GJR Publication
OPEN ACCESS

Global Journal of Research in Engineering & Computer Sciences
ISSN: 2583-2727 (Online)
Volume 06 | Issue 04 | July-August | 2026
Journal homepage: https://gjrpublication.com/gjrecs/

Research Article

Design And Implementation of a Machine Vision-Based Conveyor Sorting System with Real-Time Multi-Object Detection Under Variable Lighting Conditions

*Muhammed Bello Umar [1] and Olusola Oluwaseun Jude [2]

[1,2] Department of Mechatronics Engineering, Nigerian Defence Academy Kaduna, Nigeria.

DOI: 10.5281/zenodo.21741507 **Submission Date: 29 May 2026 | Published Date: 01 Aug. 2026**

***Corresponding author: Muhammed Bello Umar**
Department of Mechatronics Engineering, Nigerian Defence Academy Kaduna, Nigeria.

Abstract

The increasing demand for efficient waste management has encouraged the adoption of intelligent sorting systems capable of reducing human effort while improving sorting accuracy. This study presents the design and implementation of a Raspberry Pi-based machine vision conveyor sorting system for the real-time detection of recyclable materials under variable lighting conditions.

A laboratory-scale prototype was developed using a Raspberry Pi 4 Model B, an OV5647 camera module, Arduino Uno, conveyor mechanism and a rotating four-compartment collection bin. A custom dataset consisting of bottle, can and nylon objects was created and annotated using Rob flow. The YOLOv8 Nano object detection model was trained for 50 epochs and deployed on the Raspberry Pi using the Picamera2 library for real-time inference.

The trained model achieved a precision of 98.47%, recall of 100.00%, mAP@0.5 of 99.50% and mAP@0.5:0.95 of 94.46%. During testing, the system successfully detected the three object classes and generated the corresponding GPIO outputs for conveyor control. The average inference speed recorded on the Raspberry Pi CPU was approximately 0.96 frames per second, demonstrating the feasibility of deploying lightweight deep learning models on low-cost embedded hardware.

The developed system provides an effective and affordable solution for intelligent waste sorting and offers a practical foundation for future improvements in embedded machine vision and industrial automation.

Keywords: *Machine Vision, Raspberry Pi, YOLOv8, Object Detection, Conveyor Sorting, Embedded Artificial Intelligence.*

I. INTRODUCTION

The rapid increase in municipal and industrial solid waste has become a significant environmental challenge in many developing countries, creating the need for more efficient and sustainable waste management practices [24], [25]. Traditional waste sorting methods rely heavily on manual labour, making the process slow, inconsistent and potentially hazardous because workers are exposed to contaminated and recyclable materials [24], [25].

Recent advances in machine vision and artificial intelligence (AI) have created new opportunities for automating waste identification and sorting with improved speed, consistency and accuracy [5], [7], [8]. Machine vision systems employ cameras and image processing algorithms to recognise objects based on their visual characteristics, while deep learning techniques have demonstrated excellent performance in object detection and classification tasks [5], [6], [8], [19]. These technologies reduce human intervention in repetitive sorting operations and improve the efficiency of waste segregation.
Among modern object detection algorithms, the You Only Look Once (YOLO) family has gained widespread acceptance because it performs object localisation and classification simultaneously in a single inference stage, enabling real-time detection [1]–[4]. The lightweight YOLOv8 Nano (YOLOv8n) model is particularly suitable for embedded systems with limited computational resources [4].

The Raspberry Pi 4 Model B has become a widely adopted embedded computing platform for intelligent automation because of its affordability, low power consumption and compatibility with machine vision and deep learning

frameworks [9], [10], [12], [13]. When integrated with the Raspberry Pi Camera Module, it provides a practical platform for developing low-cost intelligent inspection and sorting systems suitable for educational, research and small-scale industrial applications [9], [10].

This project presents the design and implementation of a Raspberry Pi-based machine vision conveyor sorting system capable of detecting three categories of recyclable waste materials—plastic bottles, aluminium cans and nylon sachets—in real time. The developed system employs a custom-trained YOLOv8 Nano model to identify waste objects conveyed through a controlled detection chamber. Detection results are transmitted through Raspberry Pi GPIO outputs for communication with an Arduino-controlled sorting mechanism. Although the complete conveyor prototype was constructed, this study primarily focuses on the implementation and evaluation of the machine vision subsystem under different lighting conditions.

II. LITERATURE REVIEW

Automated sorting systems have become increasingly important in modern manufacturing and waste management because they improve operational efficiency, reduce labour costs and enhance the consistency of sorting processes [14], [15], [24]. Advances in machine vision and artificial intelligence (AI) have further transformed conventional sorting systems by enabling machines to identify and classify objects based on their visual characteristics with minimal human intervention [5], [7], [8].

This chapter reviews the concepts and technologies relevant to the development of the proposed machine vision-based conveyor sorting system. It discusses automated conveyor sorting systems, machine vision, artificial intelligence, the YOLO object detection algorithm, embedded computing using the Raspberry Pi and previous studies related to intelligent waste sorting. The chapter concludes by identifying the research gap addressed by this study.

2.2 Automated Conveyor Sorting Systems

Conveyor sorting systems are widely used in manufacturing and recycling industries to transport and separate products according to characteristics such as size, shape, colour, material or destination [14], [15]. Productivity is greatly improved by enabling continuous material flow while reducing manual handling.

Conventional conveyor sorting systems typically employ sensors such as ultrasonic, proximity and colour sensors to identify objects. Although these sensors perform effectively under specific operating conditions, they are generally limited to detecting a small number of object characteristics and cannot reliably distinguish between visually similar materials [14].

Machine vision provides a more flexible alternative by analysing complete images instead of individual sensor signals. This enables multiple objects features to be evaluated simultaneously, resulting in improved classification accuracy and greater adaptability to different operating environments [5], [15].

2.3 Machine Vision

This is the application of cameras, image acquisition devices and computer algorithms to automatically inspect, measure or identify objects [5], [6]. Unlike conventional sensing techniques that depend on a single measured parameter, machine vision analyses complete images to extract meaningful information for decision-making.

