DOI: https://doi.org/10.62985/j.huit_ojs.vol26.no2E.411

DEEP LEARNING-BASED VISION SYSTEM FOR AUTOMATED NIPA PALM FLESH SEPARATOR USING YOLOv11 AND OPC-UA INTEGRATION

Song-Toan Tran[1,2,*], Truc-Ly Le[1], Phuc-Hau Nguyen[1], Thien-Nhan Mai[1], Quoc-Kien Lam[1]

[1] *College of Engineering and Technology, Tra Vinh University, Vietnam*

[2] *AI center, Tra Vinh University, Vietnam*

*Email: *tstoan1512@tvu.edu.vn*

Received: 6 January 2026; Revised: 16 March 2026; Accepted: 10 April 2026

ABSTRACT

In line with the trend of sustainable development and the efficient exploitation of natural resources, the demand for processed products derived from agricultural produce is steadily increasing. With advancements in technology, the application of machinery and automation equipment in production has become an essential requirement to enhance productivity and product quality. In this study, the application of the YOLOv11 deep learning model combined with a Mitsubishi FX5U Programmable Logic Controller (PLC) via an OPC Unified Architecture (OPC-UA) server is proposed for a nipa palm flesh separation machine. The image of the nipa palm flesh is fed through the YOLOv11 model for analysis and quality assessment (immature, acceptable, mature). Based on the model's results, the corresponding signal is transmitted to the PLC to control the actuators that perform the separation of the coconut flesh. An OPC-UA intermediate server is used to enable two-way data exchange between the Python program on the computer and the PLC, while also facilitating easy expansion and integration with a monitoring system. The training results of the nipa palm flesh recognition model using YOLOv11 achieved a mAP50 of up to 97.3%. A multi-threading algorithm in Python allows for the simultaneous processing of coconut flesh quality recognition and data exchange with the PLC through the OPC-UA server. The research findings contribute to the construction and development of industrial systems integrated with image processing technology.

Keywords: Nipa palm, PLC, OPC-UA server, YOLOv11.

1. INTRODUCTION

In the trend of sustainable development and increasing value for agricultural products, the automation of processing stages plays a vital role in enhancing productivity, ensuring product quality, and reducing dependence on manual labor. The agricultural processing industry, in general, faces numerous challenges, including high requirements for product uniformity, stable quality control, and the ability to scale production. In this context, the application of emerging technologies such as machine vision, deep learning, and automated control systems is becoming an inevitable trend in smart production lines [1].

Nipa palm is a high-value product used in food and beverages [2]. Currently, the classification of fruit quality and the extraction of the flesh are performed almost exclusively through manual methods, which rely on the subjective sensory experience of workers. This dependency not only precludes standardized quality control but also severely limits industrial-scale productivity. A critical research gap exists in the quantification of the fruit's physical and mechanical properties; current literature lacks defined parameters for maturity stages and fails to offer automated solutions capable of adapting to the non-uniform morphology and complex lignified shell of the Nipa fruit. Therefore, investigating the integration of computer vision with precision mechanical extraction systems is crucial to bridging the automation gap in processing specialized agricultural products and optimizing the overall value chain.

In recent years, the integration of image processing with deep neural networks has demonstrated significant efficacy in agricultural recognition tasks, often achieving high accuracy rates and surpassing traditional hand-crafted feature methods. Among these frameworks, the YOLO (You Only Look Once) family has emerged as a prominent solution due to its superior balance between high precision and real-time processing speeds. Successive iterations of YOLO have been extensively deployed for surface anomaly detection and automated quality inspection across various products [3]-[7]. Consequently, implementing YOLO for nipa palm meat quality recognition not only streamlines the evaluative process but also establishes a robust foundation for seamless integration with industrial control systems and smart production lines.

In parallel with the advancement of computer vision, Programmable Logic Controller (PLC) maintains a central role in industrial automation systems due to their stability, high reliability, and ability to operate in harsh environments. PLC are widely used to manage actuators, ensuring system synchronization and safety. However, data exchange between computer-based image processing applications and traditional PLC faces limitations without a suitable communication architecture. The OPC Unified Architecture (OPC-UA) protocol was developed to address this need, allowing platform-independent, bi-directional data exchange and easy scalability, while supporting integration with industrial monitoring and data acquisition systems. Recent studies have demonstrated the great potential of integrating deep learning, PLC, and OPC-UA in smart manufacturing systems [8]-[11]. Using OPC-UA as an intermediary layer decouples the intelligence processing from the real-time control, thereby increasing system flexibility and scalability. Nevertheless, in the field of nipa palm processing research applying both deep learning and PLC control remains very limited.

