ECE445

SENIOR DESIGN LABORATORY

# FINAL REPORT

Vision-Guided Sorting and Pickup Cleaning Robot

Team #19

Zihan Zhou [zihanz16]
Dailin Wu [dailin2]
Jinyang Chen [jc129]
Tinghao Pan [tinghao4]

Advisor: Meng Zhang

May 15, 2026

# Abstract

Recyclable material sorting is a repetitive task that is difficult to perform consistently in a small laboratory or educational setting. This project develops a compact vision-based sorting prototype for three common recyclable categories: paper-based waste, aluminum beverage cans, and plastic bottles. The system is intended to demonstrate a complete sorting workflow, from visual recognition to embedded control and physical actuation, within a low-cost table-top platform.

The main classification task is handled by a camera-based vision pipeline. For each detected object, the software crops a fixed region of interest and extracts visual features related to color, brightness, edges, contours, and reflection patterns. These features are then used by a Random Forest classifier to identify the material category. A lightweight auxiliary sensor is used only to determine whether an object is present before image processing begins, reducing unnecessary classification of empty frames.

After a material category is predicted, the result is sent to an Arduino UNO through serial communication. The Arduino drives the servo-based sorting mechanism and guides the object toward the corresponding output bin. The final design combines computer vision, embedded control, and mechanical actuation in a modular structure that can be tested and adjusted subsystem by subsystem. While the prototype is not intended for industrial recycling, it provides a practical demonstration of automated recyclable material sorting at a small scale.

# Contents

# 1. Introduction

## 1.1 Problem Statement and Motivation

Recyclable material sorting is a common task in waste management, but it is difficult to perform consistently in a small educational or laboratory-scale setting. Items such as paper tubes, aluminum beverage cans, and plastic bottles can be visually distinct to a human operator, but manual sorting is repetitive, time-consuming, and dependent on attention. A compact automated sorter can make the process more repeatable while demonstrating the integration of sensing, computer vision, embedded control, and electromechanical actuation.

Large industrial recycling systems often use high-speed conveyors, optical sensing arrays, pneumatic ejectors, and complex mechanical structures. Although these systems are effective at large scale, they are not suitable for a senior design prototype because they are expensive, difficult to maintain, and physically larger than the available workspace. This project therefore focuses on a table-top sorting system that demonstrates the complete pipeline from object detection to material classification and physical separation.

The main technical challenge is not only to classify the object, but also to coordinate sensing, decision making, communication, and mechanical motion. The system must determine whether an object is ready for recognition, capture and classify the object, send the result to an embedded controller, and actuate the sorting mechanism at the correct time.

## 1.2 Proposed Solution and Core Functionalities

This project proposes a vision-based recyclable material sorting system for a compact table-top prototype. The system is designed to classify and sort three predefined recyclable material categories:

- paper-based waste, such as paper tubes or cardboard sleeves;
- aluminum beverage cans, such as printed soda cans;
- plastic bottles, such as transparent drink bottles.

The core function of the system is camera-based material classification. A camera captures the object inside a fixed observation region, and the software extracts compact visual features from the image, including color statistics, brightness statistics, edge information, highlight

ratio, dark-area ratio, contour information, and normalized color ratios. These features are used to train a Random Forest classifier with OpenCV and scikit-learn [3, 4].

An auxiliary optical sensor is used only to detect whether an object is present in the observation region before image classification is triggered [2]. This sensing stage reduces unnecessary processing of empty frames and helps coordinate image capture with mechanical actuation, but it is not responsible for material classification.

After classification, the Raspberry Pi or computer sends a numeric class code to the Arduino UNO through serial communication [1]. The Arduino then moves the sorting servo to the corresponding calibrated angle. Unknown or low-confidence predictions return the mechanism to a safe neutral position.

### 1.3 Subsystem Overview

At the top level, the project is divided into five connected subsystems: vision perception, embedded control, mechanical sorting, auxiliary presence sensing, and power/support hardware. Figure 1 gives the overall system pipeline, while Table 1 summarizes the function and main interconnects of each subsystem.

Table 1: Subsystem overview and interconnect summary.

| **Subsystem** | **Purpose** | **Main Interconnects** |
|---|---|---|
| Vision perception | Captures the object image, extracts visual features, and classifies the material category. | Camera frame, ROI crop, OpenCV feature extraction, Random Forest model |
| Embedded control | Receives the predicted material code and converts it into servo commands. | Serial communication, Arduino UNO, servo PWM/control signal |
| Mechanical sorting | Guides the object into the corresponding output bin according to the predicted class. | Servo-driven gate or chute, top/middle/bottom acrylic layers |
| Auxiliary presence sensing | Detects whether an object is present before the vision classifier is activated. | Optical sensor output, controller input, trigger signal |
| Power and support hardware | Provides stable power and mechanical support for sensing, actuation, and computation. | External servo supply, common ground, Raspberry Pi or computer, wiring |

The auxiliary sensor is not used as the primary material classifier. Its role is limited to object-presence triggering and timing coordination. The final material decision is made by the camera-based classifier because paper, aluminum, and plastic objects differ in shape, texture, reflection, transparency, and contour as well as color.

