số cân bằng giữa 2 thành phần (mặc định = 1)

Classification loss được tính bằng Binary Cross Entropy nhằm dự đoán xác suất tồn tại của lane tại mỗi anchor:

$$L_{cls} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \quad (25)$$

Chú thích:

N: tổng số anchor

$y_i \in \{0,1\}$: nhãn thật biểu thị anchor thứ i có lane hay không

p_i: xác suất mô hình dự đoán anchor thứ i có lane

Regression loss sử dụng Smooth L1 Loss để ước lượng vị trí các điểm lane theo row-anchor và column-anchor:

$$L_{reg} = -\frac{1}{M}\sum_{j=1}^{M} SmoothL1(x_j - \hat{x}_j) \quad (26)$$

Trong đó:

$$SmoothL1(d) = \begin{cases} 0.5d^2, khi\ |d| < 1 \\ |d| - 0.5, ngược\ lại \end{cases}$$

Chú thích:

M: tổng số điểm lane được gán nhãn (row-anchor + column-anchor)

x_j: tọa độ thật của điểm lane thứ j

$\hat{x}_j$: tọa độ mà mô hình dự đoán

d: độ lệch (error), tức là khoảng cách giữa dự đoán và giá trị thật

Hai thành phần mất mát được tối ưu đồng thời trong suốt quá trình huấn luyện để mô hình học được hình dạng làn đường trong nhiều điều kiện môi trường khác nhau.

Khi mô hình đạt độ hội tụ ổn định, trọng số được lưu lại dưới dạng checkpoint (.pth). Các đường cong hàm mất mát và kết quả đánh giá mô hình sẽ được trình bày chi tiết trong Chương 4.

3.4.4. Tích hợp mô-đun UFLDv2 vào hệ thống

Mô-đun UFLDv2 sau khi tinh chỉnh được tích hợp vào hệ thống tổng thể với vai trò cung cấp thông tin hình học của làn đường theo thời gian thực. Đầu vào của mô-đun là từng khung hình từ video camera hành trình, và đầu ra là hai tập điểm mô tả lane-left và

lane-right đã được nội suy thành đường cong liên tục. Các thông tin này được chuẩn hóa theo định dạng cố định để đảm bảo khả năng tương thích với các mô-đun khác trong hệ thống.

Bên cạnh đó, việc tích hợp mô-đun UFLDv2 giúp hệ thống duy trì tính ổn định trong các tình huống lái phương tiện thực tế, nơi điều kiện ánh sáng và chất lượng vạch kẻ đường thay đổi liên tục. Mô-đun hoạt động độc lập nhưng truyền dữ liệu làn đường theo thời gian thực cho toàn bộ pipeline, đóng vai trò quan trọng trong việc xây dựng hệ thống cảnh báo an toàn dựa trên camera.

3.5. Phát hiện và cảnh báo phương tiện phía trước

Trong hệ thống đề tài, chức năng phát hiện và cảnh báo phương tiện phía trước được xây dựng nhằm hỗ trợ đánh giá tình huống giao thông phía trước phương tiện của người lái. Chức năng này hoạt động trực tiếp trên từng khung hình video thu từ camera hành trình và được triển khai theo thời gian thực. Hệ thống sử dụng mô hình YOLOv8 huấn luyện sẵn để nhận dạng phương tiện **[15]**, sau đó kết hợp với các điều kiện hình học và ngưỡng quyết định nhằm đưa ra cảnh báo phù hợp.

3.5.1. Nhận dạng phương tiện

Với mỗi khung hình đầu vào tại thời điểm t, ảnh màu thu được từ camera hành trình được biểu diễn dưới dạng:

$$I_t \in \mathbb{R}^{H \times W \times 3} \tag{27}$$

Trong đó H và W lần lượt là chiều cao và chiều rộng của ảnh.

Mô hình YOLO thực hiện phép ánh xạ từ ảnh đầu vào sang tập các đối tượng được phát hiện trong khung hình:

$$f: I_t \rightarrow B_t \tag{28}$$

Trong đó, $\mathcal{B}_t$ là tập các bounding box do mô hình YOLOv8 xác định tại thời điểm t:

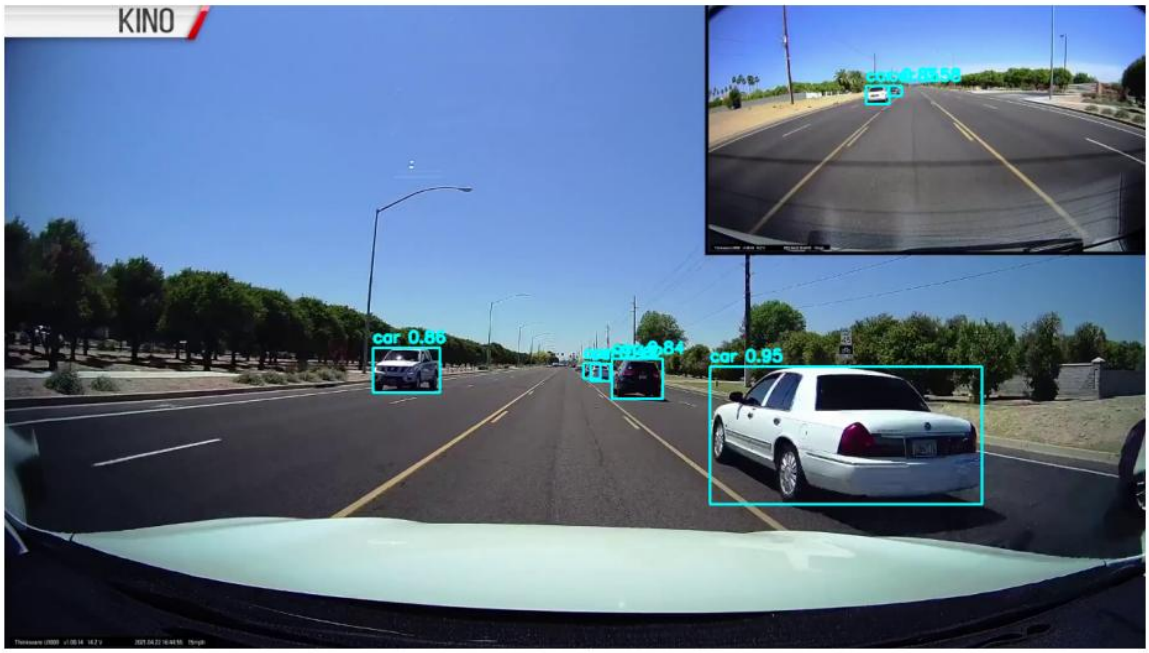
$$\mathcal{B}_t = \{(b_i, c_i, s_i)\}_{i=1}^{N} \tag{29}$$

với:

- $b_i = (x_{i1}, y_{i1}, x_{i2}, y_{i2})$: tọa độ bounding box thứ i
- c_i: nhãn lớp phương tiện
- $s_i \in [0,1]$: độ tin cậy của dự đoán

Trong quá trình xử lý, các bounding box có độ tin cậy nhỏ hơn một ngưỡng xác định δ sẽ bị loại bỏ:

$$s_i < \delta \tag{30}$$



Hình 3.18. Sử dụng YOLOv8 để nhận dạng phương tiện

3.5.2. Lựa chọn phương tiện đang ở phía trước

Không phải phương tiện nào xuất hiện trong khung hình cũng liên quan đến nguy cơ va chạm. Do đó, hệ thống cần xác định chiếc phương tiện phía trước, phương tiện có khả năng gây nguy hiểm trực tiếp cho phương tiện của người lái. Tiêu chí lựa chọn gồm hai bước:

a) Lọc theo vùng quan tâm

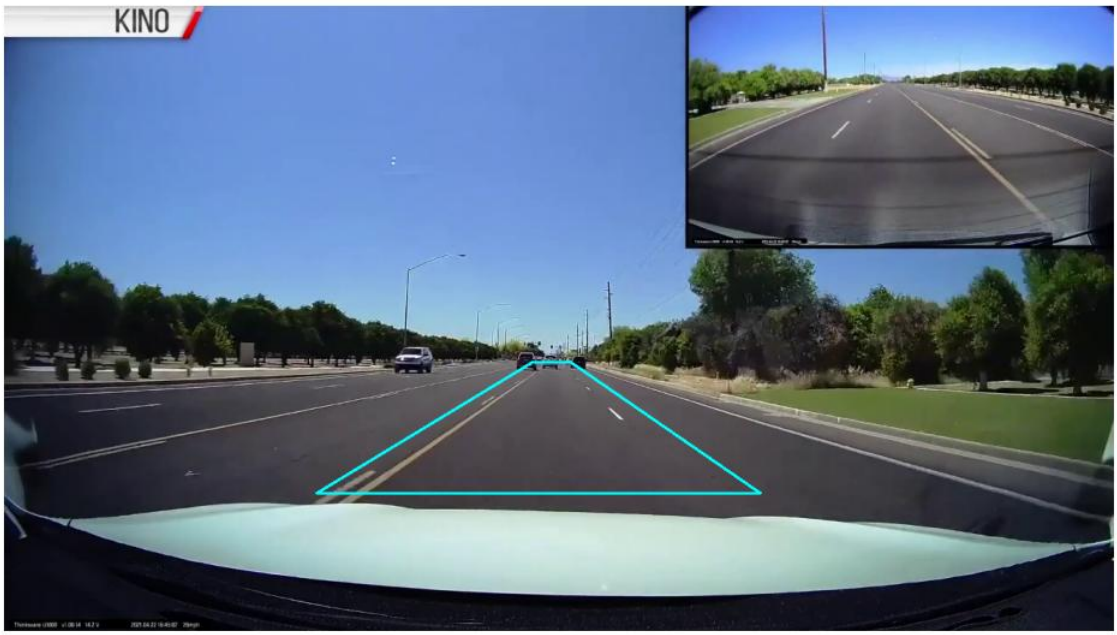
Hệ thống sử dụng một vùng ROI dạng hình thang để mô phỏng khu vực đường ngay phía trước đầu phương tiện. Cách lọc này giúp loại bỏ hiệu quả:

- Phương tiện ở làn trái/phải không liên quan
- Phương tiện di chuyển ngược chiều ở phía bên kia đường
- Các phương tiện bên lề đường

Một phương tiện được xem là nằm phía trước nếu tỉ lệ diện tích bounding box nằm trong ROI thỏa điều kiện:

$$\frac{\text{Area}(b_i \cap \text{ROI})}{\text{Area}(b_i)} \geq \tau \tag{31}$$

với τ là ngưỡng diện tích, giúp loại bỏ các phương tiện ở làn bên cạnh hoặc không nằm trong hướng di chuyển chính.



Hình 3.19. Vùng ROI mô phỏng khu vực đường phía trước phương tiện.

b) Chọn phương tiện gần nhất

Sau khi áp dụng các điều kiện lọc, hệ thống tiến hành xác định phương tiện phía trước có khả năng gây nguy hiểm cao nhất.

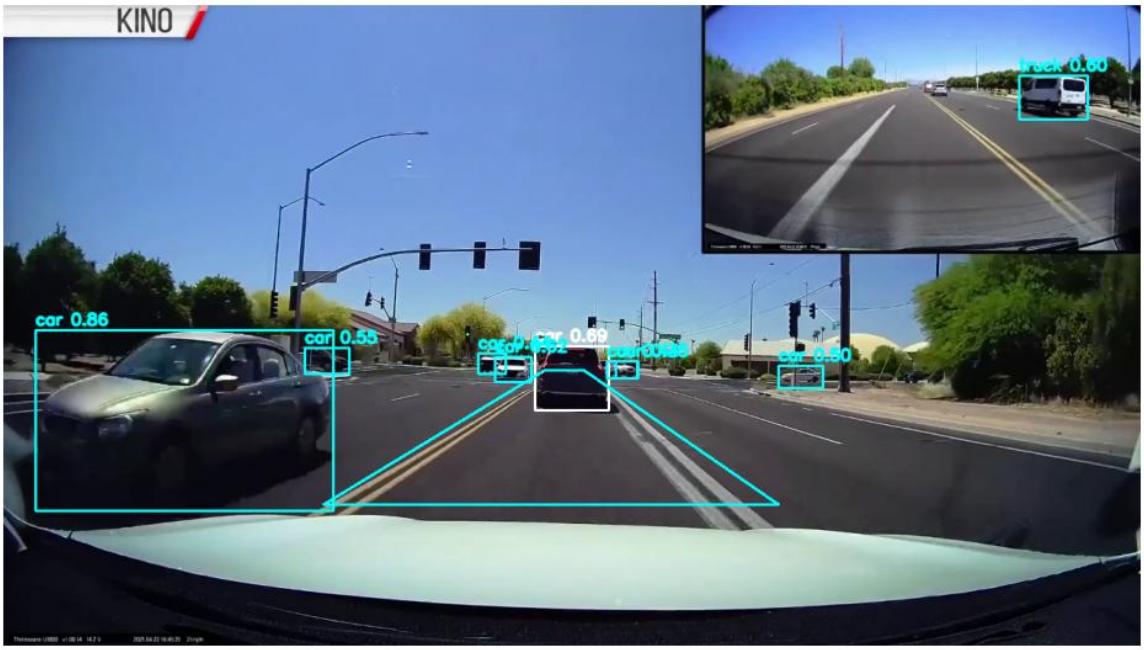
Chiều cao của bounding box được tính bằng:

$$h_i = y_{i2} - y_{i1} \tag{32}$$

Phương tiện phía trước được xác định là phương tiện có chiều cao bounding box lớn nhất:

$$i^* = \arg\max_i (h_i) \tag{33}$$

Cách lựa chọn này tuân theo nguyên lý của mô hình camera pinhole: phương tiện càng gần camera thì kích thước biểu kiến của nó trong ảnh càng lớn, tức là bounding box càng cao. Nhờ vậy, tại mỗi thời điểm, hệ thống luôn ưu tiên phương tiện có mức độ rủi ro cao nhất và phản ánh đúng tình huống giao thông phía trước xe.



Hình 3.20. Quy trình lọc và chọn phương tiện phía trước

3.5.3. Ước lượng khoảng cách

Do hệ thống sử dụng camera hành trình nên không thể biết chính xác khoảng cách giữa hai phương tiện. Thay vào đó, nhóm đã sử dụng phương pháp ước lượng tương đối dựa trên mối quan hệ giữa kích thước biểu kiến của vật thể trong ảnh và khoảng cách của nó đến camera **[16]**.

Khoảng cách được tính xấp xỉ theo công thức:

$$D \approx \frac{\alpha}{h_{i^*}} \tag{34}$$

Trong đó:

- D: khoảng cách ước lượng đến phương tiện phía trước
- h_{i^*}: chiều cao bounding box của phương tiện được chọn
- α: hệ số hiệu chỉnh phụ thuộc đặc tính camera và vị trí lắp đặt

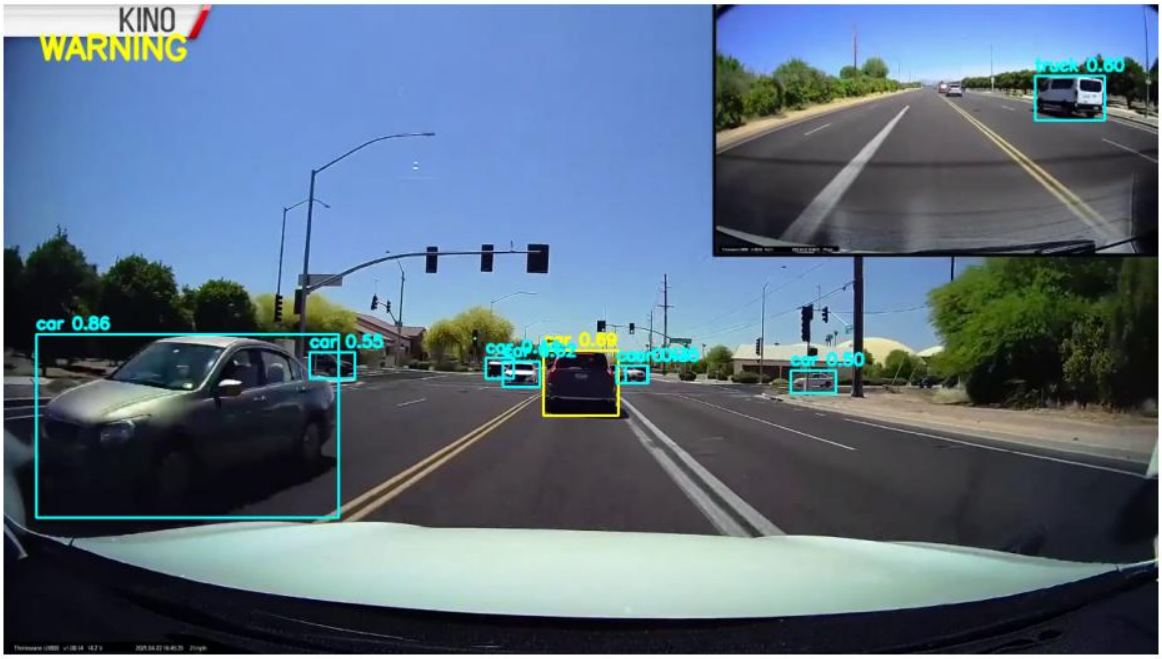
Mặc dù chỉ cho ra kết quả tương đối, nhưng cách tiếp cận này vẫn đủ tin cậy để phân loại mức độ nguy hiểm, đặc biệt trong bài toán cảnh báo va chạm sớm.

3.5.4. Đánh giá mức độ nguy hiểm

Dựa trên khoảng cách ước lượng D, hệ thống tiến hành phân loại mức độ cảnh báo theo ba cấp độ nhằm phản ánh mức độ rủi ro trong tình huống giao thông.

$$\text{Trạng thái}(D) = \begin{cases} \text{Nguy hiểm}, & D < D_1 \\ \text{Cảnh báo}, & D_1 \leq D < D_2 \\ \text{An toàn}, & D \geq D_2 \end{cases} \tag{35}$$

Trong đó D_1 và D_2 là các ngưỡng khoảng cách được xác định thông qua quá trình thử nghiệm thực tế. Nhờ cơ chế phân cấp rõ ràng này, hệ thống có thể đánh giá nhanh mức độ nguy cơ va chạm và hiển thị phản hồi trực quan tương ứng, giúp người lái nhận biết tình huống kịp thời mà không bị phân tán sự chú ý.



