International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 02 Issue: 12 December 2025
Page No: 4427-4433
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0651

Development of Vision Based Sorting System Using Machine Learning for Automated Material Classification

Dravid Rajan T[1], Arockia Justin J[2], Dhinakaran M[3], Akash V[4], Amuthan K[5], Fawaz A[6]
[1]UG Scholar, Mechatronics Engineering, Hindusthan College of Engineering and Technology, Coimbatore, Tamil Nadu, India.
[2,3,4,5,6]UG Scholar, Mechatronics Engineering, Hindusthan College of Engineering and Technology, Coimbatore, Tamil Nadu, India.
Emails: *dravidrajan4@gmail.com[1], arockiajustin95gmail.com[2], dhina2566@gmail.com[3], akashvenkatesan29@gmail.com[4], amuthan2838@gmail.com[5], cupidmaster18@gmail.com[6]*

Abstract
The Development of a Vision-Based Sorting System Using Machine Learning for Automated Material Classification aims to design and implement an intelligent sorting mechanism that automates the identification and segregation of materials based on visual features. The proposed system integrates a camera-based vision module, machine learning algorithms, and a robotic sorting mechanism to classify materials efficiently. Using a deep learning model trained on a custom dataset, the system can Detect and categorize objects in real time. The detected information are transmitted to an Arduino-based control unit, which operates a conveyor belt and robotic arm to place each item in its corresponding bin. This prototype demonstrates the potential of computer vision and automation in reducing manual labor and increasing accuracy in industrial sorting applications.

Keywords: *Vision-Based Sorting; Machine Learning; Robotic Arm Automation; Material Classification.*

1. Introduction
In many industries, sorting and classification of materials are commonly performed using traditional sensor-based systems and still performed manually, which leads to human error, and inconsistency in results. Traditional sorting systems rely on fixed sensors that can only detect limited features. Even though the Existing Systems are simple to implement, they are limited by their ability to detect only predefined attributes. Sensor-based systems also struggle to handle variations in lighting, object positioning, and overlapping items on a conveyor. Any small disturbance can affect detection accuracy. Additionally, these systems require manual calibration whenever the material type changes, which reduces flexibility. To improve accuracy and automation, there is a need for an intelligent system that can visually detect and classify materials in real time. Therefore, there is a need to address the problem by developing a vision-based sorting system that uses machine learning and image processing techniques to automatically recognize and classify the materials. Recent improvement in machine learning have enabled the development of more intelligent systems that can automatically identify objects based on visual data. Such systems allow for more accurate, adaptable, and faster sorting in real-time environments compared to traditional methods.

1.1. Scope of the Work
The proposed system is designed as a small-scale prototype that can classify and sort-coloured blocks. However, the same concept can be extended for industrial purposes such as classification of materials (plastic, metal, paper), or inspecting manufactured parts. The prototype is designed to automatically detect and sort-coloured objects using a combination of machine learning, computer vision, and embedded control [1-3].

1.2. Objective of the Project
The main objectives of this project are:
1. To design and develop an automated sorting system using image processing.
2. To train a YOLO-based machine learning model for colour-based material-classification.
3. To integrate a Camera for image capture and real-time object detection.

4. To control a robotic arm through Arduino Uno for automated sorting.

2. Method

2.1. System Architecture

The system is designed to identify and sort materials automatically using a camera, machine learning, and a robotic arm. It mainly consists of four parts: an image capturing unit, a processing unit, a control unit, and an actuation unit. The ESP32-CAM module captures the image of the object placed on the conveyor belt. The image is then sent to a computer, where the YOLO model processes it to identify the color and type of material. Once the object is classified, the result is passed to the Arduino Uno, which controls the robotic arm to sort the object into the correct bin. The system works in a continuous loop. As each new object appears on the conveyor, the camera captures it, the YOLO model classifies it, and the robotic arm performs the sorting action. This setup helps to achieve a small-scale demonstration of vision-based automation [4-10].

2.2. Hardware Units

ESP32-CAM: Used for capturing images. It includes both a camera and a Wi-Fi module, making it easy to send data to the computer.

Arduino Uno: Acts as the main controller for the both robotic arm and Conveyor Belt. It receives commands from the computer and work accordingly.

