IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 59, No. 3, March, 2026, pp. 739-751
Journal homepage: http://iieta.org/journals/jesa

Design and Experimental Validation of a PID-Controlled Cartesian Robot with HSV-Based Vision for Automated Product Classification

Van-Hung Pham[1], Truong-Son Do[2], Thi-Mai-Phuong Dao[2*], Ngoc-Khoat Nguyen[3]

[1] School of Electrical and Electronic Engineering (SEEE), Hanoi University of Science and Technology (HUST), Hanoi 100000, Vietnam
[2] Faculty of Automation, School of Electrical and Electronic Engineering (SEEE), Hanoi University of Industry (HAUI), Hanoi 100000, Vietnam
[3] Faculty of Control and Automation (FCA), Electric Power University (EPU), Hanoi 100000, Vietnam

Corresponding Author Email: daophuong@haui.edu.vn

https://doi.org/10.18280/jesa.590317

Received: 16 January 2026
Revised: 6 March 2026
Accepted: 14 March 2026
Available online: 31 March 2026

Keywords:
Cartesian robot, image processing, Proportional-Integral-Derivative control, product classification, Hue-Saturation-Value color space, automated sorting, machine vision

ABSTRACT

Traditional product classification methodologies predominantly rely on physical sensors to differentiate items by geometry and mass. While computationally efficient, these conventional systems are inherently susceptible to environmental noise—most notably electromagnetic interference (EMI), which can compromise classification fidelity and elevate error rates. This study presents a robust alternative integrating high-speed image processing with a 3-degree-of-freedom (3-DOF) Cartesian robotic platform. The proposed system features an Arduino Mega 2560-controlled architecture utilizing Nema17 stepper motors and a pneumatic vacuum end-effector. Performance was validated through the classification of 150 polychromatic circular specimens under standard ambient lighting. Empirical results yield a classification accuracy exceeding 92%, supported by a vision processing latency of 29.0 ms and a total software-to-communication overhead of 0.5 s. The integrated Proportional-Integral-Derivative (PID) controller maintained trajectory precision with a mean error below 2% of total displacement (steady-state error < 3.5 mm). Furthermore, stress testing delineated the system's operational boundaries, identifying sensitivity thresholds regarding specular reflection and background clutter. This research demonstrates that vision-integrated robotics can effectively bypass the limitations of physical sensing in EMI-prone industrial environments.

1. INTRODUCTION

In the era of rapid industrialization and modernization, the demand for intelligent and automated production systems is growing significantly. Among various industrial robots, the widespread adoption of Cartesian robots is attributed to their simple mechanical structure, ease of control, and high accuracy in repetitive tasks such as classification and pick-and-place operations. However, traditional systems mainly rely on basic sensors or pre-programmed trajectories, lacking adaptability to variations in production lines. Image processing, on the other hand, is a rapidly evolving field that aims to enable machines to analyze and understand visual information effectively [1]. It has been successfully applied in numerous domains such as face recognition [1], object detection [2], and product classification [3]. Integrating image processing [4] techniques into Cartesian robotic systems offers the potential to improve automation, precision, and operational flexibility in product classification tasks.

Several previous studies have explored the application of computer vision in robotic systems. For example, Pagano et al. [5] developed an automated system using vision-guided robotics for flexible adhesive application in shoe manufacturing. Arévalo-Hernández et al. [6] introduced a vision-assisted pick-and-place robotic machine, while Kuncan et al. realized controlling servo motors for a Cartesian robot [7] using image data and Rexroth Programmable Logic Controller (PLC). Other works, such as the study [8], investigated a Neuro-Fuzzy Approach to Soft Material Cutting with Cartesian Robots, and studies like [9-11], applied Cartesian robots for agricultural tasks and fruit classification using camera systems. Despite demonstrating significant potential, most of these studies are relatively narrow in scope, focusing on either object recognition or motion control and lack a comprehensive integration and quantitative validation between image processing and robotic control.

Additionally, some advanced research has explored human-robot interaction (HRI) through innovative techniques such as contactless electromagnetic force feedback [12] or remote control via virtual interfaces [13]. While these approaches provide high flexibility, they are often complex, costly, and not suitable for small- to medium-scale industrial applications. While Cartesian robots [14] are well-established for their high precision and cost-effectiveness in linear tasks, challenges remain in their dynamic control and seamless integration with vision systems. Previous research has addressed issues such as trajectory errors and mechanical limitations, and while advanced methods like Convolutional Neural Networks

(CNNs) [15-17] have been employed for recognition, their computational complexity can be a barrier for low-cost, real-time applications. This highlights a distinct research gap: there is a need for a holistically designed system that is not only low-cost but also features a tightly coupled control and vision architecture, validated with rigorous quantitative metrics in both simulation and real-world experiments. Many existing works either focus on complex control theory without accessible experimental validation or on vision algorithms without fully considering the dynamic performance of the integrated robotic system.

Motivated by these limitations, this study proposes a fully integrated solution that combines a 3-degree-of-freedom (3-DOF) Cartesian robot with real-time image processing for the purpose of product color-based classification with geometric filtering [18-20]. The remainder of this paper is organized as follows: Section 2 details the overall system architecture, including the dynamic modeling and Proportional-Integral-Derivative (PID) control strategy. Section 3 presents the proposed real-time image processing algorithm based on the Hue-Saturation-Value (HSV) color space. Section 4 discusses the experimental setups, simulation results, and real-world performance evaluation. Finally, Section 5 concludes the paper and outlines future research directions.

To explicitly highlight the contributions and novelty of this research compared to existing literature, a quantitative comparison of recent vision-guided robotic sorting systems is presented in Table 1. Unlike methods relying on expensive deep learning models or open-loop controls, the current approach ensures high reliability through a custom closed-loop PID and a lightweight HSV pipeline, achieving a highly cost-effective and fully validated physical prototype.

Table 1. Comparative analysis of vision-guided robotic sorting systems

Ref.	Control Method	Vision Method	Real Prototype	Cycle Time	Accuracy	Hardware Cost
[21]	Open-loop	Deep Learning (CNN)	Yes	Slow	98.0%	High
[22]	CNC-based	Feature Recognition	Yes	Medium	~90.0%	High
[8]	Neuro-Fuzzy	Not specified	Yes	Not reported	N/A	High
This Work	Closed-loop PID	Lightweight	Yes	Fast (4.0 s)	92.7%	Low

Note: PID = Proportional-Integral-Derivative; HSV = Hue-Saturation-Value; CNN = Convolutional Neural Network

2. SYSTEM ARCHITECTURE AND METHODOLOGY

This section details the architecture of the proposed system, the mathematical model governing the robot's dynamics, and the control strategy employed. A systematic approach was taken, beginning with hardware integration, followed by dynamic modeling, and culminating in the design and tuning of the feedback controller.

2.1 System architecture

The overall architecture of the integrated system is depicted in the flowchart in Figure 1. The process begins with image acquisition by the camera, followed by processing on a PC to identify and locate target products. Positional data is then transmitted to a microcontroller, which actuates the Cartesian robot to perform the sorting task [23-26]. The physical system block diagram is shown in Figure 2.

The Cartesian robot control system includes the following key components:

- **Camera:** This is the primary input source providing image information for the system. The camera assists the system in recognizing and classifying products on the production line and calculating the necessary coordinates to control the actuator mechanism's operations [27, 28].
- **Image processing (PC):** Initially, the PC handles image processing to determine the coordinates of detected objects. Following this calculation, the coordinate data is transferred to the Arduino Mega microcontroller through a serial UART interface, operating at a baud rate of 19200 bps, which in turn directs the actuators. This baud rate was determined to be sufficient for transmitting coordinate data without introducing significant latency for the pick-and-place cycle time.
- **Microcontroller:** The microcontroller (Arduino Mega 2560) controls the actuator mechanism of the robot, especially the stepper motors controlling the robot's axes. It receives signals from the PC via the serial link, decodes them, and executes corresponding control actions.
- **Power Supply:** Provides electrical energy to the system, transforms voltage and current, protects and stabilizes power sources, and supplies backup power to ensure continuous and stable operation.
- **Cartesian Robot:** This is the main actuator mechanism used for grasping and sorting products. It utilizes a vacuum system and stepper motors to move its axes to the desired gripping points.
- **Encoder:** An electro-mechanical device that converts rotary or linear motion into digital signals or pulses. Among various types, magnetic encoders [29, 30] utilize magnetic fields to sense rotational or linear displacement, offering robustness and resistance to environmental factors like dust or moisture, which are beneficial in industrial settings. Encoders are fundamentally used to detect the position, direction, and speed of a motor by counting the number of revolutions or increments of the shaft, providing crucial feedback for precise motion control. To achieve high accuracy while maintaining a low overall cost, our system was specifically engineered to operate in a true closed-loop manner. High-resolution magnetic encoders were integrated directly into the rear shafts of the standard stepper motors. To securely accommodate these sensors and ensure rigorous alignment with the rotating shafts, customized mounting brackets were designed and fabricated using 3D printing technology. During operation, real-time positional data from these magnetic encoders is continuously fed back to the Arduino Mega microcontroller. The embedded PID controller calculates the spatial error and dynamically adjusts the step and direction pulse frequencies sent to the motor drivers, effectively compensating for external disturbances and preventing step loss.

