NGHIÊN CỨU VÀ PHÁT TRIỂN HỆ THỐNG XE TỰ HÀNH ỨNG DỤNG TRÍ TUỆ NHÂN TẠO

DEVELOP AUTONOMOUS CAR SYSTEM USING ARTIFICIAL INTELLIGENCE

Hà Thị Kim Duyên[1,*], Lê Mạnh Long[1], Nguyễn Đức Duy[1], Phan Sỹ Thuần[1], Nguyễn Ngọc Hải[1], Nguyễn Thị Tú Uyên[1], Ngô Mạnh Tiến[2]

TÓM TẮT

Bài báo này trình bày phương pháp xây dựng hệ thống xe tự hành có gắn camera thực hiện nhiệm vụ điều hướng tự động trong môi trường có vạch kẻ đường và biển báo giao thông. Xe tự hành sử dụng mô hình CNN cho nhiệm vụ nhận dạng vạch kẻ đường, thuật toán Adaboost Cascaded cho nhiệm vụ nhận dạng biển báo giao thông, hệ thống được lập trình nhúng trên nền tảng phần cứng xử lý hiệu năng cao chuyên dụng cho AI là TX2 Jetson và hệ điều hành lập trình cho robot ROS. Thực nghiệm cho thấy xe tự hành hoạt động đảm bảo các chỉ tiêu chất lượng đặt trước nhờ các hệ thống nhận dạng đạt tỉ lệ chính xác cao. Kết quả đạt được thể hiện sự hiệu quả của hướng nghiên cứu ứng dụng trí tuệ nhân tạo trong nhiệm vụ điều hướng tự động của xe tự hành trong các môi trường phức tạp.

Từ khóa: *Học sâu, Robot Omni, hệ điều hành Robot (ROS), điều hướng.*

ABSTRACT

This paper presents the method of constructing an autonomous car system with a camera to perform the task of automatic navigation in environments with road markings and traffic signs. Autonomous car uses the CNN model for the task of iquandentifying road markings, the Adaboost Cascaded algorithm for the task of identifying traffic signs. Our experiments show that autonomous car operates to ensure pre-set quality criteria thanks to the high accuracy rate identification systems. The results show the effectiveness of the research direction to apply artificial intelligence in the task of automatic navigation of autonomous cars in complex environments.

Keywords: *Deep Learning, Omni Robot, Robot Operating System (ROS), navigation.*

[1]Khoa Điện tử, Trường Đại học Công nghiệp Hà Nội

[2]Viện Vật lý, Viện Hàn lâm Khoa học và Công nghệ Việt Nam

[*]Email: ha.duyen@haui.edu.vn

Ngày nhận bài: 10/01/2021

Ngày nhận bài sửa sau phản biện: 20/6/2021

Ngày chấp nhận đăng: 25/10/2021

1. GIỚI THIỆU

Trong những năm gần đây trên thế giới, cùng với việc thông minh hóa robot là nhu cầu phát triển các robot di động đã dẫn đến sự bùng nổ trong nghiên cứu phát triển các hệ thống robot di động tự trị (autonomous mobile robot). Robot di động là một máy tự động có khả năng di chuyển đến đích trong một môi trường nào đó. Đây là điều khác với các robot công nghiệp (như tay máy có khớp quay và đầu kẹp) được gắn với một không gian cố định. Khởi đầu bằng các xe vận tải tự động AGV (autonomous guided vehicles) đã được ứng dụng rất thành công trong công nghiệp, các robot di động thông minh hiện nay vẫn đang trong thời kỳ phát triển và được coi là có tiềm năng trong tương lai gần. Các tiến bộ công nghệ mới trong việc phát triển các thiết bị cảm biến (sensor) và khả năng tính toán của các hệ xử lý đã thúc đẩy mức độ tự trị trong sự vận hành các robot di động. Mặt khác, những đòi hỏi ứng dụng của robot di động trong các môi trường khác như dịch vụ, giải trí, y tế, an ninh, quân sự cũng hứa hẹn có những tiến bộ quan trọng trong quá trình thiết kế và phát triển các hệ thống này.

