NGHIÊN CỨU ỨNG DỤNG THUẬT TOÁN HỌC SÂU KẾT HỢP CẢM BIẾN KINECT TRONG PHÂN LOẠI VẬT THỂ

RESEARCH FOR THE APPLICATION OF DEEP LEARNING COMBINED KINECT SENSOR IN OBJECT CLASSIFICATION

Bùi Thanh Lâm[1], Nguyễn Đức Quang[1], Nguyễn Văn Trường[1], Phan Đình Hiếu[1], Vũ Tuấn Anh[1,*], Lưu Vũ Hải[1]

DOI: https://doi.org/10.57001/huih5804.2023.142

TÓM TẮT

Trong bài báo này, một hệ thống nhận diện vật thể sử dụng cảm biến Kinect và mô hình học sâu để xử lý hình ảnh đối tượng được đề xuất. Mô hình học sâu được áp dụng với cơ sở dữ liệu thu thập từ thực tế để đưa ra các đặc tính như hình dạng, màu sắc. Bên cạnh đó sử dụng cảm biến Kinect còn giúp thu nhận thông tin chiều sâu của đối tượng một cách dễ dàng. Các vật thể có tính chất khác nhau được thử nghiệm như: hình tròn, hình vuông, hình tam giác, màu đỏ, màu xanh, màu vàng với chiều cao khác nhau. Hệ thống đề xuất có khả năng phân loại đặc điểm của mỗi đối tượng với độ chính xác cao với chi phí tiết kiệm. Kết quả thực nghiệm cho thấy các đối tượng được phân loại với độ chính xác 92% với thời gian trung bình nhận diện mỗi vật thể là 50ms. Qua đó thể hiện khả năng ứng dụng trong thực tế của hệ thống đề xuất trong công việc phân loại vật thể.

***Từ khóa**: Học sâu, phân loại vật thể, cảm biến Kinect.*

ABSTRACT

In this paper, an objective classification system using a Kinect sensor and deep learning model to process object images is proposed. The deep learning model is carried out with the data collected from reality to give features such as shape and color. Besides that, using the Kinect sensor also helps to acquire depth information of the object easily. Objects of different properties are tested such as circles, squares, triangles, red, blue, and yellow with different heights. The proposed system is capable of classifying the characteristics of each object with high accuracy and low cost. Experimental results show that the objects are classified with an accuracy of 92% with the average time to recognize each object being 50ms. Thereby demonstrating the practical applicability of the proposed system in object classification.

***Keywords:** Deep learning, object classification, Kinect.*

[1]Trường Đại học Công nghiệp Hà Nội
[*]Email: bak.haui@gmail.com
Ngày nhận bài: 20/3/2023
Ngày nhận bài sửa sau phản biện: 25/4/2023
Ngày chấp nhận đăng: 25/8/2023

1. GIỚI THIỆU

Nghiên cứu các hệ thống nhận diện thông minh dựa trên thuật toán học sâu là xu hướng của thế giới trong những năm gần đây. Mô học sâu đã được phát triển để có thể đảm nhận các nhiệm vụ phức tạp như nhận dạng giọng nói, nhận diện đối tượng [1, 2]. Các thuật toán nhận diện đối tượng dựa trên mô hình học sâu đã mang lại nhiều hiệu quả [3, 4]. Các kỹ thuật nhận diện này được chia thành hai loại cơ bản [5]: thứ nhất bao gồm các thuật toán hai giai đoạn như RCNN [6], SPP-net [7], Fast-RCNN [8] và Faster-RCNN [9],… sử dụng vùng đề xuất hoặc tìm kiếm có chọn lọc để tạo ra vùng đề xuất, sau đó phân loại và hồi quy trên phân vùng này. Mô hình cho phép phân loại với độ chính xác cao, tuy nhiên thời gian xử chậm. Mô hình còn lại bao gồm SSD [10] và YOLO [11], sử dụng phương pháp hồi quy lần lượt dựa trên một mạng nơ ron duy nhất để dự đoán biên dạng và xác suất tồn tại đối tượng. Tốc độ xử lý được cải thiện đáng kể đối với mô hình này tuy nhiên độ chính xác không cao bằng mô hình hai giai đoạn.

Bên cạnh các thuật toán thông minh, camera là bộ phận quan trọng ảnh hưởng đến độ chính xác và tốc độ xử lý trong một hệ thống thị giác máy tính. Khác với các loại camera thông thường cảm biến Kinect có khả năng thu nhận thêm độ sâu bên cạnh thông tin ảnh RGB. Khi so sánh với các thiết bị Time-off-light trước đây, Kinect có tính kinh tế vượt trội với độ phân giải chấp nhận được. Nhiều nghiên cứu đã áp dụng Kinect vào các ứng dụng như nhận dạng hoạt động cử chỉ người [12], hỗ trợ robot di chuyển [13], nhận dạng đối tượng trong không gian 3-D [14]. Robot theo dõi sử dụng Microsoft Kinect và Matlab cho thấy kết quả tốt trong việc xác định mục tiêu đã chọn trong hình ảnh và tính toán khoảng cách với thông tin trả về từ Microsoft Kinect [15]. Cảm biến Microsoft Kinect được dùng để phát hiện các chuyển động để điều khiển robot Khepea III [16].

