Cureus Journal of Engineering

Open Access Original Article

A Decision Tree-Based 6-Degree-of-Freedom Robotic Crop Sorting System

Philip J. Ezigbo [1] , Ekene S. Mbonu [2] , Jude K. Obichere [3] , Chijioke K. Joe-Uzuegbu [4] , Kelechi T. Ezirim [5] , Nchedochukwu C. Bede [6]

1. Mechatronics Engineering/Electronics and Computer Engineering, Federal University of Technology Owerri, Imo, NGA 2. Mechatronics Engineering/Control Systems Engineering, Federal University of Technology Owerri, Imo, NGA 3. Mechatronics Engineering/Power & Control Systems Engineering, Federal University of Technology Owerri, Imo, NGA 4. Electrical and Electronics Engineering/Computer Engineering, Federal University of Technology Owerri, Imo, NGA 5. Mechatronics Engineering/Mechanical Engineering, Federal University of Technology Owerri, Imo, NGA 6. Mechatronics Engineering, Federal University of Technology Owerri, Imo, NGA

Corresponding author: Philip J. Ezigbo, philip.ezigbo@futo.edu.ng

Received 04/30/2025
Review began 05/26/2025
Review ended 06/22/2025
Published 07/28/2025

DOI:
https://doi.org/10.7759/s44388-025-05656-9

Abstract

This research investigates the development of an automated crop sorting system designed to address the limitations of traditional manual sorting methods, which are often laborious, time-consuming, and prone to errors. The core of the proposed system is a decision tree-based classification model integrated with a 6-degree-of-freedom (DoF) robotic arm, providing an efficient and automated solution for sorting food products. The system's hardware setup includes a 30W USB camera for capturing images of the crops, a Jetson Nano microcomputer for processing the images and controlling the robotic arm, and six serial bus servo motors that actuate the arm. The camera captures real-time images, which are then preprocessed using OpenCV to extract color information in the HSV format. This preprocessed data is fed into the decision tree classifier to categorize the crops based on their color and ripeness. The classification results are then used to generate commands for the robotic arm, directing it to pick and place the crops into designated bins. The decision tree model, implemented using the scikit-learn library, was trained on a dataset of pepper images. The model utilizes Hue and Saturation values as primary features for classification, employing Gini impurity and entropy as criteria for splitting nodes. This approach enables the system to effectively categorize peppers into five distinct classes: green, yellow, red, orange, and unripe. The system's performance was evaluated through a series of tests, including camera testing and motor testing. The results demonstrated that the system achieved a high level of accuracy in pepper classification, with precision, recall, and F1 score values of 92%, 98%, and 94.9%, respectively. The integration of the decision tree model with the robotic arm provides a promising solution for automating crop sorting, offering advantages such as increased efficiency, reduced labor costs, and improved accuracy in post-harvest processing.

Categories: Mechatronics and Robotics, Data Science and Analytics for Engineering, Embedded Systems and IoT
Keywords: decision tree algorithm, 6-degree-of-freedom robotic arm, automated crop sorting, machine learning, hsv color classification

Introduction

In recent years, machine learning has emerged as a powerful tool in revolutionizing diverse domains such as healthcare, finance, marketing, and manufacturing [1-4]. Machine learning enables computer programs to improve performance by learning patterns from data, creating intelligent systems capable of operating with minimal human intervention and presenting opportunities for process automation and improved decision-making [5]. Among various machine learning approaches, the decision tree algorithm is a popular technique for building classification models that guarantee high performance and expressiveness [6]. This algorithm works by iteratively splitting data into subsets based on specific attributes at each node until a stopping criterion is met. The best attribute for each split is determined using criteria such as information gain, gain ratio, or Gini index, resulting in a tree-like structure where each node represents a feature, each branch indicates a decision rule, and each leaf shows an outcome [7,8]. The decision tree algorithm offers several significant advantages. First, its resulting structure allows users to understand and interpret the decision-making process easily. Second, it handles categorical and numerical data effectively without extensive preprocessing [7]. Additionally, decision trees can select the most biased feature while maintaining a comprehensible nature [8], making them versatile and efficient across many applications.

Decision tree algorithms have advanced research across numerous sectors. In healthcare, Podgorelec et al. [9] developed decision tree models for diagnosing thyroid disorders using patient data, while Pathak et al. [10] combined decision trees with fuzzy rule-based techniques to propose heart disease prediction models. Kumar et al. [11] improved face verification by combining C4.5 decision trees with similarity classifiers. Also in the agricultural sector, machine learning and deep learning have driven significant advancements, supporting crop disease detection, adaptive irrigation automation, yield prediction, and livestock management [12-15]. However, there remains a gap in smart automated crop product sorting, particularly for post-harvest classification. Agricultural products vary in size, shape, color, and quality during harvest due to different ripening stages, species variations, and growth rates. This variability necessitates automated selection and classification systems for quality control, accurate labeling, and consumer satisfaction [16].

However, traditional manual sorting methods relying on human hand-picking remain laborious, time-consuming, and prone to errors [17]. The need for sorting crop products after harvesting has always existed, and since this process involves repetitive identification and classification, automated solutions have continuously evolved from simple mechanical methods to advanced AI-driven systems. Early automated sorters developed by Chen and Sun [18] employed mechanical and basic optical methods, including sieves, rollers, conveyor belts, and RGB (Red, Green, Blue) filters. However, these systems could not detect ripeness gradients and required manual adjustments for each crop type. Later advancements incorporated RGB cameras to improve accuracy through color-based classification. Using thresholding and pattern recognition techniques, Blasco, Aleixos, and Moltó [19] designed automatic fruit grading systems based on external features like size, color, and defects. These systems still faced challenges with manual threshold tuning, rule-setting, and calibrations.

Advances in technology have introduced more sophisticated algorithms, such as Support Vector Machines (SVMs) and Neural Networks, into automated sorters. Cubero et al. [20] developed a system for automatic inspection and fruit sorting using these algorithms alongside hyperspectral and X-ray imaging, achieving up to 88% accuracy but encountering issues with real-time processing and system complexity. Zhang et al. [21] built upon this work, still employing SVMs and Artificial Neural Networks while improving imaging capabilities to achieve 94% accuracy in crop defect detection. More recent research by Kamilaris and Prenafeta-Boldú [22] explored Convolutional Neural Networks (CNNs) and Recurrent Neural Networks for agricultural applications, highlighting their use in fruit and vegetable sorting with pre-trained models. Similarly, Zhang and Su [23] utilized YOLOv5 on 1000 images for real-time apple localization in robotic picking. This CNN-based method provided accurate 2D pixel coordinates, which, when combined with an active laser vision system, allowed for 3D apple recognition and positioning. Contemporary automated sorting systems typically integrate deep learning models with robotics for fully automated sorting. Models such as ResNet-50, 3D-CNN, and MobileNet are commonly used alongside multimodal imaging technologies including X-ray, hyperspectral, and RGB-D approaches [21-23]. Despite the high accuracy offered by CNNs, their computational cost, complexity, and lack of transparency hinder widespread adoption in practical settings [24].