Không kể hoạt động của các bộ phận gắn trên robot di động, bài toán dẫn đường cho sự di chuyển của đế robot từ một điểm xuất phát tới đích một cách an toàn, được gọi tắt là "điều hướng cho robot di động", là bài toán chính yếu trong các nghiên cứu về robot di động hiện nay. Không khác nhiều so với hành vi của con người trong việc điều hướng, muốn giải quyết được bài toán này thì robot phải tự xác định được vị trí của mình trong môi trường (positioning), xác lập được bản đồ môi trường nếu cần thiết (mapping), vạch ra được quỹ đạo đi tới đích (path planning) và xuất ra cách thức điều khiển đi trên quỹ đạo (path control) và tránh vật cản (obstacle avoidance) trên đường đi. Để làm được điều đó, một hệ thống điều hướng của robot di động thông thường có thể chia thành 4 khối [1]: hệ thống cảm biến, hệ thống nhận thức, hệ thống lập kế hoạch di chuyển và hệ thống điều khiển động cơ. Robot di động cảm nhận về môi trường bên ngoài bằng nhiều cảm biến khác nhau gắn trên nó. Dữ liệu từ cảm biến được hệ thống nhận thức xử lý, kết hợp thành các thông tin có ý nghĩa. Gần đây các công trình nghiên cứu xây dựng hệ thống điều hướng của robot di động chỉ dựa trên một nguồn dữ liệu đầu vào là hình ảnh đang ngày càng xuất hiện nhiều và nhận được sự quan tâm lớn [2, 3], trong đó dữ liệu hình ảnh và các phương pháp trích xuất thông tin hữu ích từ hình ảnh phục vụ cho hệ thống điều hướng đã đạt được nhiều thành tựu quan trọng.

Xe tự hành là một trong những trường hợp đặc biệt của robot di động, vì bài toán điều hướng của xe tự hành không

đơn thuần là dẫn đường từ một điểm xuất phát đến đích, mà còn cần tuân thủ chặt chẽ các tín hiệu giao thông xuất hiện trong quá trình di chuyển. Để làm được điều đó, dữ liệu hình ảnh trở thành một nguồn dữ liệu đầu vào không thể thiếu đối với hệ thống. Các phương pháp ứng dụng dữ liệu hình ảnh trong xe tự hành tập trung vào giải quyết hai bài toán quan trọng: nhận dạng vạch kẻ đường và nhận dạng biển báo giao thông. Trong bài toán nhận dạng nhận dạng vạch kẻ đường, đã có nhiều công trình nghiên cứu sử dụng mạng nơ-ron nhân tạo (ANN) để đảm bảo được tính chính xác và có hiệu quả cao. Tuy nhiên việc sử dụng mạng ANN mất nhiều thời gian để học do mạng phải xử lý, học từng điểm ảnh của dữ liệu đầu vào. Mạng CNN ra đời để khắc phục nhược điểm đó nhờ việc sử dụng các lớp tích chập đặt phía trước các lớp nơ-ron nhân tạo thông thường để trích xuất các đặc trưng của ảnh đầu vào, giúp quá trình học của mô hình nhanh hơn rất nhiều, đồng thời đảm bảo tính chính xác cao. Trong bài toán nhận dạng biển báo giao thông, có một số nghiên cứu về chủ đề này đạt được những kết quả khả quan. Bài báo [4] trình bày phương pháp phát hiện và nhận dạng các biển báo giao thông đường bộ sử dụng kết hợp các kỹ thuật phân đoạn ảnh, phát hiện biên và phân tích hình dáng đối tượng để phát hiện vùng ứng viên có thể là biển báo giao thông. Sau đó, rút trích đặc trưng HOG và huấn luyện mạng Nơ-ron nhân tạo để nhận dạng biển báo cho kết quả nhận dạng đạt tỉ lệ 94%. Tuy nhiên, công trình này chưa được tối ưu một cách hiệu quả. Trong nghiên cứu [5], tác giả sử dụng phương pháp Haar-like kết hợp thuật toán tăng tốc Adaboost cho việc phát hiện ảnh và sử dụng phương pháp PCA cho nhiệm vụ phân loại. Phương pháp này đã chứng minh được kết quả chính xác rất cao.

Mục tiêu của bài báo này tập trung vào giới thiệu một xe tự hành do nhóm tác giả phát triển, ứng dụng mô hình CNN cho nhiệm vụ bám làn đường, sử dụng thuật toán Machine Learning Adaboost trong bài toán phân loại biển báo giao thông đường bộ, sau đó tích hợp, lập trình nhúng trên nền tảng máy tính nhúng Jetson TX2 và hệ điều hành robot (Robot Operating System - ROS), vi điều khiển STM32 tạo thành một hệ thống xe tự hành chạy thử nghiệm.