Các nghiên cứu kết hợp Kinect sensor và thuật toán học sâu đang phát triển và đạt được thành công nổi bật. Douglas và cộng sự đề xuất một hệ thống thị giác máy tính nhằm mục đích phát hiện vật cản trên đường đi của mobile robot [17]. Hệ thống được lập trình trên Nvidia Jetson TX2, sử dụng cảm biến Microsoft Kinect và YOLO. Kết quả thực nghiệm cho thấy sự hiệu quả của thuật toán khi sai số độ sâu là bé hơn 3,64%. Choi

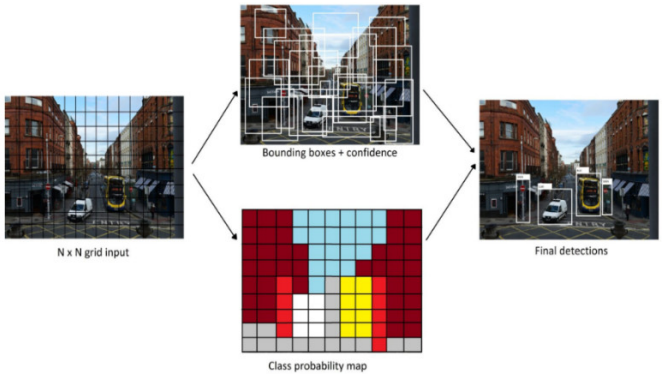
và cộng sự [18] đã áp dụng hệ thống trên trong việc phân tích trạng thái của đàn lợn con trong thời gian thực. Các tác giả sử dụng độ sâu đo đạc từ camera Kinect và mạng YOLO để tách các con lợn nằm chồng chéo lên nhau. Hệ thống đề xuất đã hoạt động tốt với độ chính xác trong việc phân loại là 91,96%. Trong nghiên cứu này, nhóm tác giả đề xuất một hệ thống phân loại các vật thể khác nhau sử dụng thuật toán học sâu kết hợp camera Kinect. Trong đó, camera Kincet được dùng để thu nhận ảnh RGB và độ sâu (RGBD) trong thời gian thực. Mục đích của việc phân loại là lấy chính xác vị trí và phân loại của đối tượng để gửi tín hiệu điều khiển cho robot gắp và đặt. Các đối tượng sử dụng trong mô hình thực nghiệm là các phôi hình vuông, hình tròn, hình tam giác với các độ sâu và màu sắc khác nhau. Kết quả thực nghiệm chứng minh hiệu quả của hệ thống đề xuất khi sai số phân loại các đặc tính của vật đạt 92%.

2. PHƯƠNG PHÁP

Trong nghiên cứu này, tác giả tập trung vào tốc độ phân loại của thuật toán. Qua các nghiên cứu [19, 20] cho thấy mô hình YOLO v5 được đánh giá là mô xử lý nhanh so với các thuật toán học sâu đã đề cập. Vì vậy trong bài báo, mô hình YOLOv5 được lựa chọn để thực hiện phân loại đối tượng.

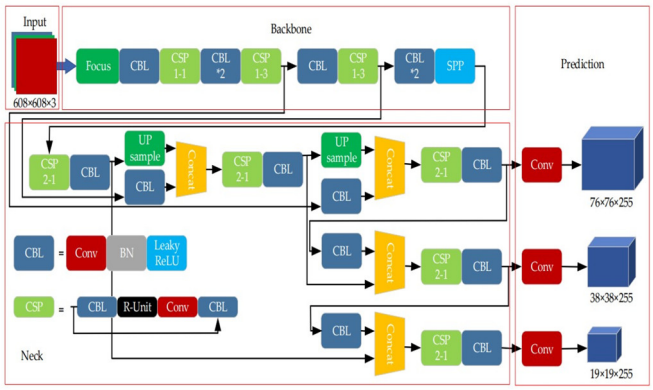
Nguyên lý và kiến trúc mạng YOLO

YOLO là một thuật toán tiên tiến ứng dụng trong phát hiện đối tượng dựa trên mô hình mạng nơ-ron tích chập (CNN). YOLOv5 được phát triển vào năm 2020 từ các phiên bản YOLOv1 - YOLOv4. Với nhiều cải tiến về tốc độ của thuật toán R-CNN, YOLOv5 được xem là một trong những lựa chọn hàng đầu cho mô hình phân loại đối tượng theo thời gian thực. Quá trình tính toán của YOLOv5 được thực hiện như sau [21]: ảnh đầu vào được chia thành M×M ô. Có tổng cộng M×M×K hộp giới hạn (bounding box) được tạo thành từ M×M ô. Mỗi ô sẽ được dự đoán thông tin về đối tượng và đặt hộp giới hạn. Mỗi hộp giới hạn chứa 5 thông số: Tọa độ tâm (x, y), chiều dài và chiều rộng (w, h), cuối cùng là độ tin cậy. M×M ô tiếp tục dự đoán về xác suất tồn tại của đối tượng. Hai thông số độ tin cậy của hộp giới hạn và xác suất tồn tại được nhân với nhau để tính điểm cho mỗi hộp dự đoán (prediction box). Hộp dự đoán được loại bỏ các pixel không ở vị trí cực đại toàn cục (IoU) để phát hiện vật thể cuối cùng. Nguyên lý làm việc của YOLOv5 có thể thấy ở hình 1.