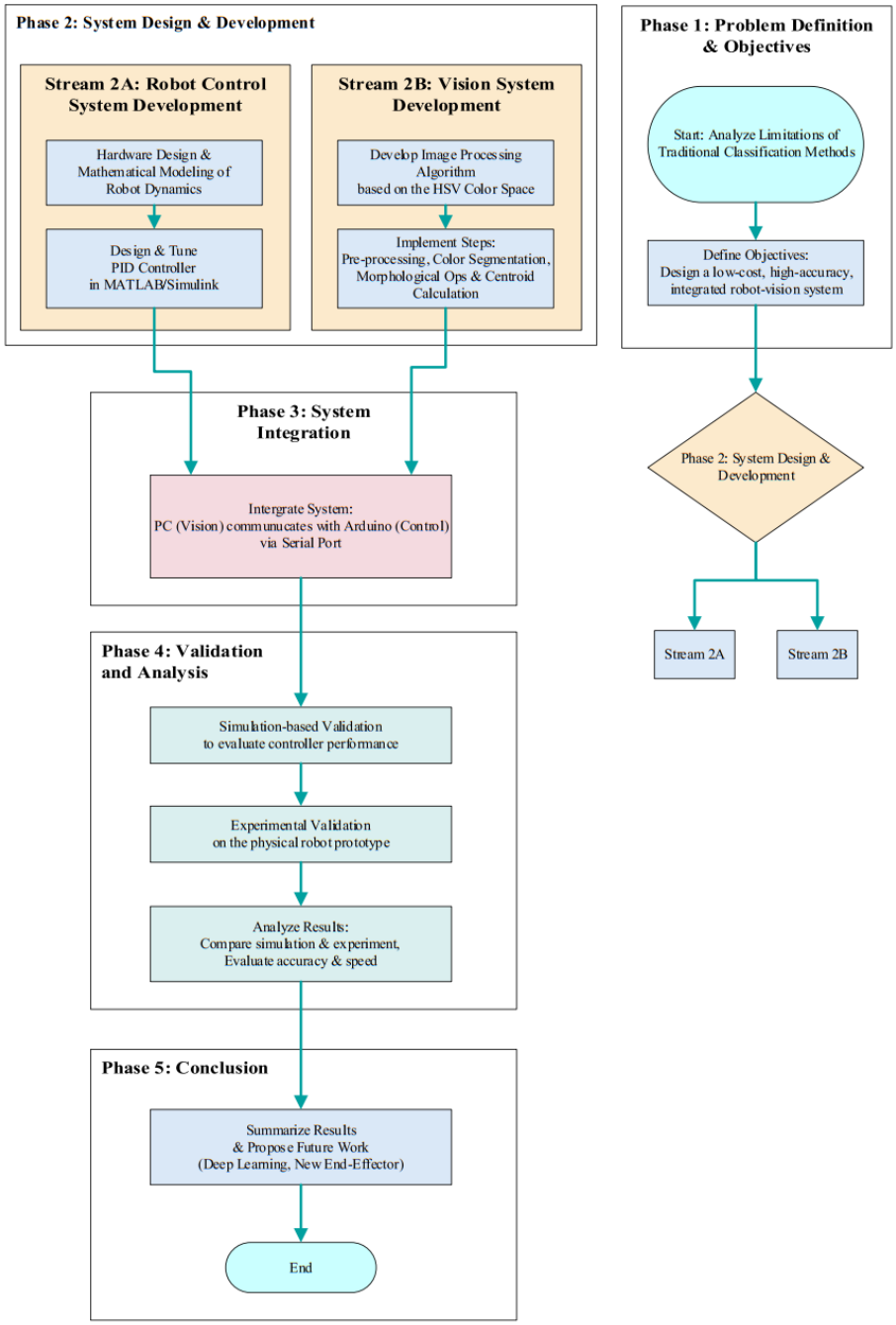

Figure 1. The research flowchart

Table 2. Communication protocol

Feature	Details
Physical Layer	UART (Serial Communication)
Baud Rate	19200 bps
Data Format	ASCII String
Frame Format (PC -> Robot)	"X,Y,Z"
Frame Format (Robot -> PC)	"Message\n"
Units	Robot Coordinates (Integer)
Update Rate	Event-Driven
Packet-loss Handling	Basic Error Logging
Timeout	1 second

Table 2 details the communication protocol established between the host PC and the robot controller. The system utilizes a UART physical layer operating at a baud rate of 19200 bps to exchange ASCII strings. Data transmission is event-driven, with the PC sending integer target positions in an "X, Y, Z" coordinate format, while the robot returns status updates via "Message\n" strings. To maintain system reliability and prevent operational hangs, the protocol incorporates basic error logging for packet loss and a strictly enforced 1-second timeout mechanism.

Table 3. System hardware specifications

Component/ Parameter	Specification
Working Area	X: 300 mm, Y: 340 mm, Z: 100 mm
Actuators	Stepper Motors Nema17 (equipped with magnetic encoders AS5600)
End-Effector	Pneumatic vacuum suction cup
Payload Capacity	1.0 kg
Vision Sensor	Logitech C270
Estimated Total Cost	$350

To provide a comprehensive overview of the system's mechanical and electrical capabilities, the detailed hardware specifications, including the physical constraints, actuator types, and overall estimated cost, are summarized in Table 3.

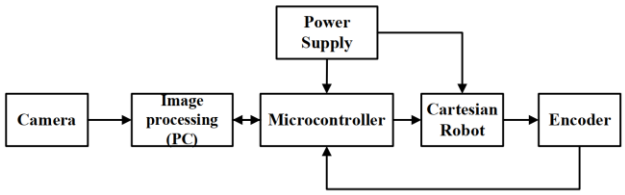

Figure 2. A block diagram of the entire system

2.2 Mathematical modeling

To design an effective controller, a dynamic model of the 3-DOF Cartesian robot was developed. The general equation of motion for a robotic manipulator is given by:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \quad (1)$$

where, q is the vector of joint positions, $M(q)$ is the mass-inertia matrix, $C(q,\dot{q})$ is the matrix of Coriolis and centrifugal forces, G(q) is the vector of gravitational forces, and tau is the vector of applied joint torques/forces. For a Cartesian robot with three prismatic joints (d2, d3, d4 corresponding to the Y, X, Z axes), this model simplifies significantly. The axes are decoupled, the Coriolis/ centrifugal terms are negligible ($C(q,\dot{q})$ approx0), and the mass matrix M is constant. The dynamic equations for each axis can be written as:

$$\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\end{bmatrix} = \begin{bmatrix}m_1+m_2+m_3 & 0 & 0\\ 0 & m_2+m_3 & 0\\ 0 & 0 & m_3\end{bmatrix}\begin{bmatrix}\ddot{d}_2\\ \ddot{d}_3\\ \ddot{d}_4\end{bmatrix} + \begin{bmatrix}0\\ 0\\ m_3\end{bmatrix}g \quad (2)$$

where, τ_1, τ_2, and τ_3 are the actuator forces applied to each respective link. As represented in the matrix, this simplified dynamic model assumes that the effects of mechanical friction are negligible. Symbols $\ddot{d}_2$, $\ddot{d}_3$, and $\ddot{d}_4$ represent the linear accelerations of the moving joints.

The specific physical parameters used to construct and simulate this dynamic model include the masses of the three moving links, specifically m_1 = 1.15 kg, m_2 = 0.6 kg, and m_3 = 0.55 kg, along with the standard gravitational acceleration g = 9.81 m/s². Where, m_x, m_y, m_z are the effective masses of each moving axis, b_x, b_y, b_z are the viscous friction coefficients, g is the acceleration due to gravity, and tau_x, tau_y, tau_z are the forces applied by the actuators on the X, Y, and Z axes, respectively. These equations form the basis of the "ROBOT MANIPULATOR" block within our MATLAB/Simulink simulation, as shown in Figure 3.

2.3 Control strategy

For the position control of each axis, a PID feedback controller was selected. While advanced control methods such as fuzzy logic [31-37] or neural networks [38-41] can offer enhanced performance in complex scenarios, the PID controller remains a highly effective and widely adopted choice for industrial applications due to its robustness, simplicity of implementation, and low computational overhead [42, 43]. For the defined pick-and-place task, where the robot follows point-to-point trajectories, a well-tuned PID controller is sufficient to achieve the required accuracy and dynamic response without the complexity of more advanced algorithms, aligning with the project's goal of developing a cost-effective system [44].

The Simulink model for the closed-loop control system is shown in Figure 3. Each axis has an independent PID controller that computes the required actuator force (tau) based on the error between the desired position (Setpoint) and the actual position feedback. The output of the controller is then fed into the robot's dynamic model (Figure 4) to simulate the physical response.