### 1.4 High-Level Requirements

To evaluate whether the final prototype meets the project objective, the system is defined by the following high-level requirements. These requirements are later mapped to verification procedures in Section 3 and Appendix A.

**Object-Presence Triggering:** The auxiliary sensing module shall distinguish between an empty observation region and an occupied observation region under controlled lighting. When no object is present, the system shall remain in the idle state and avoid unnecessary image classification or sorting commands.

**Triggered Real-Time Material Classification:** After an object is detected, the vision subsystem shall capture the object in the camera region of interest and classify it into paper-based waste, aluminum beverage can, plastic bottle, or unknown. The classifier shall operate fast enough for the table-top sorting process and shall use a confidence-based rule for uncertain cases.

**Reliable Arduino Command and Servo Response:** The vision software shall transmit the predicted class code to the Arduino through serial communication. The Arduino shall correctly interpret the command and move the sorting servo to the corresponding calibrated angle. Unknown or uncertain cases shall return the mechanism to a safe neutral position.

**Physical Sorting Functionality:** The mechanical structure shall guide objects from the sensing region to the appropriate output bin according to the classification result. The servo-driven chute or gate shall provide repeatable angular positioning and smooth object movement.

**Practical Low-Cost Integration:** The full prototype shall use readily available components, including an Arduino UNO, servo actuators, an auxiliary presence sensor, a camera, and a Raspberry Pi or computer for model inference. The system shall remain compact, easy to assemble, and suitable for demonstration within the available project schedule.

### 1.5 Block-Level Design Changes

The project was revised toward a simpler and more reliable vision-centered architecture during development. The auxiliary optical sensor was kept as a lightweight object-presence trigger, but the main material-recognition task was assigned to the camera-based classifier. This change reduces classification uncertainty because material type depends on shape, transparency, reflectivity, and contour rather than color or reflected intensity alone.

The vision block was also simplified from a heavy object-detection model to a fixed-region feature-based classifier. Since the object appears in a controlled observation area, handcrafted image features with a Random Forest classifier provide a more practical solution for the available schedule and dataset size. The mechanical design was organized into a modular three-tier structure so that sensing, routing, electronics, and collection can be tested separately before full-system integration.

# 2. Design

## 2.1 Equations and Simulations

### 2.1.1 Auxiliary Object-Presence Trigger

An auxiliary optical sensor is used to determine whether an object has entered the observation region. During calibration, the system records the background sensor response $s_{\mathrm{bg}}$. During operation, the measured response is denoted as $s_{\mathrm{meas}}$, and the absolute response difference is

$$\Delta s = |s_{\mathrm{meas}} - s_{\mathrm{bg}}|. \tag{1}$$

The binary object-presence decision is defined as

$$d = \begin{cases} 1, & |s_{\mathrm{meas}} - s_{\mathrm{bg}}| \geq s_{\mathrm{th}}, \\ 0, & |s_{\mathrm{meas}} - s_{\mathrm{bg}}| < s_{\mathrm{th}}, \end{cases} \tag{2}$$

where $s_{\mathrm{th}}$ is the calibrated detection threshold and $d$ is the object-presence flag. Equations 1 and 2 define only the trigger rule for activating the camera-based classifier; they do not perform material classification.

### 2.1.2 Camera Region of Interest

The camera input is handled using OpenCV. The default camera resolution is 1280 by 720 pixels. Instead of processing the entire frame, the system defines a fixed region of interest:

$$\mathrm{ROI} = (x_f, y_f, w_f, h_f) = (0.20, 0.15, 0.60, 0.70). \tag{3}$$

The ROI in Equation 3 corresponds to the central part of the camera image where the object is expected to appear. Cropping the image reduces background interference and makes the training and real-time prediction conditions more consistent.

### 2.1.3 Confidence-Based Unknown Handling

Let $p_k$ denote the predicted class probability for class $k$ produced by the Random Forest classifier. The final predicted class code is assigned by

$$\hat{k} = \begin{cases} \arg\max_k p_k, & \max_k p_k \geq p_{\text{th}}, \\ 0, & \max_k p_k < p_{\text{th}}, \end{cases} \tag{4}$$

where $p_{\text{th}}$ is the confidence threshold and code 0 represents the unknown or neutral state. Equation 4 prevents uncertain samples from being forced into one of the three known material classes.

### 2.1.4 Servo Angle Mapping

The Arduino converts the received class code into a calibrated servo angle. The initial mapping is

$$\theta(c) = \begin{cases} 90^\circ, & c = 0, \\ 45^\circ, & c = 1, \\ 90^\circ, & c = 2, \\ 135^\circ, & c = 3, \end{cases} \tag{5}$$

where $c = 0$ is the unknown or neutral state, $c = 1$ is paper-based waste, $c = 2$ is aluminum beverage can, and $c = 3$ is plastic bottle. The same mapping is summarized later in Table 4.

## 2.2 Design Alternatives

### 2.2.1 Auxiliary Sensor Triggering vs. Sensor-Based Classification

An early design option was to use a simple optical sensor as the primary material classifier. This approach was rejected because the target classes are not determined by a single optical response. Aluminum cans may have printed surfaces, paper objects may vary in color, and transparent plastic bottles are strongly affected by reflections and background lighting. The selected design therefore uses the auxiliary sensor only as an object-presence trigger, while the camera-based classifier performs material recognition.

