Dr. Green

Authors: Ting-Hua Hsu, Vasudha Srinivasan, Aichen Yao
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—Dr. Green is a smart recycling device for schools that identifies and self organizes waste to prevent recycling contamination while educating users on the rules of recycling. With a vision-based recycling classifier connected to a microcontroller interface with a mechanical organizer and hardware components for interactive cues, Dr. Green makes learning about recycling a lot easier while reducing contamination and further improving existing waste organization infrastructures.**

***Index Terms*—Arduino, Classification, Computer Vision, Jetson, Recycling, Smart Bin, Waste Sorting**

1 INTRODUCTION

One of the ways people attempt to reduce their waste production is through organization, with infrastructure for trash, recycling, and compost sectors commonly established in public locations such as schools. According to the EPA, while around 75% of overall waste has the potential to be recycled, only 35% of waste in the United States is actually recycled or composted [1]. Of this already low percentage, 25% on average of recyclables are found contaminated with trash [1]. While infrastructure for sorting waste does currently exist and people are generally aware and willing to participate in proper waste organization, complicated recycling rules that differ by region combined with the need to manually separate waste often creates confusion. What makes matters worse is that a single contaminant in a recycling bin can result in the entire bin being deemed unrecyclable, resulting in its disposal in the landfill.

Our goal is to address these very issues with Dr. Green, a smart recycling device that classifies and self sorts attempted recycling waste while providing real time feedback regarding whether the item attempting to be recycled can actually be done so. Its main audience is schools, which have existing attempts at waste and recycling organizations, a large waste producing population, and are a community with a purpose to learn. A self sorting system makes attempts at waste organization easier while preventing error at the root, thus removing the need to further sort at a waste plant. Additionally, with visual and audio feedback for reinforcement regarding recycling attempts, users can learn the given region's recycling rules while reducing the chance that they make the same error later onwards. With this easy to use, self organizing system, we hope to not only educate users on proper recycling rules through reinforcement, but reduce contamination of recycling at its root by identifying and moving recycled non-recyclables to the trash.

2 USE-CASE REQUIREMENTS

The main use case requirements for Dr. Green are as follows:
1. Accurate: Dr. Green needs to be able to accurately separate recyclables from any mistaken recycled trash. Ideally, the level of accuracy needs to be at 100% to prevent any contamination of recycling.
2. Educational: Dr. Green needs to educate users on recycling rules of the region. To do this, the system's setup needs to contain visual and audible feedback alerts that are understandable and educationally reinforcing without an overly negative psychological effect on users who recycled waste incorrectly. For simplicity, we will base this prototype on Pittsburgh's recycling rules.
3. Easy to Use/Maintain: Dr. Green should be straightforward and easy to use for students, as well as easy to set up, empty out, and maintain for staff. Its sizing should be accessible to users of different heights to account for use by younger students.
4. Engaging: The total operation time to throw out a piece of waste should take no more than 5 seconds and alerts must be engaging enough to avoid disinterest of the user.
5. Safe/Hands Free: Self-operation mechanism and material of the system needs to be safe enough for younger users to prevent injury in case of improper use.
6. Sanitary: Since Dr. Green deals with waste, the device needs to avoid sanitary concerns. With a hands free set up, Dr. Green should help users avoid contact with unsanitary surfaces that most existing trash cans consist of.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

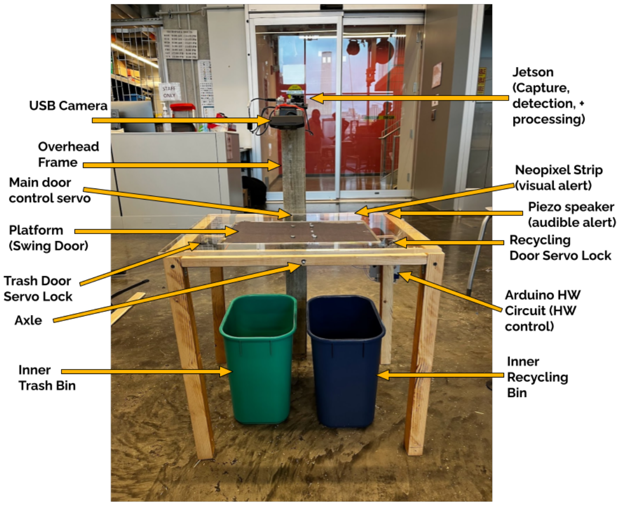


Figure 1: Dr. Green overall system (outer bin removed)

Our design consists of the following sections:

A. CV System (Object Identification)
B. Bin Control
C. Mechanics

Dr. Green's physical structure is a large recycling bin with two smaller inner bins, one for recycling and one for trash. On top of the large bin is a lid frame consisting of the user alerts as well as a swing door platform that the user can place their waste on. The door will rotate towards the respective bin that the waste on the platform will need to fall into. Connected to the lid is a vertical overhead frame to hold up a camera and processor for waste identification.