Hình 3.21. Hình ảnh minh hoạ quá trình cảnh báo

3.5.5. Hiển thị cảnh báo

Để người dùng dễ dàng nhận biết tình huống, hệ thống in cảnh báo trực tiếp lên video theo hai thành phần trực quan gồm màu sắc của bounding box và nhãn cảnh báo.

Bounding box của phương tiện phía trước được tô màu theo cấp độ cảnh báo:

- Màu đỏ tương ứng với mức "Nguy hiểm"
- Màu vàng tương ứng với mức "Cảnh báo"
- Màu xanh lục tương ứng với mức "An toàn"

Màu sắc thay đổi theo thời gian thực, giúp người lái quan sát biến động khoảng cách một cách trực quan mà không cần đọc số liệu.

Bên cạnh đó, dòng chữ trạng thái "DANGER", "WARNING" hoặc "SAFE" được in lên góc trên khung hình với màu sắc tương ứng, tạo nên một lớp cảnh báo rõ ràng và dễ nhận biết trong quá trình theo dõi video.

3.6. Đánh giá độ phức tạp và tốc độ hội tụ

3.6.1. Đánh giá độ phức tạp tính toán của các phương pháp

Trong hệ thống được đề xuất, độ phức tạp tính toán của các phương pháp được sử dụng, bao gồm xử lý ảnh truyền thống và các mô hình học sâu, được đánh giá dựa trên mức độ phụ thuộc vào kích thước dữ liệu đầu vào và khả năng đáp ứng yêu cầu xử lý thời gian thực.

Đối với các phương pháp xử lý ảnh truyền thống như tiền xử lý ảnh, phát hiện biên Canny và biến đổi Hough, chi phí tính toán chủ yếu phụ thuộc vào số lượng điểm ảnh trong mỗi khung hình. Do đó, độ phức tạp của các phương pháp này tăng gần tuyến tính theo kích thước ảnh đầu vào và có thể đáp ứng tốt yêu cầu xử lý thời gian thực, tuy nhiên độ ổn định giảm trong các điều kiện môi trường phức tạp.

Đối với các mô hình học sâu, chi phí tính toán tập trung chủ yếu ở giai đoạn suy luận. Mô hình UFLDv2 được thiết kế theo hướng gọn nhẹ, sử dụng backbone ResNet-34 kết hợp với cơ chế dự đoán dựa trên anchor, giúp giảm số lượng tham số cần ước lượng và cải thiện tốc độ xử lý. Bên cạnh đó, mô hình YOLOv8 được tối ưu cho bài toán phát hiện đối tượng với độ trễ thấp. Nhờ đó, toàn bộ hệ thống có khả năng xử lý liên tục luồng video từ camera hành trình và đáp ứng yêu cầu vận hành trong thời gian thực trên phần cứng phổ thông.

3.6.2. Đánh giá tốc độ hội tụ của mô hình học sâu

Trong quá trình huấn luyện, các mô hình học sâu được sử dụng trong đề tài, đặc biệt là UFLDv2, thể hiện khả năng hội tụ ổn định. Điều này được thể hiện qua việc giá trị hàm mất mát giảm dần theo số epoch và đạt trạng thái gần như không đổi ở các epoch cuối.

Việc fine-tune mô hình trên tập dữ liệu làn đường thực tế tại Việt Nam giúp mô hình nhanh chóng thích nghi với các đặc trưng dữ liệu đầu vào, đồng thời hạn chế hiện tượng overfitting. Quá trình huấn luyện diễn ra ổn định, không xuất hiện hiện tượng dao động mạnh của hàm mất mát, cho thấy cấu hình huấn luyện và lựa chọn tham số phù hợp với bài toán. Kết quả này khẳng định các mô hình học sâu trong hệ thống đáp ứng tốt yêu cầu thực nghiệm và có khả năng ứng dụng trong các hệ thống hỗ trợ lái xe thực tế. Xu hướng giảm của hàm mất mát theo số epoch được thể hiện rõ trong các kết quả thực nghiệm trình bày ở Chương 4.

CHƯƠNG 4. KẾT QUẢ VÀ ĐÁNH GIÁ

Chương 4 trình bày các kết quả đạt được từ những mô-đun đã được thiết kế và triển khai ở Chương 3, đồng thời đánh giá mức độ hiệu quả của từng phương pháp trong nhiều điều kiện thực tế khác nhau. Các nội dung được trình bày bao gồm: kết quả phát hiện làn đường bằng xử lý ảnh truyền thống trên MATLAB và Python, kết quả fine-tune và suy luận của mô hình học sâu UFLDv2, cũng như hiệu quả của mô-đun phát hiện phương tiện và cảnh báo phương tiện phía trước bằng YOLO.

Bên cạnh việc đánh giá bằng trực quan hình ảnh, chương này còn phân tích các chỉ số định lượng như độ chính xác, tốc độ xử lý (FPS), độ ổn định và khả năng hoạt động trong môi trường đường sá thực tế. Thông qua đó, chương 4 đóng vai trò quan trọng trong việc xác định tính khả thi, ưu điểm và hạn chế của các phương pháp được sử dụng, đồng thời làm cơ sở để đề xuất hướng cải tiến trong các nghiên cứu tiếp theo.

4.1. Kết quả nhận dạng làn đường bằng xử lý ảnh truyền thống

4.1.1. Kết quả trực quan

a) b)

Hình 4.1. Kết quả nhận dạng làn đường: (a) Python, (b) MATLAB

Hệ thống phát hiện làn đường được kiểm thử trên dữ liệu thực tế thu thập từ camera gắn trên phương tiện trong điều kiện đường nhiều làn, ánh sáng tốt và tầm nhìn rõ ràng. Kết quả cho thấy các đường màu xanh dương và xanh lá được vẽ trên hình ảnh biểu diễn biên trái và biên phải của làn phương tiện, phản ánh khả năng nhận dạng chính xác của thuật toán. Trong cả hai hình minh họa, hệ thống duy trì phát hiện ổn định ngay cả khi xuất hiện các phương tiện khác trong khung hình, chứng tỏ tính bền vững của thuật toán trước sự thay đổi môi trường. Ngoài ra, ảnh nhỏ ở góc trên bên phải cung cấp góc nhìn phụ, cũng hiển thị kết quả nhận dạng làn đường, cho thấy sự nhất quán giữa các góc nhìn. Đánh giá trực quan cho thấy các đường phát hiện bám sát mép làn thực tế, không bị lệch đáng kể, khẳng định thuật toán hoạt động tốt trong điều kiện lý tưởng.

4.1.2. Đánh giá hiệu suất MATLAB và Python

Để đánh giá hiệu quả của hệ thống phát hiện làn đường, thuật toán được triển khai trên hai môi trường MATLAB và Python với các tiêu chí cụ thể gồm thời gian xử lý, độ chính xác phát hiện và khả năng triển khai thực tế.

Từ kết quả đo ta có *Bảng 4.1* tổng kết kết quả về hiệu suất của 2 ngôn ngữ MATLAB và Python như sau:

Bảng 4.1: Bảng so sánh hiệu suất giữa MATLAB và Python

Tiêu chí	MATLAB	Python
Tổng số frame	**383**	
Thời gian thực thi	214.5810 s	8.15 s
Thời gian/ frame	0.5603 s	21.27 ms
FPS	1.78	47.01
Khả năng tích hợp thực tế	Phù hợp cho nghiên cứu và thử nghiệm thuật toán	Phù hợp cho ứng dụng thời gian thực

Nhận xét: Python cho thấy hiệu suất vượt trội so với MATLAB trong xử lý thời gian thực. Cụ thể, Python đạt trung bình 21,27 ms mỗi frame (≈47 FPS), đáp ứng tốt yêu cầu ứng dụng hỗ trợ lái phương tiện, trong khi MATLAB chỉ đạt 0,5603 giây mỗi frame (≈1,78 FPS), chậm hơn khoảng 24 lần. Nguyên nhân là Python tận dụng thư viện tối ưu như OpenCV, NumPy và dễ tích hợp với ROS, TensorFlow, phù hợp cho triển khai thực tế. MATLAB vẫn hữu ích cho nghiên cứu nhờ giao diện trực quan, nhưng hạn chế về tốc độ và khả năng tích hợp.

Kết luận: Python cho tốc độ xử lý nhanh hơn MATLAB trong khi độ chính xác phát hiện tương đương. Ngoài ra, Python phù hợp hơn cho triển khai thực tế, còn MATLAB chủ yếu phục vụ nghiên cứu và thử nghiệm.

4.1.3. Kết quả định lượng

Trong quá trình xử lý video, thuật toán sẽ phân tích từng khung hình để phát hiện hai vạch kẻ đường - gồm vạch trái và vạch phải. Đầu tiên, ảnh được chuyển sang thang xám và áp dụng bộ lọc Canny để phát hiện biên. Sau đó, ROI hình thang được thiết lập nhằm giới hạn khu vực tìm kiếm lane, giúp loại bỏ nhiễu từ các đối tượng ngoài đường. Từ ảnh biên trong ROI, thuật toán sử dụng Hough Transform để tìm các đoạn thẳng, sau đó phân loại thành lane trái (độ dốc âm) và lane phải (độ dốc dương). Nếu cả hai lane được phát hiện, bộ đếm frame phát hiện đúng sẽ tăng thêm một đơn vị.