2. CƠ SỞ LÝ THUYẾT/PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Cấu trúc phần cứng xe tự hành.

Cấu trúc phần cứng được sử dụng:

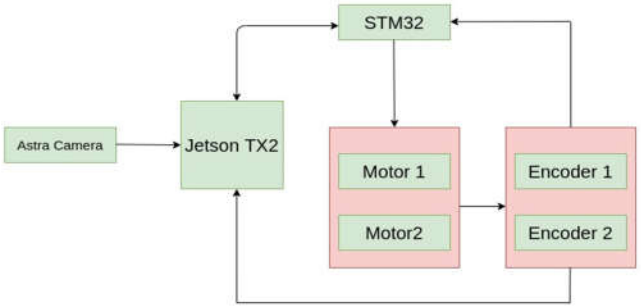
- Máy tính nhúng Jetson TX2 với cai trò xử lý trung tâm, là bộ xử lý hiệu năng cao chuyên dụng cho các xử lý trí tuệ nhận tạo (AI), Deep Learning, nó thu tập các tín hiệu từ các cảm biến, astra camera, Lindar, IMU và xử lý và gửi các tín hiệu đặt cho mạch điều khiển.

- Astra camera có độ phân giải hình ảnh RGB lên đến 1280 x 720 @ 30 khung hình / giây, độ sâu hình ảnh Res lên đến 640 x 480, tốc độ 30 khung hình / giây sẽ được sử dụng như mắt của robot để thu hình ảnh từ môi trường một cách rõ nét và chân thực nhất, là đầu vào cho các thuật toán nhận dạng vạch đường và biển báo.

- Mạch điều khiển STM32 sẽ là bộ phận nhận tín hiệu điều khiển từ Jetson TX2 rồi trực tiếp điều khiển tín hiệu đến mạch cầu MOSFET.

- Mạch cầu H sử dụng các MOSFET là mạch công suất điều khiển các động cơ DC 2 bánh di chuyển.

- Module Bluetooth để thu tín hiệu điều khiển từ điện thoại di động khi muốn điều khiển trực tiếp.



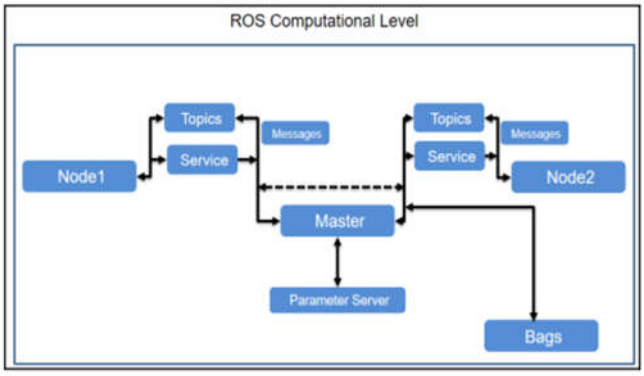
Hình 1. Sơ đồ cấu trúc phần cứng

2.2. Cấu trúc lập trình cho xe tự hành

2.2.1. Hệ điều hành lập trình Robot Operating System

Với mục đích giải quyết những thách thức đến từ sự phức tạp trong kiến trúc phần mềm của một hệ thống xe và robot tự hành, Robot Operating System (ROS) là một nền tảng quan trọng tạo điều kiện thuận lợi cho sự phát triển của các dự án về lĩnh vực này [6]. Đây là một framework chịu trách nhiệm đồng bộ hóa các module phần mềm của robot, trừu tượng các chi tiết phần cứng đối với lập trình viên, ngoài ra ROS còn cung cấp những công cụ mô phỏng và trực quan hóa mô hình robot trong môi trường ảo. Nhờ vậy các nhà phát triển có thể thuận tiện thực hiện các dự án robot trong cả giai đoạn triển khai và kiểm thử.

