Smart Plastic Container Recycling System

ECE 445 Senior Design

Design Document

9/28/2023

Team 7
Jason Wright, Jennifer Chen, Smruthi Srinivasan
Jeff Chang

Table of Contents

1. Introduction **3**
1.1 Problem 3
1.2 Solution 3
1.3 Visual Aid 4
1.4 High Level Requirements 4
2. Design **6**
2.1 Block Diagram 6
2.2 Physical Design 6
2.3 Control Subsystem 7
2.3.1 Overview 7
2.3.2 Requirements and Verification 7
2.4 Power Subsystem 8
2.4.1 Overview 8
2.5.2 Requirements and Verification 8
2.5 Sensor Subsystem 9
2.5.1 Overview 9
2.5.2 Requirements and Verification 9
2.6 User Interface Subsystem 10
2.6.1 Overview 10
2.6.2 Requirements and Verification 10
2.7 Software Design 11
2.8 Tolerance Analysis 12
3. Cost and Schedule **13**
3.1 Cost Analysis 13
3.1.1 Cost of Materials 13
3.1.2 Cost of Labor 13
3.2 Schedule 14
4. Ethics and Safety **16**
4.1 Ethics 16
4.1.1 Seek honest criticism/feedback of our technical work and acknowledge the feedback [14] 16
4.1.2 Treat all people with fairness and respect [14] 16
4.1.3 Learn and apply new skills through design and implementation [14] 16
4.2 Safety 16
4.2.1 Follow all lab safety guidelines [14] 17
4.2.2 Data Privacy [14] 17

1. Introduction

1.1 Problem

Recycling maintains a lot of benefits for the community around us, especially as we aim to tackle the effects of climate change. The benefits of recycling are countless, but recycling works to reduce waste and pollution, conserve energy and natural resources, and create and support jobs domestically. Unfortunately, a lot of people struggle with determining which materials can be recycled and where they can effectively recycle them in addition to the recycling infrastructure in the United States being outdated [1].

While other countries have effectively taught their population how to correctly recycle their items from a young age, the United States lacks education on proper recycling. This leads to contamination of other recyclables, ultimately preventing them from being recycled. In fact, estimates show that over 50% of waste ends up in landfills instead of being recycled [2]. We usually think of plastics as recyclable, but depending on the jurisdiction, some plastics may not be able to be recycled. If they are accidentally recycled, they run the risk of contaminating all of the other recyclables, which is a mistake we can no longer afford as the potential effects of climate change loom ahead.

1.2 Solution

Our solution to this problem is a device with an imaging system that reads the symbols printed on plastic containers. This device will be mounted on a user's trash system. We will have a camera sensor that works with a machine learning model (VGG16) to read the numbers printed on the plastic container and a GPS sensor that determines the location of the user using latitude and longitude coordinates. That information will be utilized to determine if that specific plastic container can be recycled in the user's location using RecycleNation API. Once the determination has been made, we will have a sorting actuator that places the plastic in the proper bin and a web application explaining more about the type of plastic and display the recycling centers nearby (if any).

1.3 Visual Aid

Figure 1: Visual Aid

1.4 High Level Requirements

1. Camera detects the plastic being positioned in front of it 95 ± 1% of the time and system is able to correctly identify the symbol listed on the plastic container 95 ± 1% of the time
2. GPS location sensor determines the user's location within a 10 meter radius and pulls the data regarding recycling in that area

3. System correctly determines 95 ± 1% of the time if the container is recyclable or not and places the container in the proper bin, web application displays specific information about plastic being recycled/specific location centers that accept this type of plastic.