Sau khi xử lý toàn bộ video, giá trị Accuracy được tính bằng công thức:

$$\text{Độ chính xác} = \frac{\text{số frame phát hiện đúng}}{\text{tổng số frame}} \times 100\% \tag{36}$$

Trong đó, "phát hiện đúng" nghĩa là thuật toán tìm được cả hai lane trong cùng một frame. Chỉ những khung hình thỏa điều kiện này mới được tính vào độ chính xác. Kết quả cuối cùng phản ánh tỷ lệ phần trăm số frame mà hệ thống nhận dạng đầy đủ hai lane so với tổng số frame, cho thấy mức độ ổn định và hiệu quả của thuật toán trong điều kiện thực tế.

Từ kết quả thu được ta có *Bảng 4.2* tổng kết độ chính xác của phương pháp xử lý ảnh truyền thống trong bài toán nhận dạng vạch kẻ làn đường ở hai trường hợp là chỉ nhận dạng một trong hai vạch kẻ đường và nhận dạng được cả hai vạch (trái và phải).

Bảng 4.2: Bảng kết quả độ chính xác của phương pháp xử lý ảnh truyền thống

Trường hợp	Độ chính xác
Nhận dạng 1 trong 2 vạch kẻ đường	96.34 %
Nhận dạng được cả 2 vạch kẻ đường	82.77 %

4.1.4. Nhận xét

Thuật toán phát hiện làn đường cho thấy hiệu quả cao trong điều kiện ánh sáng mạnh và đường rõ ràng, đảm bảo khả năng nhận dạng chính xác. Ngay cả khi xuất hiện phương tiện che khuất trong khung hình, hệ thống vẫn duy trì khả năng phát hiện ổn định, chứng tỏ tính bền vững của phương pháp. Xét về môi trường triển khai, Python là lựa chọn tối ưu cho các ứng dụng thời gian thực nhờ tốc độ xử lý nhanh và khả năng tích hợp linh hoạt với các nền tảng như ROS và TensorFlow. Trong khi đó, MATLAB lại phù hợp hơn cho nghiên cứu và thử nghiệm thuật toán nhờ giao diện trực quan và các công cụ phân tích mạnh mẽ, hỗ trợ tốt cho việc kiểm chứng và phát triển ý tưởng trong môi trường học thuật.

4.1.5. Đánh giá hệ thống

Các tiêu chí được sử dụng để đánh giá hiệu quả của hệ thống phát hiện làn đường bao gồm nhiều khía cạnh quan trọng nhằm đảm bảo tính chính xác và khả năng ứng dụng thực tế. Trước hết, độ ổn định của đường phát hiện giữa các khung hình liên tiếp được xem là yếu tố then chốt, phản ánh khả năng duy trì vị trí đường một cách mượt mà và không bị dao động lớn. Bên cạnh đó, tỉ lệ phát hiện đúng trong các điều kiện ánh sáng khác nhau, từ ánh sáng mạnh đến môi trường thiếu sáng, được sử dụng để kiểm tra tính thích ứng của thuật toán. Tốc độ xử lý, đo bằng số khung hình trên giây (FPS), là tiêu chí quan trọng để đảm bảo hệ thống có thể hoạt động gần thời gian thực. Cuối cùng, độ trễ (latency) được

đánh giá nhằm xác định khoảng thời gian từ khi nhận dữ liệu đầu vào đến khi xuất kết quả, đảm bảo đáp ứng yêu cầu của các ứng dụng hỗ trợ lái xe hoặc mô phỏng.

4.2. Kết quả nhận dạng làn đường bằng phương pháp học sâu UFLDv2

4.2.1. Kết quả trực quan

Sau khi mô hình UFLDv2 được tinh chỉnh, trọng số được sử dụng để triển khai quá trình suy luận trên các video thực tế thu từ camera hành trình. Quy trình suy luận được thực hiện theo từng khung hình, bảo đảm tốc độ xử lý thời gian thực và tính ổn định khi hệ thống hoạt động liên tục.

Đầu tiên, mô hình được nạp từ tệp trọng số (checkpoint) đã được fine-tune. Mỗi khung hình của video được đọc tuần tự và đưa qua bước tiền xử lý, bao gồm thay đổi kích thước về đúng đầu vào của mô hình, chuẩn hóa giá trị điểm ảnh và chuyển đổi sang tensor. Sau đó, khung hình được đưa qua mô hình UFLDv2 để dự đoán các điểm anchor theo hai cơ chế row-anchor và column-anchor.

Các điểm anchor đầu ra được giải mã để tạo thành tập điểm lane-left và lane-right, sau đó được nội suy thành các đường cong liên tục biểu diễn làn đường. Cuối cùng, các đường lane này được ánh xạ trở lại khung hình gốc để phục vụ trực quan hóa và truyền cho mô-đun cảnh báo phương tiện phía trước nhằm xác định vị trí tương đối giữa phương tiện và làn đường.

Hình 4.2. Minh họa kết quả suy luận làn đường trên một khung hình video thực tế

Hình minh họa cho một khung hình sau khi suy luận được thể hiện trong Hình 4.2, cho thấy mô hình có thể tái tạo hình dạng làn đường trong điều kiện thực tế và duy trì tính ổn định trong chuỗi video.

4.2.2. Kết quả định lượng

Bảng 4.3: Các chỉ số hiệu năng của mô hình UFLDv2 trên tập kiểm thử

Chỉ số	Giá trị
True Positive (TP)	980
False Positive (FP)	102
False Positive (FN)	665
Precision	0.90057
Recall	0.5957
F1-score	0.7187

$$Precision = \frac{TP}{TP + FP} \tag{37}$$

$$Recall = \frac{TP}{TP + FN} \tag{38}$$

$$F1 - score = \frac{2.Precision.Recall}{Precision + Recall} \tag{39}$$

<u>Chú thích:</u>

TP: Số mẫu thuộc lớp dương tính và được mô hình dự đoán đúng là dương tính

FP: Số mẫu thực tế thuộc lớp âm tính nhưng mô hình dự đoán nhầm thành dương tính

FN: Số mẫu thực tế thuộc lớp dương tính nhưng mô hình dự đoán nhầm thành âm tính

Precision: Tỉ lệ dự đoán dương tính mà mô hình dự đoán đúng

Recall: Tỉ lệ mẫu dương tính thực sự được mô hình phát hiện

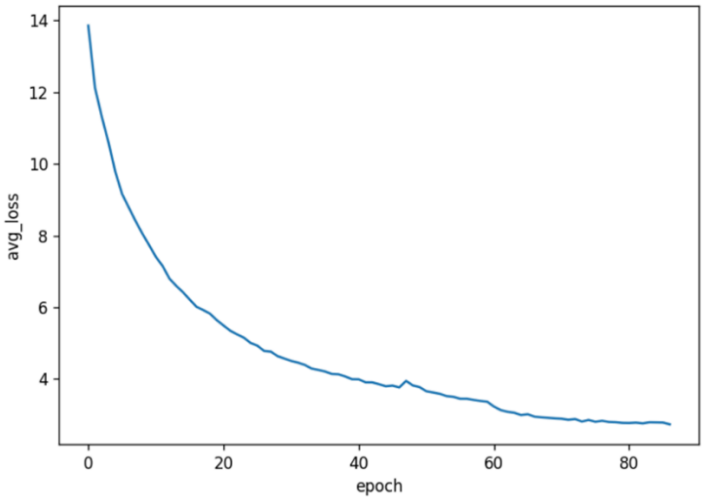
F1-score: Trung bình điều hòa giữa Precision và Recall

Bảng 4.3 trình bày các chỉ số định lượng của mô hình UFLDv2 sau khi tinh chỉnh trên tập dữ liệu 2754 ảnh làn đường thực tế. Các giá trị được tính dựa trên số lượng điểm lane (anchor points) mà mô hình dự đoán đúng hoặc bỏ sót trong tập kiểm thử.

Kết quả cho thấy mô hình đạt precision cao (≈ 90.6%), thể hiện rằng phần lớn các lane mà mô hình dự đoán đều chính xác. Đây là đặc điểm quan trọng đối với các hệ thống cảnh

báo an toàn, vốn cần hạn chế tối đa dự đoán sai (false positives) để tránh đưa ra cảnh báo không cần thiết cho người lái.

Recall đạt ≈ 59.6%, thấp hơn do mô hình có xu hướng bỏ qua các lane có chất lượng kém như lane bị mờ, đứt đoạn, bị che khuất hoặc nhiễu sáng. Đây là hiện tượng thường gặp khi fine-tune trên tập dữ liệu nhỏ và không đồng đều, đặc biệt trong điều kiện đường sá thực tế phức tạp. Tuy vậy, mô hình vẫn đảm bảo nhận dạng chính xác các lane rõ ràng - điều đủ đáp ứng yêu cầu của bài toán cảnh báo phương tiện phía trước, nơi độ tin cậy của lane phát hiện được quan trọng hơn số lượng lane phát hiện đầy đủ.