This study aims to deliver an integrated architecture suitable for industrial conditions, moving beyond simple model development. The study contributes to the broader application of deep learning in agriculture and offers an effective approach for synthesizing machine intelligence with conventional industrial automation to create smarter production systems. The primary contributions of this research are summarized as follows:

- The development of Nipa-detect based on the YOLOv11n model for the automated recognition and quality grading of nipa palm flesh.

- The implementation of an integrated framework bridging computer vision and industrial PLCs through an OPC-UA server for seamless data exchange.

- The design of a multi-threading algorithm aimed at optimizing processing latency to satisfy the stringent requirements of real-time industrial operation.

2. METHODS

2.1. System overview

The automated nipa palm flesh separation system is engineered based on an integrated architecture, combining deep learning-based image processing with PLC-based industrial control. This design ensures both high accuracy in quality classification and robust reliability in operational performance. The system's block diagram, highlighting the primary functional modules vision processing, control, and actuation is illustrated in Fig. 1.

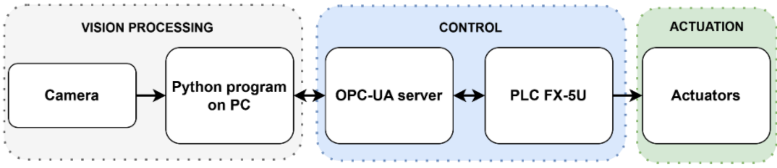


Fig. 1. The system overview

In the vision processing stage, a camera is strategically mounted to capture real-time images of the nipa palm flesh. The acquired image data is transmitted to a host computer running a Python application. The custom-trained Nipa-detect deep learning model within this application performs recognition and classifies the flesh quality into three distinct categories: young (immature), ready (acceptable), and old (mature). The Mitsubishi FX5U PLC serves as the central processing unit. It executes the control logic based on data received from the vision processing layer. The classification results are converted into corresponding control signals and transmitted to the PLC via the intermediary OPC-UA server. The PLC commands the respective actuators to sort and separate the nipa palm flesh, comprises the electromechanical components responsible for physically manipulating and separating the flesh based on the PLC's decision.

The core of the system is a rotary indexing table equipped with four pneumatic gripper units. This mechanism facilitates high automation by sequentially transporting the fruit halves at every 90 degree rotation through four functional stations: loading, splitting, image recognition, and shell ejection. Key mechanical components at the separation station include: camera, workpiece holder, pressing unit and proximity sensor, as illustrated in Fig. 2. The camera positioned at an optimized angle to capture a complete view of the nipa palm flesh secured by the workpiece holder, providing the input image for the recognition model.Workpiece holder specifically designed with two curved jaws and sharp spikes that penetrate the shell, ensuring the nipa palm halves are firmly fixed during the splitting and separation processes. Its opening and closing action is controlled by a pneumatic cylinder. Pressing unit consists of a rectangular metal plate attached to the piston rods of two pneumatic cylinders. This unit generates a direct, downward force to cleanly separate the classified acceptable flesh from its shell. Proximity sensor strategically placed at each station to verify the correct positioning and presence of the coconut fruit halves.

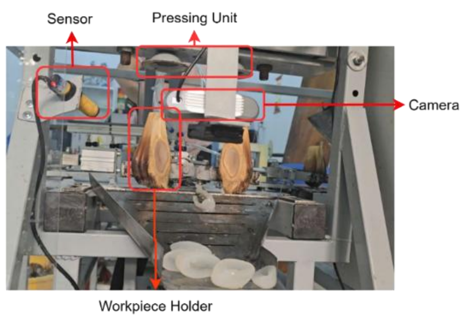


Fig. 2. The nipa palm flesh separation mechanism

2.2. Data collection and model training

2.2.1. YOLOv11

YOLOv11, the iteration in the YOLO series by Ultralytics, represents a state-of-the-art real-time object detection tool that redefines the possibilities with superior accuracy, speed, and efficiency [12]. Building upon the impressive advancements of preceding YOLO versions, YOLOv11 introduces significant enhancements in its architecture and training methodologies, establishing it as a highly versatile choice for a wide array of computer vision tasks. The YOLO framework revolutionized the field of object detection by introducing a unified neural network architecture capable of simultaneously addressing two tasks: bounding box regression and object classification. This integrated approach marked a distinct departure from traditional two-stage detection methods, allowing for end-to-end training due to its fully differentiable design.