On a high level, the Computer Vision System will capture, detect, and classify a given piece of waste placed on the platform. This information will then be sent through a wired connection to the hardware controlling unit, which will set off alerts and move the platform according to the CV system output. While we did not have time to develop past our MVP design, ideally the device would also be set up to receive notifications when one of the inner bins are full. A block diagram of the system architecture can be found in Figure 11 of the Appendix.

3.1 Detection & Classification System

In order to capture the waste items, we will be using a camera module placed directly above and facing the center of the platform door. Using a USB wire, this camera will be connected to a Jetson Xavier NX processor. On a user's command, triggered by a key press, the camera will capture an image of the current platform. The Jetson will then perform detection and classification on that image. For easier implementation, this has changed from our previous design which consisted of automatic detection of a change in the platform that would trigger the detection and classification process.

3.2 Jetson to Bin Control System

After running detection and classification, Jetson will send classification (trash vs. recyclables) to the main Arduino controller which coordinates the alert system and bin mechanics. Python's Serial module is used to deliver the binary classification result via a USB cable.

3.3 Bin Control System

Currently, this system consists of an Arduino controller which will receive classification results from Jetson. The result will be a binary value indicating if the items are all recyclables or not. Depending on that, the Arduino will control two main categories of components: Alerts and Platform Control. This has changed from our previous design that included an additional category for Reminders that would have been implemented if we made it past MVP.

3.3.1 Alerts

These components are to notify the user of whether the item they are attempting to recycle is truly recyclable or not through audible and visual cues. A neopixel strip is connected to the lid frame for the visual cue and will flash either green or red based on whether the user is correctly recycling or not. For the audible cue, a piezo speaker is also be placed on the lid frame and will either play a jingle or a buzz based on the item attempted. While not currently implemented, past MVP these components could potentially use other colors or sounds to indicate other situations such as fill level, mixed materials simultaneously present on the platform, and a lower level of confidence by the device.

3.3.2 Platform

This category is for controlling the swing platform door so that the waste is self-organized into the correct bin. There is one main servo connected directly to the door in order to control its angle of rotation towards a certain bin based on whether the given object is recyclable or not. On each side perpendicular to the main servo, there is a servo acting as a lock to the door to provide support and prevent any forceful turning of the door either by the unexpected weight of the object or the user themselves.

3.4 Mechanics

To put all these parts together, a few mechanical design components are needed. For the CV System to operate as intended, the camera and the Jetson needed to be mounted on an overhead frame. This frame needs to be tall enough that the camera can capture the entirety of the platform. The overhead structure is connected to the wooden bin frame for support, on which the alert components are mounted. An acrylic lid frame lies on top of this wooden frame to provide a clean cover to the bins below. The swing door platform, now made of plywood rather than the previous design of acrylic for its lightness and solid color, is at the center of the lid frame. The door platform has an axle rigidly attached to its bottom side, which has one side connected to the main servo to control the door's movement, and through a hole on the bin frame for support. Two support servos act as locks under each side of the bin door. An Arduino is on the inside of the bin, connected to the alerts, door controllers, and Jetson.