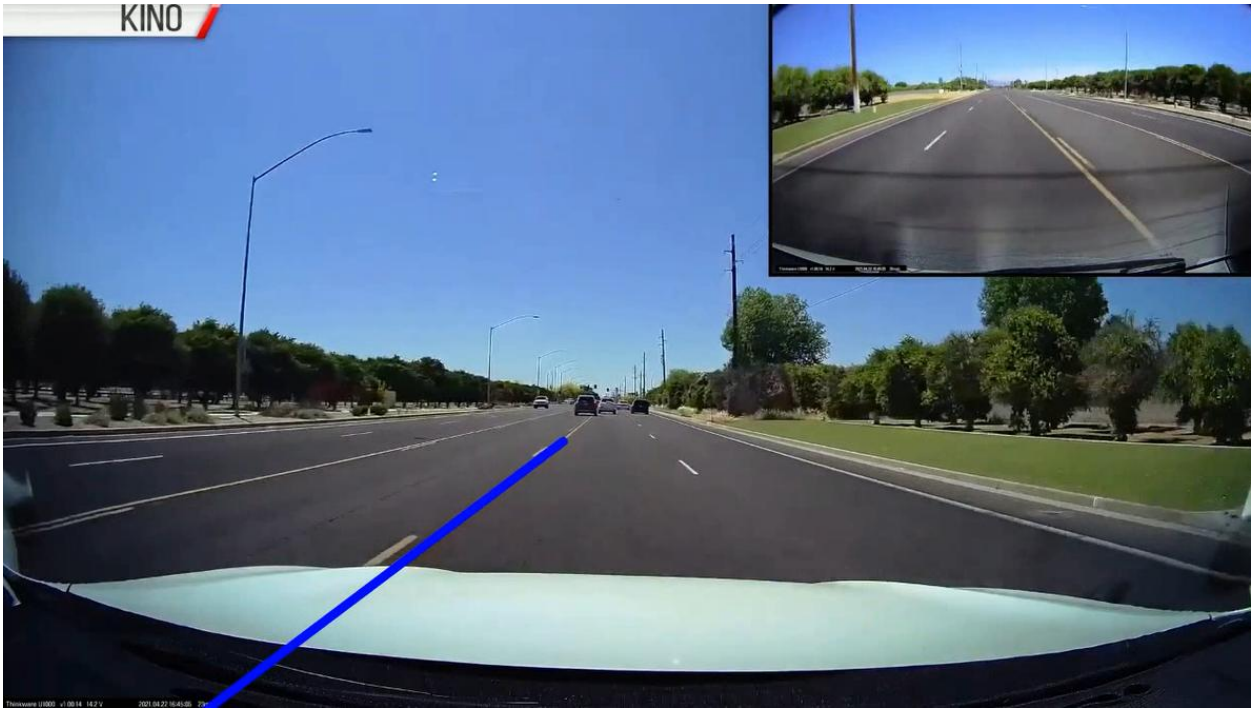


Hình 4.3. Đồ thị hàm mất mát của mô hình UFLDv2 trong quá trình fine-tune

Để minh họa quá trình tối ưu trong giai đoạn huấn luyện, *Hình 4.3* trình bày đồ thị hàm mất mát (loss) theo số epoch. Đường cong loss giảm ổn định, cho thấy mô hình đã hội tụ tốt trong quá trình fine-tune.

4.2.3. So sánh với phương pháp xử lý ảnh truyền thống

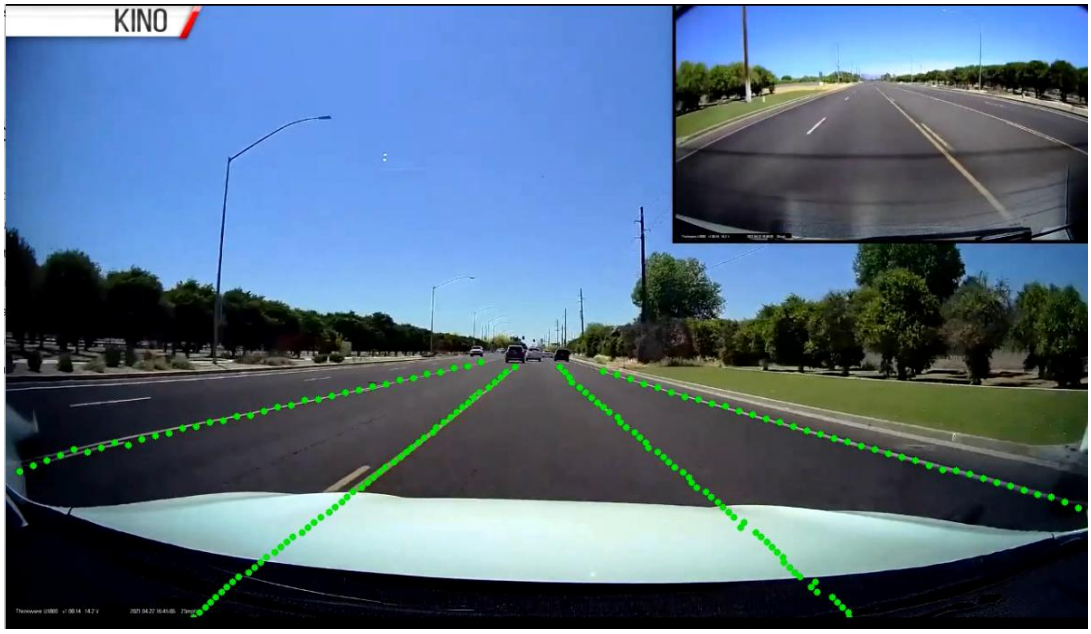
Để đánh giá hiệu quả của mô hình UFLDv2, nhóm tiến hành so sánh trực quan với phương pháp phát hiện làn đường dựa trên kỹ thuật xử lý ảnh truyền thống. Phương pháp nhận dạng làn đường bằng xử lý ảnh truyền thống được triển khai thông qua các bước lọc biên (Sobel/Canny), kết hợp với các phép biến đổi hình thái học và biến đổi Hough để suy luận các đường thẳng đại diện cho làn đường. Kết quả minh họa trong *Hình 4.4* cho thấy các thuật toán truyền thống có thể nhận dạng lane tương đối tốt trong điều kiện quan sát lý tưởng.



Hình 4.4. Kết quả phát hiện làn đường bằng phương pháp xử lý ảnh truyền thống

Tuy nhiên, trong các tình huống thực tế, phương pháp xử lý ảnh truyền thống bộc lộ nhiều hạn chế. Do mô hình hóa làn đường dưới dạng các đường thẳng liên tục, xử lý ảnh truyền thống không phản ánh đúng hình dạng cong của làn đường và rất nhạy với các yếu tố môi trường như ánh sáng mạnh, bóng cây, mặt đường loang lổ hoặc độ tương phản thấp. Điều này dẫn đến hiện tượng bỏ sót lane hoặc phát hiện sai lane, đặc biệt khi có nhiều làn hoặc khi phương tiện phía trước che khuất. Bên cạnh đó, các vạch kẻ nét đứt thường không được nhận dạng ổn định, vì các đoạn rời rạc của lane không tạo đủ điểm biên liên tục để thực hiện biến đổi Hough ghép thành một đường hoàn chỉnh. Các hạn chế trên được thể hiện rõ trong *Hình 4.4*.

Ngược lại, mô hình UFLDv2 (*Hình 4.5*) cho kết quả vượt trội với khả năng tái tạo lane-left và lane-right liên tục dưới dạng đường cong, phù hợp với hình dạng hình học thực tế của mặt đường. Mô hình vẫn giữ được độ ổn định ngay cả khi điều kiện ánh sáng không đồng đều, khi các làn bị mờ hoặc khi xuất hiện phương tiện phía trước. Điều này giúp mô-đun lane detection trở nên đáng tin cậy hơn khi đưa vào hệ thống cảnh báo phương tiện phía trước.



Hình 4.5. Kết quả phát hiện làn đường bằng mô hình UFLDv2 sau khi tinh chỉnh

Ngoài các trường hợp lý tưởng, nhóm cũng tiến hành thử nghiệm trên những đoạn đường cong thực tế tại khu vực đèo Hải Vân nhằm đánh giá khả năng xử lý trong điều kiện khó. Kết quả cho thấy phương pháp xử lý ảnh truyền thống gần như không thể duy trì lane khi góc cua lớn, mặt đường tối hoặc xuất hiện phương tiện phía trước che khuất, thể hiện rõ trong *Hình 4.6*. Các đường thẳng do phép biến đổi Hough tạo ra bị lệch đáng kể so với hình dạng lane thật, hoặc hoàn toàn không xuất hiện.

Hình 4.6. Kết quả phát hiện làn đường bằng phương pháp xử lý ảnh truyền thống trên đoạn đường cong đèo Hải Vân

Bên cạnh đó, mô hình UFLDv2 *(Hình 4.7*) vẫn duy trì được lane-left và lane-right liên tục ngay cả khi lane bị mờ hoặc có nhiều nhiễu thị giác từ môi trường xung quanh. Điều này khẳng định ưu thế của mô hình học sâu khi xử lý các tình huống lái phương tiện phức tạp mà phương pháp truyền thống không đáp ứng được.

Hình 4.7. Kết quả phát hiện làn đường bằng mô hình UFLDv2 trên đoạn đường cong đèo Hải Vân

Từ bốn hình kết quả so sánh trên, có thể thấy rằng phương pháp xử lý ảnh truyền thống chỉ phù hợp với môi trường lý tưởng, trong khi UFLDv2 thể hiện độ ổn định cao hơn và phù hợp để triển khai cho các hệ thống hỗ trợ lái phương tiện trong điều kiện thực tế.