Several architectural innovations are introduced by YOLOv11 to enhance both feature extraction capability and overall model accuracy, while the commitment to superior performance in real-time object detection is maintained. Key enhancements include the integration of the C3K2 (Cross Stage Partial with kernel size 2) block, the SPPF (Spatial Pyramid Pooling - Fast) module, and the C2PSA (Convolutional block with Parallel Spatial Attention) component. Specifically, the C3K2 block replaces the C2f block found in preceding YOLO models, contributing significantly to improved computational efficiency and faster processing speeds. Inheriting the versatility of its predecessors, YOLOv11 is designed to support a wide range of computer vision tasks, including but not limited to object detection, classification, image segmentation, keypoint detection, and Oriented Bounding Box regression. Fig. 3 illustrates the new and core functional blocks of the YOLOv11 model architecture.

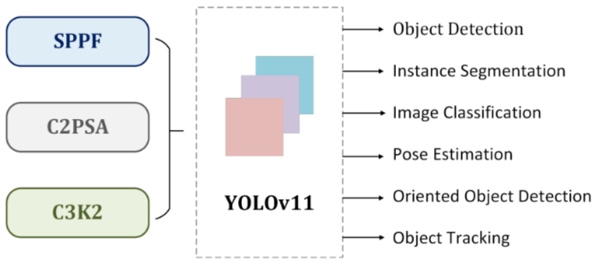


Fig. 3. Key modules of the YOLOv11 architecture [12]

Various versions of different sizes, tailored for diverse tasks, are offered by the YOLOv11 model. Within the scope of this research, the YOLOv11-nano version was selected for the object detection task to analyze nipa palm flesh images, ensuring that efficient operation is maintained in a real-time environment.

2.2.2 Dataset and training process

In deep learning-based object recognition tasks, the quality and representativeness of the dataset play a critical role in determining model performance. Consequently, in this study, the image dataset was meticulously constructed to closely reflect the actual operating conditions of the automated nipa palm flesh separation machine. The captured images simulate the equipment's operational state, wherein the nipa palm fruit is cleaved longitudinally and positioned at the observation point, mirroring the real separation process.

Table 1. The data used for training process

Categories	Total Instances	Training (80%)	Validation (20%)
Young	609	487	122
Ready	880	704	176
Old	456	364	92
Total	1945	1555	390

The dataset, meticulously constructed to reflect the actual operational conditions of the automated separation machine, comprises 1160 images containing a total of 1945 annotated instances of nipa palm flesh. The cases within these images are categorized into three distinct quality classes: young, ready, and old, which correspond to the stages of immaturity, acceptability condition, and maturity, respectively. Details regarding the data used in this study are presented in Table I. The entire annotated dataset was partitioned into an 8:2 ratio for training and validation sets to objectively evaluate the model's performance. The “young” classification is characterized by thin, highly translucent, and soft flesh due to incomplete formation. Conversely, the “ready” class possesses appropriate thickness and pliability, meeting the requirements for processing. Finally, the “old” class is visually distinguishable by an opaque white color and a firmer texture, making it unsuitable for the intended purpose. Representative samples illustrating these quality classes are provided in Fig. 4.

(a) *(b)* *(c)*

Fig. 4. Illustrations of nipa palm flesh quality classes: (a) young (immature) , (b) ready (acceptable) and (c) old (mature).

2.2.3. OPC Unified Architecture (OPC-UA)

The DeviceXPlorer OPC Server 7 is implemented as the central OPC UA Server to facilitate seamless data orchestration between the computer vision system and the Mitsubishi FX5U PLC. In this study, both the Python-based processing environment and the PLC are configured as OPC UA Clients, establishing concurrent connections to the unified server instance. The information space is structured via a hierarchical node model, where each node is directly mapped to specific registers and control bits within the FX5U PLC using the Mitsubishi MELSEC Ethernet protocol. Each data element is uniquely identified by a namespace and a Node ID, ensuring synchronized, multi-client access. The node mapping is shown in Table 2.

