JOURNAL OF SCIENCE
OF LAC HONG UNIVERSITY
ISSN: 2525 - 2186
Tạp chí Khoa học Lạc Hồng, **2025**, 24, 148-154

ỨNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH VÀ MÔ HÌNH HỌC SÂU PHÁT TRIỂN XE TỰ LÁI

Cao Thế Vinh, Nguyễn Thanh Đệ, Đặng Nguyễn Trung Nguyên, Trương Duy Thanh Nhàn, Bùi Như Việt, Lê Ngọc Khánh, Lê Tiến Lộc*

Trường Đại học Lạc Hồng, số 10, Huỳnh Văn Nghệ, phường Trấn Biên, Đồng Nai, Việt Nam

*Tác giả liên hệ: *tienloc@lhu.edu.vn*

THÔNG TIN BÀI BÁO

Ngày nhận: 11/03/2025

Ngày hoàn thiện: 22/04/2025

Ngày chấp nhận: 23/04/2025

Ngày đăng: 15/09/2025

TỪ KHÓA

Xe tự hành;
Phát hiện làn đường;
Phát hiện đối tượng;
Robot Operating System (ROS);
Exponential Moving Average (EMA).

TÓM TẮT

Trong bài báo này, chúng tôi trình phương pháp và kế hoạch phát triển xe tự hành tỉ lệ 1/10 trong thành phố thu nhỏ bao gồm việc thực hiện mô phỏng, xây dựng kế hoạch đường đi và thu thập dữ liệu làn đường và phát hiện tín hiệu giao thông sử dụng xử lý ảnh và các kĩ thuật học sâu. Robot Operating System (ROS) là một nền tảng cơ bản trong việc phát triển các hệ thống robot và xe tự lái, ROS sử dụng các phần mềm riêng lẻ giao tiếp với nhau thông qua các chủ đề và hành động. Phần mềm mô phỏng Gazebo trong ROS được sử dụng để xây dựng, thiết lập bản đồ, xe và các đối tượng để mô phỏng như trong môi trường thực tế. Làn đường được phát hiện bằng cách sử dụng xử lí ảnh, phát hiện cạnh, sử dụng thuật toán hough tranform phát hiện làn đường từ đó tính toán góc lái đưa ra dự đoán và điều hướng xe, đối tượng và các tín hiệu giao thông được tiến hành thu thập dữ liệu, gán nhãn và huấn luyện dựa trên mô hình YOLOV5, từ các dữ liệu được phát hiện tiến hành gửi tín hiệu điều khiển thông qua ROS và giao tiếp Serial để gửi xuống board mạch chấp hành điều khiển động cơ và servo điều khiển.

IMAGE PROCESSING TECHNOLOGY AND DEEP LEARNING MODELS FOR DEVELOPING SELF-DRIVING CARS

Cao The Vinh, Nguyen Thanh De, Dang Nguyen Trung Nguyen, Truong Duy Thanh Nhan, Bui Nhu Viet, Le Ngoc Khanh, Le Tien Loc*

Lac Hong University, No. 10 Huynh Van Nghe, Tran Bien, Dong Nai Province, Vietnam

*Corresponding Author: *tienloc@lhu.edu.vn*

ARTICLE INFO

Received: Mar 11st, 2025

Revised: Apr 22nd, 2025

Accepted: Apr 23rd, 2025

Published: Sep 15th, 2025

KEYWORDS

Self-driving car.
Lane detection.
Object detection.
Robot Operating System (ROS).
Exponential Moving Average (EMA).

ABSTRACT

This paper presents the methodology and development plan for a 1/10-scale self-driving car operating in a miniature city. This includes simulation implementation, path planning, and data collection for lane detection and traffic signal recognition using image processing and deep learning techniques. The Robot Operating System (ROS) is the fundamental platform for developing robotic and autonomous vehicle systems. ROS facilitates communication between individual software components through topics and actions. The Gazebo simulation software in ROS is used to build and configure maps, vehicles, and objects to create simulations replicating real-world environments. Lane detection is performed using image processing, edge detection, and the Hough Transform algorithm to identify lanes, calculate the steering angle, and p2025rovide predictions for vehicle navigation. Object and traffic signal data are collected, labeled, and trained using the YOLOv5 model. The detected data are then used to generate control signals, which are transmitted via ROS and serial communication to a control board for motor and servo actuation.

Doi:
Avaiable online at: https://js.lhu.edu.vn/index.php/lachong

1. GIỚI THIỆU

Cuộc thi Bosch Future Mobility Challenge được tổ chức hằng năm bởi công ty Bosch với mục đích thúc đẩy sự nghiên cứu và phát triển công nghệ xe tự lái trong công đồng sinh viên các nước trên thế giới. Tại đây sinh viên các trường đại học tập trung nghiên cứu công nghệ như xử lý ảnh, mô hình học sâu trong việc nhận diện làn đường, tín hiệu giao thông và các đặc trưng được phát hiện. Ngoài ra, sinh viên áp dụng những kiến thức đổi mới và thực tiễn trong việc xây dựng kế hoạch và lập đường đi tự động giúp định vị trí và điều hướng xe dựa trên thông tin như GPS, lidar. Điều này thúc đẩy phong trào nghiên cứu và phát triển công nghệ xe tự lái trong nước và trên thế giới, bài báo này nhóm tập trung vào nghiên cứu về công nghệ xử lý ảnh và các mô hình học sâu trong việc nhận diện làn đường, tín hiệu giao thông, đối tượng có thể xảy ra, mô phỏng xe di chuyển bản đồ thông minh thu nhỏ và phát triển xe thực tế với tỉ lệ 1/10.

