ECE445

SENIOR DESIGN LABORATORY

DESIGN DOCUMENT

TCS3200-Assisted Vision-Based Recyclable Material Sorting System

Team #19

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

Advisor: Meng Zhang

May 15, 2026

Contents

1. Introduction

1.1 Problem Statement

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 present, capture and classify the object, send the result to an embedded controller, and actuate the sorting mechanism at the correct time.

This project proposes a **TCS3200-assisted vision-based recyclable material sorting system**. The TCS3200 color-frequency sensor is used as an auxiliary object-presence detector, while the camera and software classifier identify the material category. This division of function prevents unnecessary image classification when no object is present and improves the timing coordination between sensing and actuation.

1.2 Solution Overview

The proposed system is a table-top sorting device for three predefined recyclable-waste 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 system uses a two-stage sensing architecture. The first stage uses the TCS3200 sensor to detect whether an object has entered the observation region [2]. The sensor converts reflected light intensity into an output frequency. If the measured frequency changes beyond a calibrated threshold, the system determines that an object is present and enables the visual recognition stage.

The second stage uses a camera and a lightweight computer vision model to classify the detected object. 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].

The embedded control path is implemented with an Arduino UNO and servo actuators [1]. After the vision software predicts the material category, the Raspberry Pi or computer sends a numeric code to the Arduino through serial communication. The Arduino then moves the sorting servo to the corresponding calibrated angle.

The TCS3200 is not used as the primary material classifier. Its role is to provide a hardware-level trigger for object presence. The final material classification is performed by the camera-based model because the target categories depend on shape, texture, reflection, transparency, contour, and overall appearance rather than color alone.

1.3 High-Level Requirements List

The project will be considered successful if the following high-level requirements are met.

Object-Presence Detection: The TCS3200 auxiliary sensing module shall distinguish between an empty sensing region and an occupied sensing region under controlled lighting. When no object is present, the system shall remain in the idle state and avoid sending unnecessary 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, a TCS3200 sensing circuit or module, 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.

2. Design

2.1 Overall System Architecture

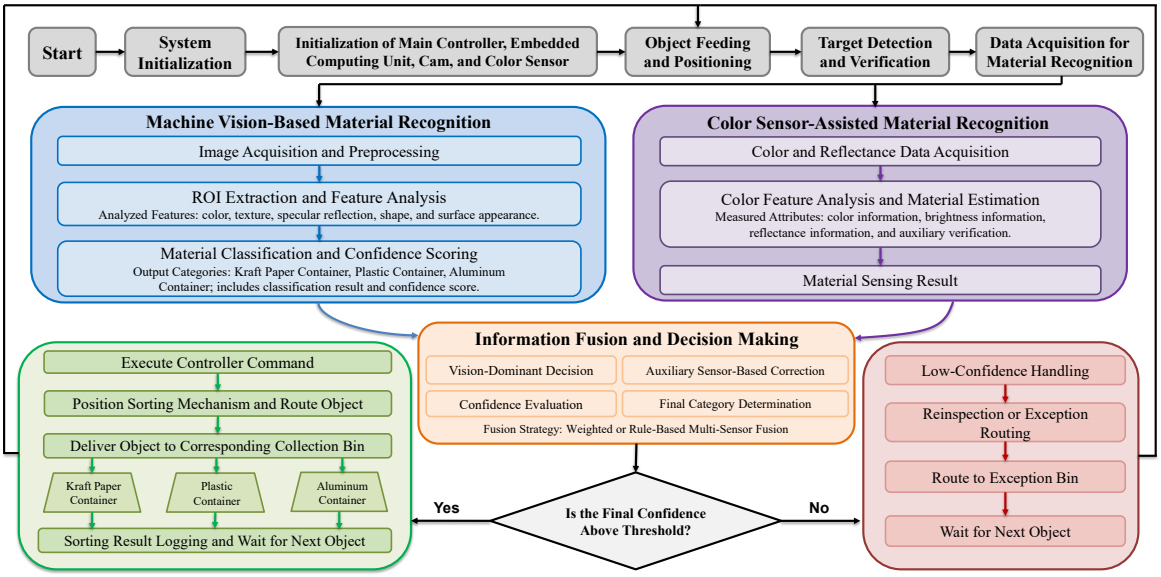

Figure 1: Vision-Based Recyclable Material Sorting Pipeline

The sorter is organized as a two-stage sensing and sorting system. The TCS3200 color-frequency sensor provides object-presence detection, while the camera-based vision model performs material classification. After classification, the vision computer sends a class code to the Arduino, and the Arduino drives the servo-based mechanism to route the object into the corresponding bin.

The TCS3200 is not used as the primary material classifier. Its role is to prevent the vision model from processing empty frames and to provide a timing trigger for image capture and servo control. 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.