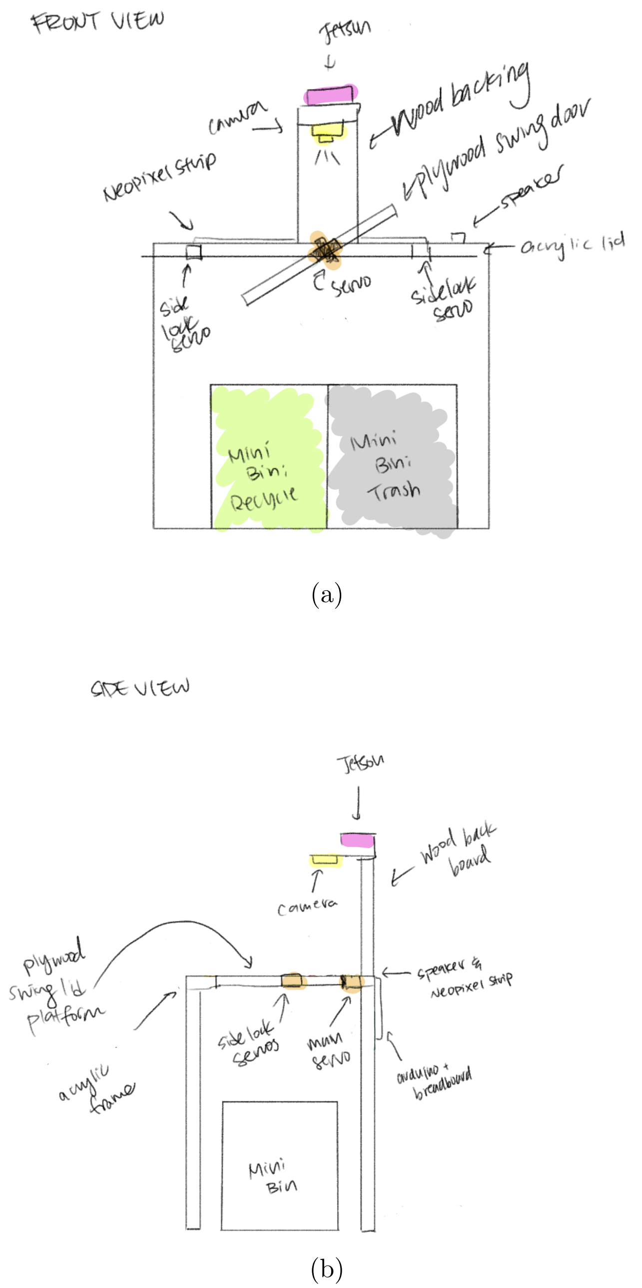


Figure 2: System description. (a) Front view of entire system (b) Side view of entire system

4 DESIGN REQUIREMENTS

Two of the most important requirements for our system are accuracy and speed. These requirements apply to the different subsystems within Dr. Green.

1. Automatic camera capture & detection

The system will not need any manual start. New items should be detected automatically. A camera, connected to Jetson, constantly captures images with a 0.3s interval in between and OpenCV will be used to continuously compare consecutive images using MSE (mean squared errors).

If a certain threshold of MSE is exceeded, it should detect a change in the image. The goal of the detection algorithm is to recognize all significant changes on the platform, for example new objects placed or removed. The rate of successful detection is expected to be 100%.

In case of two consecutive changes, which refers to new items placed and users' hands taken away, it should trigger the YOLO classification. The wait time since items are placed and users' hands removed until the start of YOLO classification should be less than 1 second.

2. Accurate YOLO classification

The YOLO algorithm detects and classifies multiple items in an image at once and outputs a single binary classification result. If any item in an image is not recyclable, the correct output is non-recyclable; if all items are recyclable, the correct output is recyclable. This result determines what alerts the users receive.

For each item detected and classified, YOLO will also generate a confidence value of the result. To prevent false positives (trash mistaken as recyclable), the CV system should consider all classified recyclables with confidence value lower than 0.85 as trash. We want to balance between contamination prevention as well as recycling efficiency; 0.85 is chosen as a reasonable threshold. Experiments over large datasets will be conducted later to test if it is indeed a good balance; the exact value will be subject to adjustment.

Overall, the goal of YOLO classification is to reach an accuracy of at least 90%. The runtime of the YOLO algorithm on any image should be less than 2 seconds.

3. Visual & sound alerts

A bar of neopixels and a microspeaker will be installed as a part of Dr. Green to provide feedback. The alerts expected for different classification results are different. If all items are correctly classified as recyclables, the neopixels will light up as green and the microspeaker will make a jingle sound. If any items are not recyclables, the neopixels will light up as red and the microspeaker will make a buzzer sound instead. The accuracy of alerts perceived, depending on the classification result produced (not necessarily the actual classification), should be 100% accurate. The alert should be set off within 1 second of the classification result, and will last for a duration of 3 seconds and be reset afterward.