Table 2. The node mapping

Node name	PLC Address	Node ID
bit_connect	M100	ns=2; s=PLC_FX5U.bit_connect
bit_com_non	M101	ns=2; s=PLC_FX5U.bit_com_non
bit_com_dat	M102	ns=2; s=PLC_FX5U.bit_com_dat
bit_com_gia	M103	ns=2; s=PLC_FX5U.bit_com_gia
state_sensor	X12	ns=2; s=PLC_FX5U.state_sensor

Regarding system reliability, the core data exchange is anchored by a connectivity monitoring mechanism centered on the bit_connect node (Node ID: ns = 2; s = PLC_FX5U.bit_connect). This node operates as a continuous signal, enabling the integrated environment to scrutinize the communication bridge between the Python-based client, the OPC UA Server, and the PLC.

The transmission of intelligence from the vision model to the hardware actuators is facilitated through a specialized triad of classification nodes: *bit_com_non*, *bit_com_dat*, and *bit_com_gia*. These nodes serve as the digital representation of the YOLOv11 inference outcomes, specifically categorizing the nipa palm meat into three maturity stages: immature, acceptable, and mature. Once the neural network completes its analysis, the Python environment writes the resulting logic state to the OPC UA Server, allowing the PLC to asynchronously retrieve the classification data and trigger the appropriate sorting mechanisms.

Synchronization between the physical sensing layer and the computational vision layer is achieved via the state_sensor node (Node ID: ns = 2; s = PLC_FX5U.state_sensor). This node acts as a primary trigger gateway that aligns the timing of specimen detection with image capture sequences. When the hardware sensor identifies a specimen within the designated inspection zone, the PLC immediately updates this node's status on the server.

2.3. Control system for the nipa palm flesh separator

To ensure real-time performance and stable communication with the PLC, the proposed system employs a multi-threaded architecture. One thread is dedicated to image acquisition and quality recognition using the Nipa-detect model, while a separate thread is responsible for data exchange with the PLC via the OPC-UA server. **Fig. 5.** presents the system flowchart, highlighting the parallelized integration of the vision-based recognition and OPC-UA data exchange modules.

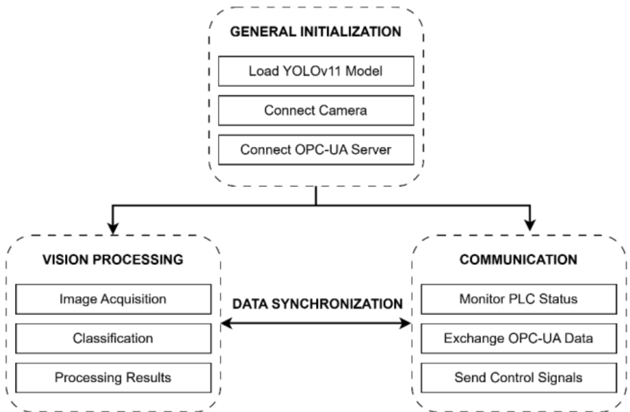


Fig. 5. Overall system flowchart, where the vision processing module and the OPC-UA communication module are implemented in parallel.

This decoupled design allows the image processing task and the industrial communication task to operate independently, consequently reducing communication latency and enhancing the system's overall responsiveness. The two threads are synchronized through shared state variables, which facilitate the reliable transmission of recognition results without compromising real-time PLC communication.

In this study, model training and operational deployment were executed on different platforms. The YOLOv11 model training phase was conducted via Google Colab utilizing an NVIDIA T4 GPU. For the deployment and inference phase, the system was tested on a local workstation to simulate standard operational conditions. This benchmark computer featured an Intel Core i5-1135G7 processor (11th Gen, 2.40 GHz) and 8 GB of RAM on a 64-bit Windows 10 Pro platform. Notably, hardware acceleration via GPU was bypassed during deployment, forcing the model to run strictly on the CPU to evaluate its efficiency under computational limitations.