Hình 1. Nguyên lý làm việc của YOLO

Mạng YOLOv5 sẽ thông qua ba thành phần chính để thực hiện nhận diện đối tượng. Ảnh đầu vào lần lượt thông qua các khối xử lý: Backbone: là một mạng nơ-ron sâu dùng để chia hình ảnh thành các ô và trích xuất các đặc tính của ảnh. Bao gồm cấu trúc Focus và CPS; Neck: Thu thập các đặc tính tại những giai đoạn khác nhau vùng với Backbone đưa vào các lớp dự đoán; Prediction: Tìm những vùng có khả năng chứa các đối tượng và tạo hộp giới hạn xung quanh các vùng có tiềm năng. Kiến trúc mạng của YOLOv5 có thể thấy ở hình 2 [21].



Hình 2. Kiến trúc mạng của YOLOv5

Hàm loss

Hàm loss trong YOLOv5 được sử dụng để đánh giá các thông tin về độ chính xác của đối tượng được phân loại [22]. Hàm loss là tổng của tiêu chí đánh giá sau:

Hàm loss độ tin cậy (Confidence loss): Hàm được tính toán trên toàn bộ ảnh M×M ô và mỗi ô bao gồm K hộp giới hạn.

$$L_{confidence} = \sum_{i=0}^{M^2}\sum_{j=0}^{K} \mathbb{R}_{ij}^{obj}\left(C_i - \hat{C}_i\right)^2 + \varphi_{coord}\sum_{i=0}^{M^2}\sum_{j=0}^{K} \mathbb{R}_{ij}^{obj}\left(C_i - \hat{C}_i\right)^2 \quad (1)$$

Trong đó C là điểm của độ tin cậy và $\hat{C}$ là độ đo đánh giá các mô hình nhận diện thực thể (IoU). $\mathbb{R}_{ij}^{obj} = 1$ khi có vật thể trong ô, ngược lại $\mathbb{R}_{ij}^{obj} = 0$

Để tăng tính ổn định cho mô hình, hệ số φ_{coord} được thêm vào để điều chỉnh giá trị cho hàm loss. Hệ số được sử dụng cho nhiều phương trình khác nhau.

Hàm loss vị trí (Localization loss): Liên quan đến dự đoán vị trí của hộp giới hạn.

$$L_{localization} = \varphi_{coord}\sum_{i=0}^{M^2}\sum_{j=0}^{K} \mathbb{R}_{ij}^{obj}\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right] + \varphi_{coord}\sum_{i=0}^{M^2}\sum_{j=0}^{K}\left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2\right] \quad (2)$$

Trong đó (x, y) là vị trí dự đoán của hộp giới hạn và $(y_i, \hat{y}_i)$ là vị trí của hộp giới hạn của dữ liệu huấn luyện. Tương tự w, h lần lượt là chiều rộng và chiều cao dự đoán của hộp giới hạn trong khi $\hat{w}, \hat{h}$ là chiều rộng và chiều cao hộp giới hạn của quá trình dữ liệu huấn luyện.

Hàm loss phân loại (Classification loss): Dự đoán về độ chính xác khi xác định vật thể. Hàm không được sử dụng khi trong ô không có đối tượng.

$$L_{classification} = \sum_{i=0}^{M^2} \sum_{c \in classes} \mathbb{R}_{ij}^{obj} \left(p_i(c) - \hat{p}_i(c)\right)^2 \quad (3)$$

Trong đó. $\hat{p}_i(c)$ xác suất của đối tượng tại lớp c tương ứng.

Theo đó ta có Hàm loss của mỗi đối tượng được tính toán như sau:

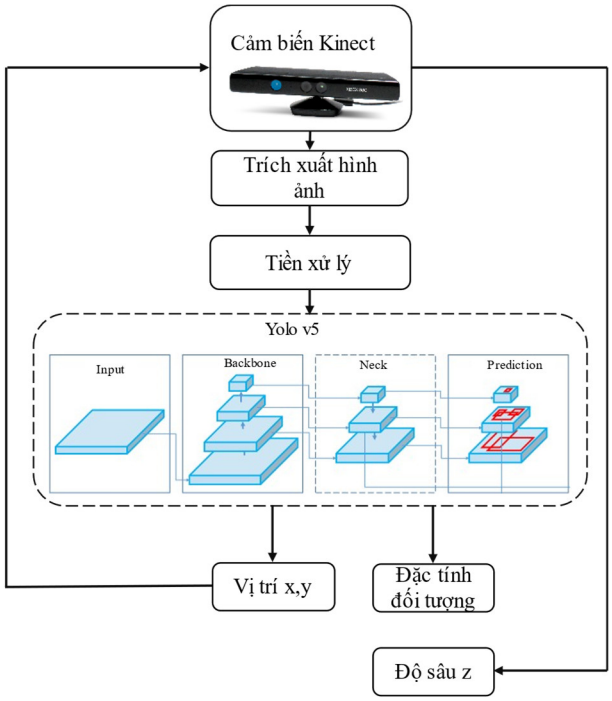
$$\begin{aligned} L &= L_{confidence} + L_{localization} + L_{classification} \\ &= \sum_{i=0}^{M^2} \sum_{j=0}^{K} \mathbb{R}_{ij}^{obj} \left(C_i - \hat{C}_i\right)^2 \\ &+ \varphi_{coord} \sum_{i=0}^{M^2} \sum_{j=0}^{K} \mathbb{R}_{ij}^{obj} \left(C_i - \hat{C}_i\right)^2 \\ &+ \varphi_{coord} \sum_{i=0}^{M^2} \sum_{j=0}^{K} \mathbb{R}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right] \\ &+ \varphi_{coord} \sum_{i=0}^{M^2} \sum_{j=0}^{K} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 \right. \\ &+ \left(\sqrt{h_i} \right. \\ &\left. \left. - \sqrt{\hat{h}_i}\right)^2\right] \sum_{i=0}^{M^2} \sum_{c \in classes} \mathbb{R}_{ij}^{obj} \left(p_i(c) - \hat{p}_i(c)\right)^2 \end{aligned} \quad (4)$$

3. XÂY DỰNG MÔ HÌNH THỰC NGHIỆM

Sơ đồ hệ thống được thể hiện ở hình 3. Trong đó hình ảnh RGB được thu nhận từ cảm biến Kinect sau đó được xử lý và làm đầu vào cho mô hình YOLO. Thông tin sau khi YOLO xử lý bao gồm thông tin về đặc tính và vị trí của đối tượng. Thông tin vị trí của vật được phản hồi lại cảm biến Kinect từ đó trích xuất ra được thông tin về chiều sâu của vật.

Quá trình trích xuất hình ảnh

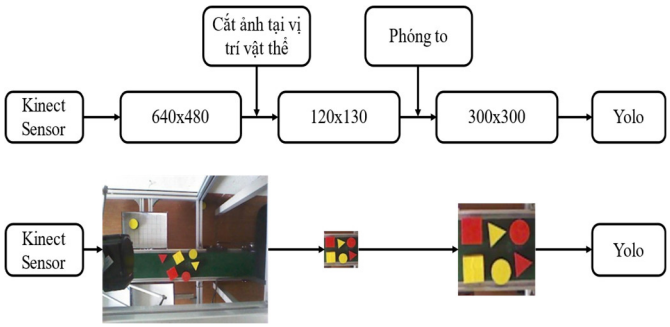
Kinect được hãng Microsoft thiết kế nhằm mục đích phục phụ người dùng tăng trải nghiệm thực tế cho các trò chơi. Tuy nhiên do nhiều ưu điểm về khả năng đo độ sâu và giá thành, sự ra đời của Kinect đã mở ra nhiều ứng dụng mới trong ngành khoa học thị giác máy tính. Tuy đã được đóng gói cả phần cứng và phần mềm nhưng nhiều thư viện đã được ra đời giúp Kinect thực hiện các nhiệm vụ khác. Nhóm tác giả sử dụng thư viện Microsoft Kinect SDK và OpenNI để thực hiện trích xuất ảnh làm đầu vào cho YOLO và lấy dữ liệu độ sâu.



Hình 3. Sơ đồ khối hệ thống

Quá trình tiền xử lý

Tiền xử lý là một quá trình quan trọng trong việc xử lý hình ảnh bao gồm việc xây dựng cơ sở dữ liệu cung cấp cho các thuật toán học sâu. Kích thước ảnh càng lớn thì độ chính xác càng cao, đồng thời dẫn đến thời gian xử lý lâu hơn. Kích thước ảnh mặc định của YOLOv5 là 640 pixel, tuy nhiên với độ phân giải này thời gian của hệ thống không được đảm bảo. Nhóm tác giả thực hiện thực quá trình xử lý để đưa kích thước ảnh chứa đối tượng ở 300 pixel. Sơ đồ xử lý được thể hiện ở hình 4.



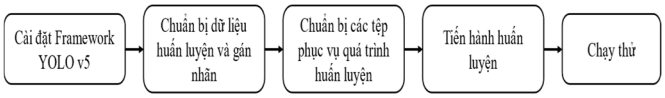
Hình 4. Sơ đồ quá trình tiền xử lý ảnh cho YOLO

Quá trình huấn luyện dữ liệu

Để huấn luyện và kiểm thử mô hình, tác giả sử dụng bộ dữ liệu bao gồm 200 ảnh cho bài toán phân loại vật thể. Mô hình được huấn luyện bằng Google Colab. Dịch vụ này cho phép truy cập vào máy tính ảo mạnh mẽ để tiết kiệm thời gian huấn luyện mô hình. Nhóm tác giả sử dụng tài liệu do Roboflow.ai phát triển hỗ trợ cho YOLOv5 [14] và sử dụng các trọng số COCO được đào tạo trước đó.