Về cơ bản, ROS có những đặc tính thiết yếu của một hệ điều hành như khả năng thực hiện các tác vụ (task) song song, giao tiếp, trao đổi dữ liệu với nhau giữa các tác vụ, quản lý dữ liệu,… Hơn thế nữa, để ROS có thể ứng dụng trong lĩnh vực robotics, ROS còn được phát triển riêng biệt về các thư viện, công cụ dành cho việc thu thập, xử lý, hiển thị, điều khiển,… ROS có thể kết hợp, tương tác với nhiều robot framework khác như Player, YARP, Orocos, CARMEN, Orca, Moos và Microsoft Robotics Studio.

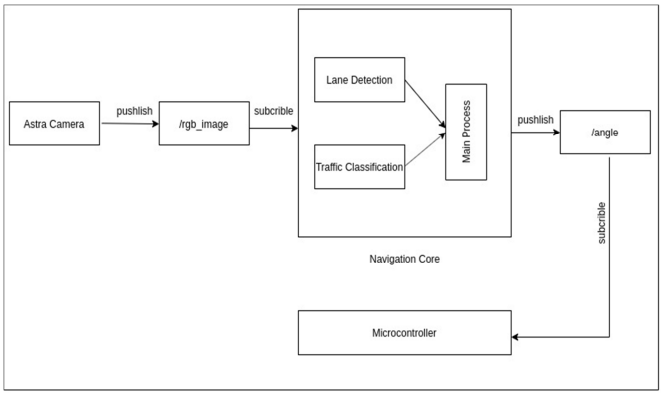


Hình 2. Cấu trúc chương trình của ROS

Cấu trúc giao tiếp của ROS được phát triển thông qua các node, các node được đóng gói trong các packages khác nhau theo từng nhiệm vụ. Hoạt động giao tiếp giữa các node dưới dạng chủ đề, tin nhắn, dịch vụ được minh họa trong hình 2.

2.2.2. Lập trình điều khiển xe tự hành trên nền ROS

Trong bài báo này, ROS đóng vai trò trung tâm điều phối giữa các module phần mềm của hệ thống theo hình 3. Các module đóng vai trò là các node mạng, thực hiện trao đổi dữ liệu thông qua cơ chế subscribe (nhận giữ liệu) và publish (cung cấp giữ liệu) tới một topic, mỗi topic chứa giữ liệu được cung cấp bởi một node duy nhất. Các cơ chế này đã được trừu tượng hóa và cung cấp các API phục vụ công việc lập trình.

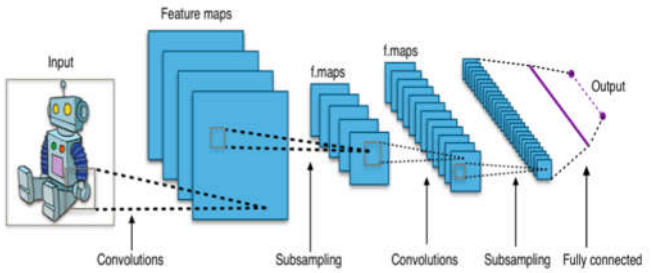


Hình 3. Hệ thống điều hướng của xe tự hành

Về kiến trúc phần mềm của hệ thống điều hướng, node Astra camera có nhiệm vụ chuyển đổi tín hiệu điện từ camera thành hình ảnh RGB, sau đó publish hình ảnh vào topic /rgb_image. Node Navigation Core subcrible topic /rgb_image để nhận hình ảnh phục vụ cho quá trình dự đoán góc lái. Tại đây, khối Lane Detection làm nhiệm vụ sử dụng mô hình CNN đã được huấn luyện dự đoán góc lái dự kiến, khối Traffic Classification sẽ sử dụng thuật toán Adaboost Cascaded để phát hiện và đưa ra phân lớp của biển báo xuất hiện trong hình ảnh. Tiếp theo, khối Main Process sẽ kết hợp đầu ra của hai khối trước đó để tính toán được góc lái cuối cùng mà cơ cấu chấp hành cần tuân theo. Góc lại này sẽ được node Navigation Core publish vào topic /angle, và nhiệm vụ còn lại của node Motor Control với sự phụ trách của vi điều khiển STM32 là nhận dữ liệu và điều hướng động cơ theo đúng yêu cầu.