A typical machine vision system consists of:

- Image acquisition device (camera)
- Lighting system
- Image processing unit
- Object detection algorithm
- Decision-making unit
- Output or control system

The performance of a machine vision system depends largely on image quality, illumination, camera resolution and the robustness of the image processing algorithm [5], [6].

2.4 Artificial Intelligence in Object Detection

Artificial intelligence (AI) enables computers to perform tasks that require human intelligence [7], [8]. Within AI, machine learning enables systems to learn patterns from data, while deep learning employs multilayer artificial neural networks to solve complex recognition problems.

Object detection combines two principal tasks:

- Object localisation
- Object classification

Unlike image classification, which only predicts the category of an object, object detection simultaneously determines both the object class and its location within an image [1], [2], [8].

Deep learning-based object detection has significantly improved detection accuracy over traditional image processing techniques because feature extraction is performed automatically by the neural network rather than being manually engineered [7], [8], [19].

2.5 The YOLO (You Only Look Once) Object Detection Algorithm

YOLO is one of the most widely adopted real-time object detection algorithms [1]–[4]. Unlike earlier detection methods that perform localisation and classification separately, YOLO predicts object locations and classes simultaneously during a single forward pass through the neural network, enabling high-speed object detection.
The major advantages of YOLO include:

- High detection speed
- Good localisation accuracy
- End-to-end learning
- Real-time performance
- Simultaneous detection of multiple objects

Several versions of YOLO have been developed, including YOLOv3, YOLOv4, YOLOv5, YOLOv7 and YOLOv8, each introducing improvements in detection accuracy, computational efficiency and network architecture **[1]–[4]**.

This study employs **YOLOv8 Nano (YOLOv8n)** because it provides an effective balance between detection accuracy and computational efficiency, making it suitable for deployment on embedded platforms such as the Raspberry Pi **[4]**.

2.6 Raspberry Pi as an Embedded AI Platform

The Raspberry Pi is a single-board computer widely used in embedded systems, robotics and intelligent automation because of its affordability, compact size and extensive software support [9], [10], [13].
The Raspberry Pi 4 Model B used in this study features:

- Quad-core ARM processor
- 2 GB RAM
- GPIO interface
- CSI camera interface
- USB connectivity
- Wireless networking

The GPIO interface enables communication with external hardware such as Arduino boards, relays, sensors and actuators, allowing machine vision decisions to control physical processes.

Although the Raspberry Pi cannot match the computational performance of dedicated GPUs, lightweight deep learning models such as YOLOv8 Nano can provide satisfactory real-time object detection performance for laboratory-scale applications [4], [9], [13].

2.7 Raspberry Pi Camera Module

The Raspberry Pi Camera Module provides direct image acquisition in real-time through the Raspberry Pi Camera Serial Interface (CSI). Compared with conventional USB webcams, the CSI interface offers lower latency and improved integration with the Raspberry Pi operating system [9], [10].

The OV5647 Raspberry Pi Camera (Fish-Eye, Night vision) Module was used in this study.

2.8 Image Dataset Preparation

The performance of deep learning models depends largely on the quality and diversity of the training dataset **[7], [8]**. A representative dataset enables the model to learn distinctive visual features associated with each object class, thereby improving detection accuracy.

For this study, a custom dataset was created using images captured with the Raspberry Pi Camera Module. Three recyclable waste categories were considered:

- Plastic bottle
- Aluminum can
- Nylon sachet

The captured images were annotated in Rob flow, where bounding boxes were manually drawn around each object. The annotated dataset was then exported in YOLO format for model training [11].

2.9 Performance Evaluation Metrics

The performance of object detection models is commonly evaluated using several quantitative metrics.

2.9.1 Precision

Precision measures the proportion of detected objects that are correctly classified.

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

where:
TP = True Positives
FP = False Positives
Higher precision indicates fewer incorrect detections.

2.9.2 Recall

Recall measures the ability of the detector to identify all relevant objects.

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

where:
FN = False Negatives.
A high recall indicates that very few objects are missed during detection.

2.9.3 Mean Average Precision (mAP)

Mean Average Precision (mAP) is the most widely accepted performance metric for object detection models. It combines precision and recall into a single measure of overall detection accuracy.
This research reports:

- mAP@0.5
- mAP@0.5:0.95

which are standard evaluation metrics used by the Ultralytics YOLO framework.

2.9.4 Frames Per Second (FPS)

Frames Per Second (FPS) measures the number of images processed per second during deployment.
Unlike precision and recall, FPS evaluates computational performance rather than detection accuracy. Higher FPS indicates faster real-time operation.

2.10 Review of Related Works

Investigations have been made by several researchers on automated waste sorting using machine vision and deep learning techniques [14], [15], [18]. Early systems relied primarily on infrared, ultrasonic and colour sensors for object identification. Although these approaches were relatively inexpensive, they were generally limited in their ability to distinguish between visually similar waste materials.

While recent studies have demonstrated that convolutional neural networks (CNNs) and YOLO-based object detectors significantly improve classification accuracy by automatically learning discriminative visual features from images [1]–[4], [7]. These techniques have been successfully applied to the identification of recyclable materials including plastics, paper, metals and glass.

Embedded implementations based on the Raspberry Pi have also been reported [9], [13]. However, many of these studies relied on external GPUs during deployment, considered only image classification rather than object detection, or evaluated their systems under limited operating conditions. Some investigations focused solely on software simulation without integrating physical conveyor mechanisms or hardware control interfaces.

The present study extends previous work by implementing a custom-trained YOLOv8 Nano model on a Raspberry Pi 4 Model B and interfacing the detection system with GPIO outputs for conveyor automation. In addition, the system was evaluated under three controlled lighting conditions to assess its performance in practical operating environments.

2.11 Research Gap

The reviewed literature indicates that although deep learning has significantly improved automated object detection, many existing systems rely on high-performance computing hardware, increasing implementation costs and limiting their suitability for educational and small-scale industrial applications [4], [9], [13].

Furthermore, relatively few studies demonstrate the complete implementation of a low-cost embedded machine vision system combining a Raspberry Pi, a custom-trained YOLOv8 model and GPIO interfacing for conveyor automation under varying lighting conditions.