4. Self-organized recycling

Alerts and swinging will happen simultaneously, both within 1 second since the classification result was computed. Depending on the classification result, items should be placed into the correct bin with 100% precision and the operation time should be less than 1 second. To prevent contamination, if any item is not classified as recyclable or if not reaching the confidence threshold, all items will be thrown into trash. Simultaneously as the alert goes off, the platform holding items should either swing to the trash bin or recycling bin. The servo will naturally adjust the platform back to its neutral state and be ready for the next use.

5. Total operation time

As explained in each section above, the combined operation time since users removed their hands from the platform to the start of alerts and recycling should be limited to under 5 seconds.

6. User safety & education

Using the two-phase detection algorithm, the product makes sure the swinging door will not accidentally hurt users' hands. The self-organized recycling mechanism avoids users' physical contact with the device for the purpose of sanitation. Education is a core goal; all alerts are chosen with the consideration of not raising mental shaming of users, especially as users would mainly be children and teenagers in schools.

5 DESIGN TRADE STUDIES

5.1 Subsystem A: CV System

5.1.1 Parallel Detection & Classificication

To avoid booting the YOLO model every time (12 second) for a platform change, we considered running detection and classification in parallel and reading classification results when a change is detected. However, because the same camera stream cannot be accessed in both programs simultaneously, we have to use a virtual camera stream as the source parameter for YOLO [6]. Although we were able to implement that, there is a 16 second delay in image update with the virtual camera stream. As a result, the classification result outputted is significant behind in time compared to the actual platform at real time. With our current "keypress" approach, detection & classification are only run once per user input and YOLO's input source is a still image, which is more accurate and energy efficient compared to the camera stream. Therefore, we have decided that the "keypress" approach is better overall in time (12s vs. 16s), accuracy, and sustainability.

5.1.2 YOLOv5

We chose YOLOv5 due to the fact that it is one of the best real time object detection and processing models out there, as well as the fact that it is implemented in PyTorch, which makes it easier to integrate. It helps us fulfill the use-case requirements about accuracy.

5.1.3 Trashnet Dataset

We chose Trashnet dataset[10], which includes various different types of recycling as well as a trash class. Ultimately we only want two classes, recyclable and non-recyclable. But since there's such a large variety of different objects that could be part of the waste class, we decided to more accurately train the recyclables would be better. If there are more classes of recyclables, then the features of those waste objects would be learned more precisely. We could pair this with the CV, so that if the ML model says no detection, but the CV says an object is there, we would classify it as trash. We also had to change the dataset according to Pittsburgh recycling rules, taking out the plastics section and replacing it with the bottles from the original dataset, since nothing in Trashnet plastics can be recycled other than the bottles. We had also tried TACO [11] dataset, which finely classified trash in to 57 different classes. We found that the model didn't work, since it's hard for the model to learn such big differences.

5.1.4 Jetson Xavier NX

For the processing unit, we were considering three main options, the Raspberry Pi, Jetson Nano, and the Jetson Xavier NX. Between the Raspberry Pi and the Jetson series, we chose Jetson because it has a GPU(graphics processing unit) with far more processing power and libraries that makes it more suitable for AI/ML applications compared to just the CPU in the RPI. In our case, since the CV subsystem deals intensively with images, a capable GPU is necessary.

We acknowledge Raspberry Pi is more affordable and did consider using it and running the YOLO algorithm on AWS EC2. However, the combined cost actually may not be lower and the added implementation complexity motivated us to choose Jetson and run YOLO locally on the device. Moreover, if running YOLO on a cloud server like EC2, network latency for Arduino to receive classification results would also be negative for the total operation time. With Jetson, classification results would just be transported using the USB cable from Jetson to Arduino and it leads to almost no delay.

When looking within the Jetson series, while the Nano is cheaper, smaller, and more lightweight, its processing speed and power as well as memory space is far less than the Xavier NX. Due to our project needing to deal with huge amounts of data that requires a good amount of memory and speed as well as the fact that both Jetsons are available in the inventory, we decided the Xavier would be the best choice for this project.

5.2 Subsystem B: Bin Control

5.2.1 Neopixel

We decided to go with a strip of pixels rather than a singular one to make the alert more apparent. By having a whole row of lights glow, it will catch the attention of the user more easily.