The performance of the PID controller is critically dependent on its gains (K_p, K_i, K_d). In this study, the PID parameters for each axis were determined through rigorous empirical heuristic tuning (trial-and-error) directly on the physical hardware. The gains were iteratively adjusted to optimize the response, specifically to achieve smooth deceleration and minimize settling time for typical pick-and-place motions, while ensuring absolute robustness against missed steps. The final discrete-time control parameters, including the strategic use of PD-like control on specific axes to intentionally prevent integral windup.

Therefore, using a PID controller is seen as a good choice to achieve the goals of high accuracy and speed in Cartesian robot systems.

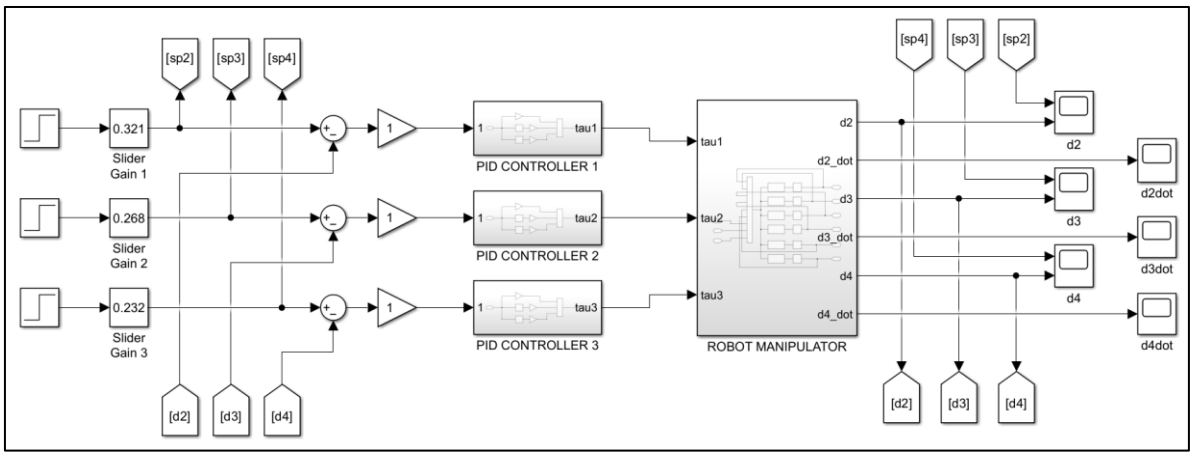

Figure 3. The control system built in MATLAB/Simulink

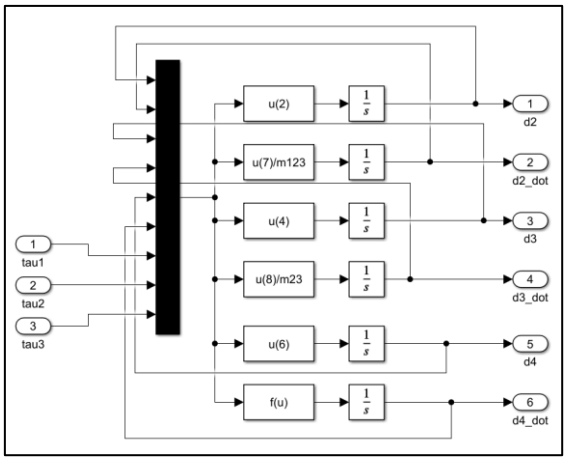

Figure 4. System mathematical model

where, tau1, tau2, tau3 are the input control forces for the Y, X, and Z axes, respectively. The function blocks u(7)/m123, u(8)/m23, and f(u) represent the implementation of the dynamic equations derived previously, calculating the acceleration for each axis based on the input forces and system parameters (masses and frictions).

The Integrator blocks are used to integrate acceleration to get velocity, and velocity to get position, thus simulating the robot's physical movement.

2.4 Control signal mapping and actuator interface

While the dynamic model and force-based (tau) PID control described in Section 2.2 and 2.3 were utilized within the MATLAB/Simulink environment to verify mechanical stability under payload and friction constraints, the physical actuator interface requires a different approach. Standard stepper motor drivers do not accept direct force or current commands; instead, they operate on step and direction pulses. To bridge this gap, our physical implementation employs a position-loop PID controller mapped directly to step frequency. The positional error e(t), calculated as the difference between the desired setpoint and the actual position feedback from the magnetic encoders, is processed by the discrete PID algorithm embedded on the Arduino microcontroller.

To protect the mechanical system, prevent stepper motor stalling, and strictly respect the motor's pull-out torque limits, saturation constraints are applied to the output command.

3. IMAGE PROCESSING FOR PRODUCT CLASSIFICATION

Once the robot control system was established, a robust image processing pipeline was developed to identify, classify, and locate products within the robot's workspace. This pipeline is executed on the host PC and is designed for real-time performance. The process consists of several sequential steps, from image acquisition to coordinate extraction.

3.1 Image acquisition and pre-processing

An image of the workspace is first captured by the system's camera at a resolution of 640 × 480 pixels. To reduce computational load and smooth out high-frequency noise, a Gaussian blur with a 5 × 5 kernel is applied as a pre-processing step. This subtle blurring helps improve the consistency of the subsequent color segmentation process.

3.2 Hue-Saturation-Value color space segmentation

Instead of using the traditional Red-Green-Blue (RGB) color space, this study opts to use the HSV color space [45-49].

The HSV color space divides colors into three separate components:

Hue (Color): Represents the primary color (red, blue, yellow, etc.).

Saturation (Saturation): The purity or intensity of the color.

Value (Brightness): The intensity of light.

Using the HSV color space offers several advantages over RGB for this application. The Hue component is largely invariant to changes in lighting conditions, such as shadows or highlights, which primarily affect the S and V components. This decoupling makes it significantly more robust and easier to define a specific color range for segmentation compared to the highly correlated R, G, and B channels. To classify products by color, the captured BGR image is first converted to the HSV color space using the OpenCV library [50-55], as illustrated in Figure 5.

Specific lower and upper HSV threshold values are then defined for each product color. For example, to detect red objects, a typical HSV range might be [0, 120, 70] to [10, 255, 255]. A binary mask is created for each color by retaining only the pixels that fall within these predefined ranges. This step effectively isolates the objects of interest from the background.

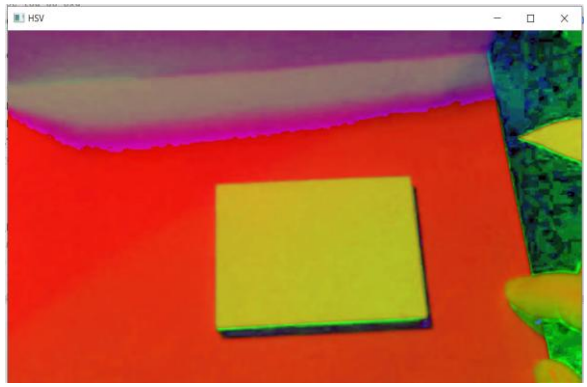

Figure 5. Image in Hue-Saturation-Value (HSV) color space

To ensure the reproducibility of the classification results, the final optimized parameters for the vision pipeline are summarized in Table 4.

Table 4. Vision system operational parameters

Parameter Group	Specific Metric	Value / Range
HSV Thresholds (Red)	Hue / Saturation / Value	[0, 93, 78] to [5, 255, 255]
HSV Thresholds (Green)	Hue / Saturation / Value	[30, 101, 34] to [70, 247, 210]
Morphology	Kernel Size	5×5 pixels
Object Size Filter	Minimum Area Threshold	1200 pixels
Bounding Box Limits	Min / Max (Width & Height)	40 to 120 pixels
Lens Correction	Reprojection Error	0.4198 pixels

Note: HSV = Hue-Saturation-Value

3.3 Morphological operations and contour detection

The initial binary masks may contain small imperfections or noise. To refine these masks, morphological operations are applied. An 'erosion' operation is first used to remove small, isolated white pixels (noise), followed by a 'dilation' operation to restore the main object's size and close any small gaps in it. This two-step process, known as 'opening', effectively cleans the mask while preserving the shape of the target objects.

Once a clean binary mask is obtained, the cv2.findContours() function from OpenCV is used to identify the continuous boundaries of the objects. This function returns a list of contours, with each contour being a vector of points that outlines a specific object. These contours are essential for localizing and classifying the objects based on their shape properties.

While color is the primary classification feature, the system employs geometric filtering to enhance reliability. By analyzing contour properties such as area and bounding box dimensions (as detailed in Table 4), the algorithm ensures that only target objects of specific sizes are processed, effectively rejecting non-target artifacts or background noise.

3.4 Object localization and data transmission

For each valid contour detected, the 'moment' of the contour is calculated. From these moments, the centroid coordinates (cx, cy) of the object can be determined. This centroid represents the object's center of mass in the 2D image plane and serves as the target coordinate for the robot's end-effector.

The system then classifies the object based on which color mask it was detected in. The final output of the image processing pipeline is a set of coordinates and a class label (e.g., 'red', 'green') for each object found. This information is formatted into a string and transmitted via the serial port to the Arduino Mega, which then commands the robot to execute the appropriate pick-and-place sequence.