This study addresses these limitations by developing an affordable Raspberry Pi-based machine vision conveyor sorting system capable of detecting three recyclable waste categories in real time using CPU-only inference. The system further

demonstrates practical integration between machine vision and external control hardware through Raspberry Pi GPIO communication.

III. MATERIALS AND METHODS

This chapter presents the materials, design procedure and implementation methodology adopted for the development of the machine vision-based conveyor sorting system. It describes the hardware components, software tools, design considerations, dataset preparation, model training, system integration and testing procedures. The overall objective was to develop a low-cost embedded machine vision system capable of detecting selected recyclable materials in real time and providing output signals for an automated sorting mechanism.

3.2 System Overview

The developed system consists of two major subsystems:

1. Machine vision subsystem, responsible for image acquisition, object detection and GPIO output generation.
2. Conveyor sorting subsystem, responsible for transporting objects and positioning the appropriate collection compartment based on the detected object.

The Raspberry Pi serves as the central processing unit of the machine vision subsystem. Images captured by the Raspberry Pi Camera are processed by the trained YOLOv8 Nano model. Once an object is recognised, the Raspberry Pi activates the corresponding GPIO output, which is intended to communicate with the Arduino-based control unit for sorting. Fig 3.1 shows the block diagram of the complete system.

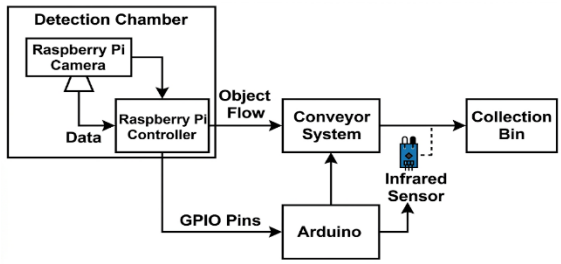

Figure 3.1: Block Diagram of the Machine Vision Conveyor Sorting System

3.3 System Components

The prototype was constructed using readily available components to minimise cost while maintaining reliable operation. The major hardware components are summarised in Table 3.1.

Table 3.1 Hardware Components

S.N	COMPONENT	FUNCTION
1	Raspberry Pi 4 Model B (2GB RAM)	Embedded Processing unit
2	Raspberry Pi Camera OV5647	Image Acquisition
3	Arduino Uno	Conveyor control
4	Standard Servo Motor	Controls the gate mechanism
5	Standard Servo Motor	Drives the conveyor belt
6	NEMA 17 Stepper Motor	Rotates the collection bin
7	Stepper Motor Driver	Controls the stepper motor
8	Ultrasonic Sensor	Detects object position at conveyor exit
9	16 × 2 LCD with I²C Module	Displays system status
10	12 V LED Light	Provides illumination inside the detection chamber
11	Buck Converter	Converts 12 V supply to 5 V
12	12 V, 5 A Power Supply	Main power source
13	Breadboard and Jumper Wires	Circuit prototyping
14	MDF Board	Base support
15	Acrylic Sheets	Detection chamber and conveyor enclosure

3.4 Mechanical Design

A laboratory-scale conveyor system was fabricated using Thousand Board Feet (MBF) as the structural base and transparent acrylic sheets to construct the detection chamber. The system was designed to insert the objects individually into the inspection region where images were captured by the Raspberry Pi Camera.

A standard servo motor was used to drive the conveyor mechanism, while another servo motor operated a gate mechanism that temporarily opened to allow detected objects to proceed towards the collection section.

At the discharge end of the conveyor, a four-compartment rotating collection bin was constructed. The collection bin was driven by a NEMA 17 stepper motor through a stepper motor driver. Based on the detected object category, the Arduino rotates the collection bin to align the appropriate compartment beneath the conveyor before the object is released.

An ultrasonic sensor positioned near the conveyor exit detects the arrival of an object and momentarily pauses conveyor movement to improve sorting accuracy.

Plate 3.1 Fabricated Conveyor Housing

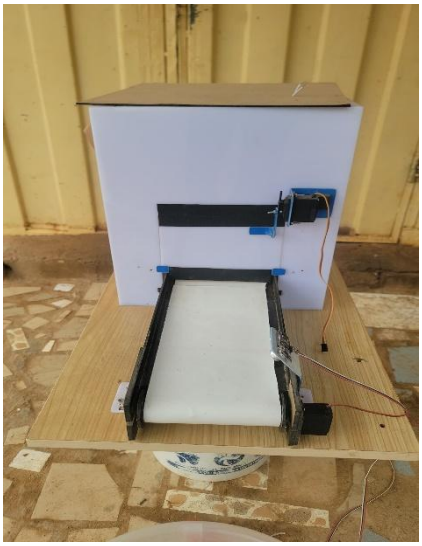

Plate 3.2 Conveyor Housing, Belt and Ultrasonic sensor

Plate 3.3 Conveyor Housing, Belt, Arduino uno, LCD display, jumper wires and Ultrasonic sensor

Plate 3.4 Complete system showing the collecting bin

3.5 Machine Vision Hardware

The machine vision subsystem consists of the Raspberry Pi 4 Model B and the OV5647 Raspberry Pi Camera Module.

The camera was mounted at the top of the detection chamber to capture images of objects moving along the conveyor belt. A 12 V LED lighting system with adjustable intensity was installed to minimise illumination variations during image acquisition. Images were captured at a resolution suitable for real-time inference while maintaining acceptable processing speed.

The Raspberry Pi processed captured images using the trained YOLOv8 Nano model and generated GPIO signals corresponding to the detected object category.

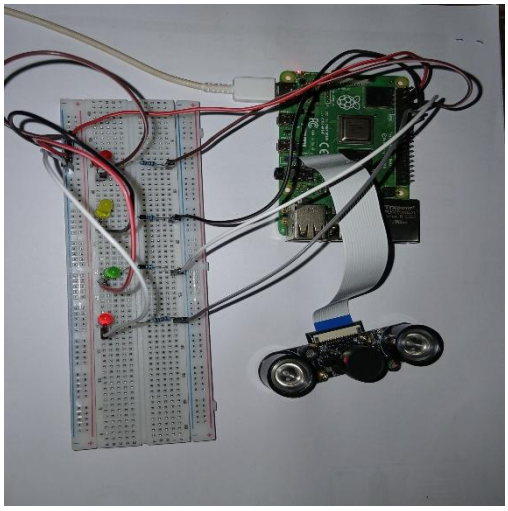

Plate 3.5 Raspberry Pi, Pi Camera GPIO setup

3.6 Design Considerations

The proposed machine vision-based conveyor sorting system was designed by integrating mechanical, electronic and intelligent vision components into a compact automated platform. The design emphasised reliable object detection, ease of implementation, low cost and stable operation under varying lighting conditions.