4.2.4. Nhận xét

Kết quả thực nghiệm cho thấy mô hình UFLDv2 sau khi tinh chỉnh thể hiện hiệu năng vượt trội so với phương pháp xử lý ảnh truyền thống. Về trực quan, UFLDv2 có khả năng giữ lane ổn định trong nhiều điều kiện thực tế như ánh sáng mạnh, bóng cây, mặt đường loang lổ hoặc xuất hiện phương tiện phía trước. Các đường lane được mô hình tái tạo dưới dạng đường cong liên tục, phù hợp với hình dạng thật của mặt đường, trong khi xử lý ảnh truyền thống thường cho kết quả gián đoạn hoặc sai lệch.

Về định lượng, mô hình đạt precision cao, phản ánh tính tin cậy của các lane được phát hiện. Mặc dù recall thấp hơn do mô hình có xu hướng bỏ qua các lane mờ hoặc nhiễu, hành vi này vẫn phù hợp với mục tiêu của hệ thống cảnh báo phương tiện phía trước, nơi việc giảm dự đoán sai quan trọng hơn việc nhận dạng đầy đủ mọi lane.

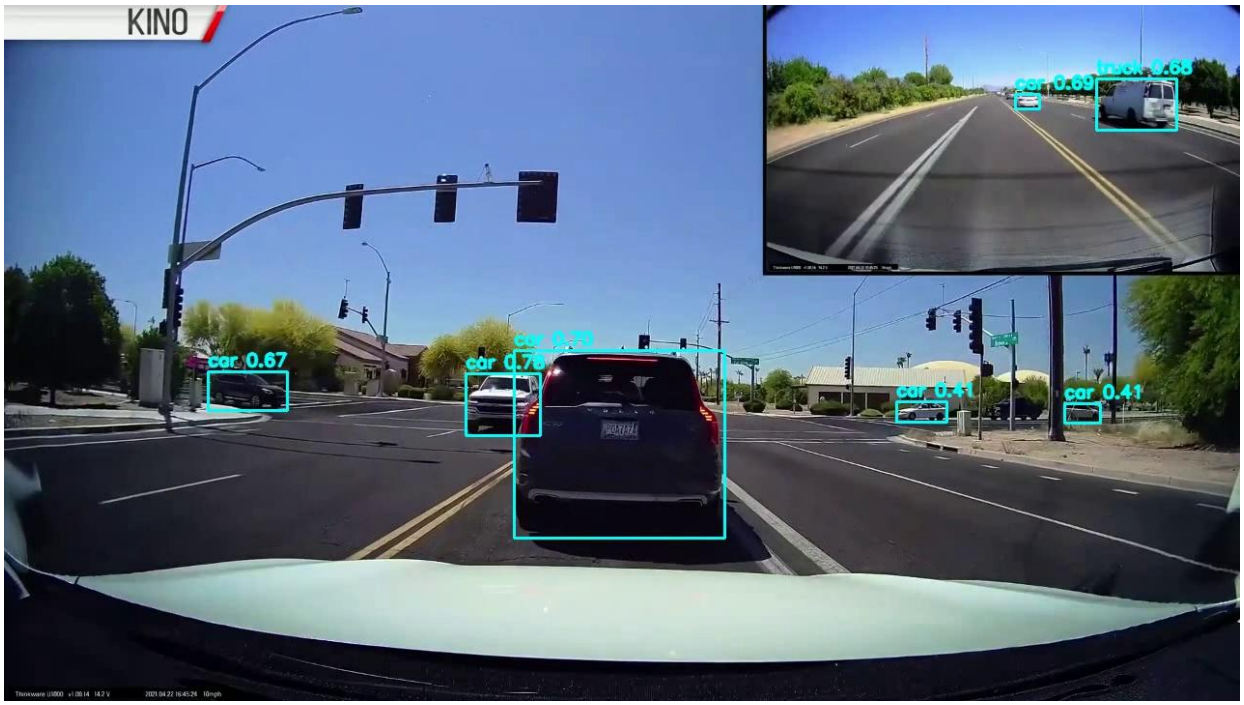
Nhìn chung, UFLDv2 đáp ứng tốt yêu cầu của bài toán phát hiện làn đường trong hệ thống cảnh báo an toàn. Phương pháp xử lý ảnh truyền thống có thể dùng như baseline minh họa nhưng không đủ ổn định để triển khai trong môi trường thực tế. Mô hình UFLDv2 sau khi fine-tune cho thấy sự phù hợp rõ rệt với dữ liệu địa phương và là lựa chọn phù hợp để tích hợp vào hệ thống giám sát giao thông dựa trên camera.

4.3. Kết quả phát hiện và cảnh báo phương tiện phía trước

4.3.1. Kết quả phát hiện phương tiện

Trong đề tài, nhóm sử dụng mô hình YOLOv8n - phiên bản nhẹ nhất của YOLOv8 - nhằm đảm bảo tốc độ xử lý cao khi chạy trên môi trường Python thời gian thực. Mặc dù kích thước mô hình nhỏ, YOLOv8n vẫn cho chất lượng phát hiện tốt đối với các phương tiện phổ biến trên video từ camera hành trình.

Quan sát trực quan cho thấy các bounding box được tạo ra khá ổn định, bám sát biên dạng phương tiện. Các phương tiện ở xa vẫn được phát hiện nhờ cơ chế PAN-FPN trong kiến trúc YOLOv8. Đồng thời, các tình huống như phương tiện bị che khuất một phần, nhiều phương tiện xuất hiện cùng lúc hoặc ánh sáng thay đổi đều được mô hình xử lý tương đối tốt. Đây là cơ sở quan trọng để hệ thống tiếp tục thực hiện bước chọn phương tiện phía trước và ước lượng khoảng cách.



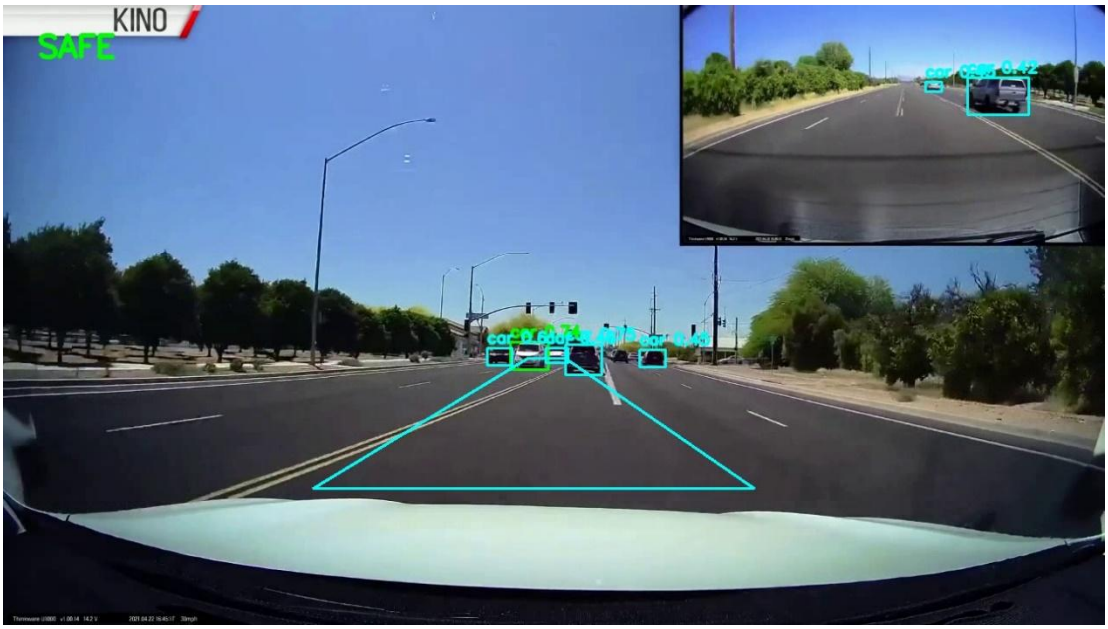
Hình 4.8. Hình ảnh minh họa kết quả nhận dạng phương tiện

Theo bảng benchmark chính thức của Ultralytics trên bộ dữ liệu COCO val2017 với độ phân giải đầu vào 640 px, mô hình YOLOv8n đạt mAP@0.5:0.95 ≈ 37.3%. Tốc độ suy luận đạt khoảng 0.99 ms/frame (khoảng >1000 FPS) trên thiết bị GPU A100 qua TensorRT **[15]**.

Ở mức ứng dụng thực tế của đề tài, YOLOv8n chạy trên Google Colab với GPU T4 đạt tốc độ trung bình 71.46 FPS ở độ phân giải đầu vào 960 pixel. Đây là tốc độ suy luận thuần của hệ thống tổng thể phản ánh khả năng phát hiện phương tiện rất nhanh và hoàn toàn đáp ứng yêu cầu thời gian thực của hệ thống.