3. RESULTS AND DISCUSSIONS

3.1. Nipa-detect model training results

The Nipa-detect model was trained on the Google Colab platform utilizing a T4 GPU. The training process was executed over 150 epochs to ensure stable model convergence. The post-training performance of the model was evaluated using standard metrics common in object detection tasks, including Precision, Recall, mAP@0.5, and mAP@0.5:0.95. These evaluation metrics are formally defined in Equations (1) through (4).

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$AP_i = \sum_n (Recall_n - Recall_{n-1}) \times Precision_n \tag{3}$$

$$mAP = \frac{1}{n}\sum_{i=1} AP_i \tag{4}$$

where TP represents the set of correctly predicted positive class objects, FP is the set of objects incorrectly predicted as positive, and FN is the set of objects incorrectly predicted as negative. AP_i denotes the average precision of the model for a specific object class. An AP_i value closer to 1 indicates higher model accuracy. N represents the number of object classes being recognized, which is N = 3 in this study. mAP@0.5 (mean Average Precision at 50%) is the average of the AP values calculated at an Intersection over Union (IoU) threshold of 50%. mAP@0.5:0.95 (mean Average Precision over 50% to 95%) is the average of the AP_i values calculated across various IoU thresholds, ranging from 50% to 95%.

Table 3. Model training results

Metrics	Precision (%)	Recall (%)	mAP@50 (%)	mAP@50:95 (%)
Results	93.3	94.7	97.3	83.7

The evaluation results on the validation set are presented in Table 3. The model achieved a precision of 93.3%, demonstrating reliable predictive capability regarding the quality of the nipa palm coconut flesh. The recall metric reached 94.7%, indicating that the model possesses good object detection capability and successfully minimizes instances of missed detections. The mAP@0.5 value of 97.3% further substantiates the model's high effectiveness in object detection and classification at the IoU = 50 threshold. Even when applying the more rigorous evaluation criterion, the mAP@0.5:0.95 score achieved 83.7%, showing that the model maintained good performance, thus demonstrating a relatively accurate object localization and classification capability under varying conditions.

Fig. 6 illustrates the convergence process of the Nipa-detect model during training. The loss functions on both the training and validation sets show a stable decreasing trend over the epochs. This indicates that the model continuously improves its capability to localize object bounding boxes and accurately reflects an increasingly precise ability to distinguish between the object classes.

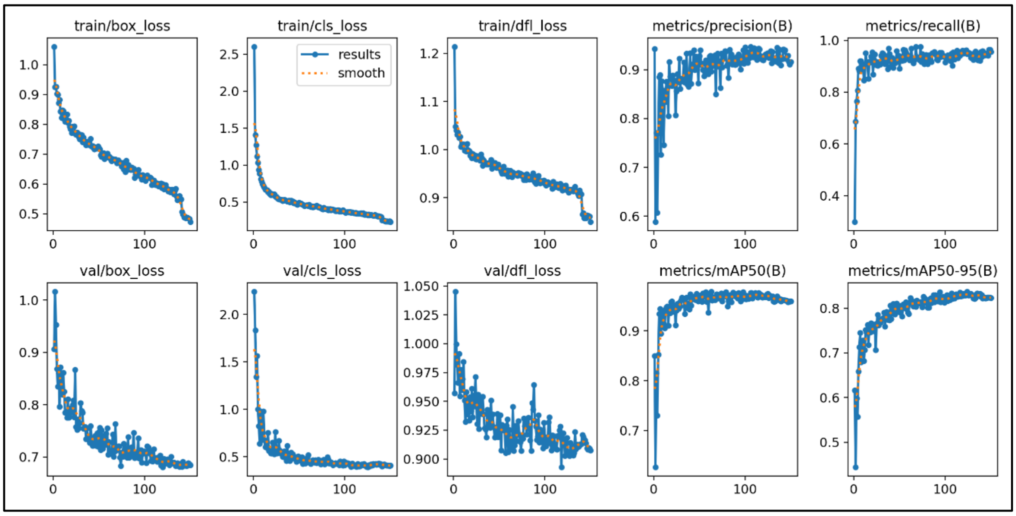
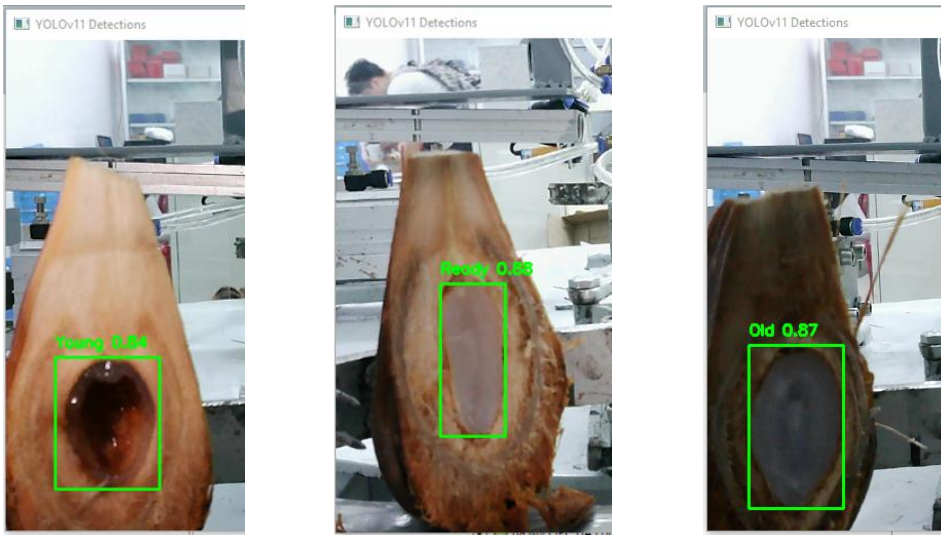