### 2.2.2 Deep Learning Detector vs. Feature-Based Random Forest

A deep learning detector such as YOLO could be used for material classification, but the first prototype operates in a controlled environment with one object inside a fixed observation region. A handcrafted-feature Random Forest classifier is easier to train, requires less data, and is more interpretable during debugging. For this reason, the feature-based model is selected for the current prototype.

### 2.2.3 Fully Embedded Inference vs. External Vision Computer

Running all vision processing directly on the Arduino is not feasible because the Arduino UNO has limited memory and processing capability. The selected design performs image capture, feature extraction, and classification on a Raspberry Pi or computer, then sends only a compact numeric class code to the Arduino. This division keeps the embedded controller simple and reliable.

### 2.2.4 Summary of Corrective Design Choices

The final architecture separates object triggering, material classification, embedded control, and mechanical actuation. This modular approach makes subsystem-level testing easier and reduces the risk that a failure in one component prevents the entire prototype from being debugged.

## 2.3 Design Description and Justification

### 2.3.1 Auxiliary Sensing Module

The auxiliary sensing module detects whether an object is present before the vision classifier is activated. Its output is compared with a calibrated background response using Equations 1 and 2. This module reduces unnecessary image processing and helps coordinate capture timing, but it is not a primary perception or classification module.

### 2.3.2 Vision Perception Module

The vision perception module captures the object image, crops the predefined ROI, resizes the image to 320 by 240 pixels, and extracts a compact feature vector. The feature vec-

tor includes mean and standard deviation of BGR and HSV channels, grayscale brightness statistics, edge ratio from Canny edge detection, highlight ratio, dark-area ratio, maximum contour area, contour count, and normalized color ratios.

These features are selected because the target materials have different visual characteristics. Paper-based objects are usually matte and fibrous, aluminum cans often produce strong highlights and printed edges, and transparent plastic bottles show reflections and contour patterns. The classifier is trained with balanced class weights and a stratified train-test split.

The vision dataset was collected using the same camera position and lighting condition as the final prototype. Four image classes were used in the training process: background, paper-based waste, aluminum beverage cans, and plastic bottles. As summarized in Table 2, the dataset contained 2735 labeled images in total, including 70 background images, 961 paper images, 813 aluminum-can images, and 891 plastic-bottle images.

The images were collected with small variations in object position, rotation, distance from the camera, and surface orientation to improve classifier robustness. For each image, the predefined region of interest was cropped and converted into the feature vector described above. The dataset was divided into training and testing subsets using a stratified split so that each material class was represented in both sets. The held-out test-set performance of the trained Random Forest classifier is reported later in Section 3.

Table 2: Summary of collected vision image data.

| **Data Source** | **Background** | **Paper** | **Aluminum** | **Plastic** | **Total** |
|---|---|---|---|---|---|
| All collected images | 70 | 961 | 813 | 891 | 2735 |

### 2.3.3 Embedded Control Module

The embedded control module is implemented with an Arduino UNO. The vision computer sends an ASCII integer followed by a newline. The Arduino reads the command, maps it to a servo angle, and drives the sorting mechanism. The serial command protocol is listed in Table 3.

### 2.3.4 Mechanical Sorting Module

The mechanical structure uses a modular three-tier assembly. The top layer provides the object feeding and sensing region. The middle layer performs directional routing through

a servo-driven guide chute or gate. The bottom layer supports electronics, wiring, power components, and material collection.

### 2.3.5 Power and Hardware Module

Servo power must be handled separately from Arduino logic power when possible. The final prototype should use an external 5 V supply for the servos, while the Arduino ground and servo supply ground must be connected together. This common-ground connection ensures that the Arduino control signal and the servo power supply share the same voltage reference.

## 2.4 Subsystem Diagrams and Schematics

### 2.4.1 System Architecture Overview

Figure 1 shows the overall pipeline of the proposed recyclable material sorting system. The system is centered on camera-based material classification. The auxiliary sensor provides an object-presence trigger, the vision software predicts the material class, the Arduino receives the predicted code, and the servo mechanism routes the object to the corresponding output bin.