Quá trình huấn luyện dữ liệu được thể hiện qua hình 5. Sau khi cài đặt Framework của YOLOv5, các dữ liệu ảnh được

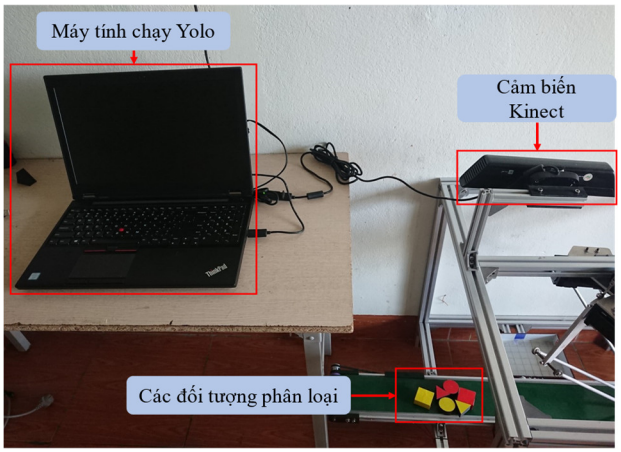
thu tập từ nhiều môi trường khác nhau sau đó các đối tượng được khoanh vùng và gán nhãn đặc tính. Sau khi thực hiện gán nhãn, các tệp thực thi được sinh ra như YOLO.data, train.txt,…. Có thể tiến hành thực hiện huấn luyện và thử nghiệm hệ thống.



Hình 5. Quá trình huấn luyện và thử nghiệm hệ thống YOLOv5

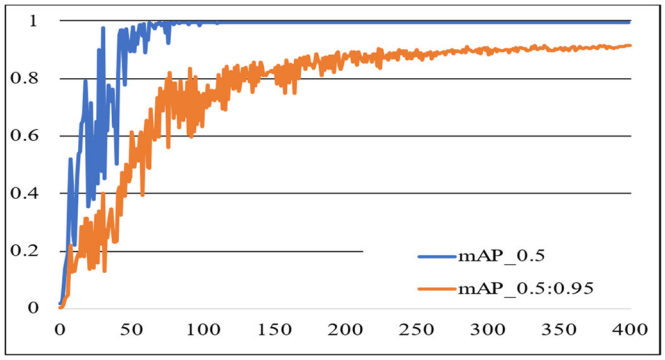
4. KẾT QUẢ PHÂN LOẠI ĐỐI TƯỢNG

Sau khi thực hiện huấn luyện, tác giả thực hiện lấy dữ liệu hình ảnh thời gian thực qua Kinect sensor và đưa ra các thử nghiệm. Phần cứng cấu hình thiết bị thực hiện mô hình sử dụng CPU có cấu hình: XEON E3 -1505, RAM 16GB, Quadro M2000M để huấn luyện YOLO. Hệ thống thực nghiệm được thể hiện ở hình 6.



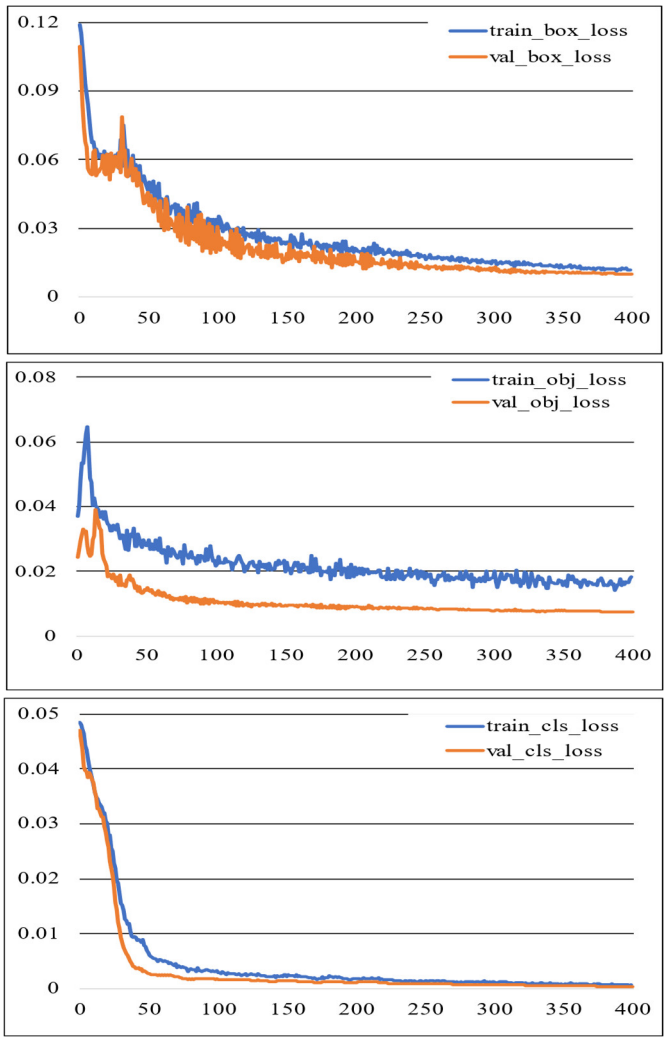
Hình 6. Hệ thống phần cứng thực nghiệm

Độ chính xác trung bình của các lớp trong trường hợp IoU là 0,5 (mAP_0.5) và trường hợp IoU là 0,5 đến 0,95 (mAP_0.5:0.95) được thể hiện trong hình 7. Kết quả cho thấy độ chính xác của mô hình được cải thiện đáng kể khi chỉ số mAP_0.5 đạt 90% sau 300 lần huấn luyện và mAP_0.5:0.95 đạt 98% sau 100 lần huấn luyện.