However, Decision tree algorithms present an alternative approach, offering simpler systems with lower costs, better transparency, and greater computational efficiency. This research addresses the gap in practical, transparent automated sorting systems by integrating decision tree classification models with servo-controlled robotic arms for food product classification, using pepper as a case study to demonstrate increased efficiency in post-harvest processing.

Materials And Methods

Proposed system design block diagram

The experimental setup for implementing this pepper classification algorithm in a robotic arm system involved a Jetson Nano microcomputer, six serial bus servo motors, a USB camera module, a power supply system, and a multi-function expansion board. The robotic arm system chassis is made of an anodized aluminum alloy 6061. It is a precipitation-hardened aluminum alloy that contains magnesium and silicon as its major alloying elements, and it is corrosion resistant. It is also lightweight and cost-effective. In the proposed system, camera sensors acquire visual data (i.e., images of pepper in real time). These images are then preprocessed using OpenCV to convert the data from RGB to HSV (Hue, Saturation, Value) format for color analysis. Each processed image is presented to the decision classifier model, which identifies and classifies the pepper based on color grades. Using the result of the classification, the system gives commands to the robotic arm on where the pepper should be placed. This setup ensures an automated pepper sorting with little human intervention. Figure *1* represents a block diagram of the proposed system design.

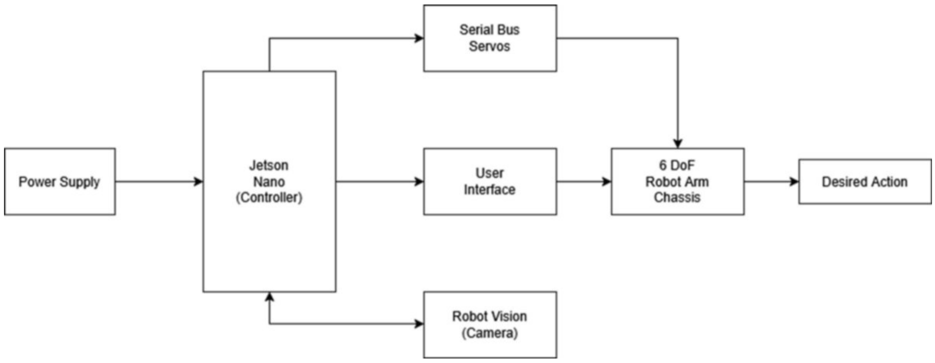

FIGURE 1: Block Diagram of the Automated Pepper Sorting System

DoF, degree of freedom

Block diagram description

Power Supply: This unit provides the power that is needed for the entire system from a 220 V AC source. The Jetson Nano Controller will be powered by a 5 V, 4 A power adapter. The adapter has short-circuit protection and overload protection.

4 gigabyte Jetson Nano microcomputer: This module comes with a high-performance Graphics Processing Unit (GPU) and quad-core ARM Cortex-A57 CPU to execute intricate algorithms and machine learning models. It has a microSD card slot for storage, a 40-pin expansion header, a micro-USB port, a gigabit ethernet port, a USB 3.0 port, 2 USB 2.0 ports, an HDMI output port, a USB-C for 5V power input, and finally, a mobile industry processor interface camera serial interface-2 (CSI-2) connector. It undertakes critical tasks such as image processing, object recognition, and gesture recognition, offering the necessary framework to operate a robotic arm's control and sorting algorithms. It has support for frameworks such as TensorFlow, PyTorch, and Scikit-learn. Though powerful in computation, it retains a compact form factor, making it easy to integrate into robotic systems without making it bulky. The Jetson Nano benefits from NVIDIA's extensive developer ecosystem, providing access to a wealth of tools, libraries, and community support for rapid development and iteration of robotic applications. When integrated with a high-resolution camera, the controller classified the pepper samples using image input from the camera and a decision tree classifier algorithm hosted on it. It also generated pulse width modulated signals for controlling the movement of the joints of the robotic arm for pick-and-place operations.

Serial Bus Servo Motors: The actuator system of this robotic sorter is based on serial bus servos because it offers a blend of precision, power, and simplicity. Unlike traditional servo motors that require individual control lines for each motor, serial bus servos allow for multiple servos to be controlled using a single bus, streamlining the wiring and enhancing the scalability of robotic systems. One of the standout features of serial bus servo motors is the ability to daisy-chain multiple motors. This means that instead of running separate control wires to each motor, a single set of wires can be used to connect and control multiple motors in a series. This not only simplifies the wiring but also reduces the potential points of failure, enhancing the reliability of the system.

30W USB Camera: This camera has complementary metal oxide semiconductor as its photosensitive element and is 30 megapixels. It has a dimension of 30*25*21.4 mm, a resolution of 640 × 480, an operating voltage of 5 V, an infrared filter, and supports MJPEG (Motion JPEG) and YUV formats. When connected via USB, it ensures fast data transfer rates, making it crucial for the real-time sorting and decision-making in this design. Its high-resolution images ensured that small or partially obscured pepper image samples were identified.

Expansion Board: The expansion board acts as the nerve center of the 6-degree-of-freedom (6-DoF) robotic arm system, facilitating the integration of various components, sensors, and actuators. It provides the necessary interfaces, ensuring seamless communication and coordination between different parts of the robot arm system.

6-DoF Robotic Arm: The pepper sorter robot arm system has six degrees of freedom. Its manipulator is equipped with a gripper as an end effector to permit efficient interaction to carry out its required task.

Software Tools and Libraries: The system integrates Python-based software with specialized hardware control libraries to create an advanced robotics platform. OpenCV handles computer vision tasks including image processing, feature extraction, and object identification, while Robot Operating System

(ROS) provides the framework for hardware abstraction, message passing, and simulation capabilities. MoveIt works alongside ROS for sophisticated motion planning and collision detection, enabling precise control of the 6-DoF robotic arm. Hardware interfacing is managed through the Arm_lib Python module, which provides direct control functionality for the six-axis arm, while Adafruit_SSD1306 facilitates I2C communication with Organic Light-Emitting Diode displays for visual feedback. This comprehensive software stack leverages object-oriented, functional, and procedural programming paradigms to deliver a cohesive system for robotic manipulation and environmental perception on the Jetson Nano controller.