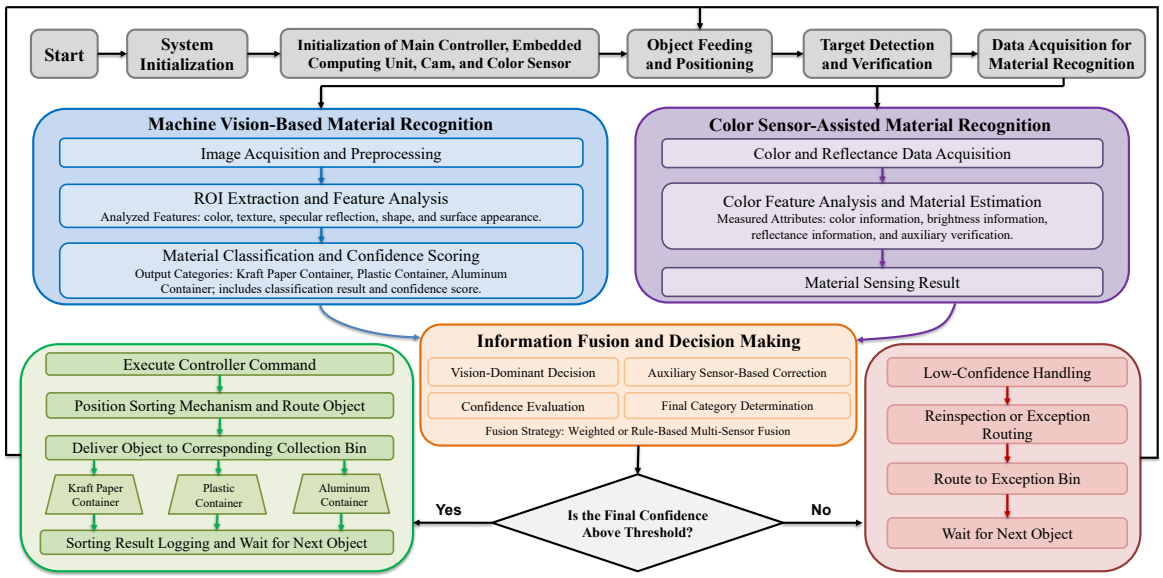


Figure 1: Vision-based recyclable material sorting pipeline.

### 2.4.2 Mechanical Structure

The mechanical sorting subsystem adopts a three-tier acrylic structure that supports object feeding, intermediate routing, and final collection within a compact vertical layout. This arrangement reduces the footprint of the prototype and enables gravity-assisted object movement from the inlet region to the output bins.

Figure 2 shows the assembled CAD rendering of the sorter and illustrates the overall geometry of the three-layer structure. Figure 3 provides an exploded view of the main mechanical parts, including the inlet plate, top layer, middle layer, bottom layer, slide, and support studs. This view makes the layer-by-layer assembly easier to understand. Figure 4 shows the labeled physical prototype and identifies the major components used in the final integrated system, including the object feeding tube, upper acrylic guide plate, auxiliary sensor module, camera, main routing platform, servo motor, secondary servo/gate actuator, Arduino UNO, support rods, backboard, and output bins.

In the final design, the upper region provides the object feeding and sensing area, the middle region performs directional routing through the servo-driven mechanism, and the lower region supports electronics, wiring, and output collection. Together, these three views present the mechanical design from complementary perspectives: overall assembled form, structural decomposition, and real fabricated implementation.

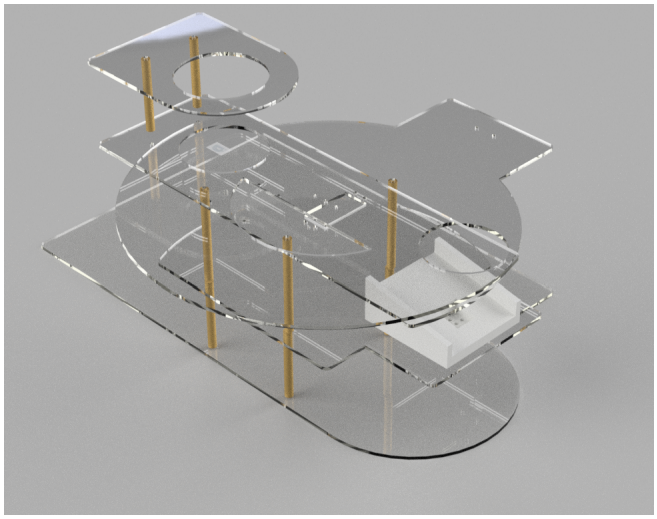

Figure 2: Assembled CAD rendering of the three-tier mechanical structure.

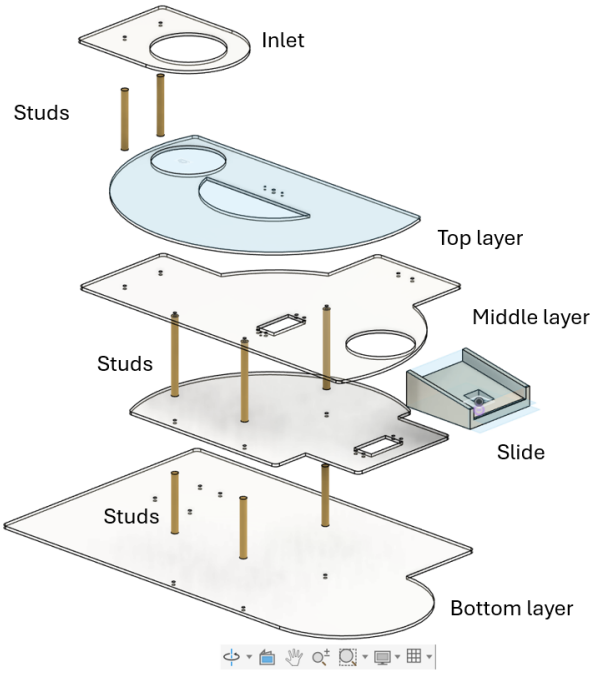


Figure 3: Exploded view of the mechanical structure, showing the inlet, top layer, middle layer, bottom layer, slide, and support studs.