3.6.1 Conveyor Design

The conveyor was designed to transport waste objects individually through the camera's field of view at a moderate speed, allowing sufficient time for image acquisition and object classification before reaching the sorting point.

3.6.2 Detection Chamber Design

An enclosed detection chamber was incorporated into the conveyor system to minimise the influence of external environmental lighting during image acquisition. The enclosed chamber therefore provides a controlled imaging environment where lighting conditions can be deliberately adjusted while maintaining repeatable experimental conditions throughout training and testing.

3.6.3 Camera and Processing Unit

The Raspberry Pi Camera was mounted above the conveyor to obtain a clear top-down view of the objects while minimising perspective distortion. A Raspberry Pi 4 Model B (2 GB RAM) was selected as the processing unit because it provides sufficient computing capability, GPIO support and compatibility with Python and the YOLOv8 framework. The lightweight YOLOv8 Nano model was adopted due to its balance between detection accuracy and computational efficiency on embedded hardware.

3.6.4 Sorting Mechanism

The sorting mechanism consists of a servo-operated discharge gate and a rotary collection bin driven by a NEMA 17 stepper motor. The bin was divided into four compartments corresponding to bottles, cans, nylon and exclusions.

The angular displacement of the collection bin is determined by

$$\theta = \frac{360^\circ}{N}$$

where:

θ= angular displacement (degrees)

N= number of compartments.
Since the bin contains four compartments,

$$\theta = \frac{360^\circ}{4} = 90^\circ$$

Thus, the stepper motor rotates the collection bin by 90° after each successful classification. A short delay was also incorporated before opening the servo gate to ensure that the detected object reached the discharge position before sorting.

3.6.5 Power Supply

The prototype was powered using a regulated 12 V, 5 A DC power supply. The maximum available power is given by

$$P = VI$$

Therefore,

$$P = 12 \times 5 = 60 \text{ W}$$

A DC–DC buck converter was used to regulate the supply voltage to 5 V for the Raspberry Pi and other low-voltage electronic components, ensuring stable and safe operation.

3.6.6 Illumination Monitoring and Control

To ensure consistent image acquisition, the detection chamber was equipped with an internal LED lighting system whose brightness could be adjusted continuously using an analogue control. The lighting level was monitored by the Arduino Uno, which converted the analogue input into a digital percentage ranging from 0% to 100% and displayed the corresponding value on a 16 × 2 LCD module.

This arrangement enabled the operator to regulate and monitor the relative illumination level inside the detection chamber throughout system operation. Maintaining controlled illumination helped reduce the effects of shadows and non-uniform lighting, thereby improving image consistency and supporting reliable object detection.

3.6.7 Experimental Evaluation under Variable Ambient Lighting

In addition to the controlled illumination provided inside the detection chamber, the developed system was evaluated under three external lighting conditions to determine its robustness to changes in ambient illumination. The surrounding environment was illuminated using three 10 W LED bulbs, which were switched to obtain the following experimental conditions. Table 3.2 shows the description.

Table 3.2 Lighting Conditions Used During Experimental Evaluation

SN	LIGHTING LEVEL	LED	DESCRIPTION
1	L1	3	High Illumination
2	L2	2	Medium Illumination
3	L3	1	Low Illumination

During each experiment, the same waste objects, camera position and conveyor arrangement were maintained while only the surrounding ambient lighting was varied. This approach enabled the influence of changing environmental illumination on the performance of the trained YOLOv8 model to be assessed under controlled conditions.

3.6.8 System Integration

The developed system integrates the conveyor, camera, Raspberry Pi, Arduino Uno, lighting unit, servo motors, stepper motor and YOLOv8 object detection model into a unified automated platform. During operation, waste objects are inserted into the detection chamber, classified by the trained model and directed into the appropriate collection compartment through coordinated control of the actuators.

3.7 Software Development Environment

The software development was carried out using Python due to its extensive support for computer vision and deep learning libraries.

The principal software tools employed in this research include Python 3.13, Ultralytics YOLOv8n, Pytorch 2.12.0, OpenCV, Picamera 2, Roboflow and Raspberry Pi OS.

Development and testing were performed remotely through Secure Shell (SSH), allowing the Raspberry Pi to be controlled directly from a Windows computer.

3.8 Dataset Preparation

A custom image dataset was created specifically for this study. Images were captured using the Raspberry Pi Camera under different lighting conditions to improve the robustness of the trained model.

Three object categories were considered namely plastic bottles, aluminum cans and sachet nylon because they are among the most common recyclable solid waste materials in the study environment. Multiple samples of each class were collected, differing in colour, size, brand, shape and level of deformation to improve the robustness of the trained model. Plate 3.6 shows a collection of the sample.

Plate 3.6: Representative waste objects used for dataset.

Approximately ninety images were manually annotated using Roboflow, with approximately thirty images representing each object class. Bounding boxes were drawn around each object before exporting the dataset in YOLO format.

The dataset was divided into training and validation subsets to facilitate model development and performance evaluation.

3.9 Model Training

The YOLOv8 Nano model was selected because of its lightweight architecture and suitability for deployment on embedded computing platforms.

Model training was performed using the Ultralytics YOLO framework for 50 epochs. The trained model was automatically saved, and the best-performing weights were exported as best.pt.

Training generated performance metrics including precision, recall, mAP@0.5 and mAP@0.5:0.95, which were subsequently used to evaluate model performance.

3.10 System Deployment

Following training, the best-performing model was deployed on the Raspberry Pi.

Unlike desktop implementations that often rely on USB webcams, the Raspberry Pi Camera was accessed using the Picamera2 library.