2.3. Hệ thống nhận diện làn đường ứng dụng (*Convolution Neural Network (CNN)*

Mạng nơ-ron tích chập (CNN) [7] là một mô hình deep learning có khả năng xây dựng các hệ thống phân loại với độ chính xác cao. Cấu trúc cơ bản của CNN gồm các lớp tích chập (Convolution layer), lớp phi tuyến (Nonlinear layer) và lớp lọc (Pooling layer). Các lớp tích chập kết hợp với các lớp phi tuyến sử dụng các hàm phi tuyến như ReLU hay Tanh để tạo ra thông tin trừu tượng hơn (Abstract/higher-level) cho các lớp tiếp theo. Cấu trúc cơ bản của một mạng nơ-ron tích chập được biểu diễn trong hình 4.

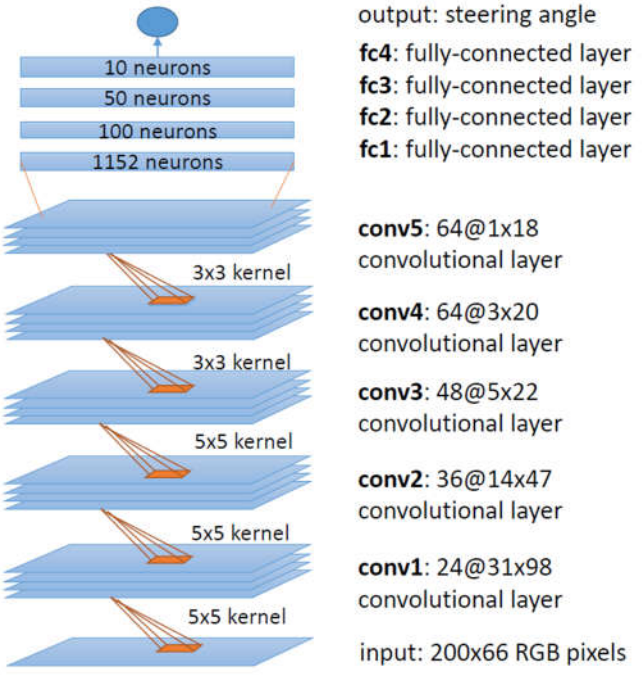


Hình 4. Cấu trúc cơ bản của một mạng CNN

Lớp tiếp theo là kết quả tích chập từ lớp trước đó, vì vậy CNN có được các kết nối cục bộ vì mỗi nơ-ron ở lớp tiếp theo sinh ra từ một bộ lọc được áp đặt lên một vùng cục bộ của lớp trước đó. Mỗi lớp như vậy được áp đặt các bộ lọc khác nhau. Một số lớp khác như lớp pooling/subsampling dùng để lọc lại các thông tin hữu ích hơn bằng cách loại bỏ các thông tin nhiễu. Trong suốt quá trình huấn luyện, CNN sẽ tự động học các tham số cho các lớp. Lớp cuối cùng được gọi là lớp kết nối đầy đủ (Fully connected layer) dùng để phân lớp dữ liệu.

Kiến trúc mạng đề xuất

Trong nghiên cứu này, nhóm tác giả đề xuất xuất một kiến trúc mạng dựa trên kiến trúc CNN NVIDIA DAVE-2 [8], kiến trúc mạng đề xuất được biểu diễn trong hình 5.



Hình 5. Mô hình mạng CNN dự đoán góc quay

Mô hình có 9 lớp với 250000 tham số cần chỉnh định. Ảnh đầu vào là ảnh RGB kích thước 200x66 pixel. Với 5 lớp tích chập, ảnh đầu vào sẽ được trích xuất và sau đó thu được các giá trị đặc trưng nhất. Qua 4 lớp fully connected ta có đầu ra là 3 node là các góc giá trị đặt gửi xuống bộ điều khiển để điều khiển xe. Đó là các giá trị -30 độ (rẽ trái), 0 độ (đi thẳng) và 30 độ (rẽ phải). Lưu ý rằng đối với mỗi khối ở các tầng từ đầu tới F6 ta sử dụng hàm kích hoạt Sigmoid dạng:

$$f(x) = \frac{1}{1+e^{-x}} \quad (1)$$

2.4. Phân loại biển báo ứng dụng CASCADE ADABOOST

Quá trình nhận dạng biển báo giao thông bao gồm 2 bài toán nhỏ là phát hiện và nhận dạng. Trong bài báo này, nhóm tác giả đề xuất sử dụng thuật toán Cascaded Adaboost với đặc trưng Haar-like [9] để xây dựng các bộ phân lớp cho từng loại biển.