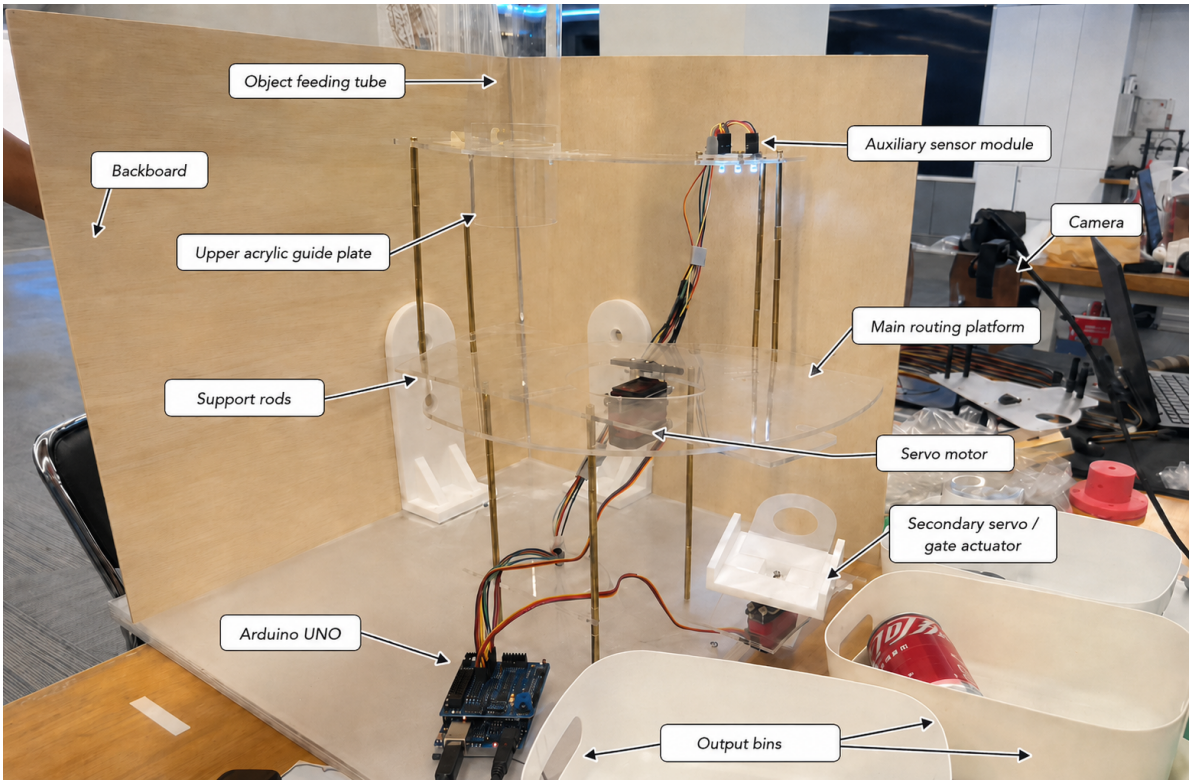


Figure 4: Labeled physical prototype of the vision-guided sorting and pickup cleaning robot.

### 2.4.3 Auxiliary Sensing Circuit

Figure 5 shows the auxiliary object-presence sensing circuit used in the prototype. The circuit provides a lightweight trigger signal for detecting whether an object is present in the observation region. Its output is used only to enable the camera-based recognition process and is not used as the final material-classification result.

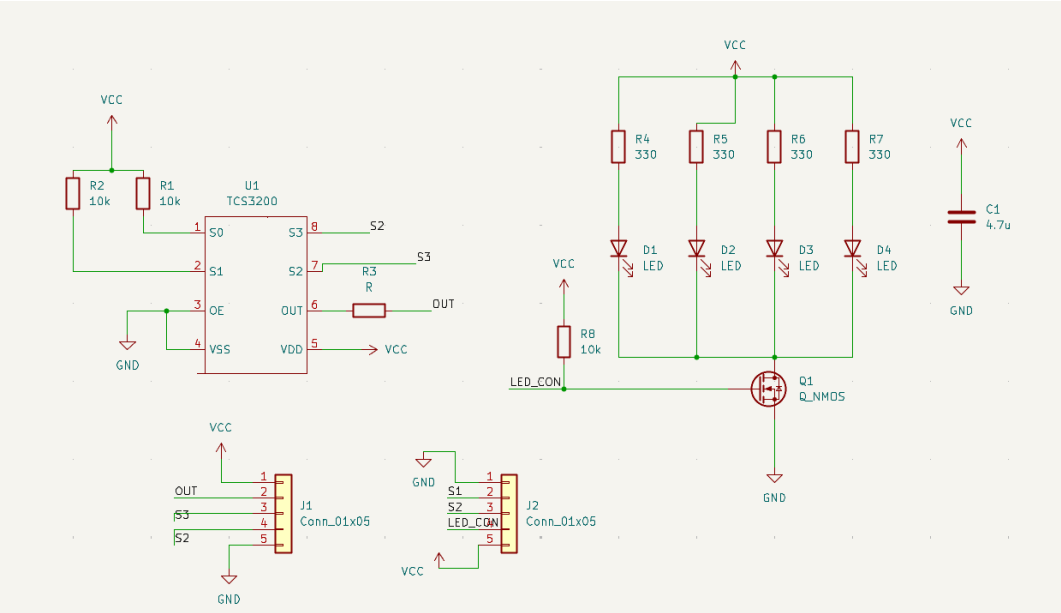


Figure 5: Auxiliary object-presence sensing circuit.