Hiện nay thuật toán Hough Transform được ứng dụng khá nhiều trong lĩnh vực nhận diện làn đường cho xe tự lái và hoạt động khá hiệu quả. Yadi Li, Liguo Chen [1] và các cộng sự thực hiện nhận diện làn đường bằng việc chuyển đổi không gian màu sang ảnh xám, phát hiện cạnh bằng thuật toán canny và ứng dụng Hough Transform phát hiện làn đường. Bài báo có cách giải thích cụ thể về việc xử lý ảnh và nhận diện làn đường, tuy nhiên việc chuyển sang ảnh xám sẽ gây ra nhiễu khi phát hiện các vật thể hay các chi tiết gần giống với làn đường, và bài báo chỉ dừng lại ở việc nhận diện làn đường mà chưa ứng dụng vào thực tế.

Trước đó Seonyoung Lee [2] và các cộng sự đã thực hiện phát hiện làn đường dựa trên phép biến đổi Hough Tranform và dựa vào đó xây dựng hệ thống cảnh báo lệch làn (LDWS), và họ cũng chỉ ra rằng phép biến đổi Hough cũng đòi hỏi lượng tính toán lớn và đưa ra đề xuất khối biến đổi Hough bao gồm bộ đệm 4x4 bit, và khối biến đổi Hough xử lý các khối pixel 4x4 liền kề cùng lúc giúp giảm độ nhiễu, loại bỏ bớt các phép toán Cos và Sin làm cho phép toán xử lý nhanh hơn. Kiến trúc mạch đề xuất của họ có thể giảm 50% cho số lượng bộ nhân so với việc tính toán Hough Transform thông thường.

Để làm giảm bớt đi độ nhiễu và tăng khả năng nhận diện chính xác làn đường. Gurjashan SinghPannu [3] đã đưa ra giải pháp phân vùng quan tâm làn đường và phép biến đổi Hough giúp cải thiện độ chính xác bằng cách tạo một lớp ảnh phủ để che đi những vùng không có khả năng phát hiện làn đường. Vùng nhận diện được làn đường có xu hướng hình thang vì làn đường càng nhìn càng xa thì càng nhỏ lại và tiếp điểm nhau ở cuối ảnh. Nhược điểm của bài nghiên cứu còn hạn chế vì sử dụng raspberry pi B+ trong việc nhận diện làn đường, như trong bài báo [2] có đề cập đến việc phép biến đổi hough cần lượng tính toán cao vì vậy việc sử dụng phần cứng Raspberry Pi B+ là không đủ để áp dụng vào thực tế.

ROS là một hệ thống phần mềm giúp quản lý và giao tiếp các phần mềm riêng lẻ với nhau, ngoài ra ROS còn hỗ trợ mô phỏng các ứng dụng bằng phần mềm Gazebo và hệ thống giám sát bằng phần mềm Rviz [4]. Trong [5], các tác giả đã thực hiện việc tính toán góc lái dựa vào phép biến đổi Hough, mô phỏng di chuyển xe tự hành dựa trên phần mềm mô phỏng Gazebo và chuẩn hoá góc lái dựa trên fuzzy logic, kết quả cho ra hiệu suất tốt trong môi trường mô phỏng, hạn chế vẫn chưa áp dụng vào trong cấu trúc phần cứng thực tế.

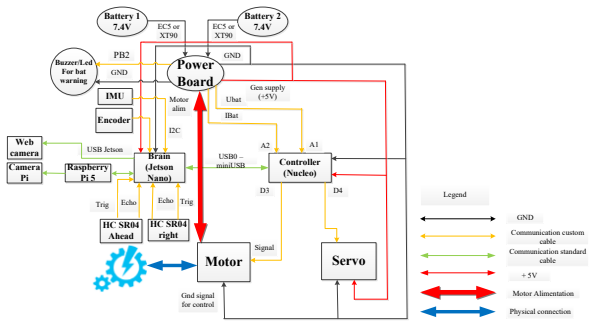
Dựa trên những tính ứng dụng cao của thuật toán Hough Transform. Nhóm đã tiến hành tổng hợp, tính toán và đưa ra giải pháp tối ưu thuật toán để có khả năng nhận diện chính xác làn đường và trả về góc lái ổn định. Ngoài ra chúng tôi còn tiến hành nâng cao khả năng nhận thức của xe bằng việc sử dụng mô hình học sâu YOLOV5 để huấn luyện tín hiệu giao thông, các đối tượng trên đường đi kết hợp cảm biến để nâng cao độ an toàn của xe. Hình 1 là mô hình xe tự hành tỉ lệ 1/10 mà nhóm chúng tôi đã thiết kế và nghiên cứu phát triển.