3.5 Camera calibration and coordinate mapping

To ensure high precision in object localization and successful pick-and-place operations, a rigorous camera-to-robot calibration process was implemented. This involves two critical steps: lens distortion correction and coordinate transformation from the 2D image plane to the 3D robot workspace. Lens Distortion Correction: The system utilizes a Logitech C270 fixed-focus camera mounted orthogonally at a constant height above the workspace. To eliminate inherent lens distortions, we performed a calibration procedure using 20 images of a standard 9×6 OpenCV chessboard pattern. Using the cv2.calibrateCamera() algorithm, we extracted the camera's intrinsic matrix (K) and distortion coefficients (D):

$$K = \begin{bmatrix} 1425.87 & 0 & 691.16 \\ 0 & 1421.19 & 402.10 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$$D = \begin{bmatrix} 0.0937 \\ 0.4523 \\ 0.0067 \\ 0.0096 \\ -0.5745 \end{bmatrix}^T \quad (4)$$

The calibration achieved a highly accurate mean re-projection error of 0.4198 pixels, confirming effective rectification. Coordinate transformation (Pixel-to-mm): Since the workspace is a flat 2D plane with a constant Z-height, the mapping from undistorted pixel centroids (c_x, c_y) to physical Cartesian coordinates (X, Y) in millimeters is performed using a Homography transformation matrix (H). The transformation follows:

$$s \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H \begin{bmatrix} c_x \\ c_y \\ 1 \end{bmatrix} \quad (5)$$

where, s is a scaling factor. To validate this mapping, physical targeting tests were conducted over 10 random points. The system demonstrated a mean grasping positional error of ± 1.5 mm, which is well within the tolerance of the pneumatic vacuum end-effector, ensuring reliable sorting performance.

3.6 System integration and safety measures

To ensure reliability in industrial deployment, several software and hardware safety layers were integrated into the control architecture.

- Boundary handling and soft limits: The workspace is strictly constrained within software to prevent physical collisions with the robot's frame. All target coordinates (X, Y, Z) received via the serial port are validated against the hardware limits: X∈[0,300], Y∈[0,340], and Z ∈ [0,100] (mm). If a coordinate falls outside this bounding box, the command is rejected, and an error flag is sent back to the host PC.
- Serial communication reliability: To mitigate risks associated with packet loss or cable disconnection, the Arduino Mega firmware includes a communication watchdog. If no valid data frame is received within a 2-second timeout, the system enters a "Safe Mode," immediately disabling all actuator drivers and the vacuum pump.
- Failure handling and limitations: In the current version, the system prevents "ghost" operations by validating detection results; if the vision algorithm cannot locate an object, the robot remains stationary at the home position. However, detecting physical grasp failures (e.g., losing suction during transport) currently remains a challenge. Advanced protocols to detect such failures via image processing and initiate automatic retry sequences are identified as critical enhancements for future iterations of this platform.

4. RESULTS AND DISCUSSION

This section presents and analyzes the performance of the proposed system. First, the simulation results for the PID control strategy are detailed. Next, the experimental results from the physical prototype are presented, covering both control performance and classification accuracy. Finally, a comprehensive discussion analyzes these results, compares them with existing work, and addresses the system's limitations.

4.1 Simulation results

Numerical simulations were conducted in

MATLAB/Simulink package to validate the dynamic model and evaluate the performance of the designed PID control system prior to physical implementation. The robot was commanded to follow step inputs corresponding to typical pick-and-place positions. The position and velocity responses for each axis (Y, X, and Z) are shown in Figures 6-11 [56, 57].

As observed in the figures, the simulated response for each axis demonstrates excellent tracking performance. The output signal (e.g., 'd2') rapidly follows the desired setpoint with minimal overshoot and settles quickly, indicating a well-tuned controller. The velocity profiles show a rapid acceleration followed by a smooth deceleration, which is desirable for fast and stable motion. A 3D visualization of the robot model moving within the simulated environment is shown in Figure 12.