### 2.4.4 Serial Command Protocol

Table 3 summarizes the command protocol used between the vision computer and the Arduino. Code 0 is reserved for the neutral or unknown state, while codes 1–3 correspond to the three target recyclable material classes.

Table 3: Serial command protocol.

| Code | Meaning | Arduino Action |
|---|---|---|
| 0 | Unknown or neutral | Move to neutral angle |
| 1 | Paper-based waste | Move to paper outlet angle |
| 2 | Aluminum beverage can | Move to aluminum outlet angle |
| 3 | Plastic bottle | Move to plastic outlet angle |

### 2.4.5 Servo Angle Assignment

Table 4 lists the initial servo angle assignment for the neutral state and the three target material classes. These values are starting points and must be calibrated after the physical chute and output bins are assembled.

Table 4: Initial servo angle assignment.

| Code | Class | Initial Servo Angle |
|---|---|---|
| 0 | Unknown or neutral | 90° |
| 1 | Paper-based waste | 45° |
| 2 | Aluminum beverage can | 90° |
| 3 | Plastic bottle | 135° |

### 2.4.6 Software Module Summary

The `vision_sorter` folder is organized into small scripts with clear roles. Table 5 summarizes the major software modules and their functions.

Table 5: Software modules.

| File | Purpose |
|---|---|
| `config.py` | Defines paths, labels, camera settings, ROI, confidence threshold, and serial settings. |
| `features.py` | Crops the image and extracts numeric image features. |
| `collect_images.py` | Captures labeled images for each material class. |
| `train_model.py` | Trains the Random Forest classifier and saves model files. |
| `realtime_predict.py` | Runs live prediction, logging, and optional serial output. |
| `serial_test.py` | Sends a chosen numeric code to Arduino for communication testing. |
| `tcs3200_test.ino` | Measures auxiliary sensor response and helps calibrate the trigger threshold. |
| `arduino_serial_sorter.ino` | Receives serial class codes and moves the sorting servo. |

As shown in Table 5, the software structure separates configuration, feature extraction, image collection, model training, real-time prediction, serial testing, and Arduino control.

# 3. Requirements and Verification

The verification process evaluated the system at the same subsystem boundaries used in the design: auxiliary triggering, vision classification, Arduino communication, servo actuation, mechanical sorting, and full-system integration. Compared with the original verification plan, this revised section reports quantitative results collected from the final prototype setup.

## 3.1 Auxiliary Trigger Verification

The auxiliary trigger was verified by comparing the sensor response in empty and occupied observation states. During the test, the background response was first recorded under the final lighting condition. Representative paper, aluminum-can, and plastic-bottle samples were then placed in the sensing region, and the measured response difference was compared with the calibrated threshold.

This verification confirmed only the object-presence function of the auxiliary sensing module. Material classification was evaluated separately using the camera-based classifier. The auxiliary trigger was not used as the final material-classification result.

Table 6 lists the major design risks and mitigation methods related to sensing, classification, timing, power stability, and mechanical alignment.

Table 6: Major design risks and mitigation methods.

| **Risk** | **Mitigation** |
|---|---|
| False auxiliary trigger or missed object | Calibrate the background response and detection threshold under final lighting. |
| Camera ROI misalignment | Mount the camera and auxiliary sensor to observe the same sensing region. |
| Classification error | Collect training images in the final setup and use confidence-based unknown handling. |
| Servo timing or angle error | Calibrate servo angles and object release timing through repeated trials. |
| Power instability | Use an external servo supply with common ground to the Arduino. |

### 3.2 Computer Vision Verification

The computer vision subsystem was evaluated using a stratified held-out test set rather than the full physical sorting process. The test set consisted of approximately 30% of the collected images and was separated from the training set before model training. Based on the collected dataset summarized earlier, the held-out test set included 820 images in total, including 21 background images, 288 paper images, 244 aluminum-can images, and 267 plastic-bottle images.

For each test image, the predicted label was compared with the ground-truth label. As shown in Table 7, the classifier correctly predicted 806 out of 820 held-out test images, resulting in a test-set classification accuracy of 98.3%. This result evaluates the material-recognition performance of the vision model only. The complete physical sorting performance, which also includes serial communication, servo motion, object sliding, and bin entry, is evaluated separately in the full-system integration test.

Table 7: Vision classification results on the held-out test set.

| **Class** | **Number of Test Images** | **Correct Predictions** | **Test Accuracy** |
|---|---|---|---|
| Background | 21 | 21 | 100.0% |
| Paper-based waste | 288 | 284 | 98.6% |
| Aluminum beverage can | 244 | 239 | 98.0% |
| Plastic bottle | 267 | 262 | 98.1% |
| **Overall** | **820** | **806** | **98.3%** |

Most classification errors occurred between aluminum cans and plastic bottles because both classes can produce strong highlights and reflection changes under the camera lighting condition. These results show that the vision classifier is reliable under the controlled image collection setup, while stable lighting and consistent object placement remain important for physical operation.