Robotic-Arm: A servo-controlled Arm that performs the sorting task from the instructions of Arduino.

Conveyor Belt: A DC motor-based belt used to move objects for inspection by the instruction of Arduino.

Power Supply Unit: Delivers the required voltage and current to the ESP32-CAM, Arduino, Conveyor belt and servos.

2.3. Software Design

The software part handles image processing, object detection, and communication with the hardware.

The system mainly uses Python, YOLO, Roboflow, VS Code, and Arduino IDE for software development.

Steps in the Software Process

1. **Dataset Preparation:** Images of different colored materials are collected and labeled in Roboflow for training.
2. **Model Training:** The labeled dataset is trained using the YOLO model, which learns to detect and classify colors.
3. **Real-Time Detection:** During operation, the ESP32-CAM sends live images to the YOLO model running in Python on the laptop.
4. **Communication with Arduino:** After classification, the Python code sends a simple signal (like ‘R’, ‘G’, or ‘B’) through the serial port to the Arduino Uno.
5. **Robotic Control:** The Arduino program receives the signal and drives the servos of the robotic arm to move the object to the correct bin.

This combination of hardware and software allows real-time object recognition and automatic sorting with minimal human help.

2.4. Working Principle

The Working Principles are

1. The Conveyor Moves the Object.
2. The ESP32-CAM captures images of objects moving on the conveyor belt.
3. The captured frames are processed by a YOLO-based object detection model running on a laptop using Python.
4. The trained model identifies the Color of the Block in this prototype.
5. The result of classification is sent to the Arduino Uno through serial communication.
6. The Arduino controls the robotic arm to pick and place the object in its respective bin.
7. The entire system works automatically as the next object reaches the camera, the process repeats.

Each step is repeated continuously to enable real-time sorting on the conveyor belt.

2.5. Algorithm and Model Details

A deep learning-based object detection algorithm that divides an image into a grid and predicts bounding boxes and class probabilities directly from the image. YOLO means You Only Look Once. It offers real-time performance suitable for embedded vision applications. In this project, YOLO is used to detect color blocks trained from a dataset from Roboflow.

Steps: Prepare dataset images for red, green, and blue blocks, And Label images in Roboflow, Train YOLO

IRJAEH
International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 02 Issue: 12 December 2025
Page No: 4427-4433
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0651

model with labelled data, Use trained weights for real-time detection.

2.6. Software Programming

Table 1 Software tools and Platform Used

Software Tool	Purpose / Function
Roboflow	For dataset creation, annotation, and preprocessing
YOLOv8	For training and testing the object detection model
Python (VS Code)	For image processing, serial communication, and integration
Arduino IDE	For programming and controlling the servo motors
ESP32-CAM Web Interface / Libraries	For camera configuration and image streaming
OpenCV (Python Library)	For real-time image handling and frame processing

1. Dataset Preparation using Roboflow

To enable the YOLO model to detect and classify objects, a custom dataset was created using Roboflow, an online platform for dataset management and annotation.

Steps:

1. **Image Collection:** Multiple images of color blocks (Red, Green, and Blue) were captured under various lighting conditions using the ESP32-CAM.
2. **Uploading to Roboflow:** The captured images were uploaded to the Roboflow workspace.
3. **Annotation:** Each object (color block) was labeled with its respective class name – Red_Block, Green_Block, Blue_Block.
4. **Dataset Export:** The annotated dataset was exported in YOLO format and downloaded for training, Table 1.