2. Design

2.1 Block Diagram

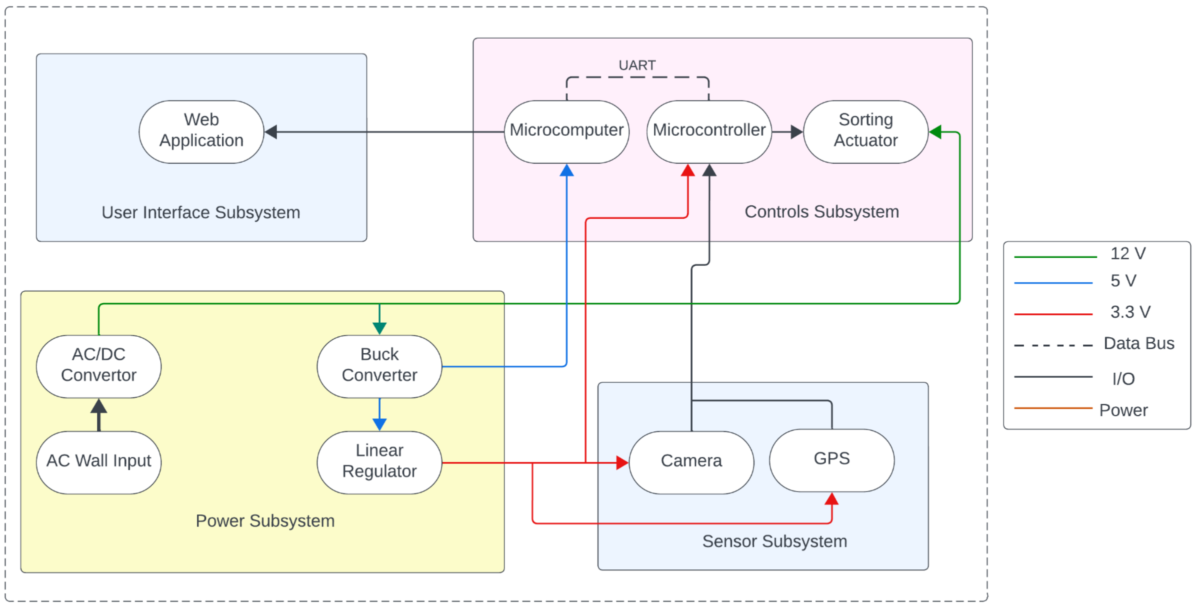

Figure 2: Block Diagram

2.2 Physical Design

The sorting mechanism of our system will be a simple platform that the user can place the desired plastic object onto. Our camera will sit several inches back from where the object is placed in order to view the entire item. Our pcb and other components will be encased and mounted behind the user side of the system next to the camera. All of this will be mounted above two small trash cans, one on either side of the sorting mechanism. Once the system has decided if an object is recyclable, our 12V stepper motor will tilt the platform through the use of a chain attached to an axel. The use of a stepper motor allows us more precise position control, allowing it to return to the level position after each sort. Additionally, the chain system allows us to customize the sensitivity of our rotation through the use of simple gears. Our motor contains 200 steps per revolution, giving a base precision of 1.8° ± 5%, which we can increase as needed.

2.3 Control Subsystem

2.3.1 Overview

The control subsystem consists of our microcontroller (STM32F103C8T6), a microcomputer (Raspberry Pi 3), and our 12V stepper motor and driver. The motor and driver will directly take 12V input from the wall converter, the microcomputer will take 5V input from the buck converter [3] [4], and the MCU will use 3.3V from our LDO [5] [6]. The MCU has 3 functions: data collection from the sensor subsystem, communication with the microcomputer, and control of the stepper motor. The stepper motor will be used for physical sorting items by tilting the platform where items are placed once they have been identified as recyclable or not from our other systems. The raspberry pi will be loaded with the trained image recognition software and be able to connect to a web application for the purpose of retrieving location specific recycling data and providing information to the user.

2.3.2 Requirements and Verification

Requirement	Verification
• Stepper motor capable of rotating the amount required to move the item and return to level position ± 1.8° (1 step)	• Test sorting motion on objects of different weights and shapes to find necessary angle difference (estimated 45°) • Check angle of return position and confirm new items will be able to stand
• MCU can communicate with multiple other components and collect data for analysis in an organized way with use of one of several supported protocols	• MCU receives data input from camera and gps modules with minimal losses (>98% success rate) • MCU communicates back and forth with microcomputer, can be verified through raspberry pi output • Display the data inputs to the U/I to verify
• Microcomputer interfaces with the internet and retrieves data in a reasonable time (<5 seconds)	• Run a script that times the data retrieval and outputs the time

	• Display script output on the U/I and check the time is <5 seconds and the information retrieved is accurate