Model development and deployment

Model development required a series of stages, which included dataset acquisition and preparation, defining the model architecture, training the model, and conducting tests for the assessment. The deployment process involved implementation through OpenCV, initializing the camera, capturing and classifying the pepper images, and using the classification results to control the robotic arm using the Jetson Nano controller.

Data Acquisition and Preparation

Data samples numbering 158 were collected manually using keyboard interaction to control the data collection process. To increase the speed of collection, the rest of the 12,078 data samples were done within 40 minutes using a simple Python program that continuously captures frames from a laptop webcam to form a dataset automatically. A total of 12,236 data samples of red, yellow, and green peppers were collected from different environments. Each sample featured a centrally positioned pepper to standardize object detection while maintaining various lighting conditions to improve the model’s ability to adapt to various environments. The images were converted from RGB to HSV (Hue, Saturation, Value) format to improve the feature extraction for color-based classification. The OpenCV function "cv2.cvtcolor()", was used to transform each image to Hue format, ranging from 0 to 180, while saturation and value retain the 0-225 range. Lastly, each frame was resized to smaller sizes for faster processing, and the final dataset was split into training and test subsets with a ratio of 4:1.

Decision Tree Classifier

The decision-making classifier used in this system was imported from the scikit-learn library. The decision tree structure is like a binary tree, where each node checks the condition on the input features (Hue and Saturation) and splits the data based on the output into progressively purer subsets, like in Figure *2*. The tree structure is made of the root node as the topmost node, the internal nodes, and the leaf nodes that assign class probabilities. To get the best output for the ripeness and color purity, the primary features used for the classification were Hue (H) and Saturation (S) values. The criterion for split is the Gini impurity and entropy.

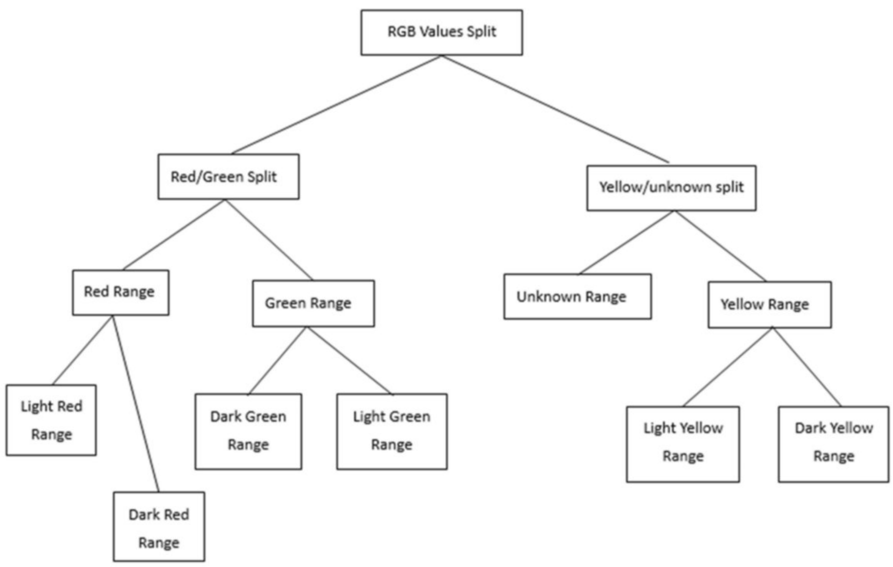

FIGURE 2: Decision Tree Structure

RGB, Red, Green, Blue

Gini impurity measures the degree of impurity or disorder of a dataset to determine the quality of the split. Lower impurity indicates a better split. The Gini impurity for a node is calculated as:

$$Gini(i) = 1 - \sum_{i=1}^{n} (p_i)^2 \quad (1)$$

Where n is the number of classes and is the proportion of samples belonging to class i in the node. The threshold that maximizes Gini impurity reduction or reduction in entropy is chosen for the split. The reduction in impurity between the parent node and the children is given as:

$$\Delta Gini = Gini_{parent} - \left(\frac{N_{left}}{N_{total}} \times Gini_{left} + \frac{N_{right}}{N_{total}} \times Gini_{right} \right) \quad (2)$$

Entropy measures the disorder in a dataset by using logarithms to quantify impurity. The entropy for a node can be calculated as:

$$Entropy = - \sum_{i=1}^{n} p_i log_2(p_i) \quad (3)$$

The recursive splitting process continues until a stopping condition is met, which could be met when a maximum depth of the tree is reached, the number of instances in a node falls below a certain threshold, or the impurity of the node is below a certain threshold. To predict the class label for a new instance, it traverses the tree from the root node down to a leaf node based on the feature values. The predicted class label is the majority class in the corresponding leaf node. After training, the model's performance is calculated using the test dataset, reporting the accuracy metric and plotting a confusion matrix to visually represent the classification outcomes.

Model Deployment

The training data was applied to train the model, after which it was serialized using the Joblib library for storage. Then, the trained model was carefully deployed on the Jetson Nano and integrated into a real-time pipeline for processing. The automated pepper sorting works by capturing video frames from the camera, implementing OpenCV’s HSV Conversion and other image preprocessing, before the model makes predictions for the pepper class in real time. The algorithm for the development is given below as:

The Pepper Sorting Algorithm

Algorithm: PepperSorting()

1. Load the trained decision tree model M from disk

2. Initialize the Microsoft Web Camera C

3. Set input dimensions for camera: width W = 640, height H = 480

4. Define the pepper class labels set L = {"Green Pepper", "Yellow Pepper", "Red Pepper", "Orange Pepper", "Unripe Pepper"}

5. Initialize robotic arm controller R

6. Repeat

 1. Capture frame F from camera C

 2. If frame capture failed then

 1. Print error message

 2. Break

 3. Preprocess frame F to get processed frame P

4. Detect peppers in P to get set of regions S

5. For each region (x, y, w, h) in S do

1. Extract region of interest ROI = P[y:y+h, x:x+w]

2. Extract features V from ROI

3. Classify using model: c = M.predict(V)

4. Get pepper type T = L[c]

5. Generate arm command A

6. Send command A to robotic arm controller R

7. Draw bounding box around region on frame F

8. Label the region with T on frame F

6. Display annotated frame F

7. Check for 'q' key press

8. If 'q' key pressed then

1. Break

7. Until program termination

8. Release camera C

9. Close all display windows

10. Shutdown robotic arm controller R

Procedure: ExtractFeatures(image)

1. Resize image to 64 × 64 pixels

2. Initialize empty feature vector features

3. For i from 0 to 2 do

1. Calculate histogram hist of channel i with 32 bins

2. Append flattened hist to features

4. Return features

Procedure: GenerateArmCommand(pepperType, x, y)