Captured images were processed continuously by the YOLOv8 model. Once an object was detected, the Raspberry Pi generated a corresponding GPIO output according to the object class.

The GPIO assignments used are presented in Table 3.2.

Table 3.2 GPIO Assignments

SN	OBJECT	GPIO PIN	PHYSICAL PIN
1	Bottle	GPIO17	Pin 11
2	Can	GPIO27	Pin 13
3	Nylon	GPIO22	Pin 15
4	Exclusions	GPIO23	Pin 16

These GPIO outputs provide the interface required for communication with the Arduino-based conveyor control system.

3.11 System Operation

The operational sequence of the developed system is summarised as follows:

1. The object is inserted into the detection chamber.
2. The Raspberry Pi Camera captures an image.
3. The conveyor drives the object out of the chamber.
4. The captured image is processed by the YOLOv8 Nano model.
5. The detected object is classified as Bottle, Can or Nylon.
6. The Raspberry Pi activates the corresponding GPIO output.
7. The Arduino receives the detection signal.
8. The rotating collection bin is positioned accordingly.
9. The gate servo opens, allowing the object to enter the appropriate compartment.
10. The conveyor resumes operation for the next object.

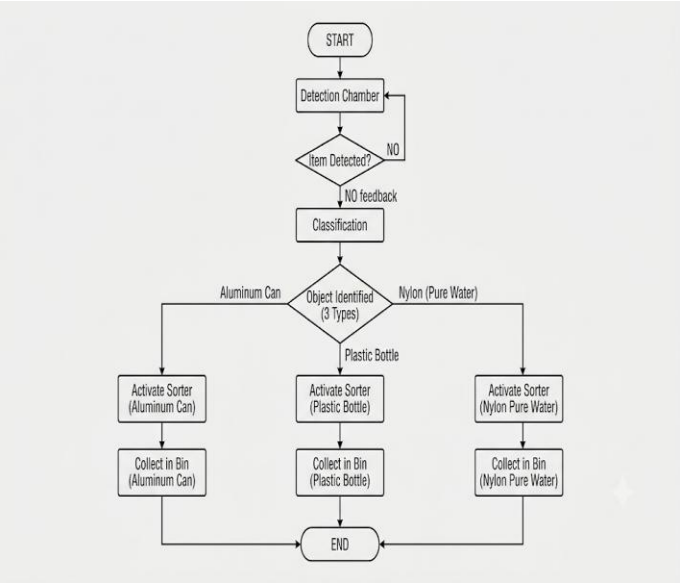

Figure 3.2: Flowchart of the Conveyor Sorting Process

3.12 Experimental Procedure

System evaluation was carried out under three different lighting conditions to assess the robustness of the developed machine vision system.

Objects belonging to each class were placed individually on the conveyor and transported into the detection chamber. Detection accuracy, confidence scores and GPIO responses were observed through the Raspberry Pi terminal.

Training performance was evaluated using precision, recall and mean Average Precision (mAP), while deployment performance was assessed by measuring the average inference speed in frames per second (FPS).

IV. RESULTS

The YOLOv8 Nano model was trained for 50 epochs in approximately 2.04 hours using the prepared dataset. The model automatically generated performance curves and validation metrics throughout the training process.

The final training results are summarised in Table 4.1.

Table 4.1 Final Training Performance

S. N	PERFORMANCE METRIC	VALUE
1	Epochs	50
2	Training Time	2.04 Hours
3	Precision	98.47%
4	Recall	100.00%
5	mAP@0.5	99.50%
6	mAP@0.5:0.95	94.46%

The obtained results indicate that the trained model achieved excellent detection performance on the validation dataset. The high precision value indicates that the model generated very few false detections, while the recall value of 100% shows that all validation objects were successfully identified during evaluation.

Similarly, the mAP@0.5 value of 99.50% demonstrates a high level of detection accuracy, whereas the mAP@0.5:0.95 value of 94.46% confirms that the model maintained good localisation performance across different Intersection over Union (IoU) thresholds.

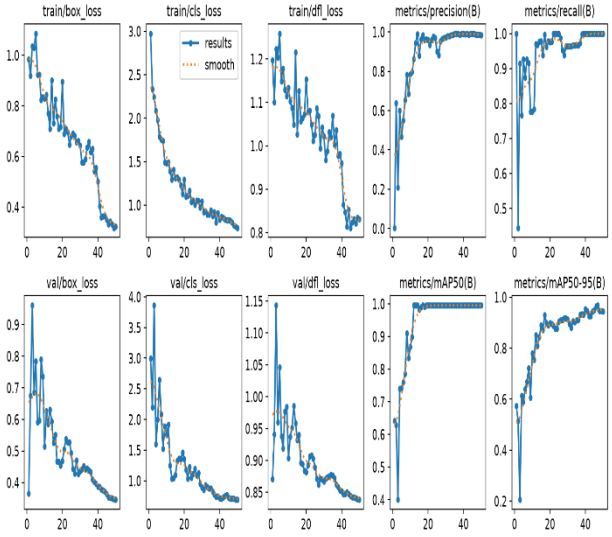

Figure 4.1 Training Performance Curves*(results.png)*

Figure 4.1 shows the learning behaviour of the YOLOv8n model over 50 training epochs. The training and validation losses decreased progressively, while the precision, recall and mAP values improved and stabilised towards the end of training, indicating successful convergence without significant overfitting.

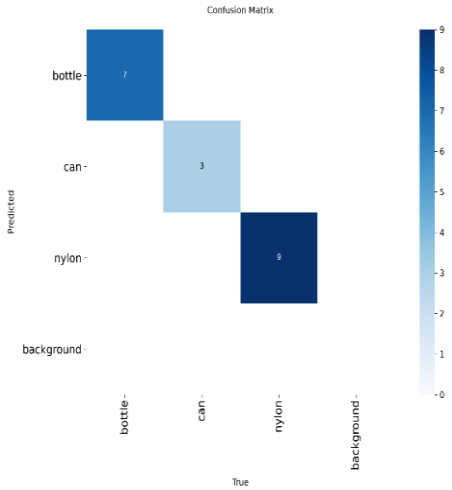

Figure 4.2 Confusion matrix of the trained YOLOv8n (*confusion_matrix.png)*

The confusion matrix indicates that most validation samples were correctly classified into their respective classes, with only a few instances of misclassification. This agrees with the high precision and recall achieved by the trained model.