2.4.1. Thuật toán Adaboost

Adaboost dựa trên kĩ thuật Boosting với ý tưởng là gán cho mỗi mẫu một trọng số W và tại mỗi bộ phân lớp sẽ tăng trọng số cho mẫu sai, giảm trọng số cho mẫu đúng. Sau đó tạo bộ phân lớp mới theo hướng tập trung vào các mẫu sai [10]. Quá trình huấn luyện được mô tả cụ thể:

Bước 1: Cho tập ảnh huấn luyện $(x_1,t_1)...(x_n,t_n)$ với $t_i \in \{-1 \quad 1\}$. Khởi tạo trọng số cho mỗi mẫu huấn luyện $w_{n(1)} = 1/N$ với $n = 1,N$

Bước 2: Thủ tục Boosting

For m = 1…M

Xây dựng bộ phân lớp yếu y_m:

Với mỗi đặc trưng j xây dựng bộ phân lớp y_j có độ lỗi E_j theo công thức (2):

$$E_j = \sum_1^n w_n^{(m)} * I(y_m(x_n) \# t_n) \quad (2)$$

$$\text{với } I(y_m(x_n) \neq t_n) = \begin{cases} 1 & (y_m(x_n) \neq t_n) \\ 0 & (y_m(x_n) \neq t_n) \end{cases}$$

Chọn bộ phân lớp y_j có độ lỗi nhỏ nhất ta được y_m.

Cập nhật lại trọng số cho bộ phân lớp sau bằng cách tập trung vào các mẫu sai theo công thức (3):

$$w_n^{(m+1)} = w_n^{(m)} . e^{\alpha_m . I(y_m(x_n) \neq t_n)} \quad (3)$$

Với: $\alpha_m = \ln \frac{1-\varepsilon_m}{\varepsilon_m}$

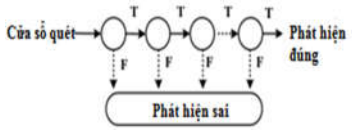
$$\varepsilon_m = \frac{\sum_1^N w_n^{(m)} * I(y_m(x_n) \neq t_n)}{\sum_1^N w_n^{(m)}} \quad (4)$$

Bước 3: Bộ phân lớp cuối cùng là tổng của M bộ phân lớp.

2.4.2. Cấu trúc phân tầng Cascade

AdaBoost là một bộ phân lớp mạnh tuy nhiên nhược điểm của nó là trên mỗi bộ phân lớp yếu ta phải duyệt tất cả các cửa sổ trên ảnh dẫn đến thời gian tính toán lâu. Để cải thiện nhược điểm này người ta thường sử dụng cấu trúc phân tầng.

Cấu trúc phân tầng được Viola and Jones giới thiệu lần đầu tiên [11] cho bài toán phát hiện khuôn mặt. Ý tưởng của cấu trúc là ở tầng hiện tại chỉ xem xét đến những cửa sổ thỏa mãn ở tầng trước. Hình 6 minh họa sơ đồ một cấu trúc phân tầng cho bộ phát hiện đối tượng. Với cấu trúc này, giải thuật nhanh chóng loại bỏ những ứng viên không thuộc lớp đó.



Hình 6. Sơ đồ cấu trúc phân tầng

3. KẾT QUẢ NGHIÊN CỨU/TÍNH TOÁN/MÔ PHỎNG VÀ THẢO LUẬN

3.1. Hệ thống nhận dạng làng đường

3.1.1. Huấn luyện mạng

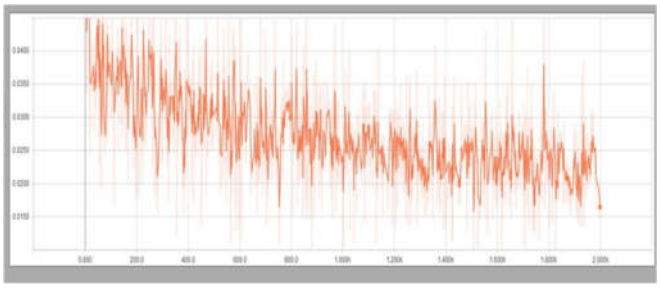
Nhóm tác giả tiến hành huấn luyện với phương pháp back propagation. Sử dụng tập ảnh huấn luyện với 1000 ảnh để học và 400 ảnh để giám sát (dùng để chỉnh định các hyperparameter của mô hình). Mục tiêu là hàm sai số (loss function).

$$E = \frac{1}{N}\sqrt{\sum_1^N (y - y_0)^2} \quad (5)$$

Tiến dần giá trị 0, với y là giá trị do mô hình dự đoán được, y_0 là giá trị được gán cho của tập đầu vào.