5.2.2 Speaker

We decided to include an audible cue rather than just visual to add learning reinforcement and be more inclusive in bin use. We chose a piezo speaker as it would not only be easy to integrate with our Arduino, but give us the ability to control the pitches being emitted. This would allow us to further reinforce the user's actions and provide feedback through "positive" and "negative" tones.

5.2.3 Arduino

In order to control our hardware and mechanical components, we needed some sort of microcontroller. While there are many possible options, we decided to go with an Arduino as it is available in the inventory, is quick to set up, can be easily simulated, and works well with multiple components such as the servo for the swing door, the neopixels, and the piezo speaker which don't need any complex libraries or wifi connectivity mechanisms that other microcontrollers such as ESP32 consist of.

5.2.4 Servo

In regards to our trap door control, we decided to go with a servo as opposed to our initial design with an actuator. This change was made alongside our change to the trapdoor itself. Since we changed our door movement from a linear open-close to more of a left-right swing, a servo would have more control due to its angular rotational motion as opposed to the linear movement of an actuator.

5.3 Subsystem C: Mechanics

5.3.1 Swing Door

We have a multi stream swing door for the bin. With a swing door, we can have two compartments instead of one, essentially supporting recycling and trash disposal. This set up essentially gives the system the ability to self organize the waste, leaving the user only having to receive the provided feedback. This makes the time needed for user engagement a lot shorter and therefore more likely that people will use the device. This change also better fulfills the hands-free sorting and cleanliness aspect of the use case requirements. Instead of the user having to pick up their mis-recycled garbage to throw in the trash bin, the device will take care of it by tilting the platform to slide the trash into the correct bin.

5.3.2 Door Platform Material

We deeply considered the tradeoffs between a wooden versus acrylic swing door platform. Our current implementation uses plywood. Previously, we had designed our device to use an acrylic lid due to its durability, thickness, easiness to clean, and uniformity in appearance with the lid frame material. However, when we tried implementing this, we found that it was far heavier than we had previously expected, resulting in opposing torque when trying to recenter it. Additionally, the acrylic material is translucent, which while good for demo purposes did not provide an adequate background for the new CV system. On the other hand, the plywood is light enough for the servo to control turns accurately, and while potentially less durable, is still quite strong and a lot smoother than expected, indicating that cleanliness should be less of a problem than we thought previously.

5.3.3 Frame Material

We decided to use wood for the overhead and bin frame, and acrylic for the lid frame. This is because while acrylic is sturdier and has a cleaner finish, it is expensive and harder to work with, and can only be cut with a laser cutter. On the other hand, wood is cheaper, easier to work with and can be cut and drilled through with hand tools. Due to the fact that the bin and overhead frames are not directly facing user interaction and require significant construction to put the parts together, we went with wood for these parts. On the other hand, we used acrylic for the lid frame as for one, the length and width of the plywood pieces readily available at TechSpark were smaller than the size of our main bin, while the acrylic that we had ordered prior to thinking about wood fit the size requirement. Additionally, the acrylic is thick and sturdy yet smooth, making it durable and easily cleanable. Additionally, having a see-through acrylic frame would be more ideal to show how our device operates during a demo.

5.3.4 Axle Clamp to Platform Door

While we initially had chosen to order 0.75 inch clamps to connect our 0.25 inch axle with the door in order to provide enough padding for the door and lid frame to align given the servo position, due to the heaviness and smoothness of the acrylic combined with these clamps along with internal padding, the door would slip out of the initially stabilized connection when trying to return to center. However, while smaller clamps of 0.25 inches do not provide enough padding for alignment and would require an additional piece of material to act as a height buffer, they are more rigid as they fit the axle size. When we implemented this with our new plywood door, our system was a lot more stable and durable, resulting in us going with this as our final setup.