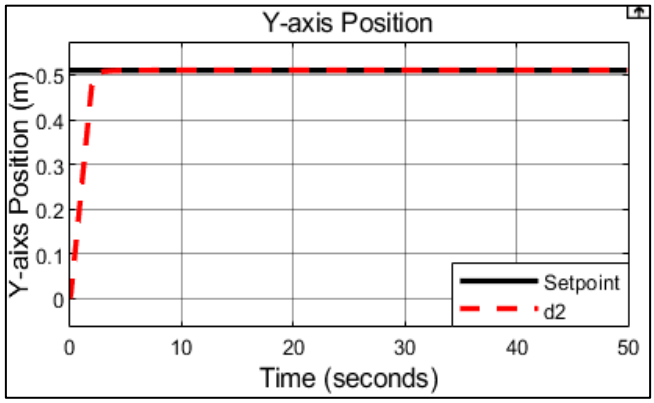

Figure 6. The response in position along the Y-axis

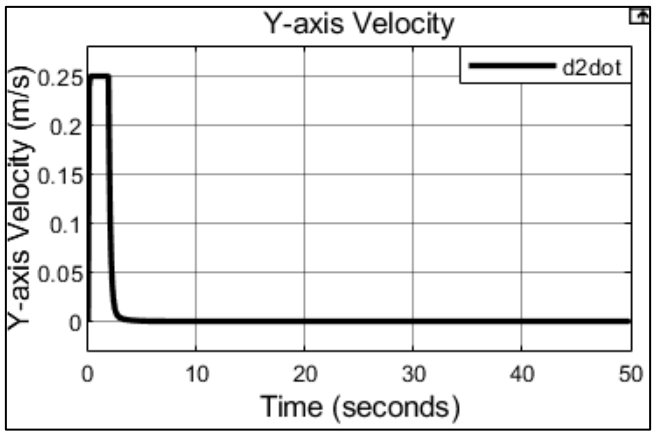

Figure 7. The response in velocity along the Y-axis

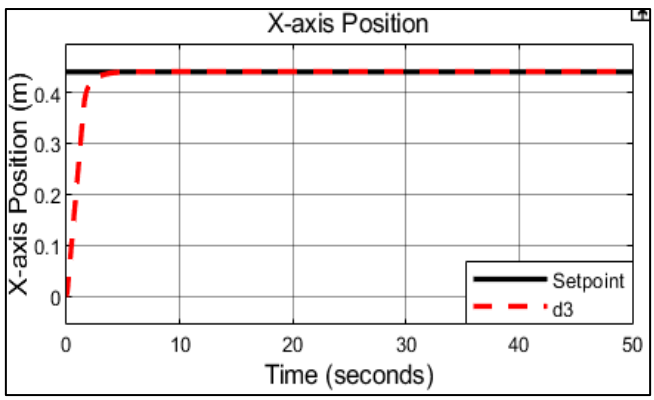

Figure 8. The response in position along the X-axis

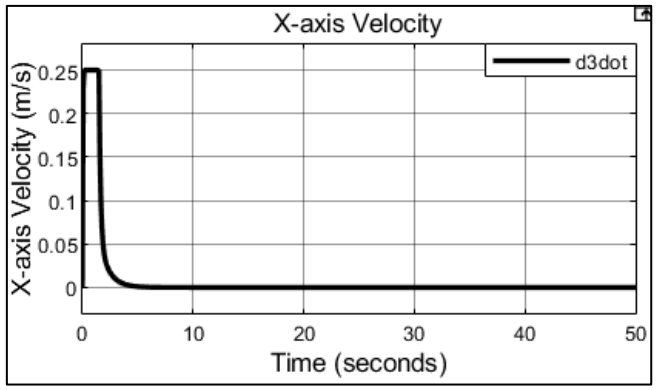

Figure 9. The response in velocity along the X-axis

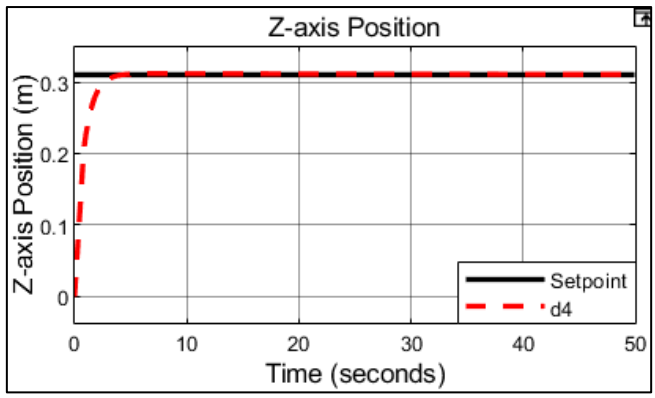

Figure 10. The response in position along the Z-axis

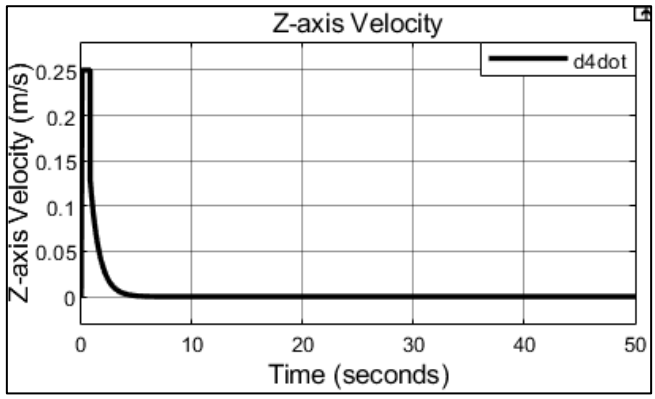

Figure 11. The response in velocity along the Z-axis

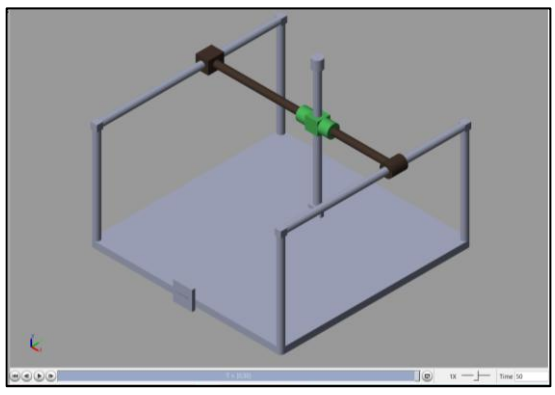

Figure 12. Simulating the robot model in MATLAB

To further test the controller's robustness, square wave inputs were used to simulate repetitive back-and-forth

movements, as shown in Figures 13-15. The controller consistently demonstrates its ability to track these dynamic changes with high fidelity, though a slight tracking lag is visible during rapid transitions, as expected.

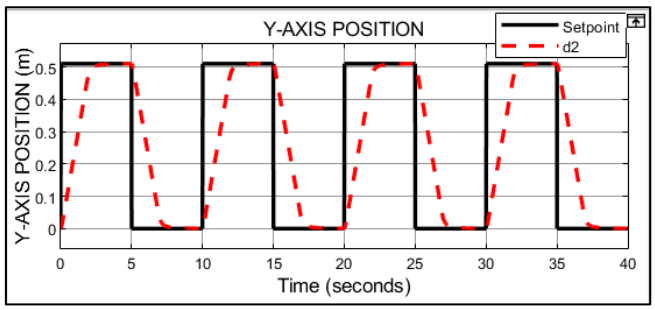

Figure 13. Response of Y-axis position with Proportional-Integral-Derivative (PID) controller

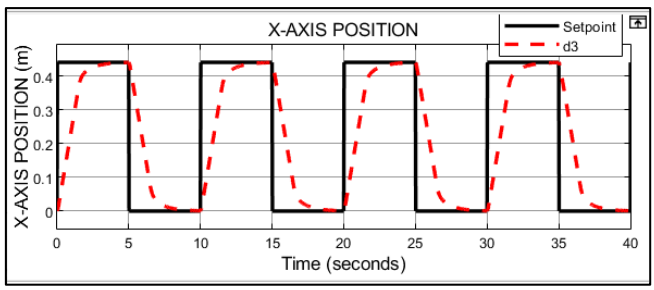

Figure 14. Response of X-axis position with Proportional-Integral-Derivative (PID) controller

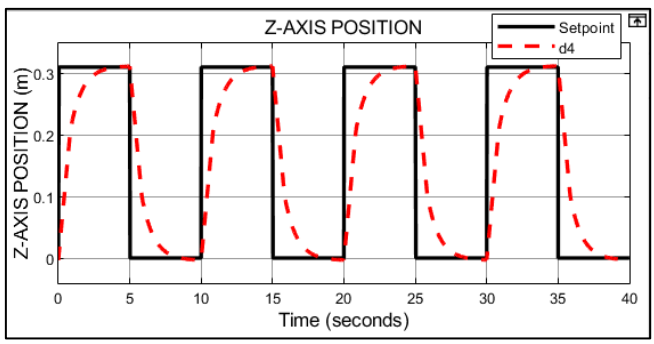

Figure 15. Response of Z-axis position with Proportional-Integral-Derivative (PID) controller

4.2 Experimental results

The validated control and image processing algorithms were deployed on the physical prototype, as shown in Figure 16. To facilitate real-time operation and parameter monitoring during the trials, a custom graphical user interface (GUI) was utilized (Figure 17). Experiments were conducted to evaluate two key aspects: the real-world performance of the PID motion controller and the accuracy of the vision-based classification system.

- **Control system performance**

The robot was commanded to follow the same step trajectories used in the simulation. The actual position responses on the Y and X axes were captured and are shown in Figures 18 and 19. The experimental results closely mirror the simulation, confirming the accuracy of the dynamic model and the effectiveness of the controller. However, the physical system exhibits slightly longer settling times and larger tracking errors, which can be attributed to unmodeled dynamics such as static friction, mechanical backlash, and motor nonlinearities.

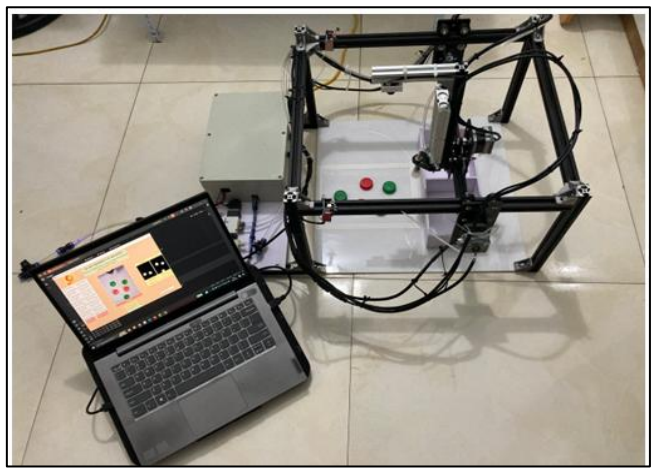

Figure 16. The real system model for experiments

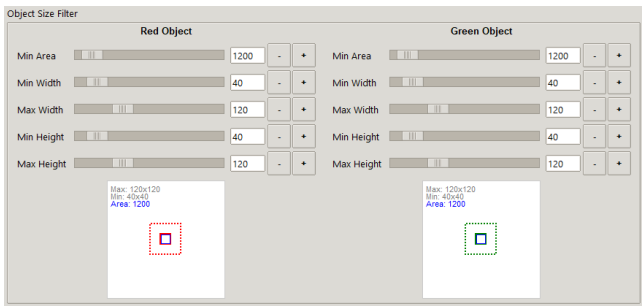

Figure 17. Adjust the operating parameters of the vision system on the software interface

Despite these minor unmodeled dynamics, the physical system demonstrated exceptional stability, largely due to the critical interplay between the custom software PID algorithm and the hardware characteristics. First, the Proportional gain (K_p) dynamically maps the positional error to the step frequency, inherently generating a smooth deceleration profile as the end-effector approaches the target, thereby preventing mechanical shocks. Second, finding a stable PID parameter basin was highly straightforward, yielding near-zero overshoot. This hardware-assisted stability occurs because once the PID algorithm accurately drives the positional error to zero and halts the pulse output, the substantial holding torque of the stepper motors instantly locks the mechanism, naturally suppressing any inertia-driven overshoot typical of DC servo systems.

To provide a highly auditable and standardized evaluation of the tracking performance, the classical step-response metrics for the physical prototype are systematically reported in physical units in Table 5.

Table 5. Physical step-response performance metrics

Metric	X-axis	Y-axis
Overshoot (%)	~ 0%	~ 0%
Rise Time (s)	2.2	2.0
Settling Time (s)	4.0	3.5
Steady-State Error (mm)	< 3.5	< 2.8

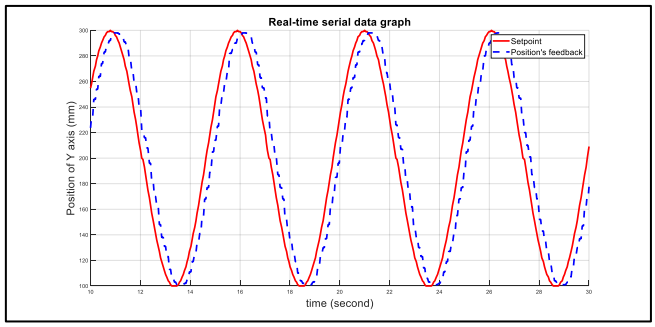

Figure 18. The response of position along the Y-axis

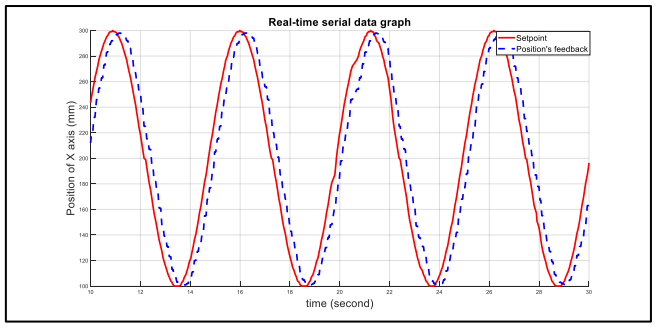

Figure 19. The response of position along the X-axis

- **Image processing and classification performance**

The performance of the image processing algorithm was evaluated using a dataset of 150 images of red and green circular objects placed in various positions within the workspace under typical laboratory lighting conditions. The system's task was to correctly classify the color of each object and determine its coordinates. The user interface displaying the real-time detection results is shown in Figure 20.