Table 1: Control Subsystem RV Table

2.4 Power Subsystem

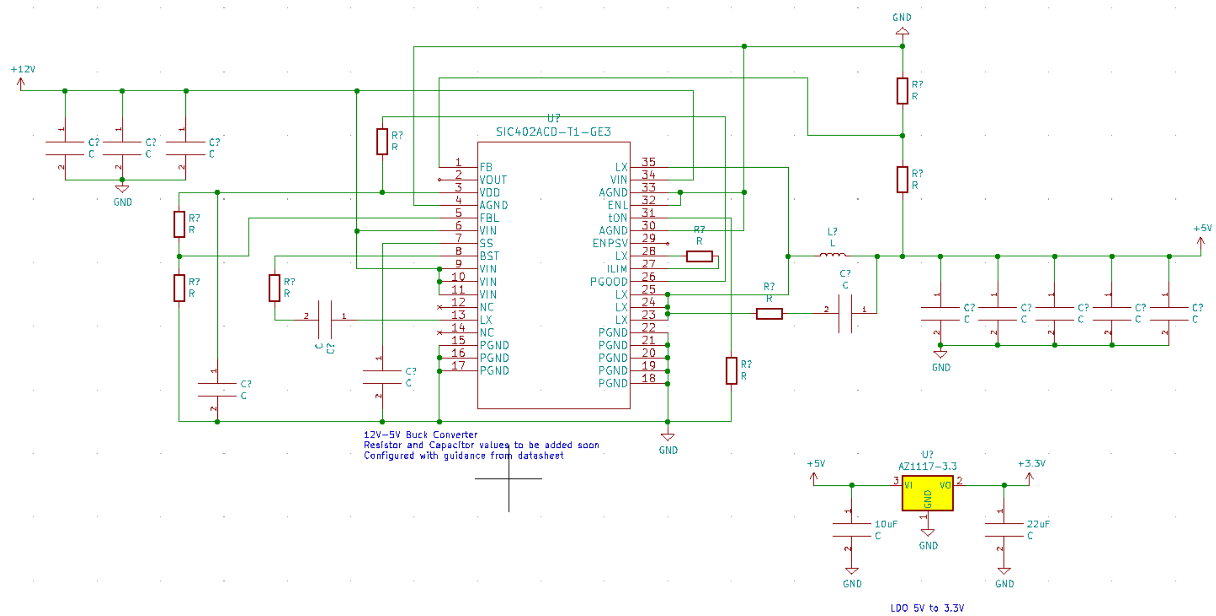

Figure 3: Buck Converter and LDO Schematic

2.4.1 Overview

The power subsystem consists of three major components. First, 120 VAC power is rectified to a 12V DC value by a 30W converter (Qualtek QFWB-30-12-US01). The 12V power can be passed directly to the stepper motor and its driver. It is also sent to a buck converter that steps the power down to 5V. The 5V power is sent directly to the raspberry pi, and to a linear, low-dropout regulator (LDO). The LDO steps the voltage down one more time to 3.3V which can be passed to the microcontroller, camera, and gps. Both DC-DC converters will need exterior resistance, capacitance, and inductor values according to their individual datasheets.

2.5.2 Requirements and Verification

Requirement	**Verification**
• Ripple on the 5V system stays under	• Ripple can be measured with an

250 mV, as needed by the raspberry pi and can support expected current draw by the LDO and raspberry pi (1.5 A)	oscilloscope, current support is available on datasheets
• Ripple on the 3.3V system is kept under 300 mV for continuous MCU operation, current must support all 3.3V components	• Ripple can be measured with an oscilloscope, current support is available on datasheets

Table 2: Power Subsystem RV Table

2.5 Sensor Subsystem

2.5.1 Overview

The sensor subsystem contains the hardware necessary for the device to capture an image of recycling symbol on the container via the camera and for it to locate the user's coordinates using a GPS. The camera will be positioned to take an image of the user's plastic container and it will transmit the image data to our microcontroller. A GPS will also be located on the sensor subsystem to determine the user's location and those coordinates will be sent to the microcontroller for processing.