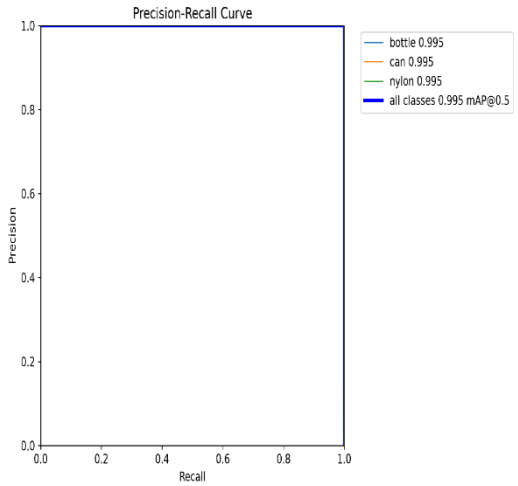

Figure 4.3 Precision-Recall curve *(BoxPR_curve.png)*

The Precision–Recall curve is concentrated near the upper-right corner of the graph, indicating excellent detection performance. This observation is consistent with the high precision (98.47%), recall (100%), and mAP@0.5 (99.50%) achieved by the trained YOLOv8 model. The limited number of validation samples also contributes to the compact appearance of the curve.

4.2 Deployment on Raspberry Pi

Following successful training, the model was deployed on a Raspberry Pi 4 Model B running Raspberry Pi OS.
Deployment was carried out using the Picamera2 library for image acquisition. The trained model (best.pt) was loaded successfully and processed live camera frames continuously. The Raspberry Pi generated GPIO outputs corresponding to the detected object class, thereby providing the interface required for communication with the Arduino-controlled sorting mechanism.

4.3 Real-Time Detection Results

The developed system was evaluated using representative samples of each object class.
Objects were introduced individually into the detection chamber while the Raspberry Pi continuously analysed the captured images.

The system successfully detected:

- Bottle
- Can
- Nylon

During testing, the terminal displayed the detected object class together with its confidence score and measured processing speed. Fig 4.1 shows the real-time detection result.

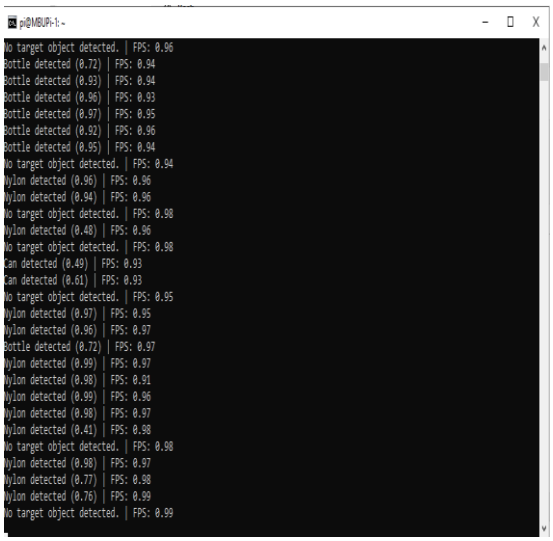

Plate 4.1 Real-time object detection using the Raspberry Pi implementation.

Typical terminal outputs included:
Bottle detected (0.93)
Can detected (0.61)
Nylon detected (0.94)
No target object detected.
The corresponding GPIO output associated with each detected object was activated automatically.

4.4 Inference Speed

Real-time deployment performance was evaluated by measuring the processing speed of the Raspberry Pi during live detection.

Table 4.2 summarises the observed deployment performance.

Table 4.2 Deployment Performance

S.N	Parameter	Value
1	Platform	Raspberry Pi 4 Model B
2	Processor	Quad-Core ARM CPU
3	Inference Device	CPU

Average Processing Speed ≈0.96 FPS

The obtained frame rate is considerably lower than GPU-based implementations reported in the literature. This result is expected because inference was performed entirely on the Raspberry Pi CPU without hardware acceleration.

Although the achieved frame rate is modest, it proved adequate for the laboratory-scale conveyor prototype developed in this study, where objects move individually through the detection chamber at controlled speeds.

4.5 Performance Evaluation under Variable Lighting Conditions

The performance of the developed machine vision-based conveyor sorting system was evaluated under three illumination levels to determine the influence of lighting intensity on object detection performance. Illumination was provided using three 10W LED lamps. The lighting conditions were varied by switching on one, two or three lamps while maintaining identical camera position, conveyor speed and object placement throughout the experiment.

The lighting conditions used during the evaluation are presented in Table 4.4.

Table 4.4 Lighting Conditions Used During Performance Evaluation

S.N	LIGHTING LEVEL	LED	DESCRIPTION
1	L1	3	High Illumination
2	L2	2	Medium Illumination
3	L3	1	Low Illumination

The trained YOLOv8 model was tested using representative plastic bottles, aluminum cans and nylon sachets that were not restricted to the training images. Detection performance was observed in terms of recognition consistency, confidence and response time under each lighting condition. The lighting condition detection for L1, L2 and L3 are shown in Plates 4.2, 4.3 and 4.4 respectively.