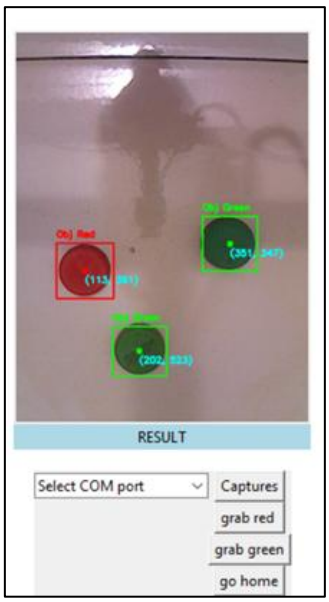

Figure 20. The user interface on the computer

The system achieved an overall classification accuracy of 92.7%. The detailed distribution of these results is illustrated in the confusion matrix in Table 6. Furthermore, comprehensive evaluation metrics, including precision, recall, and F1-score, are summarized in Table 7.

To provide a rigorous evaluation of the system's operational speed, the previously reported "0.5 seconds per product" was further analyzed. This duration represents the complete software and communication cycle, measured from the moment the camera capture command is initiated until the Arduino microcontroller successfully receives the coordinate payload via the serial UART interface.

The vision pipeline was executed on a host PC (Intel Core i5, 16GB RAM) running Python 3.9 and OpenCV 4.5.3. Through rigorous profiling over multiple runs, it was found that the pure vision processing component (HSV masking and centroid extraction) is remarkably efficient, taking an average of only 29.0 ms (± 5.1 ms). The remainder of the 0.5s software cycle is attributed to serial transfer overhead and buffer synchronization between the PC and the Arduino.

The comprehensive timing breakdown, including the physical robot motion and vacuum actuation delays, is summarized in Table 8. The results indicate that while the computational vision task is nearly instantaneous, the overall throughput is primarily limited by the mechanical constraints of the Cartesian frame and the pneumatic vacuum system, resulting in a true full cycle time of approximately 4.0 seconds.

Table 6. Confusion matrix

	Predicted Red	Predicted Green	Total Actual
Actual Red	70	5	75
Actual Green	6	69	75
Total Predicted	76	74	150

Table 7. Classification performance metrics

Class	Precision	Recall	F1-Score
Red	0.921	0.933	0.927
Green	0.932	0.920	0926

Table 8. System timing breakdown per product cycle

Operational Stage	Mean Time (ms)	Std. Deviation (ms)
1. Software & Communication Cycle (0.5s)		
Vision Processing (Acquisition & Process)	29.0	5.1
Serial Transfer & Arduino Sync	~471.0	-
2. Physical Actuation Cycle		
Robot Motion (X/Y/Z axes)	~2800.0	~100.0
Vacuum Actuation (Grasp / Release)	~700.0	~50.0
Total Full Cycle Time	~4000.0	-

4.3 Error analysis and robustness testing

To rigorously evaluate the external validity of the proposed HSV-based vision system, two additional stress-test scenarios were conducted: a Lighting Variation test and a Background Clutter test. These experiments aim to identify the operational boundaries and failure cases of the color-based classification approach.

Figures 21 and 22 illustrate sample results from these stress tests. In Figure 21, the algorithm successfully isolates targets against complex textures (e.g., wood grain, metallic parts). However, Figure 22 explicitly showcases failure cases. Severe

flash glare washes out the object's color, causing the HSV values to fall outside the predefined range and resulting in missed detections. These results confirm that while efficient, the HSV approach requires controlled illumination to maintain peak performance.

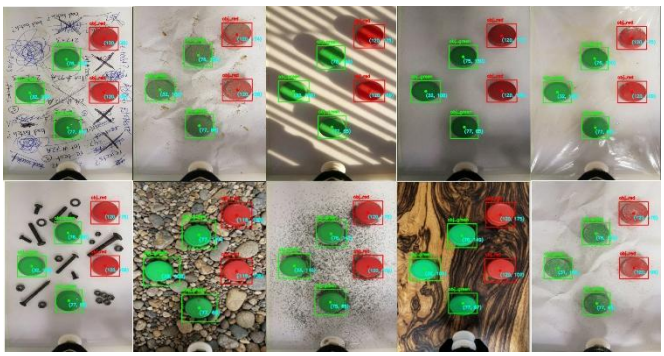

Figure 21. An illustration of stress tests – successful isolate targets

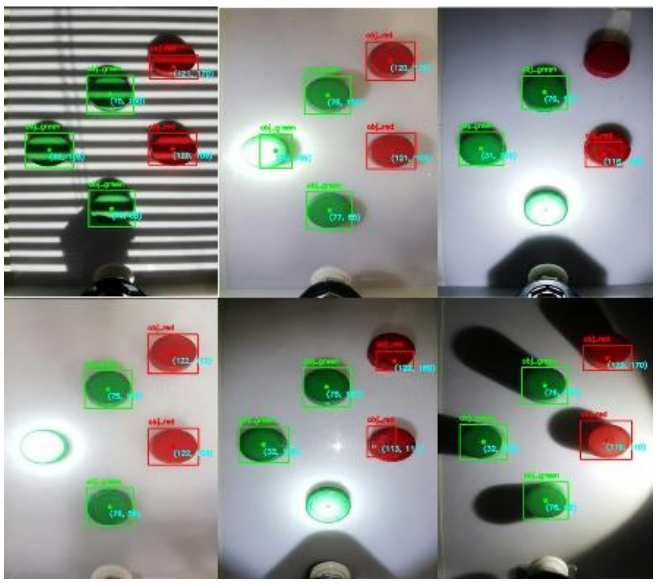

Figure 22. An illustration of stress tests – failure cases

To quantify the robustness of the HSV algorithm, a sensitivity analysis was performed by shifting the Hue (H) and Saturation (S) thresholds by ± 10%. As shown in Table 9, the system maintains high accuracy within a narrow band, but performance degrades significantly when Saturation thresholds are too restrictive, leading to missed detections under shadows.

Table 9. Hue-Saturation-Value (HSV) threshold sensitivity analysis

Threshold Shift (Δ)	Classification Accuracy (%)	Primary Failure Mode
-10%(More inclusive)	89.4%	False positives (Background noise)
0%(Baseline)	92.7%	Optimal performance
+10%(More restrictive)	84.2%	False negatives (Missed objects in shadows)

Regarding mechanical reliability, a repeatability test was conducted by commanding each axis to return to a reference point 10 times from different directions. The mean bidirectional positioning error was measured at ±0.8 mm for the X/Y axes and ± 0.5 mm for the Z-axis. This high level of repeatability confirms that the custom closed-loop PID control effectively mitigates mechanical backlash and friction during long-term operation.

4.4 Global discussion and comparative analysis

The experimental results confirm the feasibility and effectiveness of the proposed integrated system. The close correlation between simulated and actual control performance validates the dynamic model and the empirical PID tuning methodology. As detailed in Table 9, the controller demonstrates high precision, maintaining a mean tracking error of less than 3% in physical experiments, which is highly acceptable for industrial pick-and-place tasks. Regarding the vision pipeline, the initial claim of 0.5 seconds was found to be a conservative estimate for the entire software-communication cycle. Detailed profiling in Table 10 reveals that the core HSV-based algorithm is exceptionally efficient, requiring only 29.0 ms per frame. This efficiency allows the system to remain responsive even on low-cost hardware. While the 92.7% accuracy achieved under controlled lighting is promising, our new stress tests (Section 4.3) provide a more nuanced understanding of the system's limits. The identified failure cases under extreme glare underscore that while the system is cost-effective, it requires consistent illumination for maximum reliability.

Table 10. Comparative performance of vision algorithms on the same dataset (150 images)

Method	Accuracy (%)	Mean Processing Time (ms)
RGB Thresholding (Baseline 1)	~ 82.5%	25.4
HSV without Morphology (Baseline 2)	~ 87.8%	26.1
Proposed (HSV + Morph + Filter)	92.7%	29.0

Note: HSV = Hue-Saturation-Value

The selection of the HSV color space over the traditional RGB model is motivated by its inherent robustness to environmental lighting drift; the decoupling of chromaticity from luminance allows for more stable color segmentation in non-ideal industrial settings. Furthermore, the addition of morphological opening operations and area filtering is critical for real-world reliability. These post-processing steps effectively eliminate small, isolated noise pixels and small artifacts that would otherwise introduce significant jitter in the centroid coordinates. By achieving an accuracy of 92.7% within a 29.0 ms window, this integrated vision pipeline provides a superior balance between precision and computational efficiency compared to standard baselines.