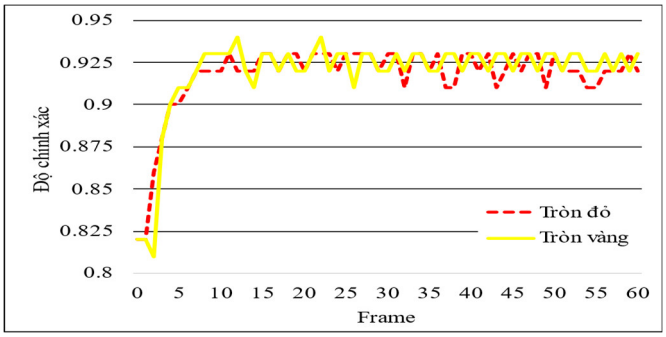
Hình 7. Chỉ số độ chính xác trung bình qua quá trình huấn luyện

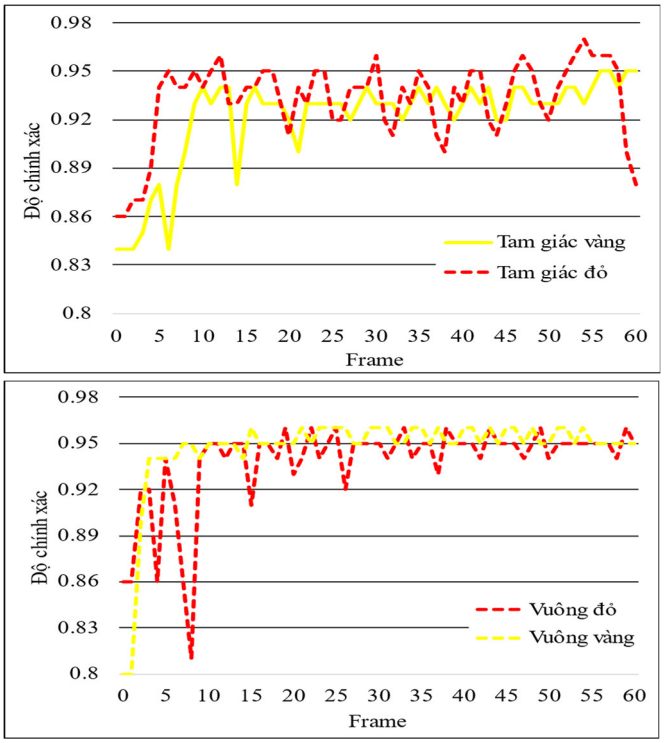
Độ hội tụ trong quá trình huấn luyện và kiểm thử của hàm loss được trình bày ở hình 8. Chỉ số box loss thể hiện mức độ phát hiện được vị trí tâm của giới hạn bao phủ đối tượng; object loss biểu diễn cho xác suất mà đối tượng tồn tại trong khu vực đề xuất; class loss đưa ra mức độ dự đoán đúng lớp của một đối tượng nhất định. Các chỉ số box loss, object loss và class loss đều có xu hướng giảm mạnh sau đó ở mức ổn định. Có thể thấy chất lượng thuật tán phát hiện vật được cải thiện đáng kể và đạt độ chính xác cao sau 400 lần huấn luyện.



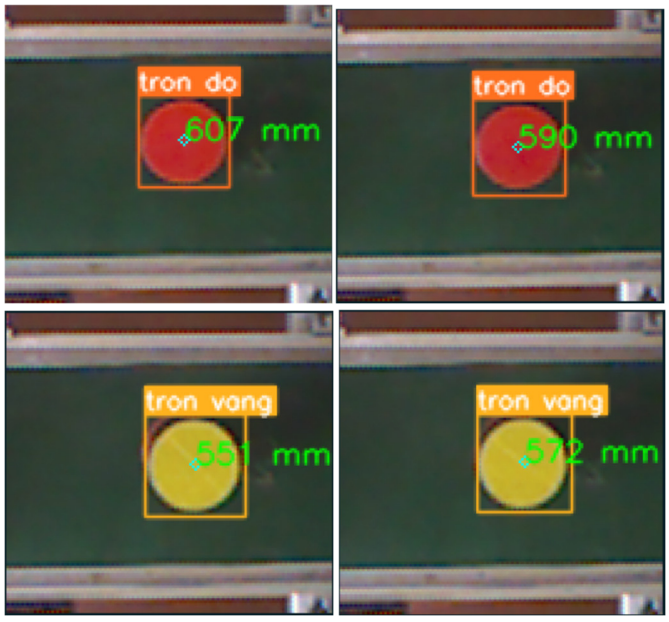
Hình 8. Các chỉ số đặc tính của quá trình huấn luyện (train) và thử nghiệm (val) của tập dữ liệu đầu vào lượt là box loss, object loss và class loss

Độ chính xác khi nhận dạng các loại phôi khác nhau có thể thấy ở hình 9. Trong khu vực ổn định (khi vật thể ở giữa khung hình) độ chính xác của các vật thể trung bình đạt 92%. Các phôi đi qua vùng nhìn thấy của camera là 60 khung hình. Video thời gian thực có giá trị khung hình trên giây (FPS) trung bình là 33.