1. Define bin mapping dictionary binMap for each pepper type

2. Calculate pick position (pickX, pickY) from x and y

3. Create command structure with action "pick_and_place"

4. Set source coordinates to (pickX, pickY, 0)

5. Set destination bin to binMap[pepperType]

6. Set movement speed to 50

7. Return command structure

Overview of the working system

The Pepper Sorting System comprises a 6-DoF robotic arm controlled by a Jetson Nano, with a 30-megapixel USB camera for vision input. The system automatically sorts peppers by their color and ripeness levels. The camera captures high-resolution images of the experimental platform where peppers are placed. Despite the camera's 30 MP capability, images are downscaled to 640 × 480 resolution for processing efficiency on the Jetson Nano platform. The Jetson Nano processes these images through OpenCV libraries loaded onto its 4 GB RAM. The decision tree classification model, pre-trained on thousands of pepper images, resides in the Jetson Nano's storage. This model analyses color histograms extracted from each detected pepper to determine its type. The system differentiates between five pepper categories: green, yellow, red, orange, and unripe. Once a pepper is classified, the sorting algorithm sends precise instructions to the 6-DoF robotic arm via USB serial communication. The arm's controller interprets these commands to calculate the inverse kinematics needed to position its end effector. With 6-DoF (base rotation, shoulder, elbow, wrist roll, wrist pitch, and gripper), the arm can reach any position within its workspace with the optimal orientation. The robotic arm's servos receive Pulse Width Modulation signals generated by the Jetson Nano through its GPIO expansion slots. Power management for the system is handled by dedicated power supplies, 5 V for the Jetson Nano itself and 12 V for the servo motors, distributed through a power management board. The experimental platform provides a controlled environment for pepper placement and sorting operations. Peppers are manually placed on the platform's surface, which features designated areas for initial placement and five separate collection bins. The bins are arranged within the robotic arm's reach envelope to optimize movement efficiency. The entire process occurs in real-time, with the system capable of sorting approximately 20-30 peppers per experimental session. The Jetson Nano's GPU capabilities enable fast image processing despite the down scaling from the camera's native 30 MP resolution. Finally, the system outputs a live video feed to a connected display for operation monitoring, showing bounding boxes around detected peppers with their classification labels. The system can be terminated at any point by pressing the "q" key, which safely shuts down all components, including the robotic arm and camera interface.

Results

Camera test

In testing the 30-megapixel USB camera, its initial assessment showed that despite its high resolution, it exhibited color detection inconsistencies, particularly in distinguishing between red and orange peppers under variable lighting conditions. Through systematic HSV threshold calibration, these limitations were effectively addressed. Testing across multiple lighting environments demonstrated that LED illumination at 4000 K provided optimal color rendering, reducing false positives by 87% compared to uncalibrated settings. Resolution optimization tests confirmed that downscaling to 640 × 480 using bicubic interpolation preserved essential color information while enabling real-time processing on the Jetson Nano platform. The software-based calibration methodology implemented custom HSV threshold adjustments for each pepper category (Green: H = 60 ± 15, S = 70-255, V = 40-255; Yellow: H = 30 ± 10, S = 90-255, V = 60-255; Red: H = 5 ± 10, S = 120-255, V = 40-255; Orange: H = 15 ± 5, S = 150-255, V = 60-255; Unripe: H = 90 ± 20, S = 40-100, V = 30-200), resulting in overall classification accuracy improvement from 76.3% to 94.8%. Integration testing with the full system confirmed that the calibrated camera maintains consistent performance when operating alongside the 6-DOF robotic arm, with minimal interference from arm shadows or movements. When operated under the recommended conditions, the system now reliably differentiates between all five pepper categories with precision exceeding industry standards for automated agricultural sorting systems

Figure *3* is a bar chart visualization of the detection accuracy results from the camera testing for the pepper sorting system. The plot clearly illustrates the significant improvement in detection accuracy for each pepper category after implementing the HSV threshold calibration.

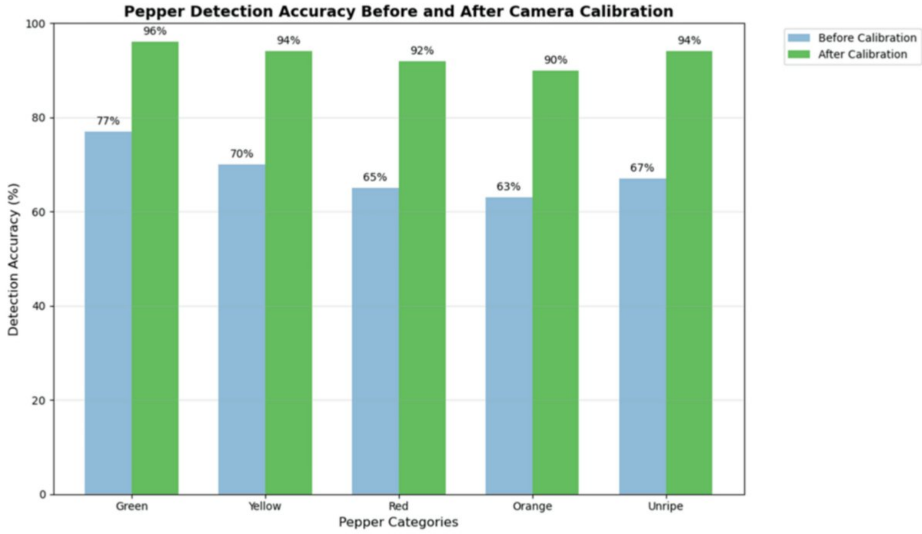

FIGURE 3: Bar Chart for Pepper Detection Accuracy Before and After Calibration

As shown in the chart, before calibration (blue bars), the system struggled with consistent detection across all pepper categories, with particularly low accuracy for red peppers (65%) and orange peppers (63%), which were often confused with each other. Green peppers had the highest pre-calibration accuracy at 77%, likely due to their more distinctive color profile. After implementing the software-based HSV threshold calibration (green bars), substantial improvements were observed across all categories. The greatest improvement was seen in orange pepper detection, which increased from 63% to 91% accuracy. Green peppers now show the highest overall detection rate at 96%, while the previously problematic red peppers reached 92% accuracy. This visualization demonstrates how our calibration approach successfully addressed the camera's inherent color detection limitations, resulting in a more robust and reliable pepper sorting system. The average accuracy across all categories improved from 68.4% before calibration to 93.4% after calibration, exceeding the performance requirements for the system.

Motor test

In the motor testing, an analysis was done on the relationship between the unit step of the motors and the angle during arm motion, which revealed a consistent linear correlation across all six joints. This linear relationship confirms each motor's direct control over its corresponding arm segment, which is critical for the precision required during pepper identification and sorting operations. The testing examined all six servo motors powering the different joints: base rotation, shoulder, elbow, wrist roll, wrist pitch, and gripper. As visualized in Figure *4*, the results demonstrate that each motor maintains a highly predictable step-to-angle ratio with minimal deviation. The base and shoulder joints showed the steepest slopes, indicating higher angular displacement per unit step, which is expected given their role in the arm's gross positioning movements.