3.1.2. Kết quả huấn luyện mạng

Sau 2000 vòng lặp, kết quả huấn luyện chính là đồ thị của hàm sai số E được thể hiện ở hình 7.



Hình 7. Kết quả huấn luyện

Có thể thấy sau 2000 vòng lặp hàm sai lệch giảm xuống còn 0,0153 tương ứng sai số trung bình là 1,53% và độ chính xác của mô hình là 98,47%. Đây là một kết quả có thể chấp nhận được.

Tiến hành nhận dạng với mô hình đã được huấn luyện ta có kết quả như bảng 1.

Bảng 1. Kết quả huấn luyện nhận dạng làn đường bằng CNN

	Ảnh dãn nhãn rẽ trái	Ảnh gián nhãn đi thẳng	Ảnh dán nhãn rẽ phải
Dự đoán rẽ trái	98,5%	0,5%	1%
Dự đoán đi thẳng	0,48%	99%	0,52%
Dự đoán rẽ phải	1,2%	0,89%	97,91%

Với kết quả trên kết luận được rằng mô hình đạt độ chĩnh xác cao nhất với ảnh được dán nhãn đi thẳng với độ

chính xác 99%, tiếp sau đó là ảnh rẽ trái với 98,5% và ảnh rẽ phải với 97,91%. Với kết quả này khi mô hình được áp dụng với xe mô hình xe tự hành đã đảm bảo kết quả bám đường chính xác.

3.2. Hệ thống phân loại biển báo

Trong nghiên cứu này, nhóm tác giả tiến hành nhận dạng 4 loại biển: biển hạn chế tốc độ, biển rẽ trái, biển rẽ phải và biển dừng (stop) tương ứng xây dựng 4 bộ phân lớp.

Hình 8. Biển báo cần nhận

Dữ liệu để huấn luyện bao gồm 3020 ảnh nền không chứa biển báo, còn lại mỗi loại sử dụng 1200 ảnh.

Các thông số được chọn cho bộ huấn luyện Cascade Adaboost như sau: false positive rate, detection rate và cascade false positive có giá trị lần lượt là 0,5; 0,5 và 0,005.

Các bộ phân lớp sau đó được kiểm th với 200 ảnh trong bộ test chuẩn của cuộc thi [11] và kết quả được thể hiện trên bảng 2.

Bảng 2. Kết quả huấn luyện phân lớp biển báo bằng thuật toán Cascade Adaboost

	Biển hạn chế 40km/h	Biển rẽ trái	Biển rẽ phải	Biển dừng
Số lượng kiểm tra	50	50	50	50
Số phát hiện	60	55	54	61
Số nhận dạng đúng	45	47	46ử	43
Độ chính xác	90%	94%	92%	86%

3.3. Thử nghiệm mô phỏng

Trong phần này, hệ thống điều hướng sẽ được tiến hành mô phỏng trên phần mềm giả lập được cung cấp bởi Unity. Trong quá trình mô phỏng, phần mềm giả lập sẽ trả về dữ liệu hình ảnh do camera phía trước của xe ghi lại. Hình ảnh này sẽ được sử dụng làm đầu vào của mô hình CNN đã được training và thuật toán phân loại biển báo để dự đoán góc quay cần thiết, sau đó truyền lại giá trị cho trình giả lập để điều hướng xe đi theo góc quay đó trong khung hình tiếp theo.

Hình 9. Trình giả lập mô phỏng xe tự hành

Hình 10. Hình ảnh trả về từ camera

Tại hình 11, nhận thấy đường có xuất hiện góc cua trái đồng thời không có sự xuất hiện của biển báo, hệ thống điều hướng dự đoán góc quay là -30 độ.