2.5.2 Requirements and Verification

Requirements	Verification
• The image the camera produces must be at least 244 x 244 to ensure that the symbol can be read properly by the VGG model.	• Place a plastic container in front of the camera sensor • Verify that the pixel count is at least 244 x 244 pixels using OpenCV
• The GPS must provide coordinates within a 10 meter radius given that the device is in a location free of large obstacles.	• Place device complete with the GPS sensor in a location with known coordinates • Display the GPS coordinates sent by the device on the web page (for verification purposes only)

	• Calculate the distance between the true location and GPS coordinates • Verify the calculated distance is within a 10 meter radius

Table 3: Sensor Subsystem RV Table

2.6 User Interface Subsystem

Figure 4: Web Application Front End [7]

2.6.1 Overview

The user interface will consist of a full stack web application where information about the specific plastic the user is attempting to recycle is displayed, in addition to the locations of specific recycling centers that accept the type of plastic inputted. The user interface will receive this information from the microcomputer and use that to display the necessary information. The left side of the page will display the plastic type that is pulled from our microcomputer software and information about it and the right side of the page will display a map that pinpoints the locations using the RecycleNation API [8] and Google Maps API [9]. The web application will be built using React.js for the front end and Django on the back end.

2.6.2 Requirements and Verification

Requirements	Verification
• The web application must display information of the type of plastic the user is attempting to recycle	• Navigate to the web page • View the left hand side for plastic type and the corresponding information • Verify the plastic type displayed matches the type printed on the container
• The web application must be able to display recycling locations within a 30 mile radius, if applicable to a user's location.	• Navigate to the web page • View the right hand side for the map visualization • Verify that the locations (if any) displayed accept the plastic type

Table 4: User Interface Subsystem RV Table

2.7 Software Design

The machine learning model that we have selected is the VGG16 Model [10]. This model is a deep Convolutional Neural Network that consists of 16 layers of convolutional layers, max pooling layers and dense layers. We will use a model that is pre-trained on ImageNet [11] and fine tune the model using the Plastic Identification Symbol dataset [12]. This model will be loaded onto our microcomputer and it will be fine tuned once it is connected to the controls subsystem.

The image that the camera sensor takes will be our input image into the network. As a requirement of our control subsystem, we will verify that the image is 244x244 pixels beforehand. This image then passes through a series of convolutional layers of 3x3 filter size that learn specific features of the image and max pooling layers that selects the maximum value of our image feature map. This works to increase efficiency and reduce computational complexity, which is important since a downside of using the VGG model is that since it has 16 layers it can take a long time to train. It then goes through 3 fully connected layers that work on classifying the image and outputting the category it most likely belongs to.

As mentioned, we will be utilizing a pre-trained model and fine tuning it in order to fit our project's needs. To implement this in our project, we will be utilizing PyTorch and OpenCV libraries. An example classification for our use case is first, the camera captures an image of a water bottle's recycling symbol. The image will then be sent to the microcontroller and then the microcomputer where it will be inputted into the VGG model network. The image will pass layer by layer as the network learns its features and performs a classification. The final output will be a list of class names and the corresponding probabilities that this image belongs to the specific class, with Plastic #1 being at the top with the highest probability.

2.8 Tolerance Analysis

The primary cause for concern relating to design tolerance is the voltage regulation of the various components. Our power source is meant to be the standard 120 VAC wall supply. We will use a converter currently available for purchase (Qualtek QFWB-30-12-US01, 30 W converter) to rectify this to 12 VDC, the voltage needed by our stepper motor. This voltage has a maximum ripple of 200 mV, well within the range acceptable by our motor and driver, so we will not regulate this with anything more complex than capacitors.

Our MCU, camera module, and gps sensor all run off of 3.3 V inputs. To minimize power lost in our voltage regulators, we will first step our 12 V power down to 5V using a buck converter (Vishay Siliconix SIC402ACD-T1-GE3). Our microcomputer can be powered directly from the 5 V, with some additional regulation according to its datasheet, and it is expected to draw well under 1A (less than half of what we can supply). From the 5 V, we can then make use of linear low-dropout regulators. The MCU, gps, and camera will be powered through two AZ1117CH-3.3TRG1 LDO with a dropout voltage of 1.3V. This single regulator is capable of supplying 1 A of maximum output current. The MCU draws a maximum of 50 mA under full load, plus a maximum of about 100 mA for the gps and camera modules combined. Other power dissipation will come from connections, and some potential I/O additions, such as buttons for manual control. We can estimate the junction temperature of an LDO with the following equation:

$$T_j = i_{out}(V_{in} - V_{out})(\Theta_{jc} + \Theta_{ca}) + T_a \tag{1}$$

From the datasheet of our LDO, we find that our maximum operable temperature is 125°C, our Θ junction is 15 °C/W, and we estimate our current to potentially peak at 200 mA. If we then estimate a case-to-ambient thermal resistance of 100 °C/W and use a hot-day temperature of

35°C for ambient temperature, we get a final junction temperature of 70.7 °C, well below our maximum operating temperature.