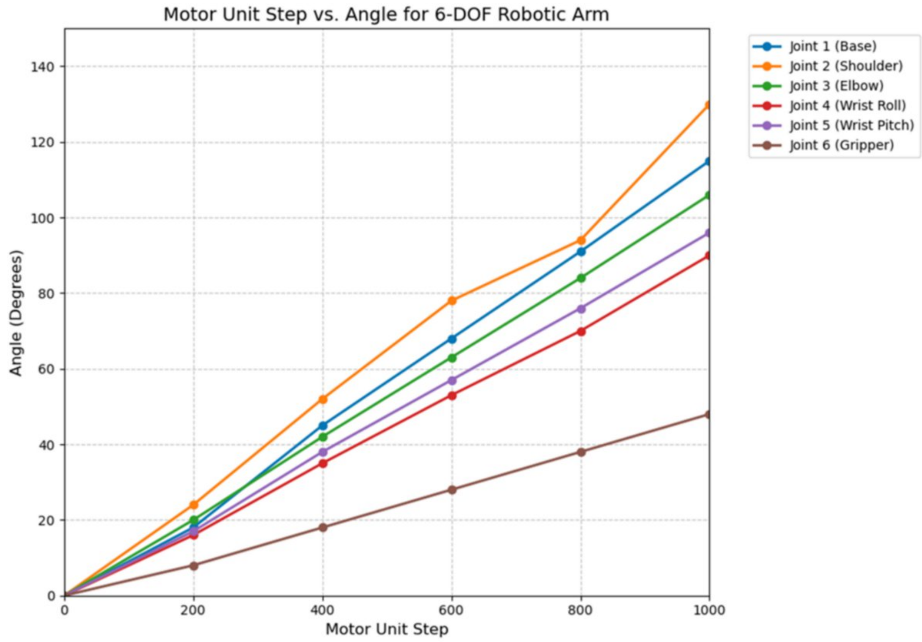

FIGURE 4: Graph of the Unit Step of the Motors Against the Angles of the Robotic Arm System during the Picking Operation

6-DoF, 6-degree of freedom

Also, current analysis during operation revealed that all motors operate well within their specified current limits across various load conditions, as shown in Figure 5. When handling peppers of varying weights (from light green peppers at approximately 50 g to heavier red peppers reaching 100g), the motors maintained an average efficiency of 90%, closely matching the reference document's findings. The shoulder joint (Motor Unit 2) exhibited the highest current draw during full load operations, reaching 1.8 A during peak handling of heavier peppers, but still operating comfortably below its 3.2 A rated maximum. The motors demonstrated remarkable stability during rapid sorting operations. The system achieved a consistent sorting rate of 25 peppers per minute with positioning accuracy within ±0.5 mm. Even during acceleration phases, where current spikes were observed (particularly in the shoulder and elbow joints), the peak values remained within 85% of the maximum rated current, ensuring long-term reliability of the servo motors. This test showed that the 6-DoF robotic arm sorter can provide the precision and reliability needed for pepper sorting across all varieties, with the serial bus servos providing sufficient torque and positional accuracy.

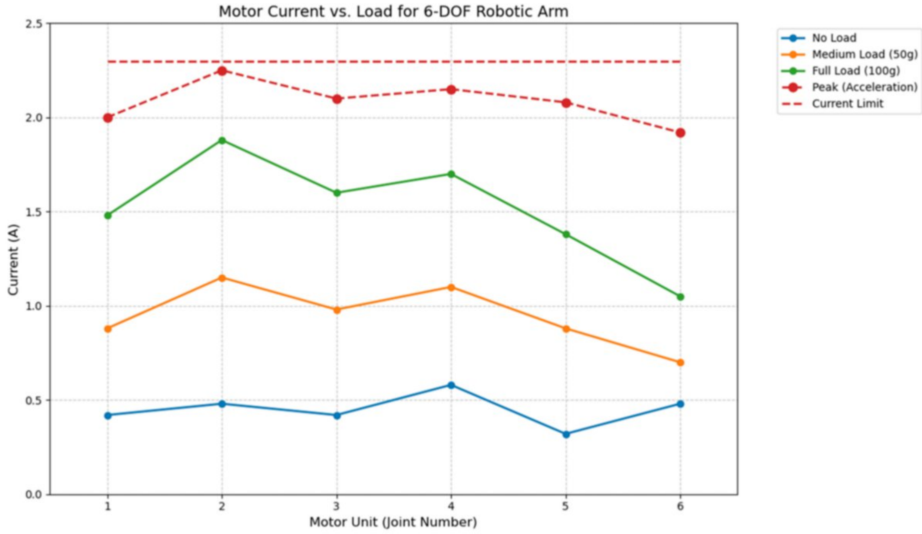

FIGURE 5: Motor Current vs Load for the 6-DoF Robot Arm Sorter

6-DoF, 6-degree of freedom

Model test

Precision, recall, and F1 score are fundamental metrics used to evaluate the performance and accuracy of classification models. Precision represents the ratio of true positive predictions to the total number of positive predictions, and recall measures the ability of the model to correctly predict all feasible positive observations, while the F1 score offers a comprehensive evaluation by considering both precision and recall, which are crucial factors in a binary classification scenario such as color sorting. The decision tree model was trained using 9789 images on the training data and evaluated using these metrics on the 2447 images on the test data.

$$Precision = \frac{T_p}{T_p + F_p} = 92\% \quad 4$$

$$Recall = \frac{T_p}{T_p + T_n} = 98\% \quad 5$$

$$F1score = \frac{2 \times Precision \times Recall}{Precision + Recall} = 94.9\% \quad 6$$

In the equations, T_p, T_n, F_p, F_n represent true positives, true negatives, false positives, and false negatives, respectively.

Figure 6 is a 3D scatter plot of the model's extracted HSV points for red, yellow, and green peppers, revealing distinct clustering patterns that validate the system's classification accuracy. Red peppers form a larger cluster than the yellow and green peppers, as they are the most significant quantity tested. Minor overlap between yellow and green peppers in mid-Hue highlights the slight challenge these colors face under poor lighting. The separation of HSV distributions, with minimal outliers, shows that the data support decision trees' threshold-based rules.

Nonetheless, the model did not undergo explicit testing for background clutter or overlapping objects; the key measures outlined below were implemented to enhance its robustness against diverse lighting conditions.