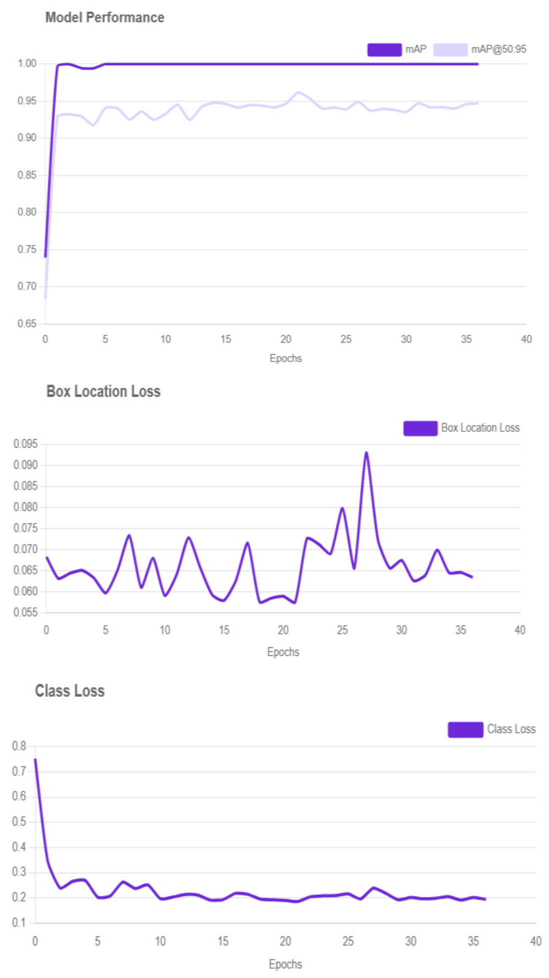

Figure 1 Model Performance Graph

The Figure 1 graphs show the training and validation accuracy improving over epochs, indicating that the model learned the color-based classification effectively. The model performed well with minimal loss and high precision.

2. Training the YOLO Model

The YOLO object detection algorithm was used for training the dataset. YOLO is chosen for its speed, accuracy, and integration with real-time applications.

Steps:

1. Install YOLO using Python (Ultralytics library).
2. Load the Roboflow dataset path in the YOLO configuration file.
3. Train the model using command-line training scripts in VS Code.
4. Evaluate model performance.
5. Save the best-performing model weights.

3. Python Program for Image Processing and Communication

The Python script is the central controller that integrates camera input, YOLO detection, and Arduino communication.

Modules Used:

1. opencv-python for video frame capture
2. serial for Arduino communication
3. ultralytics for YOLO model loading
4. time and numpy for logic control and array processing

Workflow:

1. Receive live video feed from ESP32-CAM over Wi-Fi.
2. Process each frame using YOLO for object detection.
3. Extract class label.
4. Send corresponding command to Arduino via Serial communication.
5. Repeat for each detected object.

4. Integration of Hardware and Software

The integration process connects all modules into a functioning system:

1. ESP32-CAM captures live video and streams it to Python via Wi-Fi.
2. YOLO model (in Python) processes the image and classifies objects.
3. Python serial communication sends the class label to Arduino Uno.
4. Arduino activates the Robotic Arm for sorting.

This integration enables real-time automated material classification and sorting using machine learning.

5. Data Flow and Control Flow:

Data Flow

The Data Flow represents how information moves through the system.

1. Object image from ESP32-CAM
2. ESP32-CAM sends image to YOLO model.
3. YOLO identifies object Color.
4. Python sends sorting command to Arduino
5. Arduino controls Robotic Arm by servo motor to sort object

Control Flow

The control flow defines how commands and feedback are managed.

1. System starts and initializes all modules.
2. Conveyor belt starts moving.
3. IR sensor detects object presence.
4. ESP32-CAM captures the image.
5. YOLO model processes and classifies it.
6. Python script sends signal to Arduino.
7. Arduino activates servo motor for sorting.
8. Conveyor moves next object for inspection.

The process runs continuously without manual intervention.

3. Results and Discussion

3.1. Results

All the hardware and software parts were successfully connected and tested together. We mainly checked how well the system could detect the color of the objects and how accurately it could sort them into their respective bins using the Robotic Arm, shown in Figure 2.

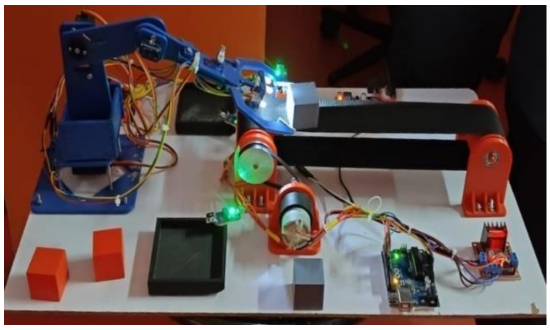

Figure 2 Prototype

The Testing setup was arranged as:

1. The conveyor belt was powered by a 60 RPM DC motor.
2. The ESP32-CAM was fixed in front of the belt to capture images of the blocks.
3. The Arduino Uno was connected to the laptop through a USB cable.
4. The trained YOLO model was running in VS Code using Python, continuously detecting objects from the ESP32-CAM stream.
5. Servo motors were placed at the end of the conveyor to push the blocks into bins based on color.