5. CONCLUSIONS

This study successfully designed, implemented, and validated an integrated system combining a 3-DOF Cartesian robot with a real-time vision system for automated product classification. By developing a dynamic model and applying a well-tuned PID control strategy, the system demonstrated

excellent motion control, with experimental results showing a mean tracking error of less than 3% and closely matching simulation performance. The computationally efficient image processing pipeline, based on the HSV color space, proved effective for the classification task, achieving an overall accuracy of 92.7%. While the core vision processing (capture to coordinate calculation) requires an average of only 29.0 ms, the complete software and communication cycle was recorded at 0.5 seconds. Experimental results confirm that the overall system throughput is primarily governed by mechanical actuation, with a total pick-and-place cycle time of approximately 4.0 seconds. However, extensive stress testing indicates that the system's performance is susceptible to environmental factors, particularly extreme specular reflections, and complex background clutter, which can degrade classification accuracy. This highlights the necessity of controlled lighting for practical industrial deployment. The key contribution of this work lies in the holistic integration and rigorous quantitative validation of a low-cost system, providing a practical and accessible framework for small- to medium-scale industrial automation.

Despite the promising results, certain limitations were identified, primarily the vision algorithm's sensitivity to significant variations in ambient lighting. To address this, future work will focus on replacing the current HSV-based pipeline with a robust object detection engine. This engine will utilize advanced deep learning models (such as YOLOv8 or Vision Transformers) trained on large-scale, diverse datasets. Such a transition will overcome the inherent weaknesses of color-based segmentation, providing higher resilience to illumination drift and specular reflections while enabling more complex multi-class classification based on intrinsic geometric features. Further research will also aim to improve the system's mechanical capabilities by designing more versatile end-effectors for handling a wider variety of products. Finally, optimizing the control and processing algorithms to further reduce the overall cycle time will be a priority to meet higher throughput demands in practical applications.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the reviewers and editors for their insightful feedback and valuable suggestions, which have greatly contributed to the improvement of this work.

REFERENCES

[1] El Fadel, N. (2025). Facial recognition algorithms: A systematic literature review. Journal of Imaging, 11(2): 58. https://doi.org/10.3390/jimaging11020058

[2] Vashisht, M., Kumar, B. (2020). A survey paper on object detection methods in image processing. In 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, pp. 1-4. https://doi.org/10.1109/ICCSEA49143.2020.9132871

[3] Kim, S., Mai, T.D., Han, S., Park, S., Nguyen, D.T., So, J., Singh, K., Cha, M. (2022). Active learning for human-in-the-loop customs inspection. IEEE Transactions on Knowledge and Data Engineering, 35(12): 12039-12052. https://doi.org/10.1109/TKDE.2022.3144299

[4] Jayaswal, K., Palwalia, D.K., Kumar, S. (2021). Performance investigation of PID controller in trajectory control of two-link robotic manipulator in medical robots. Journal of Interdisciplinary Mathematics, 24(2): 467-478. https://doi.org/10.1080/09720502.2021.1893444

[5] Pagano, S., Russo, R., Savino, S. (2020). A vision guided robotic system for flexible gluing process in the footwear industry. Robotics and Computer-Integrated Manufacturing, 65: 101965. https://doi.org/10.1016/j.rcim.2020.101965

[6] Arévalo-Hernández, J.L., Rubio-Espino, E., Ponce-Ponce, V.H., Sossa, H. (2021). Vision assisted pick and place robotic machine. International Journal of Combinatorial Optimization Problems and Informatics, 12(3): 20. https://doi.org/10.61467/2007.1558.2021.v12i3.244

[7] Kuncan, F., Öztürk, S., Keleş, F. (2022). Image processing-based realization of servo motor control on a cartesian robot with Rexroth PLC. Turkish Journal of Engineering, 6(4): 320-326. https://doi.org/10.31127/tuje.1004169

[8] Sujith, R., Kumar, R.A., Vishnu, H., Dhanesh, S., Sudheer, A.P. (2021). Experimental investigation and numerical validation of neuro fuzzy-based Cartesian robot for soft material cutting. Journal of Applied Research and Technology, 19(5): 420-436. https://doi.org/10.14482/indes.30.1.303.661

[9] Jitviriya, W., Pudchuen, N., Phunopas, A., Hayashi, E. (2020). Design and modeling of an automatic cartesian farming robot. In 2020 International Conference on Artificial Life and Robotics (ICAROB2020), B-Con Plaza, Beppu, Oita, Japan, pp. 448-451. https://doi.org/10.5954/ICAROB.2020.GS4-1

[10] Muharom, S., Wardhana, D.A.P., Septyan, M., Pambudi, A.D., Indrawan, R.W., Firmansyah, R.A., Awanda, A. (2023). Identifying strawberry ripeness level using camera for cartesian robot application based on microcontroller. In 2023 Sixth International Conference on Vocational Education and Electrical Engineering (ICVEE), Surabaya, Indonesia, pp. 7-12. https://doi.org/10.1109/ICVEE59738.2023.10348260

[11] Saadeh, M., Koutsougeras, C. (2023). Development of a cartesian robot with image processing and grasp detection. In 10th ECCOMAS Thematic Conference on Smart Structures and Materials, Patras, Greece, pp. 1246-1257. https://doi.org/10.7712/150123.9872.442489

[12] Du, G., Zhang, B., Li, C., Gao, B., Liu, P.X. (2020). Natural human–machine interface with gesture tracking and cartesian platform for contactless electromagnetic force feedback. IEEE Transactions on Industrial Informatics, 16(11): 6868-6879. https://doi.org/10.1109/TII.2020.2966756

[13] Martínez, M.A.G., Sanchez, I.Y., Chávez, F.M. (2021). Development of automated virtual CNC router for application in a remote mechatronics laboratory. In 2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Mauritius, Mauritius, pp. 1-6. https://doi.org/10.1109/ICECCME52200.2021.9590986

[14] Dağdelen, Y., Akyüz, F., Feyzioğlu, A., Toptaş, E. (2023). Design and analysis of a 4-axis cartesian robot for unloading plastic injection machines in industrial applications. Journal of Mechatronics and Artificial Intelligence in Engineering, 4(2): 104-111.