Fig. 6. The learning curves of the Nipa-detect model training process



(a) *(b)* *(c)*

Fig. 7. Results of the Nipa-detect model: (a) young, (b) ready and (c) old

Fig. 7 presents the results of the nipa palm flesh quality recognition and classification performed by the Nipa-detect model under simulated operational conditions. The model demonstrates the capability to accurately detect the location and correctly classify objects belonging to all three groups: young, ready, and old, even when variations in lighting conditions and sample shape exist. The determined bounding boxes are clearly defined and closely adhere to the object boundaries, indicating strong localization capability. This result validates the stability and applicability of the model within the automated nipa palm flesh sorting system.

3.2. The multi-threaded control algorithm

The experimental results presented in the Table 4 demonstrate a substantial enhancement in system performance through the implementation of the proposed multi-threaded architecture. Most notably, the total cycle time witnessed a drastic reduction of approximately 66.7%, falling from 105 ms in the single-threaded baseline to just 35 ms in the multi-threaded configuration.This significant improvement can be attributed to the asynchronous decoupling of the primary system tasks. In the single-threaded approach, the system is forced to execute tasks sequentially, creating a bottleneck where the communication delay (75 ms) is compounded by the inference latency (30 ms). Conversely, the multi-threaded strategy allows the YOLOv11-nano inference engine and the OPC-UA communication module to operate concurrently. By isolating these processes, the ommunication delay is reduced from a blocking 75 ms to a non-blocking 15 ms, as the data exchange no longer waits for the entire vision pipeline to reset.

Table 4. Comparison of system performance

Performance Metrics	Single-threaded	Proposed multi-threaded
Average Inference Latency	30 ms	20 ms
Communication Delay (OPC-UA)	75 ms	15 ms
Total Cycle time	105 ms	35 ms

Furthermore, the decrease in Average Inference Latency (from 30 ms to 20 ms) suggests that parallelizing the I/O operations allows the GPU/CPU to dedicate more consistent resources to the vision model without being interrupted by network handshaking protocols. With a total cycle time of 35 ms, the system comfortably exceeds the requirements for real-time responsiveness in industrial nipa palm processing. This ensures that the Mitsubishi FX5U PLC receives classification signals with high determinism, allowing the actuators to respond accurately to the physical flow of the production line without data congestion or signal lag.

3.3. Limitations of the study

Although the automated nipa palm flesh separation system achieved promising results, the current study still presents several limitations. The training dataset was primarily collected under laboratory conditions, leading to limited diversity in lighting, viewing angles, and sample morphology. This constraint may potentially impact the model's generalization capability when deployed in real-world industrial production environments. Additionally, the precision of the mechanical structure is not yet fully optimized, which affects the overall throughput and efficiency of the flesh pressing and separation process.