5.3.5 Lid Frame Design

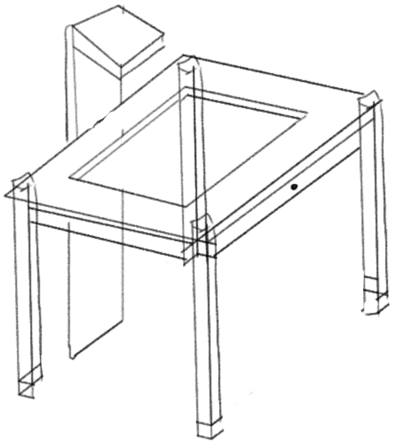


Figure 3: Final Implemented Frame Design

Figure 2 above is our latest lid frame design. It is a wooden frame structure that will encase the bin. While previously our plan was to only build the wooden bin frame if time permitted and have the lid directly lie on the bin, we realized that we would need to implement most of the pieces of the frame anyways to hold up the servos, axle, door, and hardware circuit. Additionally, the legs are needed in order to hold all the parts up above the bin so that there there is enough clearance for the objects to fall inside.

6 SYSTEM IMPLEMENTATION

6.1 Subsystem A: CV System

Our implementation plan for this subsystem involves buying a camera and Jetson, downloading existing datasets, and modifying the YOLOv5 model to fine tune it to our project.

A USB camera will be connected to the Jetson with its usb camera. The camera will be fixed at a certain angle on top of the lid and will capture an image that covers the whole device's platform continuously. As figure 4 explains, the detection algorithm is as follows. When the system was just booted, the camera will capture the background on user's command. Following that, whenever users have finished placing items and are ready for classification to begin, they will press a key to capture another image of the platform. Then Jetson will run CV detection which compares current image with the background; if a change is detected, YOLO classification will be run. In practice, the keypress will be replaced with some foot pedal that maintains hands-free for the sake of sanitation.

After running YOLO classification, the Jetson will review the labels of all identified items. We consider all items identified to be recyclables but with confidence values lower than 0.85 to be false positives. If all items are labeled as recyclables and all with confidence values equal to or higher than 0.85, the Jetson will return positive which signals all items are correctly classified recyclables. Otherwise, it will return negative. In cases when items are detected by CV but not YOLO, we will consider them as trash.

The final result of the YOLO classification model will be binary and the type of the return value will be a boolean. Using Python's serial module, the Jetson will serialize the result and send it over a USB cable to the Arduino. The Arduino will then use its Serial module to read incoming data from Jetson and coordinate the bin control system.

Instead of constantly running CV detection and YOLO classification, we chose to trigger those only based on user inputs. The major advantage of this approach is to save energy, which is supported with GPU utilization experiments. Please refer to Design Trade Studies section for more discussion on this design decision.

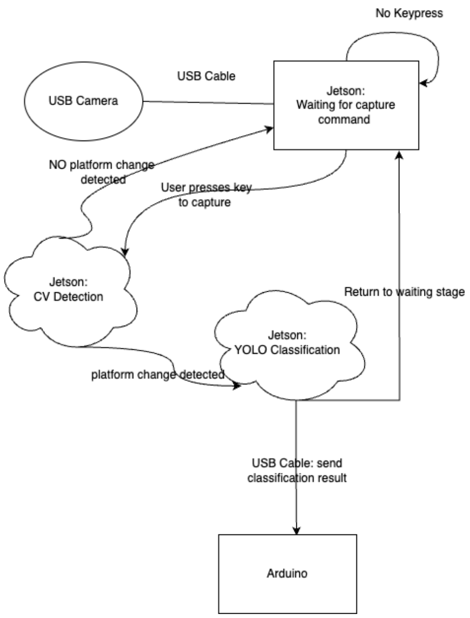


Figure 4: FSM for Detection and Classification

6.2 Subsystem B: Bin Control System

This subsystem initially consisted of an Arduino Uno connected to a Neopixel strip for the visual cue, a Piezo Speaker for the audible cue, a main servo to control the swing door, two side servos to lock the door in place, and two ultrasonic sensors to check the mini bin fill levels (past MVP). For our final implementatation we implemented everything except for the ultrasonic sensors that were meant for past MVP. We bought the Arduino, piezo, servos, ultrasonic distance sensors, neopixels, and USB cable, and self assembled and programmed the circuit. Before physical assembly, a circuit schematic was made and simulated using Tinkercad. Once all the parts had arrived and building had started, we found that certain components did not match the simulated ones, resulting in some reworking being required of the overall circuit and the libraries used. The most significant difference was that the neopixel strip was a "Dotstar" and need two instead of one control pins from the Arduino, one for clock and another for data [7].

The specifics regarding connections are described in schematic figures 8 and 9.