3. Cost and Schedule

3.1 Cost Analysis

3.1.1 Cost of Materials

This table does not include components such as resistors and capacitors as we are finalizing the quantities necessary. This table will be updated in the future to reflect the addition of these components.

Description	**Manufacturer**	**Part Number**	**Quantity**	**Cost ($)**
Microcontroller	STMicroelectronics	STM32F103C8T6	1	6.42
Camera Sensor	Olimex	OV7670	1	5.38
GPS Sensor	u-blox	NEO-M9N-00B	1	27.00
Stepper Motor	Adafruit	324	1	14.00
Trash Can	Sterilite	1.5 Gallon Trash Can	2	1.96
30W Wall Converter	Qualtek	QFWB-30-12-US01	1	9.93
LDO	Diodes Inc.	AZ1117CH-3.3TRG1	1	0.45
Buck Converter	Vishay Siliconix	SIC402ACD-T1-GE3	1	1.99
		Total		$67.13

Table 5: Materials Cost Analysis

3.1.2 Cost of Labor

Since the majority of this group is in Computer Engineering, the hourly rate listed below is based on the average salary of a Computer Engineering graduate, which is $105,352 [13]. This is approximately $52.68 an hour based on a 40 hour work week for 50 weeks a year.

Name	Hourly Rate ($)	Hours	Total ($)	Total x 2.5 ($)

Jennifer Chen	52.68	150	7902	19755
Smruthi Srinvasan	52.68	150	7902	19755
Jason Wright	52.68	150	7902	19755
			Total	$59,265

Table 6: Labor Cost Analysis

We will also be using the Machine Shop in ECEB and that work will take approximately 1 day.

3.2 Schedule

- Week 6 (9/25-9/29):
  - Design document: Everyone
  - Schematic development: Jason
  - Machine learning model selection: Jennifer and Smruthi
- Week 7 (10/2-10/6):
  - Design review: Everyone
  - PCB review and necessary modifications: Jason
  - Order parts and materials: Everyone
  - Conduct additional research on VGG: Jennifer and Smruthi
  - Final discussion with machine shop on mechanical design: Everyone
- Week 8 (10/9-10/13):
  - Order PCB: Everyone
  - Train VGG model: Smruthi and Jennifer
- Week 9 (10/16-10/20):
  - Program microcontroller: Jason
  - Fine tune model on Raspberry Pi: Smruthi
  - Begin user interface development: Jennifer
- Week 10 (10/23-10/27):
  - Final PCB revisions: Jason
  - Order PCB (if necessary): Everyone
  - Finish user interface: Jennifer
  - Integrate microcontroller with Raspberry Pi: Smruthi
- Week 11 (10/30-11/3):
  - Finalize assembly: Jason
  - Complete integration and begin testing: Jennifer and Smruthi
- Week 12 (11/6-11/10):

    - Continue testing: Everyone
- Week 13 (11/13-11/17):
    - Mock demo: Everyone
- Week 14 (11/20-11/24):
    - Final modifications: Everyone
- Week 15 (11/27-12/1):
    - Final demo: Everyone
- Week 16 (12/4-12/8):
    - Final presentation: Everyone

4. Ethics and Safety

4.1 Ethics

4.1.1 Seek honest criticism/feedback of our technical work and acknowledge the feedback [14]

Per IEEE Code of Ethics, we must follow the feedback that we receive from our TA and professor regarding the implementation of our project and make changes accordingly. Throughout the project, we will be consistently meeting with our TA to review our designs and prototypes before
implementation and seek feedback. We will
also thoroughly review and research any ideas if necessary before implementing and make sure to cite any resources that we use.