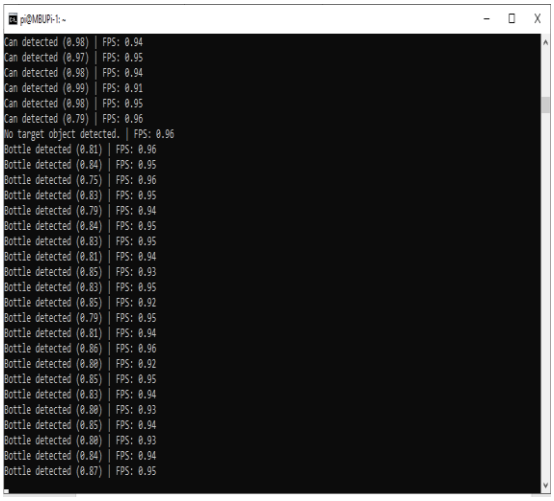

Plate 4.2: L1 (High Illumination) detection

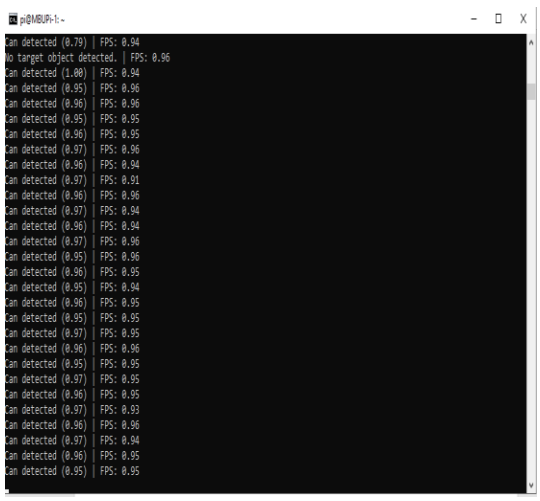

Plate 4.3: L2 (Medium Illumination) detection

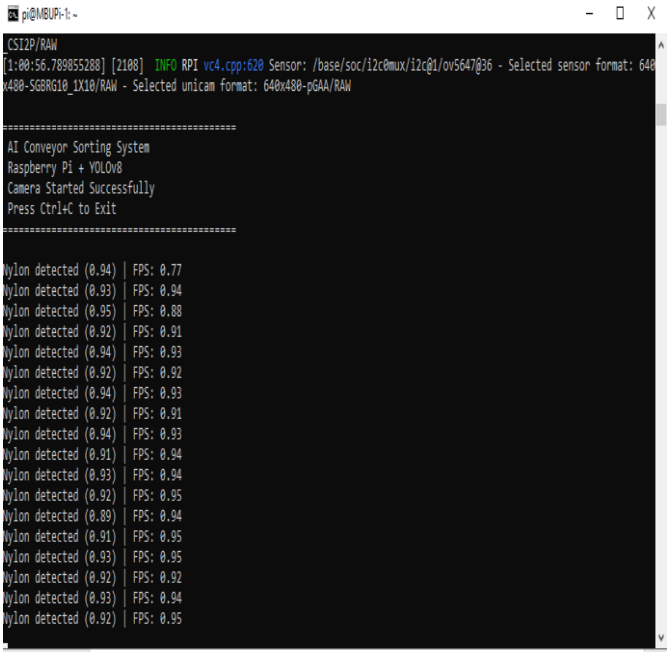

Plate 4.4: L3 (Low Illumination) detection

Below is the summary of the detection performance presented in Table 4.5.

Table 4.5 Detection Performance under Different Lighting Conditions

S.N	LIGHTING LEVEL	BOTTLE	CAN	NYLON	PERFORMANCE DECRIPTION
1	L1 (High)	Excellent	Excellent	Excellent	Fast, Stable and highly reliable detection
2	L2 (Medium)	Very Good	Very Good	Very Good	Stable detection with slight reduction in response speed
3	L3 (Low)	Good	Good	Fair	Detection maintained but with occasional delay and reduced confidence

4.6 Discussion of Results

The experimental evaluation demonstrated that the proposed machine vision system maintained reliable object detection under all three illumination levels. Although detection speed and confidence varied slightly as lighting intensity decreased, the trained YOLOv8 model consistently recognised the three target waste categories throughout the experiments.

Under high illumination (L1), where all three 10W LED lamps were switched on, the system produced the best overall performance. Plastic bottles were recognised almost immediately after entering the camera field of view. The aluminium cans were also detected rapidly, although the taller Schweppes can occasionally require a slightly longer detection time than the Coke can. Nylon sachets were detected continuously with stable confidence values. The increased illumination improved image brightness and contrast, making object features more distinguishable to the trained model.

Under medium illumination (L2), detection remained highly reliable. The bottle, can and nylon classes continued to be recognised correctly with only a slight reduction in response speed. No significant degradation in detection accuracy was observed. This indicates that the trained model was sufficiently robust to moderate reductions in lighting intensity.

Under low illumination (L3), detection performance remained satisfactory but became noticeably slower. Plastic bottles continued to be detected correctly, although smaller bottles required additional time before stable detection was achieved. The aluminium cans were also recognised successfully, but occasional interruptions in detection were observed before recognition stabilised. Nylon sachets exhibited the greatest sensitivity to reduced illumination, particularly when heavily folded or wrinkled, resulting in intermittent detection before continuous recognition resumed.

These observations indicate that lower illumination reduces image contrast and the visibility of important visual features used by the convolutional neural network during inference. Consequently, the detector required additional frames before confidently classifying certain objects. Despite this reduction in response speed, all three waste categories remained detectable, demonstrating that the trained model possessed good robustness to variations in illumination.

The inclusion of images captured under different lighting conditions during dataset preparation contributed significantly to this robustness. By exposing the model to variations in brightness during training, the network learned illumination-invariant features that enabled successful object recognition under practical operating conditions.

Object geometry also influenced detection performance. Large cylindrical bottles generally produced faster and more stable detections than transparent nylon sachets. Nylon waste frequently changed shape because of folds and wrinkles, resulting in greater appearance variation between frames. Similarly, taller beverage cans occasionally produced slight fluctuations in confidence due to reflections from their metallic surfaces.

Overall, the experimental results demonstrate that the developed system satisfies the design objective of providing reliable real-time waste object detection under variable lighting conditions. The trained YOLOv8 Nano model achieved excellent validation performance, recording a precision of 98.47%, recall of 100%, mAP@0.5 of 99.50% and mAP@0.5:0.95 of 94.46%. Although lower illumination slightly reduced detection speed, the results indicate that the model learned and maintained the distinguishing features and consistent recognition of the selected object classes effectively despite the relatively small custom dataset.

Deployment on the Raspberry Pi also demonstrated that lightweight deep learning models can be implemented successfully on affordable embedded hardware. The system consistently detected bottles, cans and nylon sachets in real time while simultaneously generating GPIO outputs for external hardware control.

The measured processing speed of approximately 0.96 FPS reflects the computational limitations of CPU-only inference on the Raspberry Pi. Although slower than desktop GPU implementations, the achieved performance was sufficient for demonstrating the feasibility of the proposed machine vision system within the scope of this research.