The Neopixel strip is connected to a power source, ground, and Digital Pins 11 and 13 on the Arduino, the SPI pins for CLK and Data input to the strip. To control the output of the neopixel strip, we used the Adafruit Neopixel library. The specific colors of the neopixels is specified using RGB values.

The Piezo Speaker is connected to ground and PWM Digital Output Pin 3 on the Arduino. PWM is needed for this implementation, as it will allow us to imitate Analog signals with the digital output, giving us control of the pitch [8].

All of the Servo motors will need a connection to power, ground, and a PWM digital output pin to control the variance of the angles they will need to turn. The main servo will be connected to pin 10, and the side servos will be connected to pins 9 and 6. The main servo's output shaft is connected directly to the swing door, with each side of the control horn screwed parallel to the sides of the door. The Servo motors will be controlled using the Arduino Servo library. Since servos have a maximum rotation angle of 180°, The center angle will be set to 90°. If the servo needs to turn left, the angle will be increased towards 180°, and towards 0° if it is to turn right. The side servos will be placed under either side of the door, with the control horns perpendicular to the door. If the side needs to be unlocked, the servo will turn 90° to essentially stop blocking the door from turning in the given direction.

Finally, the ultrasonic sensors, which were planned for past MVP, were not implemented, but would ideally need a connection to power, ground, and PWM Digital Pins. They would send and receive a signal to calculate the relative distance of objects to it. If the sensed distance of waste to the top of the bin is less than an inch, it will alert the user using the alert components. If further pins are needed based on the type of ultrasonic sensor used, we would need to additionally invest in an Arduino pin extender to account for this addition.

After these connections were made, the Arduino IDE and C++ based library were used to program the controls for each of the components on the Arduino. The program flow is described in figure 12.

Initially, the Arduino receives a binary output from the Jetson through a USB connection, 1 if recyclable, and 0 if non-recyclable. While not implemented, a potential future step would be to extend this output rather than using a singular binary value to simulate other outputs, such as mixed materials. Once this output is received, the neopixel is set either to green for "correct" and red for "incorrectly recycled"." The piezo output is set to an ascending jingle tone for "correct" or a low buzz for otherwise. The side of the bin that matches the predicted value, left for recyclable and right for not, will be unlocked. After these series of events, the platform controlling servo will turn towards the bin matching the predicted value so that the waste can fall into the bin. After testing, our angle has been set to +/- 50° from center to point the waste to fall towards the center of each of the mini bins. After the waste falls, the platform controlling servo re-centers itself. The platform locks then turn back to their original positions, and the neopixel and speakers turn off, resetting the entire system for the next disposal cycle.

An issue we have been having with this system is that after classification when the output is written to the serial input of the Arduino from the Jetson, after the first iteration the port connection resets, often resulting in it switching from "/dev/ttyACM0" to "/dev/ttyACM1" or vice versa. This results in the whole system crashing as the designated port written in the code is not available. After searching multiple sources, it seems that this is a known issue when that people have not been able to solve. While multiple solutions were attempted, such as purging ModemManager, resetting the Arduino, selecting the port with the IDE, testing different cables, etc. [9], this issue still seems to persist. However, since this issue doesn't always show up, we are able to get a good amount of working trials during testing.

6.3 Subsystem C: Mechanics

This subsystem is the implementation of the overall device structure. Our current design, as seen in figures 1 and 2, consists of a large, main recycling bin with dimensions 20.5" x 15" x 21". Inside this main bin are two mini bins, one for trash and one for recycling with dimensions 8.25" x 11.75" x 11.5". On top of the main bin will be an acrylic lid frame of dimensions 24" x 18" x 0.375". Under this is the bin frame that is made out of wood with supporting legs on each corner of the lid frame with dimensions 1.5"x 1.5" x 21.5". This allows for enough leeway for the main bin to be moved in and out of the lid frame structure for emptying it out. The lid frame will have the neopixel strip and speaker mounted on top, with the circuit and Arduino mounted to the back of the device frame. The servo is mounted on one side of the wooden table-like frame, and there is a hole on the opposing side for the axle held up by the servo to fit through that is lose enough for smooth rotation while providing support. In the middle of this wooden lid frame will be a cutout for the plywood platform door of dimensions 9.5" x 14.5" with 0.25" space in between the door and lid frame for free movement. To attach the axle to the door, two c 0.25 inch clamps are used, with 0.5 inch pieces as padding to align the door with the lid frame. The platform tilts at a 50 degree angle in both directions in order for the waste to fall into the smaller bins. For this to work, we had to make sure the it was not too long and the main bin was tall enough that the platform clears the opening of the smaller bins while turning. This is reflected in the fact that half the platform size, 7.25", is less than the the difference between the height of the main and mini bins, 9.5".