2.2 Mechanical System Design

The mechanical structure of the sorter is designed as a modular three-tier assembly, as shown in Figure 2. The structure is mainly constructed from laser-cut acrylic plates. This vertical layout reduces the footprint of the prototype and supports a gravity-assisted sorting process.

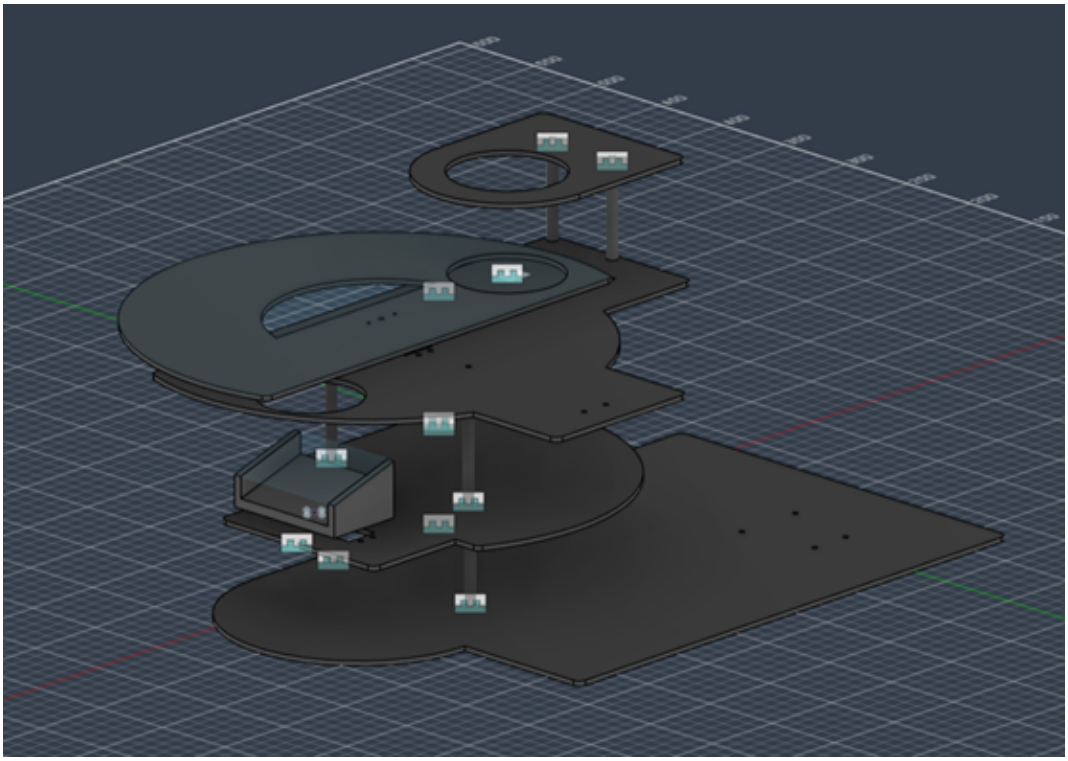

Figure 2: Three-tier mechanical structure of the recyclable material sorter.

Top layer: infeed and initial positioning. The top layer provides the object feeding path and the initial positioning area. The camera and TCS3200 sensor are mounted to observe this region so that object presence can be detected before image classification. A servo-driven mechanism can hold or release the object after classification.

Middle layer: directional routing. The middle layer acts as the routing stage. After the object is classified, the sorting servo rotates the guide chute or gate to the corresponding

outlet direction.

Bottom layer: base, electronics, and collection. The bottom layer provides structural support and mounting locations for the Arduino, PCB, power supply, wiring, and collection region.

2.3 Auxiliary Object-Presence Detection Using TCS3200

The TCS3200 sensor converts reflected light intensity into an output frequency. In this project, it detects the transition between an empty observation area and an occupied observation area. During calibration, the system records the background frequency f_{bg}. During operation, the measured frequency is denoted as f_{meas}, and the frequency difference is

$$\Delta f = |f_{\mathrm{meas}} - f_{\mathrm{bg}}|. \tag{1}$$

The object-presence decision is defined as

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

where f_{th} is the calibrated detection threshold and d is the binary object-presence flag.