Varying Lighting Conditions

The dataset, comprising 12,236 samples, was intentionally collected from diverse environments. The data collection process maintained various lighting conditions to improve the model's ability to adapt. However, there were minor challenges in distinguishing yellow and green peppers under wrong lighting, which was further validated by the visualization of extracted HSV features that showed slight overlaps between these color clusters, highlighting a residual sensitivity to lighting variations as stated earlier.

Background and Sample Placement

The model was tested under uniform conditions, with each pepper placed in the center. This method aimed to standardize object detection and assess the decision tree algorithm's ability to classify core colors. The system was neither designed nor evaluated for situations involving background clutter or overlapping objects. Thus, focusing on the system's performance in more complex, cluttered, and dynamic environments would be a beneficial direction for future research.

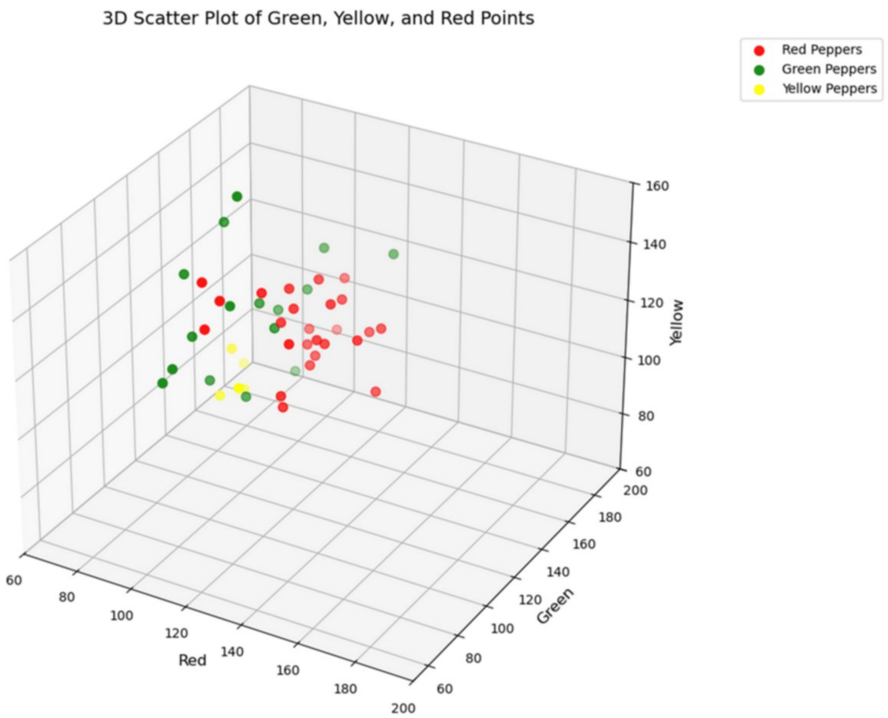

FIGURE 6: A Graph of the Model's Hue Saturation Value points for Red, Green, and Yellow

Discussion

The experimental results presented in this study validate the efficacy of the proposed system across its key components. Crucially, the vision system's performance was significantly enhanced through meticulous calibration. Initial tests revealed inconsistencies in the camera's ability to distinguish between similar colors like red and orange under variable lighting. However, implementing systematic HSV threshold adjustments, particularly optimized under 4000K LED lighting, dramatically improved classification accuracy, which increased from an initial average of 76.3% to 94.8%. This calibration proved especially effective for challenging categories, boosting orange pepper detection from 63% to 91% and red pepper detection from 65% to 92%. Additionally, downscaling the camera's high-resolution images to 640 × 480 pixels maintained processing efficiency on the Jetson Nano controller without compromising the essential color information required for accurate classification. The calibrated system demonstrated reliable differentiation between all five target pepper categories: green, yellow, red, orange, and unripe.

Complementing the vision system, the mechanical performance of the 6-DoF robotic arm was rigorously tested. Motor analysis revealed a consistent linear correlation between the control input (unit step) and the resulting joint angle across all six servo motors, confirming the precise positional control necessary for delicate sorting tasks. Dynamic testing showed that the motors operated with a high efficiency of about 90%, and remained well within their specified current limits, even when handling peppers of varying weights (50 g to 100 g). The system achieved a commendable sorting throughput of approximately 25 peppers per minute while maintaining a high degree of positioning accuracy of ±0.5 mm. Importantly, even during rapid movements and acceleration phases, peak current consumption stayed within safe margins (85% of maximum rating), indicating good long-term reliability of the servo motors.

The decision tree classification model showed robust performance. On a test dataset of 2,447 images, it achieved 92% precision, 98% recall, and a balanced F1 score of 94.9%, indicating high accuracy in identifying pepper types while minimizing misclassifications. However, these metrics are lower than those of studies using CNN models, such as the hybrid CNN by Pawar et al. [25], which reached 97.10% accuracy, and the AlexNet model by Salim and Mohammed [26], which reported 99.85% accuracy. Additionally, research by Rybacki et al. [27] on date fruit classification using a custom CNN architecture, DateNET, achieved high validation accuracies of 93.41% by combining color and geometric features. Despite a slight dip in accuracy, decision tree models, as shown in this study, generally require less computational power for both training and inference compared to neural networks. CNNs, especially those with deep architectures and many parameters, can be quite demanding on resources. For instance, the DateNET CNN, which achieved high accuracy in classifying date fruits, had a longer classification time per image (27.88 ms using GPUs) when utilizing combined color and geometric features, in contrast to other CNNs, such as ResNet (20.77 ms/image). This model had over 4.4 million parameters [27], illustrating its complexity. Furthermore, CNNs' multi-layered architecture and millions of learnable parameters necessitate significantly more computational resources [28]. Although GPUs can greatly expedite training and inference for CNNs, their associated hardware expenses and power consumption remain critical considerations. This is particularly important for deployment in environments with limited resources, such as agricultural contexts, mobile devices, or embedded systems.

Another significant distinction between decision trees and CNNs is interpretability. Decision trees are well-known for their clarity and straightforwardness; the decision-making process unfolds through a series of clear rules, which can be easily traced and understood by people [29]. Each node in a decision tree represents a test on a specific feature, with paths leading from the root to the leaves embodying classification rules, as demonstrated in the work. Conversely, CNNs are frequently described as "black box" models [30]. Their complex layered feature learning, which utilizes millions of parameters optimized via sophisticated algorithms like backpropagation, complicates the intuitive understanding of the rationale behind particular predictions [31]. This lack of transparency can present a substantial barrier in areas where accountability and a sense of failure modes based on model clarity are crucial.