For future research, we will focus on enhancing the stability of the hardware system. The dataset will be expanded and diversified to improve the model's robustness and sustainability. Furthermore, advanced exploitation of the OPC-UA protocol for data analytics will be considered to fully evolve the system toward a smart, automated solution compatible with modern industrial automation lines.

4. CONCLUSIONS

This study proposes and implements an automated nipa palm flesh separation system, integrating the Nipa-detect deep learning model with a Mitsubishi FX5U PLC controlled via an OPC-UA server. A multi-threaded algorithm was employed to optimize the system's processing speed. Experimental results demonstrate that the model achieves high accuracy in classifying the quality of the nipa palm flesh, while the multi-threaded control architecture successfully reduces latency and enhances system stability. The application of OPC-UA further improves scalability and integration capabilities with supervisory systems in an industrial environment. This study opens an effective avenue for applying computer vision in conjunction with automated control within agricultural product processing systems.

Acknowledgement: This research was fully funded by Tra Vinh University (TVU) under grant contract number 38/2025/HĐ.HĐKH&ĐT - ĐHTV.

REFERENCES

[1] A. Hassoun, "Food sustainability 4.0: Harnessing fourth industrial revolution technologies for sustainable food systems," *Discover Food*, vol. 5, Art. no. 171, 2025, doi: https://doi.org/10.1007/s44187-025-00461-z..

[2] Mekong Agricultural Products, "Nipa palm: A precious gift from the Mekong Delta." [Online]. Available: https://nongsanmekong.com/dua-nuoc-qua-tang-quy-gia-cua-mien-tay/.

[3] C. M. Badgujar, A. Poulose, and H. Gan, "Agricultural object detection with You Only Look Once (YOLO) algorithm: A bibliometric and systematic literature review," *Computers and Electronics in Agriculture*, vol. 223, Art. no. 109090, 2024, doi: https://doi.org/10.1016/j.compag.2024.109090.

[4] T.-L. Le, P.-H. Nguyen, T.-N. Mai, Q.-K. Lam, and S.-T. Tran, "An embedded system with YOLOv5 for automated drug delivery system," *Journal of Technical Education Science*, 2026, https://doi.org/10.54644/jte.2025.1758.

[5] G. Peng, K. Wang, J. Ma, B. Cui, and D. Wang, "AGRI-YOLO: A lightweight model for corn weed detection with enhanced YOLO v11n," *Agriculture*, vol. 15, Art. no. 1971, 2025, doi: https://doi.org/10.3390/agriculture15181971.

[6] L. T. Ramos and A. D. Sappa, "A comprehensive analysis of YOLO architectures for tomato leaf disease identification," *Scientific Reports*, vol. 15, no. 1, Art. no. 26890, 2025, doi: https://doi.org/10.1038/s41598-025-11064-0.

[7] C. M. Badgujar, A. Poulose, and H. Gan, "Agricultural object detection with You Only Look Once (YOLO) algorithm: A bibliometric and systematic literature review," *Computers and Electronics in Agriculture*, vol. 223, Art. no. 109090, 2024, doi: https://doi.org/10.1016/j.compag.2024.109090.

[8] S. P. Muniraj and X. Xu, "An implementation of OPC UA for machine-to-machine communications in a smart factory," *Procedia Manufacturing*, vol. 53, pp. 52–58, 2021, doi: https://doi.org/10.1016/j.promfg.2021.06.009.

[9] E. Kirdan, F. Rezabek, N. Mülbauer, G. Carle, and M.-O. Pahl, "Real-time performance of OPC UA," *arXiv preprint* arXiv:2310.17052, 2023, doi: https://doi.org/10.48550/arXiv.2310.17052.

[10] Z. Luo *et al.*, "OPC UA-based smart manufacturing: System architecture, implementation, and execution," in *Proc. 5th Int. Conf. Enterprise Systems (ES)*, Beijing, China, 2017, pp. 281–286, doi: https://doi.org/10.1109/ES.2017.53.

[11] N. Holmes, L. Katavich, and X. Xu, "Retrofitting legacy systems for Industry 4.0 via OPC UA and distributed control," *Manufacturing Letters*, vol. 44, pp. 1337–1348, 2025, doi: https://doi.org/10.1016/j.mfglet.2025.06.153.

[12] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancements," *arXiv preprint* arXiv:2410.17725, 2024, doi: https://doi.org/10.48550/arXiv.2410.17725.