Hình 1. *Xe tự hành tỉ lệ 1/10.*

2. TỔNG QUAN HỆ THỐNG XE TỰ HÀNH MINI

2.1 Cấu trúc phần cứng

Hình 2 là cấu trúc sơ đồ phần cứng của xe tự hành. Bao gồm: bộ điều khiển trung tâm, bộ điều khiển chấp hành, bộ phận tín hiệu đầu vào và bộ chấp hành.



Hình 2. *Sơ đồ cấu trúc phần cứng.*

2.1.1 Bộ điều khiển trung tâm (Jetson Nano)

Chúng tôi sử dụng Jetson Nano để làm trung tâm điều khiển để xử lý tất cả các tín hiệu của bộ phận tín hiệu đầu vào. Nhờ đó nó có thể xử lý và đưa ra những những phán đoán tình huống chính xác nhất để cho bộ điều khiển chấp hành điều khiển theo lệnh của bộ não của xe.

2.1.2 Bộ điều khiển chấp hành (Nucleo F401FE)

Nucleo F401FE là bộ phận nhận tín hiệu đã được xử lý để đưa ra lệnh điều khiển cho Servo nhận tín hiệu góc lái giúp xe có thể di chuyển ổn định trên đường, và điều khiển động cơ giúp xe có thể di chuyển.

2.1.3 Bộ phận tín hiệu đầu vào

a. Raspberry Pi 5

Rapsberry được sử dụng như là một bộ não nhỏ để thu thập các hình ảnh, dữ liệu từ các vật thể như con người, biển báo, xe cộ, vật cản,… truyền dữ liệu đã được xử lý đó về cho bộ điều khiển trung tâm thông qua Universal Asynchronous Receiver Transmitter (UART) qua đó bộ điều khiển trung tâm sẽ dựa vào những tín hiệu đó để đưa ra những phán đoán tình huống chính xác nhất.

b. Camera

Chúng tôi sử dụng Webcam là Camera Pi V2. Camera Pi có nhiệm vụ thu thập dữ liệu về các vật đối tượng và các tín hiệu giao thông, Webcam có nhiệm vụ thu thập dữ liệu làn đường và gửi đến bộ điều khiển trung tâm.

c. HC-SR04

HC-SR04 là cảm biến siêu âm được sử dụng cho mục đích phát hiện những đối tượng bất ngờ trong tình huống không thể xử lý kịp thời, và dùng để kiểm tra vùng đỗ xe trống [5].

2.1.4 Bộ chấp hành

Motor và servo

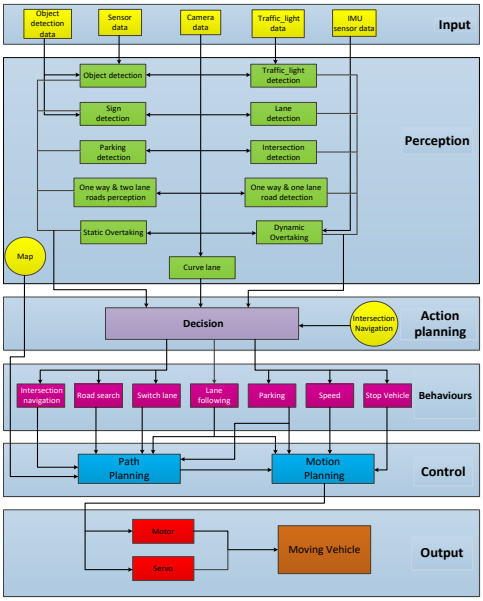
Motor dùng để giúp xe di chuyển trên các đoạn đường. Servo dùng để đánh lái và điều hướng di chuyển của xe qua các cung đường và giao lộ.

2.2 Cấu trúc phần mềm

Hệ thống điều khiển xe tự hành hoạt động bắt đầu từ việc thu thập dữ liệu tại tầng đầu vào (Input). Dữ liệu từ camera, cảm biến siêu âm, đèn giao thông và cảm biến Inertial Measurement Unit (IMU) được thu thập và cung cấp thông tin về môi trường xung quanh. Tiếp theo, tại tầng nhận diện (Perception), dữ liệu này được phân tích về nhận diện vật thể, đèn giao thông, biển báo, làn đường và giao lộ,… Ngoài ra, hệ thống còn phân tích loại đường (một chiều hay hai chiều) và xử lý các tình huống như vượt xe tĩnh hay xe động, từ đó tạo ra một mô hình tổng quát về môi trường.

Qua đó, chúng được dựa vào phần việc xử lý giao lộ (intersection) để định hình đường đi của xe. Khi xe đi tới các giao lộ, camera sẽ nhận diện các đối tượng, biển báo và giao lộ để bộ trung tâm xử lý rồi đưa ra các lệnh thực hiện cho xe hành động.