### 3.3 Arduino Communication and Servo Verification

The Arduino communication and servo mechanism were evaluated through repeated command cycles. During the test, serial codes 0 to 3 were sent from the vision computer to the Arduino, and the corresponding servo response was observed. A total of 100 command cycles were tested.

As shown in Table 8, the Arduino correctly decoded all commands, the servo reached the expected calibrated positions in all trials, and the mechanism returned to the neutral position after each sorting action. The command-response success rate was therefore 100.0% for all three tested items.

Table 8: Arduino command and servo response verification results.

| **Test Item** | **Number of Trials** | **Successful Trials** | **Success Rate** |
|---|---|---|---|
| Serial command decoding | 100 | 100 | 100.0% |
| Servo movement to target angle | 100 | 100 | 100.0% |
| Return-to-neutral motion | 100 | 100 | 100.0% |

### 3.4 Full-System Integration Verification

Full-system verification tested the complete pipeline from object presence to physical sorting. In each trial, a representative paper, aluminum-can, or plastic-bottle sample was placed in the sensing region. A trial was counted as successful only if the object was detected, classified, transmitted to the Arduino as a class code, and guided into the intended output bin.

A total of 200 physical sorting trials were performed, and 185 trials were successful. As summarized in Table 9, the resulting full-system sorting success rate was 92.5%. This success rate is lower than the held-out vision test accuracy because the full-system test also includes mechanical sliding, object placement, servo timing, and bin-entry behavior.

Table 9: Full-system sorting verification results.

| **Test Item** | **Number of Trials** | **Successful Trials** | **Success Rate** |
|---|---|---|---|
| Full-system sorting process | 200 | 185 | 92.5% |

The unsuccessful trials were mainly caused by object placement deviation, insufficient sliding smoothness, and occasional reflection-induced classification uncertainty. These results indicate that the prototype can complete the intended sorting workflow with repeatable behavior, while the mechanical feeding path and lighting stability remain the main areas for future improvement.

Table 10 summarizes the final verification results for each subsystem.

Table 10: Verification result summary.

| **Subsystem** | **Test Method** | **Result** |
|---|---|---|
| Auxiliary trigger | Test empty and occupied sensing states. | The trigger activated the vision pipeline when an object was present and remained idle for empty background cases under the calibrated lighting condition. |
| Vision classifier | Test a stratified held-out image test set. | The classifier achieved a test-set material-classification accuracy of 98.3%. |
| Serial communication | Send class codes from the vision computer to Arduino. | Arduino received and interpreted all tested serial commands correctly. |
| Servo mechanism | Command each sorting position repeatedly. | The servo reached the expected calibrated positions and returned to neutral in 100 out of 100 command cycles. |
| Full sorting system | Run the complete detection-classification-sorting process. | The system achieved a full-system sorting success rate of 92.5% over 200 physical trials. |

## 4. Cost

All costs in this section are listed in RMB. The prototype parts cost is summarized in Table 11. The listed prices are based on current purchase records for mechanical parts, actuator components, camera hardware, and the computing platform.

Table 11: Prototype parts cost.

| **Item** | **Quantity** | **Unit Cost (RMB)** | **Total Cost (RMB)** |
|---|---|---|---|
| M3 screw and nut set | 1 | 15.4 | 15.4 |
| Basswood sheet | 1 | 38.8 | 38.8 |
| Acrylic plate | 2 | 60.0 / 56.0 | 116.0 |
| Acrylic hollow circular tube | 1 | 53.8 | 53.8 |
| Acrylic adhesive glue | 1 | 22.7 | 22.7 |
| M3 brass standoff set | 1 | 36.6 | 36.6 |
| HTS-16L serial bus servo | 1 | 94.0 | 94.0 |
| DS3230 180-degree high-torque digital servo | 2 | 59.36 | 118.72 |
| Raspberry Pi 4B 8GB development kit | 1 | 713.0 | 713.0 |
| 1080P USB industrial camera | 1 | 97.0 | 97.0 |
| **Total prototype parts cost** | – | – | **1306.02** |

Labor cost is estimated using the ECE 445 cost formula

$$\text{Labor Cost} = \text{Ideal Salary} \times \text{Actual Hours} \times 2.5. \tag{6}$$

Table 12 summarizes the estimated labor cost for the four team members. The estimated total project cost is obtained by adding the labor cost to the prototype parts cost in Table 11.

Table 12: Labor cost estimate.

| **Partner** | **Main Work** | **Rate (RMB/hr)** | **Hours** | **Cost (RMB)** |
|---|---|---|---|---|
| Zihan Zhou | Vision pipeline, classifier design, report integration | 120 | 58 | 17,400 |
| Dailin Wu | Mechanical structure, assembly, servo calibration | 120 | 55 | 16,500 |
| Jinyang Chen | Arduino control, serial communication, circuit testing | 120 | 50 | 15,000 |
| Tinghao Pan | Auxiliary trigger testing, verification planning, cost documentation | 120 | 55 | 16,500 |
| **Total** | – | – | **218** | **65,400** |

As shown in Table 12, the estimated labor cost is 65,400 RMB. Together with the prototype parts cost of 1306.02 RMB from Table 11, the total estimated project cost is 66,706.02 RMB. If only purchased prototype components are considered, the direct hardware cost is 1306.02 RMB.