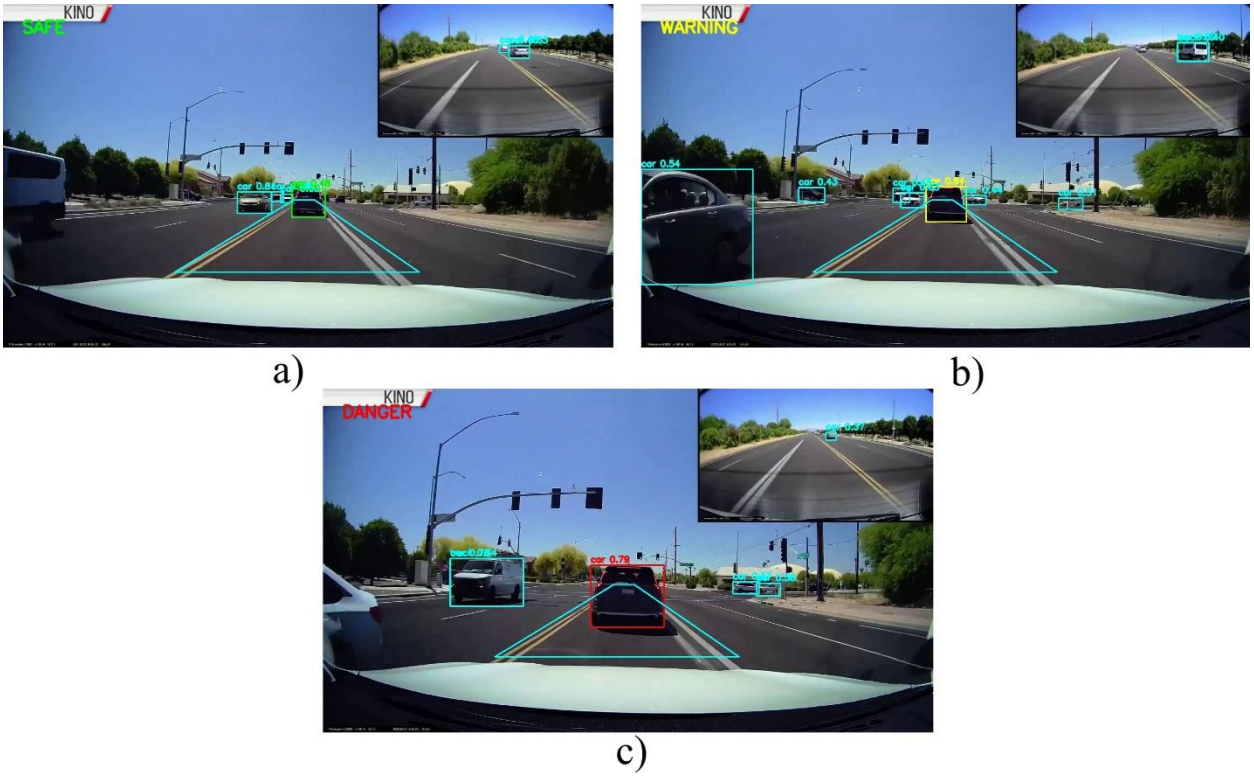
4.3.2. Kết quả cảnh báo phương tiện phía trước

Để đánh giá khả năng lựa chọn đúng phương tiện phía trước, nhóm đã tách 100 khung hình từ video thử nghiệm và tiến hành gán nhãn thủ công phương tiện thực sự nằm phía trước xe. Kết quả cho thấy hệ thống chọn chưa đúng 8 khung hình, với độ chính xác đạt 92%.



Hình 4.9. Hình ảnh chọn sai phương tiện phía trước

Quan sát trực quan trên video cho thấy, khi xe phía trước di chuyển lại gần, mức cảnh báo được cập nhật theo hướng tăng dần từ “Safe” sang “Warning” và “Danger”, phản ánh tương đối chính xác mức độ rủi ro.



Hình 4.10. Hình ảnh các mức cảnh báo (a. An toàn, b. Cảnh báo, c. Nguy hiểm)

4.3.3. Nhận xét

Qua quá trình thử nghiệm, mô-đun cảnh báo phương tiện phía trước thể hiện độ ổn định cao khi xác định đúng phương tiện nằm trong vùng phía trước xe và đưa ra cảnh báo phù hợp theo sự thay đổi khoảng cách tương đối. Tuy vậy, bên cạnh những khung hình đạt kết quả chính xác, hệ thống vẫn xuất hiện một số trường hợp sai lệch cần được phân tích để hoàn thiện mô hình trong tương lai.

Một phần các lỗi xuất phát từ mô hình YOLOv8n. Trong những khung hình thiếu ánh sáng, bị chói nắng hoặc có xe ở khoảng cách xa, YOLO đôi khi không nhận dạng được phương tiện trong vài khung hình. Việc bỏ sót này dẫn đến tình huống hệ thống không chọn được xe phía trước dù thực tế có phương tiện đang di chuyển cùng làn. Một nguyên nhân khác liên quan đến hình dạng bounding box: khi YOLO phát hiện nhưng đường bao không khớp hoàn toàn với kích thước vật thể (bbox lệch hoặc bị cắt) khiến xe phía trước bị loại nhầm.

Bên cạnh đó, một số sai lệch xuất hiện khi xe gắn camera đang chuyển làn hoặc đi vào khúc cua. Trong các khung hình này, ROI hình thang vốn được thiết kế cho chuyển động thẳng không còn trùng khớp với làn đường thực tế. Khi ROI bị lệch sang trái hoặc phải so

với hướng di chuyển, xe phía trước có thể nằm ngoài vùng ROI tạm thời, dẫn đến việc hệ thống chọn nhầm xe ở làn bên hoặc không chọn được xe nào.

Mặc dù tồn tại những trường hợp sai lệch kể trên, kết quả tổng thể cho thấy mô-đun FCDW vẫn đạt mức hiệu quả tương đối tốt với các quãng đường thẳng. Nhờ sự kết hợp giữa phát hiện phương tiện bằng YOLOv8n và cơ chế lọc dựa vào ROI, hệ thống vẫn duy trì được sự ổn định, hoạt động gần thời gian thực và phù hợp để triển khai trong các bài toán hỗ trợ lái chi phí thấp.

4.4. Đánh giá tốc độ xử lý của các mô-đun trong hệ thống

Để đánh giá khả năng hoạt động thời gian thực của hệ thống, nhóm tiến hành đo tốc độ xử lý của từng mô-đun thành phần cũng như toàn bộ pipeline. Tất cả các phép đo được thực hiện trên cùng một video đầu vào 30 FPS với tổng cộng 383 khung hình.

Các thử nghiệm được triển khai trên nền tảng Google Colab, sử dụng GPU T4 nhằm đảm bảo mô hình học sâu (YOLOv8 và UFLDv2) đạt tốc độ suy luận tối ưu. Điều này giúp kết quả đánh giá phản ánh đúng năng lực thực tế của hệ thống khi triển khai trên môi trường có gia tốc phần cứng.

4.4.1 FPS của mô-đun nhận dạng làn đường bằng xử lý ảnh truyền thống.

Phương pháp xử lý ảnh truyền thống đạt tốc độ cao (~37 FPS), vượt ngưỡng thời gian thực. Điều này phù hợp với đặc điểm của pipeline xử lý ảnh truyền thống: thuật toán đơn giản, không yêu cầu mô hình học sâu và chạy hoàn toàn bằng các phép toán nền tảng của OpenCV.

4.4.2 FPS của mô-đun nhận dạng làn đường bằng UFLDv2

UFLDv2 có tốc độ suy luận chậm hơn nhiều so với phương pháp truyền thống, tuy nhiên vẫn đạt mức chấp nhận được cho ứng dụng hỗ trợ lái tốc độ vừa (~13 FPS). Sự giảm tốc độ đến từ kiến trúc deep learning có backbone ResNet-34 và cơ chế dự đoán nhiều anchor.

4.4.3 FPS của mô-đun phát hiện và cảnh báo phương tiện phía trước (YOLOv8)

YOLOv8 cho tốc độ xử lý tốt (~26.6 FPS), gần đạt thời gian thực hoàn chỉnh. Điều này phù hợp vì YOLOv8 được thiết kế tối ưu cho tốc độ, đặc biệt trên GPU. Đây là nền tảng quan trọng để xây dựng mô-đun cảnh báo phương tiện phía trước.

4.4.4 FPS của toàn bộ pipeline

Để đánh giá khả năng hoạt động thời gian thực của hệ thống khi tích hợp đầy đủ các mô-đun, nhóm tiến hành đo tốc độ xử lý tổng thể của hai cấu hình pipeline:

- Pipeline 1: Phát hiện làn đường bằng xử lý ảnh truyền thống + YOLOv8

- Pipeline 2: Phát hiện làn đường bằng UFLDv2 + YOLOv8

Kết quả được tóm tắt trong *Bảng 4.4* sau:

Bảng 4.4. Tốc độ xử lý của toàn bộ pipeline

Cấu hình	Tổng thời gian (s)	FPS trung bình
Pipeline 1	~ 14.5	~26.40
Pipeline 2	~ 33.9	~11.30

4.4.5 Nhận xét chung

Pipeline 1 đạt tốc độ xử lý cao (~26.4 FPS), gần tương đương thời gian thực, do mô-đun lane truyền thống có chi phí tính toán thấp và không tạo thêm độ trễ đáng kể ngoài mô-đun YOLOv8.

Pipeline 2 có tốc độ thấp hơn (~11.3 FPS) khi tích hợp UFLDv2. Nguyên nhân chính là mô hình UFLDv2 có chi phí tính toán lớn hơn đáng kể so với xử lý ảnh truyền thống.