Cuối cùng, các lệnh điều khiển được gửi đến phần cứng tại tầng đầu ra (Output), nơi động cơ và servo thực hiện các hành động cần thiết. Toàn bộ quy trình hoạt động trong vòng lặp liên tục, đảm bảo xe tự hành có thể phản ứng với các thay đổi của môi trường và vận hành một cách an toàn, hiệu quả cho đến khi đạt được mục tiêu. Hình 3 dưới đây thể hiện một cấu trúc liên kết phần mềm.

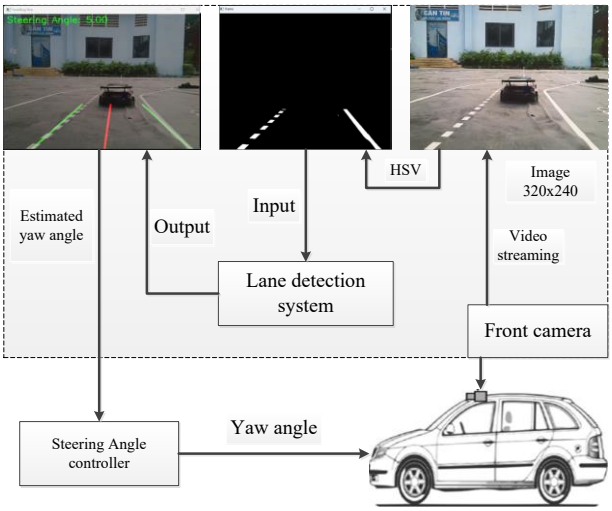


Hình 3. *Sơ đồ phần mềm.*

3. PHƯƠNG PHÁP NGHIÊN CỨU VÀ PHÁT TRIỂN

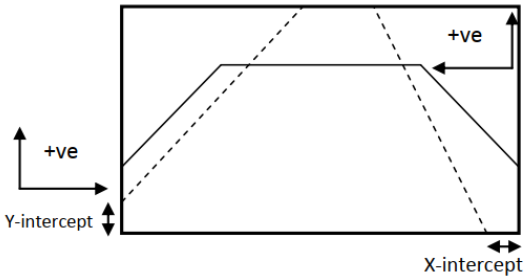
3.1 Phát hiện và theo dõi làn đường

Chúng tôi áp dụng phương pháp xử lý ảnh để theo dõi và phát hiện làn đường [7]. Khi đó, camera của Jetson xử lý những hình ảnh làn đường. Sau đó, những hình ảnh được thu thập sẽ được chuyển đổi màu từ RGB sang HSV để giúp dễ dàng tách biệt màu sắc (Hue) khỏi độ sáng (Value), giảm ảnh hưởng của ánh sáng môi trường và tăng độ chính xác khi phát hiện vạch đường. HSV giúp xử lý nhanh và hiệu quả nhờ khả năng nhận diện màu sắc ổn định hơn. Hình 4 dưới đây là cách mà bộ điều khiển trung tâm xử lý ảnh.



Hình 4. *Cấu trúc xử lý ảnh làn đường.*

Sau đó, dữ liệu hình ảnh sẽ được phân vùng quan tâm (Region of Interest – ROI). Vùng quan tâm trong xử lý ảnh từ camera trên ô tô được xác định từ dưới cùng của hình ảnh và mở rộng lên trên, thường không vượt qua một nửa chiều cao của ảnh. Điều này sẽ giúp loại bỏ khu vực không cần thiết như bầu trời, giảm tính toán và cải thiện hiệu quả. Hình 5 là cách mà chúng tôi tính toán phân vùng cần quan tâm.



Hình 5. *Phân vùng quan tâm làn đường.*

ROI có dạng hình thang để phản ánh sự thu hẹp của đường khi khoảng cách tăng dần. Tính toán bằng công thức sau:

Đỉnh trên bên trái:

$$top\_left = \begin{pmatrix} \frac{(1 - top\_width\_ratio) \times width}{2} \times \\ (top\_height\_ratio \times height) \end{pmatrix} \quad (1)$$

Đỉnh trên bên phải:

$$top\_right = \begin{pmatrix} \frac{(1 + top\_width\_ratio) \times width}{2} \times \\ (top\_height\_ratio \times height) \end{pmatrix} \quad (2)$$

Đỉnh dưới bên trái:

$$bottom\_left = (0, height) \quad (3)$$

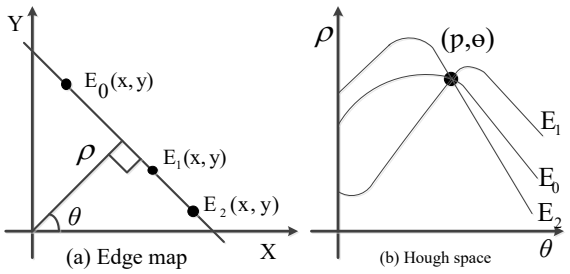
Đỉnh dưới bên phải:

$$bottom\_right = (width, height) \quad (4)$$

Qua đó, chúng tôi sử dụng thuật toán Hough Transform (HT). Đó là một phương pháp mạnh mẽ được sử dụng trong xử lý ảnh để phát hiện các đường thẳng từ hình ảnh đầu vào. Thuật toán này chuyển đổi dữ liệu điểm ảnh từ không gian tọa độ (x,y) sang không gian tham số (ρ,θ). Hough Transform được tính toán dưới công thức sau:

$$\rho = x \times \cos(\theta) + y \times \sin(\theta) \quad (5)$$

Khi áp dụng thuật toán, mỗi điểm trong không gian ảnh được ánh xạ thành một đường sin trong không gian tham số (ρ,θ). Giao điểm của các đường này trong không gian Hough thể hiện một đường thẳng trong không gian ảnh. Hình 6 thể hiện ánh xạ các điểm tham chiếu bản đồ cạnh sang không gian Hough.