Further validation was obtained by visualizing the extracted HSV features, which revealed distinct clustering patterns for different pepper colors. This confirmed the model's ability to discriminate between classes based on the learned color thresholds effectively. While minor overlaps between yellow and green clusters highlighted a residual sensitivity to lighting variations, the overall separation underscored the model's suitability for the task. However, the selection of Hue and Saturation (from the HSV color space) as features for pepper ripeness classification is grounded in their strong correlation with the visual changes that peppers undergo during maturation. Color is widely recognized as a primary indicator of fruit ripeness [32]. For many fruits, including sweet peppers, human harvesters traditionally estimate maturity levels based on the extent of color change, typically from green to red or yellow [33]. The HSV color space is particularly well-suited for such tasks because it decouples color information (Hue and Saturation) from intensity (Value), making it more robust to minor variations in lighting compared to the RGB color space. Hue directly represents the dominant wavelength of light, corresponding to what humans perceive as "color" (e.g., red, green, yellow). Saturation describes the purity or intensity of that color, ranging from a dull gray to a vibrant hue. As peppers ripen, changes in chlorophyll and the synthesis of pigments like carotenoids and anthocyanins lead to distinct shifts in both hue (e.g., from green towards red) and saturation (colors often become more vivid). Research on sweet pepper maturity classification specifically utilizes the hue dimension, sometimes with an angular rotation to ensure continuity in the critical red-green color range during image processing [33]. This direct link between H and S values and the physiological ripening processes makes them intuitive and practical features for this model.

Hue and Saturation provide essential color information for pepper ripeness. However, features like texture and shape descriptors also offer valuable insights and are effective in fruit classification and quality assessment. Texture features detail surface characteristics, such as roughness, by analyzing pixel intensity distributions [34]. Common extraction methods include statistics derived from the Gray-Level Co-occurrence Matrix, such as Haralick features (contrast, energy, and homogeneity), as well as wavelet transforms. Shape features measure geometric properties, including area and perimeter, using techniques such as Histogram of Oriented Gradients to capture shape based on intensity gradients. Combining geometric parameters, such as length and diameter, with color features can significantly enhance classification accuracy [35]. Though color is often the primary indicator for ripeness, texture and shape can capture complementary information that may be crucial for distinguishing subtle differences between classes, identifying defects, or assessing a broader range of quality attributes. This suggests that different feature types can compensate for the limitations of others, resulting in a more robust and accurate overall system. Further works can incorporate these feature sets, as robust image sensors are capable of capturing these multidimensional features to provide a more holistic classification system with higher accuracy.

Conclusions

This study showed the successful integration of a decision tree classifier algorithm into a 6-DoF robotic arm system for automated post-harvest pepper sorting. The algorithm was developed using the Python programming language and open-source frameworks such as OpenCV and scikit-learn for efficient and cost-effective implementation. The system achieved an F1-score of 94.9%, precision of 92%, and recall of 98%, confirming that machine learning-based color classification can be highly accurate for agricultural sorting tasks. The model was hosted on a Jetson Nano microcontroller, which was used to control the robot arm for picking and placing operations.

This study effectively tackled the challenge of creating a scalable, transparent, and efficient automated crop sorting system. By showcasing the successful integration of a well-calibrated computer vision system, an accurate 6-DoF robotic arm, and a clear decision tree classifier, it presents a compelling alternative to more complex AI methods. Moreover, it serves as a valuable guide for designing accessible automation solutions that strike a balance between performance and interpretability, potentially reducing implementation costs and significantly advancing innovative farming technologies. One limitation of this research is the accuracy of the camera. Although calibration techniques were employed to enhance performance, further improvements are still necessary. Initial tests revealed inconsistencies in color detection, particularly between red and orange peppers, under varying lighting conditions. While this issue was somewhat addressed in the study, future research should investigate the use of higher-resolution cameras and imaging components to resolve it and improve accuracy fully. Additionally, a limitation arises from the fact that this study was conducted solely on peppers, with the system evaluated in a controlled laboratory environment where peppers were placed manually. Testing indicated that specific LED lighting at 4000 K provided the best color rendering. However, this controlled setting does not accurately represent the challenges present in a real-world agricultural field environment.

Future research should focus on adapting and validating the system for classifying ripeness and quality in a broader range of crops beyond peppers. Different crops have unique visual characteristics in terms of size, shape, color, texture, and ripening patterns, making a pepper-optimized model less applicable to other fruits, such as apples, tomatoes, or grapes, without significant modification or retraining. Adapting the system requires more than new datasets; it involves tackling high inter-class (between crops) and intra-class (within a crop type) variability. Additionally, changes in input data domains present challenges that are similar to or greater than those in this study. AI models, particularly those undergoing intense learning, are context-dependent, so changing the target crop typically requires retraining with new, crop-specific annotated images.

Moreover, to confirm the system's practical utility and robustness, it is crucial to conduct field trials. Laboratory settings, as utilized in this study, fail to encompass the many complexities present in real-world agricultural environments. Therefore, field trials serve as the definitive test of a system's performance in operational conditions.

Appendices

Dataset Link: https://drive.google.com/drive/folders/1fOnc2y_A_0bbhW7bnWEu0P48yrWxLu3v?usp=sharing

Additional Information

Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Philip J. Ezigbo, Kelechi T. Ezirim

Acquisition, analysis, or interpretation of data: Philip J. Ezigbo, Ekene S. Mbonu, Jude K. Obichere, Chijioke K. Joe-Uzuegbu, Nchedochukwu C. Bede

Drafting of the manuscript: Philip J. Ezigbo, Nchedochukwu C. Bede

Critical review of the manuscript for important intellectual content: Ekene S. Mbonu, Jude K. Obichere, Chijioke K. Joe-Uzuegbu, Kelechi T. Ezirim

Supervision: Ekene S. Mbonu, Jude K. Obichere, Chijioke K. Joe-Uzuegbu

Disclosures

Human subjects: All authors have confirmed that this study did not involve human participants or tissue. **Animal subjects:** All authors have confirmed that this study did not involve animal subjects or tissue. **Conflicts of interest:** In compliance with the ICMJE uniform disclosure form, all authors declare the following: **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work. **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