# 5. Conclusion

## 5.1 Accomplishments

This project developed a complete design for a vision-based recyclable material sorting system. The proposed system integrates camera-based material classification, Arduino serial communication, servo actuation, auxiliary object triggering, and a modular mechanical sorting structure. The design demonstrates a complete pipeline from sensing and recognition to physical sorting.

The main accomplishment is the separation of system functions into reliable and testable modules. The vision subsystem performs the main material-classification task, the Arduino controls the mechanical response, and the auxiliary trigger helps reduce unnecessary processing when no object is present. This modular architecture makes the prototype easier to debug and better suited for a short senior design schedule.

## 5.2 Uncertainties and Unresolved Issues

Several uncertainties remain before final demonstration. The auxiliary trigger threshold must be calibrated under the final lighting condition. The camera and auxiliary sensor must be aligned so that both observe the same object region. The classifier performance depends on collecting enough training images under the final mechanical and lighting setup. The servo angles in Table 4 are initial values and must be adjusted after the chute and output bins are assembled.

Power stability is another important issue. Servo motors can draw large transient current, especially during rapid motion or near-stall conditions. The final prototype should therefore use an external servo supply and a common ground with the Arduino.

## 5.3 Future Work and Design Alternatives

Future work should improve the classifier dataset, test the system under more varied lighting conditions, and add more material categories. A deeper vision model such as YOLO could be introduced if the dataset becomes large enough and if the computing platform can support real-time inference. The mechanical design can also be improved by adding smoother object feeding, more stable chute geometry, and stronger mounting points for the camera and sensor.

### 5.4 Broader Impacts

The project has educational and environmental value. It provides a compact example of automated recycling technology and demonstrates how sensing, computer vision, embedded systems, and mechanical design can be integrated into a practical prototype. Although it is not intended to replace industrial recycling systems, it can help students understand the engineering challenges behind automated waste sorting.

### 5.5 Ethical Considerations

The project should be presented as a controlled educational prototype rather than a complete recycling solution. It is designed for three known, relatively clean material classes in a fixed setup and should not be used for hazardous waste, contaminated waste, glass, sharp objects, or unknown materials.

The vision subsystem uses a camera, so the camera should be aimed only at the sorting region. Dataset images, prediction logs, and videos should be used only for development, testing, and project presentation. The system should handle uncertain predictions by returning to the neutral state instead of forcing an incorrect sorting decision. This conservative behavior is consistent with safe and honest engineering practice.

## References


[1] Arduino Documentation, "Arduino UNO Rev3," accessed May 2026. [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3

[2] ams OSRAM, "TCS3200 Programmable Color Light-to-Frequency Converter Datasheet," accessed May 2026. [Online]. Available: https://look.ams-osram.com/m/664723bdb31f55db/original/TCS3200-DS000107.pdf

[3] OpenCV Documentation, "OpenCV-Python Tutorials," accessed May 2026. [Online]. Available: https://docs.opencv.org/

[4] scikit-learn Documentation, "RandomForestClassifier," accessed May 2026. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[5] Raspberry Pi Documentation, "Raspberry Pi 4 Model B," accessed May 2026. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

# A. Requirement Traceability Appendix

This appendix summarizes how the high-level requirements in Section 1.4 are connected to the verification procedures discussed in Section 3. The purpose of this appendix is to make the relationship between system requirements, test methods, and expected outcomes easier to review.

Table 13: Requirement traceability summary.

| **Requirement** | **Verification Method** | **Expected Outcome** |
|---|---|---|
| Object-presence triggering | Compare empty and occupied auxiliary sensor responses under the final lighting condition. | The vision pipeline is triggered only when an object is present. |
| Material classification | Test saved images and live camera samples from paper, aluminum, and plastic classes using the trained classifier. | The system predicts the correct material class or returns unknown for low-confidence cases. |
| Arduino command and servo response | Send serial codes 0–3 from the vision computer to the Arduino and observe the corresponding servo motion. | The Arduino interprets each command correctly and moves the servo to the calibrated position. |
| Physical sorting functionality | Run complete sorting trials using representative paper, aluminum, and plastic objects. | Each object is routed to the intended output bin in the correct sequence. |
| Low-cost system integration | Compare the selected components, labor estimate, and total cost summarized in Section 4. | The prototype remains low-cost, modular, and suitable for a table-top demonstration. |

## B. Team Contribution Summary

This appendix summarizes the major contributions of each team member. Table 14 lists the primary responsibility areas during the design, implementation, and debugging of the prototype.

Table 14: Team contribution summary.

| **Team Member** | **Contribution** |
|---|---|
| Zihan Zhou | Main software development, including the Raspberry Pi vision-training pipeline and the overall code framework. |
| Dailin Wu | Mechanical-control coordination, including servo-structure interaction and motion-control code development. |
| Jinyang Chen | Vision data collection and labeling, with additional support for program development and testing. |
| Tinghao Pan | Main mechanical design, prototype assembly, and physical debugging of the sorting structure. |