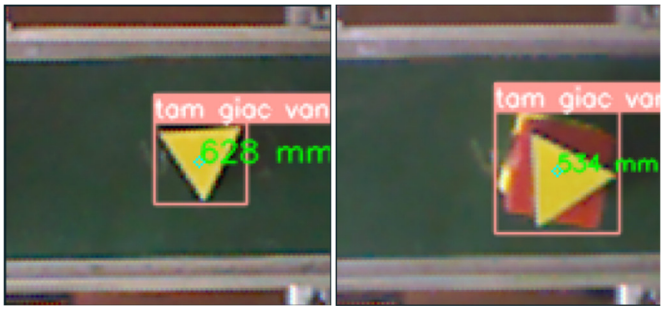


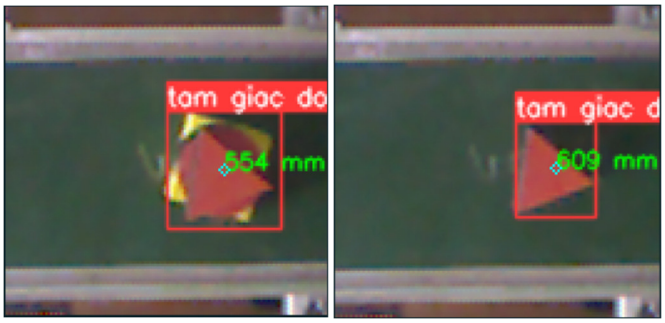


Hình 9. Độ chính xác khi tiến hành thực nghiệm cho phôi di chuyển trên băng chuyền (a) phôi tròn, phôi tam giác, phôi vuông

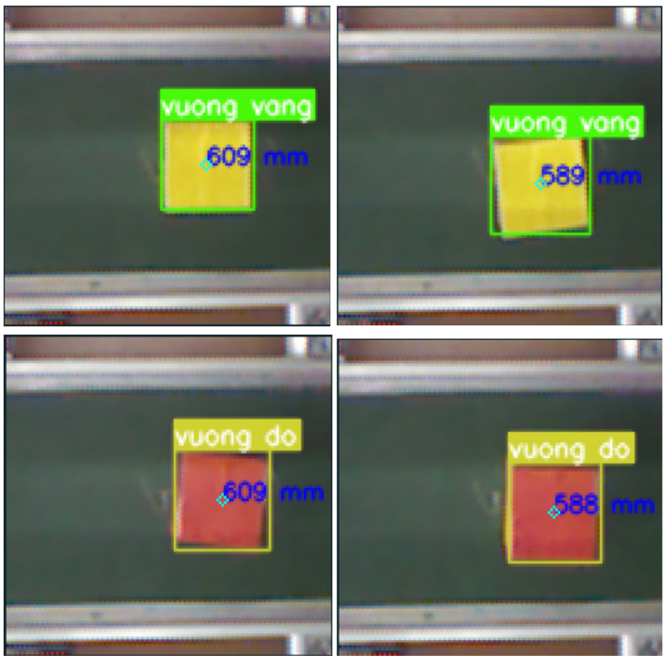


Hình 10. Nhận dạng vật thể hình tròn





Hình 11. Nhận dạng vật thể hình tam giác



Hình 12. Nhận dạng vật thể hình vuông

Kết quả trong hình 10, 11, 12 có thể thấy, hệ thống đã phát hiện đúng các đặc tính của phôi như tròn đỏ, tròn vàng ở hình 10, tam giác đỏ và tam giác vàng ở hình 11, vuông đỏ và vuông vàng ở hình 12. Có thể thấy hệ thống đã hoạt động tốt khi đã xác định chính xác đặc tính các đối tượng và lấy được dữ liệu độ sâu của mỗi đối tượng qua Kinect.

5. KẾT LUẬN

Bài báo đã thực hiện trích xuất các tính chất của phôi đồng thời với thông tin độ sâu sử dụng kết hợp Kinect và YOLO. Việc trích xuất thông tin độ sâu ở Kinect và phân loại đặc tính của phôi dựa trên YOLO đã được thực hiện thành công trên thời gian thực. Kết quả thực nghiệm phản ánh sự hiệu quả và chính xác của mô hình. Có thể thấy đây là một hệ thống phân loại đối tượng có hiệu suất cao và chi phí thấp cho các dây chuyền phân loại sản phẩm. Trong tương lai mô hình có thể cải thiện về thời gian xử lý và độ chính xác tọa độ của vật.