4.7 Challenges Encountered

The first challenge involved accessing the Raspberry Pi Camera using OpenCV's Video Capture interface. This approach proved unreliable because of compatibility issues with the camera stack used by recent Raspberry Pi operating systems.

A second challenge involved managing Python virtual environments. The Picamera2 library was initially unavailable inside the virtual environment used for YOLO deployment. This issue was resolved by configuring the Python environment to include the required system packages.

Another limitation was the relatively small size of the custom dataset. Although the trained model achieved high validation accuracy, increasing the number of training images would likely improve generalisation and reduce occasional misclassification.

Finally, the absence of GPU acceleration on the Raspberry Pi limited the achievable inference speed. Nevertheless, the system remained sufficiently responsive for the intended laboratory demonstration.

V. CONCLUSION AND RECOMMENDATIONS

This study presented the design and implementation of a machine vision-based conveyor sorting system for the real-time detection of recyclable materials under variable lighting conditions. The primary objective was to develop a low-cost intelligent sorting system capable of identifying plastic bottles, aluminium cans and nylon sachets using embedded artificial intelligence.

A laboratory-scale prototype was designed and constructed using MBF board and acrylic sheets. The system incorporated a conveyor mechanism, detection chamber, infrared sensor, standard servo motors, a rotating four-compartment collection bin driven by a NEMA 17 stepper motor, an Arduino Uno control unit and a Raspberry Pi 4 Model B equipped with an OV5647 camera module.

A custom dataset consisting of three object classes was prepared and manually annotated using Roboflow. The YOLOv8 Nano object detection model was trained for 50 epochs and achieved excellent validation performance, recording a precision of 98.47%, recall of 100.00%, mAP@0.5 of 99.50% and mAP@0.5:0.95 of 94.46%.

The trained model was successfully deployed on the Raspberry Pi using the Picamera2 library for image acquisition. During real-time testing, the system correctly detected bottles, cans and nylon sachets and generated the appropriate GPIO output corresponding to each detected object. Although inference was performed entirely on the Raspberry Pi CPU, the system achieved an average processing speed of approximately 0.96 frames per second, which proved adequate for the laboratory-scale prototype.

The implementation demonstrated that deep learning-based object detection can be successfully deployed on affordable embedded hardware for intelligent conveyor sorting applications.

5.2 Conclusion

The results obtained in this research demonstrate that the objectives of the study were successfully achieved.

The developed machine vision system accurately detected the selected recyclable materials and provided the required GPIO signals for integration with the conveyor control mechanism. The use of the lightweight YOLOv8 Nano model made it possible to perform embedded object detection on a Raspberry Pi without the need for dedicated graphics processing hardware.

The high precision, recall and mAP values obtained during model evaluation indicate that the trained detector learned the characteristics of the selected object classes effectively. Practical deployment further confirmed that the model could perform reliable real-time detection under varying lighting conditions, although occasional misclassification between visually similar objects was observed.

Overall, the study demonstrates that affordable embedded platforms can support intelligent machine vision applications for automated waste sorting. The developed prototype provides a practical foundation for future research in embedded artificial intelligence, robotics and smart recycling systems.

5.3 Recommendations

Based on the findings of this study, the following recommendations are proposed:

Future work should expand the dataset by including a larger number of images captured under more diverse environmental conditions. A larger dataset is expected to improve the model's ability to generalise and reduce occasional misclassification.

Additional recyclable materials such as paper, glass and cardboard should be incorporated to increase the practical usefulness of the system.

Future implementations should investigate the use of embedded AI accelerators, such as the Google Coral TPU or Intel Neural Compute Stick, to improve inference speed.

A higher-resolution camera with automatic exposure control may further improve object detection under challenging lighting conditions.

REFERENCES

1. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779–788.
2. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 6517–6525.
3. A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, 2020.
4. G. Jocher *et al.*, "Ultralytics YOLOv8 Documentation," Ultralytics Inc., 2024.
5. R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed., Springer, 2022.
6. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed., Pearson Education, 2018.
7. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed., Pearson, 2021.
8. I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*. MIT Press, 2016.
9. Raspberry Pi Foundation, *Raspberry Pi 4 Model B Datasheet*, Cambridge, United Kingdom.
10. Raspberry Pi Foundation, *Picamera2 Documentation*, Cambridge, United Kingdom.
11. Arduino, *Arduino Uno Rev3 Technical Specifications*, Arduino AG.
12. OpenCV Team, *OpenCV Documentation*, Open Source Computer Vision Library.
13. PyTorch Team, *PyTorch Documentation*, Linux Foundation AI.
14. Roboflow Inc., *Roboflow Annotation and Dataset Management Documentation*, 2024.
15. D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed., Pearson, 2012.
16. J. Han, M. Kamber and J. Pei, *Data Mining: Concepts and Techniques*, 4th ed., Morgan Kaufmann, 2023.
17. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
18. S. Haykin, *Neural Networks and Learning Machines*, 3rd ed., Pearson, 2009.
19. Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning," *Nature*, vol. 521, pp. 436–444, 2015.
20. M. Nixon and A. Aguado, *Feature Extraction and Image Processing for Computer Vision*, 4th ed., Academic Press, 2020.
21. M. Sonka, V. Hlavac and R. Boyle, *Image Processing, Analysis and Machine Vision*, 5th ed., Cengage Learning, 2019.
22. T. Bradski and A. Kaehler, *Learning OpenCV*, O'Reilly Media.
23. J. Craig, *Introduction to Robotics: Mechanics and Control*, 4th ed., Pearson, 2018.
24. W. Bolton, *Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering*, 7th ed., Pearson, 2021.ss
25. M. P. Groover, *Automation, Production Systems and Computer-Integrated Manufacturing*, 5th ed., Pearson, 2020.

CITATION

Umar, M. B.& Jude, O. O. (2026). Design And Implementation of a Machine Vision-Based Conveyor Sorting System with RealTime Multi-Object Detection Under Variable Lighting Conditions. In Global Journal of Research in Engineering & Computer Sciences (Vol. 6, Issue 4, pp. 99–114). https://doi.org/10.5281/zenodo.21741507