Hình 6. *Ánh xạ các điểm tham chiếu bản đồ cạnh sang không gian Hough.*

Làn đường được phát hiện bao gồm nhiều đoạn thẳng dưới dạng phương trình phi tuyến tính (5), độ dốc *m* được tính theo công thức (6) và giao điểm với trục y được xác định công thức (7).

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad (6)$$

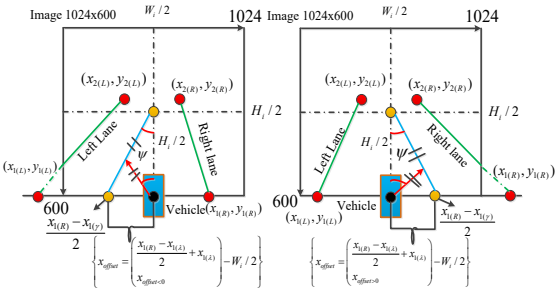
$$c = y_1 - m \times x_1 \quad (7)$$

$$y = mx + c \quad (8)$$

Làn đường trái và làn đường phải được xác định dựa trên ranh giới và vùng trung tâm của ảnh ở hình 7, các đoạn dốc âm thuộc vùng trái của ảnh và các đoạn dốc dương thuộc vùng phải của ảnh. Công thức (9),(10).

$$\left\{ x_{offset} = \left(\frac{x_{1(R)} - x_{1(L)}}{2} \right) - \frac{W_i}{2} \right\}, x_{offet} < 0 \quad (9)$$

$$\left\{ x_{offset} = \left(\frac{x_{1(R)} - x_{1(L)}}{2} \right) - \frac{W_i}{2} \right\}, x_{offet} > 0 \quad (10)$$



Hình 7. *Phương pháp tính toán.*

Làn đường trái, phải được phân biệt cần tính làn đường đại diện bằng cách lấy trung bình độ dốc và giao điểm của các đoạn thẳng hợp lệ theo công thức (11).

$$average(m, c) = \frac{1}{n} \sum_{i=1}^{n} (m_i, c_1) \quad (11)$$

Y_offset là khoảng cách cố định bằng một nửa chiều cao của ảnh hướng nhìn ra phía trước, dựa vào giá trị x_offset và y_offset tính toán góc lái của xe theo thức (12).

$$angle\_to\_mid\_radian = arctan\left(\frac{x_{offset}}{y_{offset}}\right) \quad (12)$$

Góc lái được tính toán dưới dạng radian nên cần chuyển đổi giá trị từ radian sang độ để gửi góc lái thực tế và điều khiển servo.

$$steering\_angle = \left(angle\_to\_mid\_radian \times \frac{180}{\pi}\right) \quad (13)$$

Góc lái được tính toán do nhiều yếu tố tác động làn đường được phát hiện như màu sắc và ánh sáng dẫn đến việc nhận diện làn đường có những hệ số sai lệch. Bộ lọc lượt mũ (EMA) được dùng để làm mượt góc lái trong quá trình xử lý ảnh, làm giảm nhiễu sự sai số.

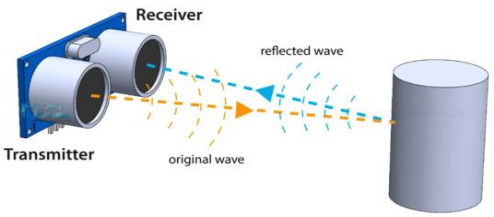
$$smoothed\_angle = \alpha \times current\_angle + (1-\alpha) \times previous\_angle \quad (14)$$

3.2 Phát hiện vật cản và trách vật cản

Chúng tôi đặt hai cảm biến siêu âm phía trước và bên hông xe kết nối với cổng vào và ra của Jetson Nano. Mỗi sóng siêu âm được nhận lại sau khi phản xạ từ vật cản, khoảng cách từ vật cản đến xe có thể được tính toán theo công thức sau:

$$\Delta t = t_1 - t_0$$
$$L = (\vartheta \times \Delta t)/2 \quad (15)$$

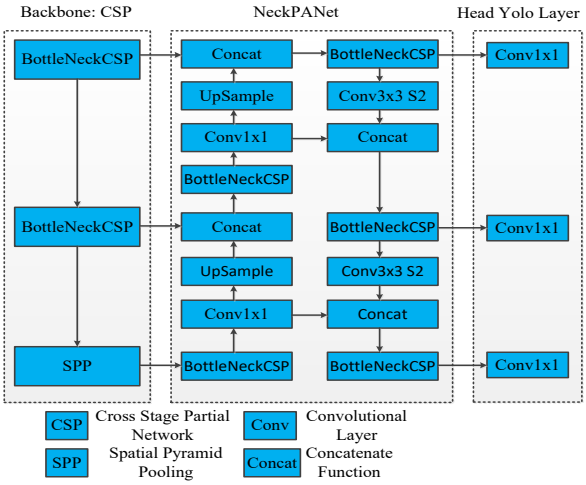
Thời gian phát sóng siêu âm được kí hiệu là t0 thời gian nhận lại sóng siêu âm phản xạ được kí hiệu là t_1, và Δt đại diện cho khoảng thời gian việc phát sóng âm và nhận lại tiếng vọng khi phản xạ trong hình 8. Khoảng cách đo được là L, vận tốc âm thanh được biểu thị là V [5].



Hình 7. *Cách hoạt động của cảm biến siêu âm.*

3.3 Phát hiện tín hiệu giao thông và đối tượng

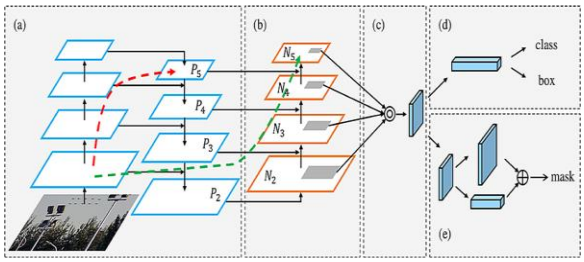
Với việc đề cao về tính chính xác cao và tốc độ phát hiện nhanh, mô hình YOLOv5 đáp ứng được yêu cầu của các ứng dụng thời gian thực. YOLOv5 có thể chia thành ba thành phần chính: backbone, neck và head [8]. Hình 9 là cấu trúc tổng thể của Yolov5.



Hình 9. *Cấu trúc YOLOV5.*

- Backbone: Là cấu trúc chính chịu trách nhiệm việc trích xuất fetures maps từ hình ảnh đầu vào và dùng Cross Stage Partial (CSP) giúp tối ưu hóa quá trình. Qua mỗi lớp convolution, kích thước của dữ liệu hình ảnh sẽ giảm đi, giúp giảm lượng tính toán và tăng tốc độ xử lý.

- Neck: YOLOv5 sử dụng MaxPooling tuần tự. MaxPooling tuần tự giúp giảm số lượng phép tính và tăng tốc độ tính toán, đồng thời vẫn đảm bảo khả năng học được các đặc trưng đa mức.



Hình 10. *Các lớp kiến trúc trong Yolov5.*

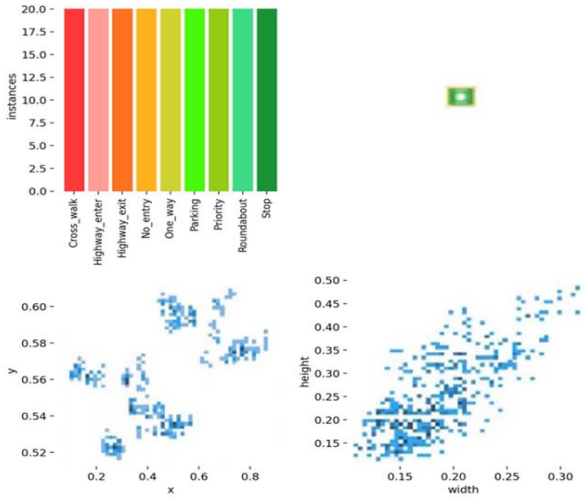
Kim tự tháp fetures map là một thành phần cơ bản trong các hệ thống nhận dạng để phát hiện các đối tượng ở các quy mô khác nhau tại hình 10. Việc xây dựng một kim tự tháp gồm các feature map, các tầng thấp có độ phân giải cao nhưng ít thông tin ngữ nghĩa, các tầng cao có độ phân giải thấp nhưng nhiều thông tin ngữ nghĩa với phương pháp skip connection để kết hợp dữ liệu từ các tầng khác nhau bằng cách cho phép bỏ qua các phép tính phi tuyến tính[9],[10].

- Head: sử dụng các feature map từ Neck để dự đoán tọa độ, xác suất và lớp các đối tượng đưa ra đầu ra cuối cùng dưới dạng vecto đại diện cho vị trí và loại đối tượng.

Để nâng cao khả năng nhận thức của xe chúng tôi sử dụng mô hình YOLOV5 để đào tạo. Chúng tôi đã tiến hành thu thập gần 1500 dữ liệu ảnh để thực hiện đào tạo các tín hiệu giao thông, người đi bộ, xe và giao lộ. Ảnh sẽ được thu thập từ Camera Pi V2 và lưu vào Raspberry Pi5 và lưu vào tập dữ liệu, dữ liệu được thu thập chúng tôi chia thành 3 phần thư mục là test, train, valid tăng khả năng nhận diện của mô hình.