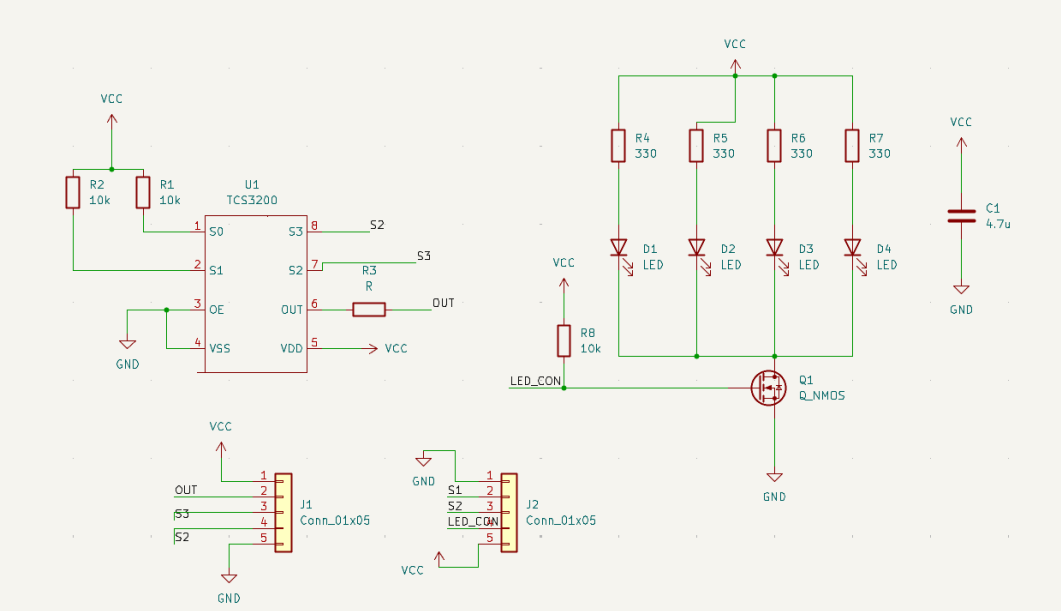

Figure 3: TCS3200 auxiliary object-presence detection circuit.

In this circuit, R1 and R2 pull the S1 and S0 pins to VCC, setting the default frequency-scaling mode. The OE pin is connected to ground so that the output remains enabled. The OUT pin is routed to the controller through a protection resistor. C1 provides local power-supply decoupling, while the LED circuit provides controlled illumination. Q1 acts as a low-side NMOS switch controlled by the LED_CON signal.

2.4 Vision Perception System

2.4.1 Camera Input and 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:

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

This region corresponds to the central part of the camera image where the object is expected to appear. Cropping the frame reduces background interference and makes the training and real-time prediction conditions more consistent.

2.4.2 Dataset Collection Workflow

The training dataset contains one folder per class: paper-based waste, aluminum beverage can, and plastic bottle. Images are collected using the same camera, lighting, background, and object position planned for the final prototype. This is important because the classifier is lightweight and depends on the visual distribution of the final setup.

The unknown state is not treated as a normal material category in the first version. Instead, it is produced by a confidence threshold. If the Random Forest classifier has low confidence for all known classes, the system sends code 0 and returns the sorting mechanism to the neutral state.

2.4.3 Feature Extraction

The feature extraction module converts each image into a compact numeric vector. The region of interest is resized to 320 by 240 pixels, and the following feature groups are computed:

- Mean and standard deviation of BGR and HSV color channels.
- Mean and standard deviation of grayscale brightness.
- Edge ratio from Canny edge detection.
- Highlight ratio and dark-area ratio.
- Maximum contour area and contour count.
- Normalized blue, green, and red channel ratios.

These features are selected because the target objects 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.

2.4.4 Classifier Training

The training script loads images from the class folders, extracts features, and trains a Random Forest classifier. The current configuration uses a fixed random seed, balanced class weights, and a stratified train-test split. The training process saves the trained classifier and the label-code mapping for real-time prediction.

A deep learning detector such as YOLO is not selected for the first prototype because the task is limited to one object inside a fixed observation region. The handcrafted-feature Random Forest approach requires less training data, is easier to debug, and is more suitable for the available project timeline.

2.5 Real-Time Prediction and Logging

The real-time prediction script loads the trained model, opens the camera, extracts features from the region of interest, predicts the class, and sends the result to the Arduino when serial mode is enabled. In preview mode, the user can see the region of interest and current predicted label. In headless mode on a Raspberry Pi [5], the script can print predictions to the terminal and send serial commands without opening a display window.

The real-time loop is triggered by object presence. When the TCS3200 indicates that an object is present, the system captures the current camera frame, computes the predicted class,

applies the confidence threshold, and sends the corresponding command to the Arduino. Optional CSV logging records the timestamp, TCS3200 object-presence flag, predicted code, text label, and confidence.

2.6 Arduino Control System

2.6.1 Serial Communication

The Arduino interface uses a simple serial protocol. The vision computer sends an ASCII integer followed by a newline. The Arduino reads the integer from the serial port and maps it to a servo command.

Table 1: 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

The Arduino may also monitor the TCS3200 output frequency or use the detection signal as an internal state-machine condition. In either case, the sorting command is executed only after an object is detected and classified.