LỜI CẢM ƠN

Bài báo này được tài trợ bởi Trường Đại học Công nghiệp Hà Nội với đề tài có mã số 02-2022-RD/HĐ-ĐHCN.

TÀI LIỆU THAM KHẢO

[1]. S. Liang, Y. Wang, Y. Lu, et al., 2019. *Cognitive SSD: A Deep Learning Engine for In-Storage Data Retrieval*. in USENIX Annual Technical Conference, pp. 395-410.

[2]. Z.Q. Zhao, P. Zheng, S.T. Xu, et al., 2019. *Object detection with deep learning: A review*. IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212-3232.

[3]. A. R. Pathak, M. Pandey, S. Rautaray, 2018. *Application of deep learning for object detection*. Procedia computer science, vol. 132, pp. 1706-1717.

[4]. Y. Xiao, Z. Tian, J. Yu, et al., 2020. *A review of object detection based on deep learning*. Multimedia Tools and Applications, vol. 79, pp. 23729-23791.

[5]. J.A. Kim, J.Y. Sung, S.H. Park, 2020. *Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition.* in 2020 IEEE international conference on consumer electronics-Asia (ICCE-Asia), pp. 1-4.

[6]. D. R. Chowdhury, P. Garg, V. N. More, 2019. *Pedestrian intention detection using Faster RCNN and SSD*. in Advances in Computing and Data Sciences: Third International Conference, ICACDS 2019, Ghaziabad, India, Revised Selected Papers, Part II 3, 2019, pp. 431-439.

[7]. P. Purkait, C. Zhao, C. Zach, 2017. *SPP-Net: Deep absolute pose regression with synthetic views*. arXiv preprint arXiv:1712.03452.

[8]. X. Wang, A. Shrivastava, A. Gupta, 2017. *A-fast-rcnn: Hard positive generation via adversary for object detection*. in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2606-2615.

[9]. B. Cheng, Y. Wei, H. Shi, et al., 2018. *Revisiting RCNN: On awakening the classification power of faster RCNN.* in Proceedings of the European conference on computer vision (ECCV), pp. 453-468.

[10]. S. Liang, Y. Wang, Y. Lu, et al., 2019. *Cognitive SSD: A Deep Learning Engine for In-Storage Data Retrieval*. in USENIX Annual Technical Conference, pp. 395-410.

[11]. P. Jiang, D. Ergu, F. Liu, et al., 2022. *A Review of Yolo algorithm developments*. Procedia Computer Science, vol. 199, pp. 1066-1073.

[12]. K. K. Biswas, S. K. Basu, 2011. *Gesture recognition using microsoft Kinect®*. in The 5th international conference on automation, robotics and applications, pp. 100-103.

[13]. A. Oliver, S. Kang, B. C. Wünsche, et al., 2012. *Using the Kinect as a navigation sensor for mobile robotics*. in Proceedings of the 27th conference on image and vision computing New Zealand, pp. 509-514.

[14]. R. Held, A. Gupta, B. Curless, et al., 2012. *3D puppetry: a Kinect-based interface for 3D animation*. in UIST, pp. 423-434.

[15]. J. Mohan, S. Ashok, 2018. *A Reliable Robot Workspace Monitoring System Using Kinect v2*. in 2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE), pp. 686-689.

[16]. Y. Wang, G. Song, G. Qiao, et al., 2013. *Wheeled robot control based on gesture recognition using the Kinect sensor*. in 2013 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 378-383.

[17]. D. H. Dos Reis, D. Welfer, M. A. De Souza Leite Cuadros, et al., 2019. *Mobile robot navigation using an object recognition software with RGBD images and the YOLO algorithm*. Applied Artificial Intelligence, vol. 33, no. 14, pp. 1290-1305.

[18]. M. Ju, Y. Choi, J. Seo, et al., 2018. *A Kinect-based segmentation of touching-pigs for real-time monitoring*. Sensors, vol. 18, no. 6, pp. 1746.

[19]. Y.H. Lee, Y. Kim, 2020. *Comparison of CNN and YOLO for Object Detection*. Journal of the semiconductor & display technology, vol. 19, no. 1, pp. 85-92.

[20]. E. Prasetyo, N. Suciati, C. Fatichah, 2020. *A comparison of YOLO and mask R-CNN for segmenting head and tail of fish*. in 2020 4th International Conference on Informatics and Computational Sciences (ICICoS), pp. 1-6.

[21]. Y. Zhang, Z. Guo, J. Wu, et al., 2022. *Real-Time Vehicle Detection Based on Improved YOLO v5*. Sustainability, vol. 14, no. 19, pp. 12274.

[22]. T. Patil, S. Pandey, K. Visrani, 2020. *A review on basic deep learning technologies and applications*. Data Science and Intelligent Applications: Proceedings of ICDSIA 2020, pp. 565-573.

AUTHORS INFORMATION

Bui Thanh Lam, Nguyen Duc Quang, Nguyen Van Truong, Phan Dinh Hieu, Vu Tuan Anh, Luu Vu Hai

Hanoi University of Industry, Vietnam