Tuy nhiên, Pipeline 2 lại cho kết quả nhận dạng làn đường ổn định và chính xác hơn, đặc biệt trong các tình huống vạch kẻ đường bị mờ hoặc biến dạng. Tùy mục tiêu sử dụng (tốc độ hay độ chính xác), có thể lựa chọn pipeline phù hợp hoặc tiếp tục tối ưu hóa mô hình UFLDv2 để cải thiện FPS.

4.5. Mô tả các file kết quả và dữ liệu lưu trữ

Trong quá trình thực nghiệm, hệ thống sinh ra nhiều loại dữ liệu và kết quả trung gian nhằm phục vụ đánh giá và phân tích. Các dữ liệu này được lưu trữ dưới dạng các tệp riêng biệt với mục đích cụ thể như sau:

- **Video kết quả đầu ra** *(output_lane_yolo.mp4):*

  Lưu các khung hình video sau khi được xử lý, trong đó hiển thị làn đường nhận dạng, bounding box phương tiện và trạng thái cảnh báo. Các video này được sử dụng để đánh giá trực quan hiệu quả của hệ thống.

- **File kết quả đánh giá định lượng** *(fps_results.csv):*

  Lưu các thông số đánh giá như độ chính xác, FPS, thời gian xử lý của từng mô-đun. Các file này phục vụ việc tổng hợp số liệu và so sánh giữa các phương pháp.

- **File dữ liệu huấn luyện và nhãn** *(image_0123.jpg, image_0123.lines):*

  Bao gồm các ảnh đầu vào (.jpg) và file nhãn tương ứng (.lines) dùng cho quá trình fine-tune mô hình UFLDv2. Dữ liệu này được sử dụng lại khi huấn luyện hoặc mở rộng hệ thống trong tương lai.

KẾT LUẬN

Trong đề tài này, nhóm đã xây dựng và đánh giá một hệ thống hỗ trợ lái xe dựa trên video camera hành trình, tập trung vào hai chức năng chính là nhận dạng làn đường và cảnh báo phương tiện phía trước. Hệ thống được triển khai theo hướng kết hợp giữa các kỹ thuật xử lý ảnh truyền thống và các mô hình học sâu nhằm khai thác ưu điểm của từng phương pháp.

Kết quả thực nghiệm cho thấy hướng tiếp cận này mang lại hiệu quả rõ rệt. Pipeline sử dụng mô hình học sâu UFLDv2 cho khả năng nhận dạng làn đường ổn định và chính xác trong nhiều điều kiện thực tế, trong khi pipeline xử lý ảnh truyền thống đạt tốc độ xử lý cao và đáp ứng tốt yêu cầu thời gian thực. Đối với bài toán phát hiện và cảnh báo phương tiện phía trước, mô hình YOLOv8 cho kết quả phát hiện nhanh và đủ độ tin cậy để tích hợp vào hệ thống hỗ trợ lái xe dựa trên camera.

Việc so sánh các pipeline cho thấy mỗi phương pháp đều có ưu và nhược điểm riêng, từ đó mở ra khả năng lựa chọn hoặc kết hợp tùy theo nhu cầu thực tế, chẳng hạn như ưu tiên tốc độ xử lý hoặc ưu tiên độ chính xác và tính ổn định của hệ thống.

Bên cạnh những kết quả đạt được, đề tài vẫn còn một số hạn chế như tốc độ suy luận của mô hình UFLDv2 chưa đạt mức thời gian thực hoàn toàn, phương pháp ước lượng khoảng cách dựa trên camera còn mang tính tương đối, và hệ thống chưa xử lý tốt các tình huống đặc biệt như mưa lớn, chói sáng hoặc đường cong gắt.

Trong tương lai, hệ thống có thể được mở rộng theo các hướng như tối ưu hóa hoặc sử dụng backbone nhẹ hơn cho UFLDv2, kết hợp kỹ thuật tracking để ổn định cảnh báo phương tiện phía trước, và áp dụng các phương pháp tăng cường chất lượng ảnh nhằm cải thiện khả năng nhận dạng làn đường trong điều kiện khó.

Nhìn chung, kết quả đạt được cho thấy hệ thống đáp ứng tốt mục tiêu đề ra và có tiềm năng ứng dụng trong các hệ thống hỗ trợ lái xe chi phí thấp dựa trên camera hành trình, đồng thời tạo nền tảng cho các nghiên cứu tiếp theo trong lĩnh vực giao thông thông minh.

TÀI LIỆU THAM KHẢO

[1] My Accident Answers, "Human Errors: What Percentage of Car Accidents Occur?," *My Accident Answers*, [Online]. Available: https://myaccidentanswers.com/statistics/what-percentage-of-car-accidents-is-caused-by-human-errors/. [Accessed: 2025].

[2] National Safety Council, "Fatigued Driving," *National Safety Council*, [Online]. Available: https://www.nsc.org/road/safety-topics/fatigued-driver. [Accessed: 2025].

[3] Colorado Department of Transportation, "AEB Tech Has Led to Fewer Rear-End Crashes, Study Finds," *Shift Into Safe News*, Mar. 2025. [Online]. Available: https://www.codot.gov/safety/shift-into-safe-news/2025/march/aeb-tech-has-led-to-fewer-rear-end-crashes-study-automotive-dive. [Accessed: 2025].

[4] U.S. Department of Transportation, "ITS Knowledge Resources – 2017-B01175," *ITS Knowledge Resources*, 2017. [Online]. Available: https://www.itskrs.its.dot.gov/2017-b01175. [Accessed: 2025].

[5] VIBLO, "Part 2 – Edge Detection với OpenCV," *VIBLO*, [Online]. Available: https://viblo.asia/p/part-2-edge-detection-voi-opencv-eW65Gv4YlDO. [Accessed: 2025].

[6] OpenCV, "Hough Line Transform," *OpenCV Documentation*, ver. 3.4. [Online]. Available: https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html. [Accessed: 2025].

[7] Viet Anh, "Hough Transform – Phát hiện đường thẳng," *vietanh.dev*, Oct. 24, 2019. [Online]. Available: https://www.vietanh.dev/blog/2019-10-24-hough-transform-phat-hien-duong-thang. [Accessed: 2025].

[8] L. Qin et al., "Ultra Fast Lane Detection v2," arXiv preprint arXiv:2206.07389, 2022.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.

[11] T.-Y. Lin, M. Maire, S. Belongie, et al., "Microsoft COCO: Common Objects in Context,"

in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 740–755.

[12] M. Yaseen, “What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector,” arXiv:2408.15857, 2024.

[13] CVAT.ai, “CVAT: Computer Vision Annotation Tool,” 2023. [Online]. Available: https://cvat.ai

[14] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," in Proc. COMPSTAT, 2010, pp. 177-186.

[15] Ultralytics, “YOLOv8 Models,” 2023. Available: https://docs.ultralytics.com/models/yolov8/

[16] J. He, Z. Zhang, and J. Wang, “Monocular Vehicle Distance Measurement Based on Pinhole Camera Model,” Sensors, vol. 18, no. 4, 2018.

PHỤ LỤC

A. Danh mục các từ viết tắt

Viết tắt	Tiếng Anh	Nghĩa tiếng Việt
ADAS	Advanced Driver Assistance Systems	Hệ thống hỗ trợ lái xe nâng cao
ROI	Region of Interest	Vùng quan tâm
FCDW	Front Car Distance Warning	Cảnh báo khoảng cách phương tiện phía trước
FPS	Frames Per Second	Số khung hình xử lý trên giây
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
UFLD	Ultra-Fast Lane Detection	Mô hình phát hiện làn đường tốc độ cao
YOLO	You Only Look Once	Mô hình nhận dạng đối tượng thời gian thực
COCO	Common Objects in Context	Bộ dữ liệu COCO
GPU	Graphics Processing Unit	Bộ xử lý đồ họa
SGD	Stochastic Gradient Descent	Thuật toán tối ưu SGD
BCE	Binary Cross-Entropy	Hàm mất mát phân lớp nhị phân
NMS	Non-Maximum Suppression	Loại bỏ cực đại không cần thiết
CVAT	Computer Vision Annotation Tool	Công cụ gán nhãn dữ liệu hình ảnh
TTA	Test-Time Augmentation	Tăng cường dữ liệu khi suy luận
DL	Deep Learning	Học sâu
GT	Ground Truth	Dữ liệu nhãn thực tế

B. Danh mục ký hiệu

Ký hiệu	Ý nghĩa
ρ	Khoảng cách vuông góc từ gốc tọa độ đến đường thẳng (trong Hough Transform)
θ	Góc nghiêng của đường thẳng so với trục hoành
G	Gradient tổng hợp
$\mathbf{G_x, G_y}$	Gradient theo trục x và y
σ	Độ lệch chuẩn của bộ lọc Gaussian
(xmin, ymin, xmax, ymax)	Tọa độ hộp giới hạn YOLO
$\mathbf{H_{bbox}}$	Chiều cao bounding box
D	Khoảng cách tương đối ước lượng đến phương tiện phía trước
$\mathbf{L_{cls}}$	Thành phần mất mát phân lớp
$\mathbf{L_{reg}}$	Thành phần mất mát hồi quy
LR	Tốc độ học