2.6.2 Arduino State Machine

The Arduino operates as a simple state machine. In the idle state, it waits for object detection and keeps the servo at the neutral position. After the object is detected and classified, the Arduino receives the class code, moves the sorting servo to the corresponding calibrated angle, and then returns the mechanism to the neutral state.

2.6.3 Servo Sorting Mechanism

The mechanical prototype uses servo actuators for object feeding, release, and sorting. After the predicted class code is received, the sorting servo rotates to a calibrated angle before the object is released into the chute. The initial angle assignment is shown in Table 2; these

values are starting points and must be calibrated after the physical chute and output bins are assembled.

Table 2: 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°

2.7 Software Structure

The `vision_sorter` folder is organized into small scripts with clear roles. This structure allows the team to test each subsystem independently before connecting the full prototype.

Table 3: 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 TCS3200 frequency and helps calibrate the detection threshold.
`arduino_serial_sorter_example.ino`	Receives serial class codes and moves the sorting servo.

2.8 Power and Hardware Considerations

Servo power must be handled carefully. The final prototype should use a separate 5 V supply for the servos, and the Arduino ground and servo supply ground must be connected

together. This common-ground connection is necessary because the Arduino control signal and the servo power supply need the same voltage reference.

Servo actuators can draw large transient current during motion or near-stall conditions. Using an external supply reduces the chance of Arduino reset, voltage drop, and unstable servo motion.

The camera and TCS3200 should be mounted rigidly and aligned to observe the same sensing region. This alignment ensures that the object detected by the TCS3200 is also visible inside the camera region of interest.

2.9 Tolerance Analysis and Design Risks

The main design risks are summarized in Table 4. These risks are related to sensing, classification, timing, power stability, and mechanical alignment.

Table 4: Major Design Risks and Mitigation Methods

Risk	**Mitigation**
False TCS3200 trigger or missed object	Calibrate the background frequency and detection threshold under final lighting.
Camera ROI misalignment	Mount the camera and TCS3200 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.

2.10 Verification Plan

Each major subsystem will be tested independently before full-system integration. The verification plan is summarized in Table 5.

Table 5: Verification Plan

Subsystem	**Test Method**	**Expected Result**
TCS3200 sensing	Test empty and occupied sensing states.	The sensor should trigger only when an object is present.
Vision classifier	Test collected images and live samples.	The system should classify target materials or mark uncertain cases as unknown.
Serial communication	Send class codes from the vision computer to Arduino.	Arduino should receive and interpret commands correctly.
Servo mechanism	Command each sorting position.	The servo should move to the calibrated position and return to neutral when needed.
Full sorting system	Run the complete detection-classification-sorting process.	The object should be guided to the corresponding output bin in the correct sequence.

This plan helps isolate errors among sensing, classification, communication, actuation, and mechanical sorting.

3. Cost

Table 6: Preliminary Bill of Materials

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	–	–	**1306.02**

4. Schedule

Only four weeks are available for the revised project, so the schedule focuses on rapid purchasing, prototype assembly, subsystem testing, and full system integration.

Table 7: Project Schedule

Week	**Main Tasks**
Week 1	Finalize the TCS3200-assisted vision architecture, purchase list, mechanical layout, and dataset plan.
Week 2	Assemble the first mechanical prototype, bring up Arduino control, connect the TCS3200 circuit, and collect initial images.
Week 3	Calibrate the TCS3200 threshold, train the first classifier, test serial communication, and tune servo angles.
Week 4	Complete the full detection-classification-sorting loop, run integration tests, and prepare the final demonstration.

The main schedule risk is the interaction between sensing, classification timing, and physical object motion. Integration must begin early so that trigger timing, camera capture, servo movement, and chute alignment can be tested together.

5. Ethics and Safety

5.1 Ethics

The project is a controlled educational prototype, not a complete recycling solution. It is designed for three known, relatively clean material classes in a fixed setup and should not be presented as a general-purpose waste classifier for mixed household trash, contaminated waste, hazardous materials, or unknown objects.

The vision subsystem uses a camera, so the camera should be aimed only at the sorting area. Dataset images, prediction logs, and videos should be used only for development, testing, and presentation of this project. Classification limitations should be reported honestly, especially for aluminum cans with varied printed surfaces and transparent plastic bottles with unstable reflections.

5.2 Safety

The main safety concerns are moving servos, electrical wiring, and test object selection. Team members should keep hands away from the chute and sorting gate while the Arduino

is powered and the real-time prediction script is running.

Electrical connections should be stable, loose wires should be avoided, and servo power should remain within rated limits. Test objects should be lightweight and safe. The system should not be tested with sharp objects, liquid-filled containers, glass, hazardous waste, or anything that could spill into electronics. When the classifier is uncertain, the system should choose the neutral state instead of forcing a sort.

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/