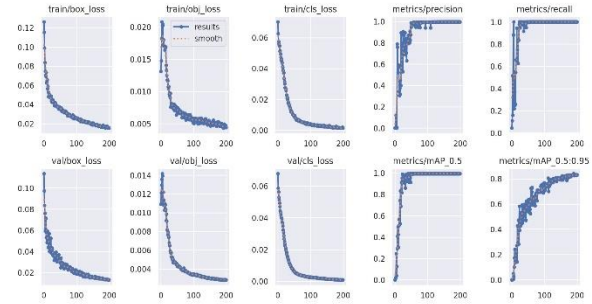
Các đối tượng cụ thể như biển báo sẽ được chia thành tập dữ liệu riêng biệt. Đối tượng như xe, người đi bộ, người đi xe đạp sẽ được lưu vào và huấn luyện thành từng phần riêng biệt giúp nhận diện chính xác mô hình, sau đó chúng tôi tiến hành gán nhãn dữ liệu đã được thu thập và lưu dưới dạng txt để giúp mô hình có thể nhận biết mô hình phát hiện là đối tượng nào. Hình 11 thể hiện biển báo, vật thể đã được gán nhãn dữ liệu hoàn tất quá trình chuẩn bị data cho quá trình huấn luyện. Sau khi hoàn tất quá trình huấn luyện chúng tôi thu được dạng biểu đồ để thống kê số lượng nhãn dữ liệu được gán trong hình 12.

Hình 11. *Gán nhãn dữ liệu.*



Hình 12. *Số lượng label thống kê từ YOLOV5.*

Sau khi có được dữ liệu bao gồm ảnh và nhãn dữ liệu chúng tôi tiến hành huấn luyện, chúng tôi tiến hành train thử nghiệm với các số lần epoch khác nhau, lần đầu tiên tiến hành đào tạo với 100 epoch mô hình có thể nhận diện được tuy nhiên do dữ liệu lớn và việc đào tạo chưa đủ vì vậy độ chính xác chưa tốt, nên chúng tôi tiếp tục tiến hành đào tạo và thu được kết quả tốt ở 600 epoch.

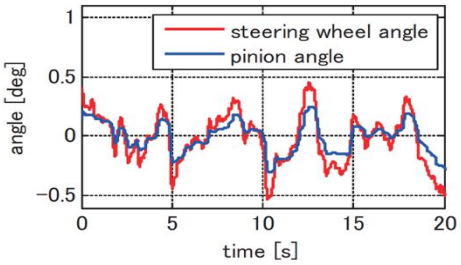


Hình 13. *Biểu đồ kết quả huấn luyện.*

Biểu đồ trên hình 13 cho thấy kết quả đào tạo khá tốt, training loss và validation loss đều giảm dần cho thấy mô hình học tốt trong quá trình huấn luyện mà không bị overfitting. Precision và recall đều bằng 1 cho thấy mô hình đưa ra dự đoán chính xác và tìm được hầu hết các mẫu dương trong dữ liệu.

4. KẾT QUẢ THỬ NGHIỆM

Sau khi thực hiện tính toán các giải pháp lọc nhiễu, tăng độ chính xác và tính toán tối ưu góc lái nhận diện làn đường dùng Hough Transform, góc lái cho ra có mức độ ổn định cao cụ thể như Hình 14. Kết quả cho thấy góc lái trước và sau khi tối ưu đạt hiểu quả tốt hơn, tránh được tình huống góc lái đánh quá nhanh và bất chợt.



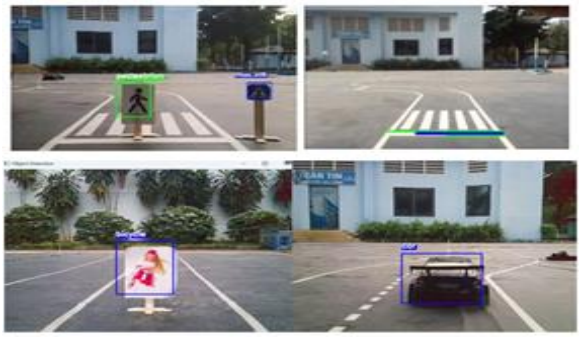
Hình 14. *So sánh hệ số góc lái ổn định và chưa ổn định.*

Hình 8. *Kết quả nhận diện làn đường.*

Hình 15 cho thấy camera của xe đã nhận diện rõ ràng để thu thập hình ảnh đó đưa vào bộ xử lý tín hiệu đầu vào, sau quá trình xử lý đã cho ra kết quả chính xác.

Hình 9. *Kết quả huấn luyện biển báo.*

Sau đó kết quả thể hiện như hình 16, khi chúng tôi cho xe chạy thử trên bản đồ thật và đặt các vật thể, biển báo, xe và con người. Trong lúc xe di chuyển khả năng nhận diện và phát hiện các đối tượng đó đúng 90%.



Hình 10. *Kết quả huấn luyện đối tượng.*

Chúng tôi đã thử nghiệm việc phát hiện làn đường trên tất cả các môi trường và điều kiện khác nhau và thu được các kết quả nổi bật thể hiện trên hình 17.