Hình 11. Hệ thống dự đoán góc quay -30 độ

Hình 12. Kết quả dự đoán góc quay 0 độ

Hình 13. Kết quả dự đoán góc quay -30 độ

Tại hình 12, hệ thống nhận diện làn đường dự đoán góc quay là 0 độ, hệ thống phân loại biển báo đã dự đoán thành công biến báo rẽ trái, hệ thống điều hướng sẽ tiếp tục cho xe đi thẳng cho đến khi không còn sự xuất hiện của biến báo thì tiến hành vào cua với góc quay -30 độ, theo hình 13.

Trên trình giả lập mô phỏng, hệ thống có thể hoạt động tốt đáp ứng thời gian thực với tốc độ 30 FPS và giúp chiếc xe có thể di chuyển một cách linh hoạt trong môi trường.

4. KẾT LUẬN VÀ KHUYẾN NGHỊ

Bài báo này trình bày phương pháp xây dựng hệ thống xe tự hành ứng dụng trí tuệ nhân tạo trên nền tảng hệ điều hành lập trình robot ROS cho bài toán điều hướng tự động trong môi trường tuân theo tín hiệu làn đường và biển báo giao thông. Mạng CNN ứng dụng cho nhiệm vụ xác định đường đi đảm bảo tính chính xác cao với tỉ lệ 98%. Với ưu điểm có kiến trúc đơn giản sử dụng các lớp Convolution để trích xuất đặc trưng của ảnh làm rút ngắn thời gian học, mạng CNN đang ngày càng được sử dụng phổ biến trong các ứng dụng về xử lý ảnh. Thuật toán Adaboost ứng dụng cho bài toán nhận dạng biển báo giao thông đảm bảo tính chính xác cao, dễ dàng cho việc triển khai trên các nền tảng máy tính nhúng. Các kết quả kiểm thử phương pháp trên trình giả lập Unity cho thấy tính hiệu quả, khả thi của phương pháp điều hướng cho xe tự hành.

TÀI LIỆU THAM KHẢO

[1]. Kocic Jelena, Jovicic Nenad, Drndarevic Vujo., 2019. *An End-to-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms*. Sensors. 10.3390/s19092064.

[2]. Lenac Kruno, Kitanov Andrej, Cupec Robert, Petrovic, Ivan, 2017. *Fast planar surface 3D SLAM using LIDAR*. Robotics and Autonomous Systems 92. 197-220. 10.1016/j.robot.2017.03.013.

[3]. Yuan Chang, Chen Hui, Liu Ju, Zhu Di, Xu Yanyan, 2018. *Robust Lane Detection for Complicated Road Environment Based on Normal Map*. IEEE Access. PP. 1-1. 10.1109/ACCESS.2018.2868976.

[4]. Truong Quoc Bao, Truong Hung Chen, Truong Quoc Dinh, 2015. *Road traffic sign detection and recognition using HOG feature and Artificial Neural network*. Journal of Science, Can Tho University.

[5]. Nguyen Van Long, 2016. *Learn and propose methods to identify and classify traffic signs in Vietnam*. Master thesis, Duy Tan University.

[7]. Albawi Saad, Abed Mohammed Tareq, Alzawi Saad, 2017. *Understanding of a Convolutional Neural Network*. 2017 International Conference on Engineering and Technology (ICET).

[8]. Bojarski Mariusz, Testa Davide, Dworakowski Daniel, Firner Bernhard, Flepp Beat, Goyal Prasoon, Jackel Larry, Monfort Mathew, Muller Urs, Zhang Jiakai, Zhang Xin, Zhao Jake, Zieba Karol, 2016. *End to End Learning for Self-Driving Cars*. Computer Vision and Pattern Recognition.

[9]. R. Lienhart, J. Maydt, 2002. *An extended set of Haar features for rapid object detection*. IEEE Image Processing.

[10]. Yoav Freund, Robert E. Schapire, 1999. *A Short Introduction to Boosting*. Journal of Japanese Society for Artificial Intelligence, 14(5):771-780.

[11]. Paul Viola and Michael Jones, 2002. *Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade*. Advances in neural information processing systems 14.

AUTHORS INFORMATION

Ha Thi Kim Duyen[1], Le Manh Long[1], Nguyen Duc Duy[1], Phan Sy Thuan[1], Nguyen Ngoc Hai[1], Nguyen Thi Tu Uyen[1], Ngo Manh Tien[2]

[1]Faculty of Electronic Engineering, Hanoi University of Industry

[2]Insititute of Physics, Vietnam Academy of Science and Technology