4.1.2 Treat all people with fairness and respect [14]

Per IEEE Code of Ethics, we must be fair and respectful to everyone in the team and our mentors. To ensure that we are all communicating well and working well as a team, we have a group text as well as a Discord server for coordinating meetings, etc. In addition, we have a shared Google drive folder where we will have all of the project related documents, resources, etc. GitHub will be used for our lab notebook as well as for collaborating on the code.

4.1.3 Learn and apply new skills through design and implementation [14]

Per IEEE Code of Ethics, we must learn any new technical skills necessary for the design and implementation of the project. All of the team members have completed the safety and CAD training, and plan to review additional resources as necessary. The team consists of hardware, application development and AI/ML experience, and we will collaborate together throughout the project development process. We will also consistently meet with our mentors and TAs for feedback.

4.2 Safety

4.2.1 Follow all lab safety guidelines [14]

Although we do not anticipate significant safety hazards in our design, we recognize that we are working with current and voltage which can pose a safety risk. To minimize this risk, we will follow the safety guidelines as we build our project.

4.2.2 Data Privacy [14]

Per IEEE Code of Ethics, we will make sure to protect any user data and consider data privacy pertaining to data collected such as the user's location. We will not store any sensitive information pertaining to a user's location and we will implement necessary measures in the web application. We will also make sure to follow any software license terms and cite any resources that we use.

References

[1] EPA. “The U.S. Recycling System” (2022), [Online]. Available: https://www.epa.gov/circulareconomy/us-recycling-system#:~:text=For%20the%20environment%2C%20recycling%3A,and%20process%20new%20raw%20materials. (visited on 09/12/2023).

[2] A. Bradford, A. Truelove, S. Broude. “Trash in America: Moving From Destructive Consumption to a Zero-Waste System” (2018), [Online]. Available: https://frontiergroup.org/resources/trash-america/#:~:text=Currently%2C%20though%2C%20the%20majority%20(,much%20material%20at%2034.6%20percent. (visited on 09/09/2023)

[3] *“SiC402A/B”,* SiC402A/B, Vishay Siliconix, Jun. 2020. [Online]. Available: https://www.vishay.com/docs/63729/sic402abcd.pdf (visited on 09/21/2023).

[4] *“Reference Board User’s Manual for SiC403 (6 A), SiC402 (10 A), and SiC401 (15 A) Synchronous Buck Regulators”,* SiC402A/B, Vishay Siliconix, Nov. 2014. [Online]. Available: https://www.vishay.com/docs/62923/sic401.pdf (visited on 09/21/2023).

[5] *“STM32F103”,* STM32F103, STMicroelectronics, Sep. 2023. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f103cb.pdf (visited on 09/21/2023).

[6] *“AZ1117C”,* AZ1117C, Diodes, Sep. 2022. [Online]. Available: https://www.diodes.com/assets/Datasheets/AZ1117C.pdf (visited on 09/21/2023).

[7] Plastics for Change. “”The 7 Different Types of Plastics”.” (2021), [Online]. Available: https://www.plasticsforchange.org/blog/different-types-of-plastic (visited on 09/14/2023).

[8] RecycleNation. “”Recycling API”.” (2023), [Online] Available: https://recyclenation.com/recycling-api/ (visited on 09/14/2023).

[9] Google Maps Platform. “”The Maps Embed API Overview”.” (2023), [Online]. Available: https://developers.google.com/maps/documentation/embed/get-started (visited on 09/14/2023).

[10] Boesch, Gaudenz. “”VGG Very Deep Convolutional Neural Networks (VGGNet) - What you need to know”.” (2023), [Online]. Available: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/ (visited on 09/21/2023)

[11] PyTorch. “”VGG16_BN”.” (2017), [Online]. Available: https://pytorch.org/vision/main/models/generated/torchvision.models.vgg16_bn.html#torchvision.models.vgg16_bn (visited on 09/21/2023).

[12] Kaggle. ““Plastic Identification Symbol”.” (2020) [Online] Available: https://www.kaggle.com/code/abhichoudhury/plastic-identification-symbol/input (visited on 09/21/2023).

[13] T. G. C. of Engineering. “”The Grainger College of Engineering - Computer Engineering”.” (2023), [Online]. Available: https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/computer-engineering (visited on 09/21/2023).

[14] IEEE. “”IEEE Code of Ethics”.” (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 09/13/2023).