Ngoài ra, chúng tôi còn thử nghiệm trên bản đồ thực và ghi lại những dữ liệu xử lý ảnh dò làn đường khi chạy thực tế. Hình ảnh và video thử nghiệm cho thấy khả năng nhận diện làn đường một cách chính xác và phán đoán tình huống đủ để đảm bảo giữ khoảng cách an toàn. Chi tiết video thực nghiệm có thể xem tại link sau:

https://youtu.be/jpgMhoUd3CQ?si=TltN_K8l7n913v3

5. KẾT LUẬN

Bài báo chúng tôi đã trình bày phương pháp nghiên cứu và phát triển xe tự hành dựa trên nền tảng hệ điều hành Robot Operating System (ROS) cho những bài toán điều hướng tự động trong một môi trường tuân theo hệ thống giao thông đường bộ. Với việc dùng phương pháp xử lý ảnh để phát hiện làn đường đảm bảo tính chính xác cao lên đến 90%. Với ưu điểm phát hiện và nhận diện rõ ràng các hướng đi, xe có thể tính toán những góc lái ổn định để có thể di chuyển chính xác trên đường. Ngoài ra việc sử dụng YOLOV5 để nhận diện các đối tượng cũng giúp ích cho xe về khả năng nhận diện và học các đối tượng đó. Các kết quả thử nghiệm trên bản đồ thật đã cho thấy tính hiệu quả và khả thi của các nghiên cứu mà chúng tôi đưa ra. Mặc dù, chúng tôi gặp không ít những khó khăn về mặt chi phí, những tác động môi trường ảnh hưởng tới sự hoàn thiện của các phương pháp,… Tiếp đến, chúng tôi đang nghiên cứu về phương pháp mới để xe có thể tối ưu và hạn chế những tác động ngoài luồng và có thể áp dụng lên một chiếc xe thật.

6. TÀI LIỆU THAM KHẢO

[1] Li, Y., & Chen, L. (**2016**). *Nighttime lane markings recognition based on Canny detection and Hough transform*. *In Proceedings of The 2016 IEEE International Conference on Real-Time Computing and Robotics.*

DOI: 10.1109/RCAR.2016.7784064

[2] Lee, S., Son, H., & Min, K. (**2010**). *Implementation of Lane Detection System using Optimized Hough Transform Circuit*. In *Proceedings of the IEEE International Conference. Korea Electronics Technology Institute, Seongnam-si*, Republic of Korea. ISBN: 978-1-4244-7456-1.

DOI: 10.1109/APCCAS.2010.5775078

[3] Pannu, G. S., Ansari, M. D., & Gupta, P. (**2015**). *Design and Implementation of Autonomous Car using Raspberry Pi*. *International Journal of Computer Applications*, 113(9), March 2015.

DOI: 10.5120/19854-1789

[4] Tran Ngoc, H., & Quach, L. D. (**2022**). *Adaptive Lane Keeping Assist for an Autonomous Vehicle based on Steering Fuzzy-PID Control in ROS*. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 13(10).

DOI: 10.14569/IJACSA.2022.0131086

[5] Roy, S., & Zhang, Z. (**2020**). *Route Planning for Automatic Indoor Driving of Smart Cars*. In *Proceedings of the 2020 IEEE 7th International Conference on Industrial Engineering and Applications*. Tongji University, Shanghai, China.

DOI: 10.1109/ICIEA49774.2020.9102061

[6] Hwang, K., Jung, I. H., & Lee, J. M. (**2022**). *Implementation of Autonomous Driving on RC-CAR with Raspberry PI and AI Server*. Webology, 19(1), January 2022.

DOI: 10.14704/WEB/V19I1/WEB19293.

[7] Hồ Văn Thu, & Lê Thanh Phúc. (**2015**). *Ứng dụng xử lý ảnh nhận dạng đường đi cho ô tô chạy tự động (Application of Image Processing to Detect Lane for Autonomous Vehicle)*. Tạp chí Khoa học Giáo dục Kỹ thuật, Trường Đại học Sư phạm Kỹ thuật TP.HCM, (31/2015), 21.

https://jte.edu.vn/index.php/jte/article/view/492

[8] Li L, Wang Z, Zhang T. *GBH-YOLOv5: Ghost Convolution with BottleneckCSP and Tiny Target Prediction Head Incorporating YOLOv5 for PV Panel Defect Detection*. Electronics. 2023; 12(3):561.

DOI: 10.3390/electronics12030561

[9] Xie, J., Pang, Y., Nie, J., Cao, J., & Han, J. (**2022**). *Latent feature pyramid network for object detection*. IEEE Transactions on Multimedia, 25, 2153-2163.

DOI: 10.1109/TMM.2022.3143707

[10] Wang, H., Qin, Q., Chen, L., Li, Y., & Cai, Y. (**2025**). *RTMDet-R: A Robust Instance Segmentation Network for Complex Traffic Scenarios*. *IEEE Transactions on Intelligent Transportation Systems*.

DOI: 10.1109/TITS.2025.3539658