Detection and Classification Output

When an object moved in front of the camera, the YOLO model detected it and displayed a bounding box around the object with a label (Grey, Orange, or Blue).The program also displayed the confidence percentage, showing how sure the model was about its detection. If the confidence was more than 80%, we considered it a successful detection, Figure 3 & 4.

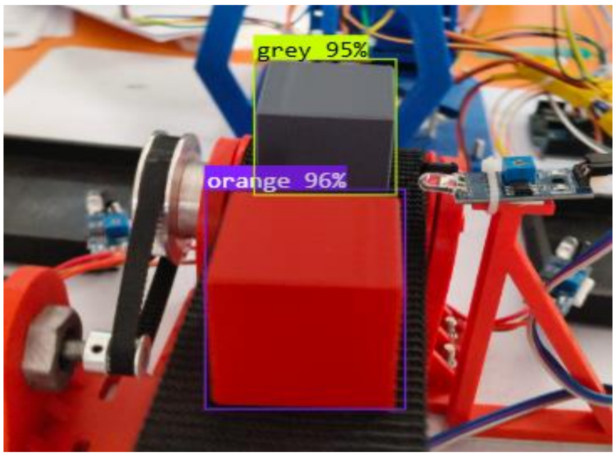

Figure 3 Output from YOLO During Detection with Confidence Percentage

Table 2 Output

Object Color	Confidence	Sorting Result
Grey	90-96%	Sorted Correctly
Orange	92-96%	Sorted Correctly
Blue	75-83%	Sorted Correctly

Accuracy of the System: We tested the system with 30 Color Blocks (10 of each color). The YOLO model correctly detected 26 out of 30 objects, giving an accuracy of 86%. The few incorrect detections happened mainly due to the Poor lighting conditions, Similar background color as the object, Slight motion blur during conveyor movement. These issues can be improved by better lighting and camera placement.

Table 3 Output

Total Samples	Correct Detections	Incorrect Detections	Accuracy
30	26	4	86%

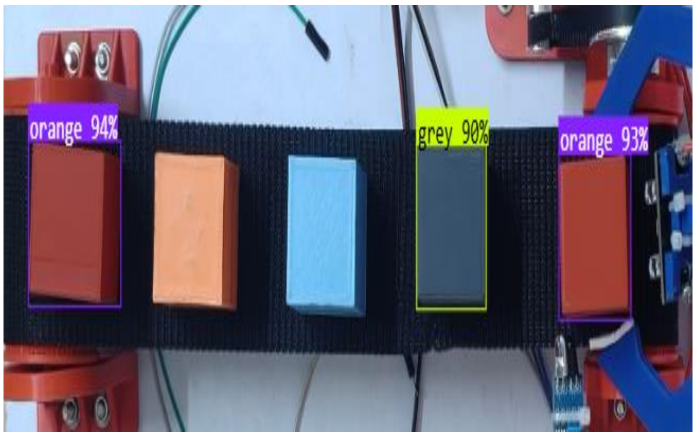

Figure 4 Object is Detected with a Bounding box and labeled as Grey, Orange

System Response Time: The average time between object detection and sorting action was around 2 to 3 seconds, which is quite fast for a prototype setup. Most of this time was taken by Capturing and processing the image through YOLO, and Sending commands to the Arduino. Still, the overall system speed was suitable for small-scale automation applications, shown in Table 2 & 3.

3.2. Discussion

From the test results, we can say that the proposed system worked efficiently for automatic color-based sorting. The combination of ESP32-CAM, YOLO, and Arduino Uno proved to be cost-effective and easy to implement. Some limitations were observed, The detection accuracy depends on light and camera position, The Wi-Fi stream from the ESP32-CAM sometimes lagged slightly, Robotic Arm accuracy decreases if the load is heavy. However, for a prototype, the performance was quite successful. Even though we used a simple prototype, the same concept can be extended to sort different materials, shapes, or even industrial components by retraining the YOLO model with new datasets. The system can be improved and upgraded in several ways in the future, The model can be trained to identify different materials, shapes, or textures, not just colors. A better-quality camera and faster microcontroller (like Raspberry Pi) can be used for smoother real-time performance. The system can be scaled up and connected with larger conveyor systems for factory or warehouse automation. The data from sorting operations can be uploaded to the cloud for remote monitoring and analytics. Using improved algorithms or edge AI hardware can make the system more intelligent and faster. Additional cameras can be used to detect objects from different angles for better accuracy.