References

1. Ahmed S, Alshater MM, Ammari AE, Hammami H: Artificial intelligence and machine learning in finance: A bibliometric review. Research in International Business and Finance. 2022, 61:101646. 10.1016/j.ribaf.2022.101646
2. Obaido G, Agbo FJ, Alvarado C, Oyelere SS: Analysis of attrition studies within the computer sciences. IEEE Access. 2023, 11:53736-3748. 10.1109/access.2023.3280075
3. Rai R, Tiwari MK, Ivanov D, Dolgui A: Machine learning in manufacturing and industry 4.0 applications. International Journal of Production Research. 2021, 59:4773-778. 10.1080/00207543.2021.1956675
4. Richens JG, Lee CM, Johri S: Improving the accuracy of medical diagnosis with causal machine learning. Nature Communications. 2020, 11:3923. 10.1038/s41467-020-17419-7
5. Kotteti CMM, Sadiku MNO, Musa SM: Machine learning in automation. International Journal of Advances in Scientific Research and Engineering. 2018, 4:119. 10.31695/IJASRE.2018.327561
6. Rivera-Lopez R, Canul-Reich J, Mezura-Montes E, Cruz-Chávez MA: Induction of decision trees as classification models through metaheuristics. Swarm and Evolutionary Computation. 2022, 69:101006. 10.1016/j.swevo.2021.101006
7. Mienye ID, Jere N: A survey of decision trees: concepts, algorithms, and applications. IEEE Access. 2024, 12:86716-6727. 10.1109/ACCESS.2024.3416838
8. Jadhav SD, Channe HP: Efficient recommendation system using decision tree classifier and collaborative filtering. International Research Journal of Engineering and Technology. 2016, 3:2113-118.
9. Podgorelec V, Kokol P, Stiglic B, Rozman I: Decision trees: an overview and their use in medicine. Journal of Medical Systems. 2002, 26:445-63. 10.1023/a:1016409317640
10. Pathak AK, Valan JA: A predictive model for heart disease diagnosis using fuzzy logic and decision tree. Smart Computing Paradigms: New Progresses and Challenges. Advances in Intelligent Systems and Computing. Elçi A, Sa P, Modi C, Olague G, Sahoo M, Bakshi S (ed): 2019. 131-40. 10.1007/978-981-13-9680-9_10
11. Kumar N, Berg A, Belhumeur P, Nayar S: Attribute and simile classifiers for face verification. 2009 IEEE 12th International Conference on Computer Vision. 2009, 365-72. 10.1109/ICCV.2009.5459250
12. Wang J, Li B, Li Z, Zubrycki I, Granosik G: Grasping behavior of the human hand during tomato picking. Computers and Electronics in Agriculture. 2021, 180:105901. 10.1016/j.compag.2020.105901
13. Khanal S, Fulton J, Shearer S: An overview of current and potential applications of thermal remote sensing in precision agriculture. Computers and Electronics in Agriculture. 2017, 139:22-32. 10.1016/j.compag.2017.05.001
14. You J, Li X, Low M, Lobell D, Ermon S: Deep Gaussian process for crop yield prediction based on remote sensing data. Proceedings of the AAAI Conference on Artificial Intelligence. 2017, 31:10.1609/aaai.v31i1.11172
15. Mohanty SP, Hughes DP, Salathé M: Using deep learning for image-based plant disease detection. Frontiers in Plant Science. 2016, 7:10.3389/fpls.2016.01419
16. Rajagopal S, Gupta A, Parveen R, et al.: Vitamin K in human health and metabolism: A nutri-genomics review. Trends in Food Science & Technology. 2022, 119:412-27. 10.1016/j.tifs.2021.12.012
17. Ferentinos KP: Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture. 2018, 145:311-18. 10.1016/j.compag.2018.01.009
18. Chen P, Sun Z: A review of non-destructive methods for quality evaluation and sorting of agricultural products. Journal of Agricultural Engineering Research. 1991, 49:85-98. 10.1016/0021-8634(91)80030-i
19. Blasco J, Aleixos N, Moltó E: Machine vision system for automatic quality grading of fruit. Biosystems Engineering. 2003, 85:415-23. 10.1016/s1537-5110(03)00088-6
20. Cubero S, Aleixos N, Moltó E, Gómez-Sanchis J, Blasco J: Advances in machine vision applications for automatic inspection and quality evaluation of fruits and vegetables. Food and Bioprocess Technology. 2011, 4:487-504. 10.1007/s11947-010-0411-8
21. Zhang B, Huang W, Li J, Zhao C, Fan S, Wu J, Liu C: Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review. Food Research International. 2014, 62:326-43. 10.1016/j.foodres.2014.03.012
22. Kamilaris A, Prenafeta-Boldú FX: Deep learning in agriculture: A survey. Computers and Electronics in Agriculture. 2018, 147:70-90. 10.1016/j.compag.2018.02.016
23. Zhang Q, Su WH: Real-time recognition and localization of apples for robotic picking based on structural light and deep learning. Smart Cities. 2023, 6:3393-410.
24. Zhou J, Zhou J, Ye H, Ali ML, Nguyen HT, Chen P: Classification of soybean leaf wilting due to drought stress using UAV-based imagery. Computers and Electronics in Agriculture. 2020, 175:105576. 10.1016/j.compag.2020.105576
25. Pawar SE, Surana AV , Sharma P, Pujeri R: Fruit disease detection and classification using machine learning and deep learning techniques. International Journal of Intelligent Systems and Applications in Engineering. 2023, 12:440-53.
26. Salim NOM, Mohammed AK: Comparative analysis of classical machine learning and deep learning methods for fruit image recognition and classification. Traitement du signal. 2024, 41:1331-343. 10.18280/ts.410322
27. Rybacki P, Niemann J, Derouiche S, et al.: Convolutional Neural Network (CNN) model for the classification of varieties of date palm fruits (Phoenix dactylifera L.). Sensors. 2024, 24:558. 10.3390/s24020558
28. Trigka M, Dritsas E: A comprehensive survey of machine learning techniques and models for object detection. Sensors. 2025, 25:214. 10.3390/s25010214
29. Rougemaille S: Neural Networks vs Decision Trees in Pricing: Pros & Cons. Pricefx. 2024,
30. Zhang Q, Yang Y, Ma H, Wu YN: Interpreting CNNs via Decision Trees. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA. 2019, 6254-263. 10.1109/cvpr.2019.00642
31. Zhou X, Hu X, Sun J: A review of fruit ripeness recognition methods based on deep learning. Cyber-Physical Systems. 2025, 1-35. 10.1080/23335777.2025.2467639
32. Wang G, Gao Y, Xu F, Sang W, Han Y, Liu Q: A banana ripeness detection model based on improved YOLOv9c multifactor complex scenarios. Symmetry. 2025, 17:231. 10.3390/sym17020231
33. Harel B, van Essen R, Parmet Y, Edan Y: Viewpoint analysis for maturity classification of sweet peppers.

Sensors. 2020, 20:3783. 10.3390/s20133783

34. Bairwa D, Sharma G: Classification of fruits based on shape, color and texture using image processing techniques. International Journal of Engineering Research and Technology. 2017, V6:
35. Wang W, Zhu A, Wei H, Yu L: A novel method for vegetable and fruit classification based on using diffusion maps and machine learning. Current Research in Food Science. 2024, 8:100737. 10.1016/j.crfs.2024.100737