https://doi.org/10.21595/jmai.2023.23553
[15] González-Arriaga, D.M., Vargas-Treviño, M.A.D., Vergara-Limon, S., De León, C.L.C.D., López-Gómez, J., Vargas-Treviño, M., Guerrero-García, J. (2023). Design of a software platform to generate convolutional neural networks for the parametric identification of a cartesian robot. IEEE Access, 11: 63371-63387. https://doi.org/10.1109/ACCESS.2023.3289078
[16] Park, Y., Lee, J., Sim, D., Cho, Y., Park, C. (2025). Designing spiking neural network-based reinforcement learning for 3D robotic arm applications. Electronics, 14(3): 578. https://doi.org/10.3390/electronics14030578
[17] Pham, D.T., Nguyen, T.V., Le, H.X., Nguyen, L., Thai, N.H., Phan, T.A., Pham, H.T., Duong, A.H., Bui, L.T. (2020). Adaptive neural network based dynamic surface control for uncertain dual arm robots. International Journal of Dynamics and Control, 8(3): 824-834. https://doi.org/10.1007/s40435-019-00600-2
[18] Satsangee, G.R., Al-Musaibeli, H., Ahmad, R. (2024). A defect detection method based on YOLOv7 for automated remanufacturing. Applied Sciences, 14(13): 5503. https://doi.org/10.3390/app14135503
[19] Naranjo-Campos, F.J., Victores, J.G., Balaguer, C. (2024). Method for bottle opening with a dual-arm robot. Biomimetics, 9(9): 577. https://doi.org/10.3390/biomimetics9090577
[20] Zhao, J. (2022). Surface defect classification with vision transformer. In 2022 3rd International Conference on Intelligent Design (ICID), Xi'an, China, pp. 124-128. https://doi.org/10.1109/ICID57362.2022.9969746
[21] Pratama, D.W., Ismail, M., Nurraudah, R., Rifai, A.P., Nguyen, H.T. (2025). Classification of metal surface defects using convolutional neural networks (CNN). Green Intelligent Systems and Applications, 5(1): 93-105. https://doi.org/10.53623/gisa.v5i1.581
[22] Li, P., Chen, M., Ji, C., Zhou, Z., Lin, X., Yu, D. (2024). An agent-based method for feature recognition and path optimization of CNC machining trajectories. Sensors, 24(17): 5720. https://doi.org/10.3390/s24175720
[23] Zhang, S., Li, S., Li, X., Xiong, Y., Xie, Z. (2022). A human-robot dynamic fusion safety algorithm for collaborative operations of cobots. Journal of Intelligent & Robotic Systems, 104(1): 18. https://doi.org/10.1007/s10846-021-01534-8
[24] Xiang, T.W., Ting, L.S., Chet, K.V., Soong, L.W. (2025). Design and development of an automated fruit quality classifier and sorter using a robotic arm. Edelweiss Applied Science and Technology, 9(10): 32-49. https://doi.org/10.55214/2576-8484.v9i10.10340
[25] Nasab, S.D.M., Beiranvand, A., Masouleh, M.T., Bahrami, F., Kalhor, A. (2022). Design and development of a multi-axis force sensor based on the hall effect with decouple structure. Mechatronics, 84: 102766. https://doi.org/10.1016/j.mechatronics.2022.102766
[26] Yadav, S.K., Shahi, S. (2024). Safe human-robot collaboration in dynamic environments: An AI-powered situation awareness perspective. International Journal of Tropical Medicine, 19(4): 92-98. https://doi.org/10.36478/makijtm.2024.4.92.98
[27] Terras, N., Pereira, F., Ramos Silva, A., Santos, A.A., Lopes, A.M., Silva, A.F.D., Cartal, L.A., Apostolescu, T.C., Badea, F., Machado, J. (2025). Integration of deep learning vision systems in collaborative robotics for real-time applications. Applied Sciences, 15(3): 1336. https://doi.org/10.3390/app15031336
[28] Zhang, J. (2024). Research on visual recognition and positioning of industrial robots based on big data technology. Applied Mathematics and Nonlinear Sciences, 9(1): 4. https://doi.org/10.2478/amns.2023.2.00193
[29] Paredes, F., Herrojo, C., Martín, F. (2021). Position sensors for industrial applications based on electromagnetic encoders. Sensors, 21(8): 2738. https://doi.org/10.3390/s21082738
[30] Wang, S., Ma, R., Cao, F., Luo, L., Li, X. (2024). A review: High-precision angle measurement technologies. Sensors, 24(6): 1755. https://doi.org/10.3390/s24061755
[31] Alnufaie, L. (2023). Nonsingular fast terminal sliding mode controller for a robotic system: A fuzzy approach. IEEE Access, 11: 75522-75527. https://doi.org/10.1109/ACCESS.2023.3288000
[32] Hu, S., Wan, Y., Liang, X., Zhang, S. (2024). Adaptive fast terminal sliding mode control of robotic manipulators based on dynamics feedforward. In Proceedings of the 2024 3rd International Symposium on Intelligent Unmanned Systems and Artificial Intelligence, Qingdao, China, pp. 344-348. https://doi.org/10.1145/3669721.3669751
[33] Liu, W., Liu, L., Zhang, D., Cheng, J. (2025). Nonsingular fast terminal sliding mode control of uncertain robotic manipulator system based on adaptive fuzzy wavelet neural network. International Journal of Fuzzy Systems, 27(3): 898-911. https://doi.org/10.1007/s40815-024-01818-9
[34] He, Y., Wang, H., Tian, Y., Zhou, X. (2023). Model-free based nonsingular fast terminal sliding mode control for Gough-Stewart platform. In 2023 42nd Chinese Control Conference (CCC), Tianjin, China, pp. 2370-2375. https://doi.org/10.23919/CCC58697.2023.10241153
[35] Li, X., Gao, X., Sun, L., Zheng, D., Shi, H., Lei, C., Hu, L., Zong, Y. (2022). The adaptive neural network fuzzy sliding mode control for the 3-RRS parallel manipulator. Advances in Mechanical Engineering, 14(9): 16878132221126112. https://doi.org/10.1177/16878132221126112
[36] Rahali, H., Zeghlache, S., Cherif, B.D.E., Benyettou, L., Djerioui, A. (2025). Robust adaptive fuzzy type-2 fast terminal sliding mode control of robot manipulators in attendance of actuator faults and payload variation. Electrical Engineering & Electromechanics, (1): 31-38. https://doi.org/10.20998/2074-272X.2025.1.05
[37] Chen, J., Zhang, H., Tang, Q., Zhang, H. (2024). Adaptive fuzzy sliding mode control of the manipulator based on an improved super-twisting algorithm. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 238(10): 4294-4306. https://doi.org/10.1177/09544062231214835
[38] Marrero, D., Kern, J., Urrea, C. (2024). A novel robotic controller using neural engineering framework-based spiking neural networks. Sensors, 24(2): 491. https://doi.org/10.3390/s24020491
[39] Aly, A.A., Hsia, K.H., El-Sousy, F.F., Mobayen, S., Alotaibi, A., Mousa, G., Le, D.N. (2022). Adaptive neural backstepping control approach for tracker design of wheelchair upper-limb exoskeleton robot system. Mathematics, 10(22): 4198. https://doi.org/10.3390/math10224198

[40] Zhang, Z., Cui, P., An, A. (2023). Adaptive neural tracking control for upper limb rehabilitation robot with output constraints. IET Cyber-Systems and Robotics, 5(4): e12104. https://doi.org/10.1049/csy2.12104
[41] Ma, J., Wang, H., Qiao, J. (2022). Adaptive neural fixed-time tracking control for high-order nonlinear systems. IEEE Transactions on Neural Networks and Learning Systems, 35(1): 708-717. https://doi.org/10.1109/TNNLS.2022.3176625
[42] Yamazaki, K., Vo-Ho, V.K., Bulsara, D., Le, N. (2022). Spiking neural networks and their applications: A review. Brain Sciences, 12(7): 863. https://doi.org/10.3390/brainsci12070863
[43] Azeez, M.I., Abdelhaleem, A.M.M., Elnaggar, S., Moustafa, K.A., Atia, K.R. (2023). Optimization of PID trajectory tracking controller for a 3-DOF robotic manipulator using enhanced Artificial Bee Colony algorithm. Scientific Reports, 13(1): 11164. https://doi.org/10.1038/s41598-023-37895-3
[44] Ghani, N.M.A., Othman, A., Hashim, A.A.A., Nasir, A. N.K. (2023). Comparative analysis of PID and fuzzy logic controllers for position control in double-link robotic manipulators. Journal of Intelligent Systems and Control, 2(4): 183-196. https://doi.org/10.56578/jisc020401
[45] Zhang, J., Li, M., Feng, Y., Yang, C. (2020). Robotic grasp detection based on image processing and random forest. Multimedia Tools and Applications, 79(3): 2427-2446. https://doi.org/10.1007/s11042-019-08302-9
[46] Al-Mashhadani, Z., Chandrasekaran, B. (2020). Autonomous ripeness detection using image processing for an agricultural robotic system. In 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, pp. 0743-0748. https://doi.org/10.1109/UEMCON51285.2020.9298168
[47] Ali, M., Kumar, S., Pal, R., Singh, M.K., Saini, D. (2023). Graph-and machine-learning-based texture classification. Electronics, 12(22): 4626. https://doi.org/10.3390/electronics12224626
[48] Xiao, F., Wang, H., Li, Y., Cao, Y., Lv, X., Xu, G. (2023). Object detection and recognition techniques based on digital image processing and traditional machine learning for fruit and vegetable harvesting robots: An overview and review. Agronomy, 13(3): 639. https://doi.org/10.3390/agronomy13030639
[49] Khediri, N., Ammar, M.B., Kherallah, M. (2021). Comparison of image segmentation using different color spaces. In 2021 IEEE 21st International Conference on Communication Technology (ICCT), Tianjin, China, pp. 1188-1192. https://doi.org/10.1109/ICCT52962.2021.9658094
[50] Zhang, X., Toudeshki, A., Ehsani, R., Li, H., Zhang, W., Ma, R. (2022). Yield estimation of citrus fruit using rapid image processing in natural background. Smart Agricultural Technology, 2: 100027. https://doi.org/10.1016/j.atech.2021.100027
[51] Chen, P., Elangovan, V. (2020). Object sorting using faster R-CNN. arXiv preprint arXiv:2012.14840. https://doi.org/10.48550/ARXIV.2012.14840
[52] Sharma, A., Pathak, J., Prakash, M., Singh, J.N. (2021). Object detection using OpenCV and Python. In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, pp. 501-505. https://doi.org/10.1109/ICAC3N53548.2021.9725638
[53] Sulistiyowati, I., Ichsan, H.M., Anshory, I. (2024). Object sorting conveyor with detection color using ESP-32 camera Python based on Open-CV. Journal of Electrical Engineering and Computer Sciences, 9(1): 61-68. https://doi.org/10.54732/jeecs.v9i1.7
[54] Hira, S., Lande, S. (2022). Detection of fruit ripeness using image processing. International Journal of Health Sciences, 6(S6): 3874-3886. https://doi.org/10.53730/ijhs.v6nS6.10146
[55] Ayyıldız, M., Çiftçi, Y.K., Kilecioğlu, K., Arslan, Ö.F. (2024). Fruit sorting automation; cartesian robot and conveyor design. Duzce University Journal of Science and Technology, 12(3): 1627-1639. https://doi.org/10.29130/dubited.1400615
[56] Noventino, T.H., Rosa, M.R., Fuadi, A.Z. (2022). PID control design and kinematic modelling of 3-DoF robot manipulator. In 2022 International Conference on Electrical Engineering, Computer and Information Technology (ICEECIT), Jember, Indonesia, pp. 88-94. https://doi.org/10.1109/ICEECIT55908.2022.10030325
[57] Hoa, T.D., Van Khiem, N., Thien, T.D. (2021). Design, simulation, fabrication and control a 3-DOF planar robotic manipulator. Journal of Technical Education Science, HCM City University of Science and Education, 64. https://doi.org/10.54644/jte.64.2021.87