Conclusion

In this system, we successfully designed and implemented a Vision-Based Sorting System using Machine Learning for Automated Material Classification. The main objective was to prove low-cost automation is possible using open-source tools and affordable components and to create a efficient prototype that can automatically detect and sort the coloured objects using image processing and machine learning techniques. The system combined the use of ESP32-CAM, YOLO model, Python, and Arduino Uno to perform real-time image detection and sorting. The ESP32-CAM captured images, the YOLO model processed and classified the objects, and the Arduino controlled the servo motors to perform the sorting action. After testing, the system achieved around 86% accuracy in object detection and sorting, which shows that machine learning-based approaches can be used effectively even in simple hardware setups. Through this work, we understood how artificial intelligence and automation can come together to make processes faster and reduce manual effort. Overall, the project met its goals and proved that low-cost automation is possible using open-source tools and affordable components.

Acknowledgements

I express my sincere gratitude to all those who supported and guided me throughout the development of this project. I extend my heartfelt thanks to my project supervisor for providing valuable insights, encouragement, and continuous guidance, which greatly contributed to the successful completion of this work. I am also thankful to the faculty members and my seniors for offering the necessary resources, technical assistance, and a conducive environment for experimentation and testing. I wish to acknowledge the support of my teammates for their cooperation and constructive suggestions throughout the project duration. Finally, I am deeply grateful to my family for their constant motivation, understanding, and support, which enabled me to pursue and complete this project successfully.

References

[1]. Prasath C. A., “Automated Waste Classification and Segregation Using YOLOv5 and Robotic Arm,” *Frontiers in Mathematical and Computational Research*, vol. 1, no. 1, pp. 8–15, 2025.

[2]. Wang A., Jiao H., Chen Z., and Yang J., “An Improved YOLO-Based Waste Detection Model and Its Integration to Robotic Gripping Systems,” *Computers, Materials & Continua*, vol. 84, no. 3, pp. 5773–5790, 2025.

[3]. Paudel P., Shrestha S., Shrestha S., Gurung S., and Adhikari S., “Automated Waste Sorting with Delta Arm and YOLOv8 Detection,” *Journal of Artificial Intelligence and Capsule Networks*, vol. 6, no. 3, pp. 299–315, 2024

[4]. Youran Huang, “Research on Garbage Sorting Robotic Arm Based on Image Vision,” *Journal of Physics: Conference Series*, 2024, article 012020

[5]. Sulistiyowati I., Ichsan H. M., and Anshory I., “Object Sorting Conveyor with Detection Color Using ESP-32 Camera Python Based on Open-CV,” *JEECS (Journal of Electrical Engineering and Computer Sciences)*, vol. 9, no. 1, pp. 61–68, 2024.

[6]. Li, X., and Zhang, Y., “Real-Time Object Classification Using YOLO for Industrial Automation,” International Journal of Advanced Computer Science and Applications, vol. 12, no. 3, pp. 210–218, 2021.

[7]. Kumar, S., and Prasad, R., “Vision-Based Robotic Arm Control for Automated Sorting,” Journal of Robotics and Mechanical Engineering Research, vol. 9, no. 2, pp. 45–52, 2022.

[8]. Patel, D., and Singh, A., “Deep Learning-Based Sorting System Using Computer Vision Techniques,” Procedia Computer Science, vol. 218, pp. 1124–1132, 2023.

[9]. Ahmed, M., and Rahman, T., “Machine Learning Approach for Material Detection Using Image Processing,” IEEE Access, vol.

8, pp. 150293–150301, 2020.

[10]. Fernandes, J., and Mathew, L., "Implementation of ESP32-CAM for Intelligent Vision Systems," International Journal of Engineering Trends and Technology, vol. 69, no. 8, pp. 180–186, 2021.