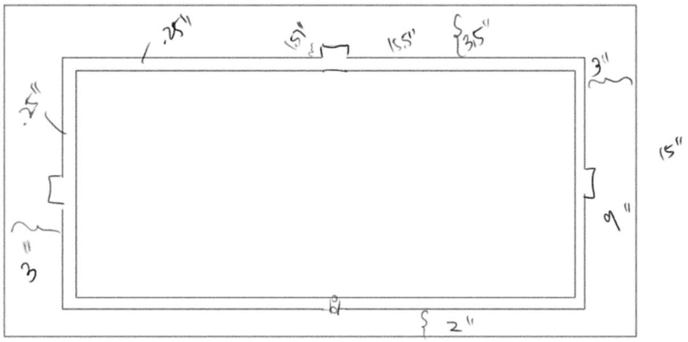


Figure 5: Bottom inside view of lid with dimensions

7 TEST & VALIDATION

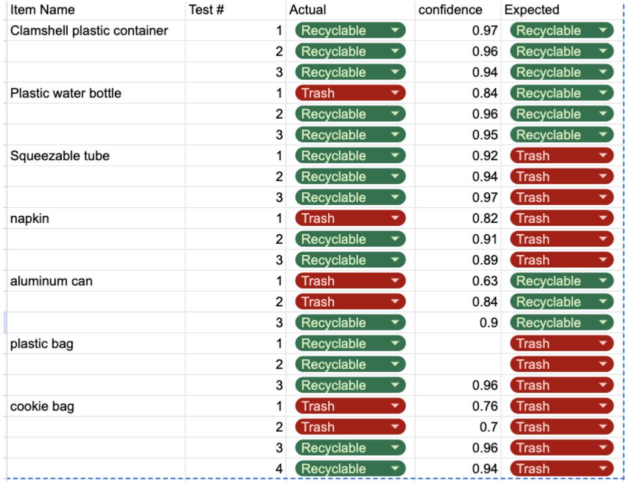
Our testing plan was to initially test each component of our device, then test their integration step by step. Since our main quantitative requirements are speed and accuracy, we made sure to test these at each step. Refer to figure 9 below for a summary of our test results.

7.1 Capture and Detection

We have tested our image capture set up so that it encapsulates every side and corner of the platform, but also doesn't include anything past that to avoid unexpected noise in the image. The timing was also tested, although the camera is quite laggy, it could take 2 seconds for the frames to update with the new image, the classification is immediate after the object appears on screen.

7.2 Model

We were only able to achieve around 70% accuracy for our model, with tests on plastic containers, plastic water bottles and aluminum cans which are recyclable, and a squeezable tube, napkin, and cookie snack bag which are non-recyclable. This is underperforming of our targeted 90%, but we have went through multiple dataset changes, as well as trying different ways to train. The classification time though is immediate.



Item Name	Test #	Actual	confidence	Expected
Clamshell plastic container	1	Recyclable	0.97	Recyclable
	2	Recyclable	0.96	Recyclable
	3	Recyclable	0.94	Recyclable
Plastic water bottle	1	Trash	0.84	Recyclable
	2	Recyclable	0.96	Recyclable
	3	Recyclable	0.95	Recyclable
Squeezable tube	1	Recyclable	0.92	Trash
	2	Recyclable	0.94	Trash
	3	Recyclable	0.97	Trash
napkin	1	Trash	0.82	Trash
	2	Recyclable	0.91	Trash
	3	Recyclable	0.89	Trash
aluminum can	1	Trash	0.63	Recyclable
	2	Trash	0.84	Recyclable
	3	Recyclable	0.9	Recyclable
plastic bag	1	Recyclable		Trash
	2	Recyclable		Trash
	3	Recyclable	0.96	Trash
cookie bag	1	Trash	0.76	Trash
	2	Trash	0.7	Trash
	3	Recyclable	0.96	Trash
	4	Recyclable	0.94	Trash

Figure 6: Model Testing

7.3 Visual/Sound Cues

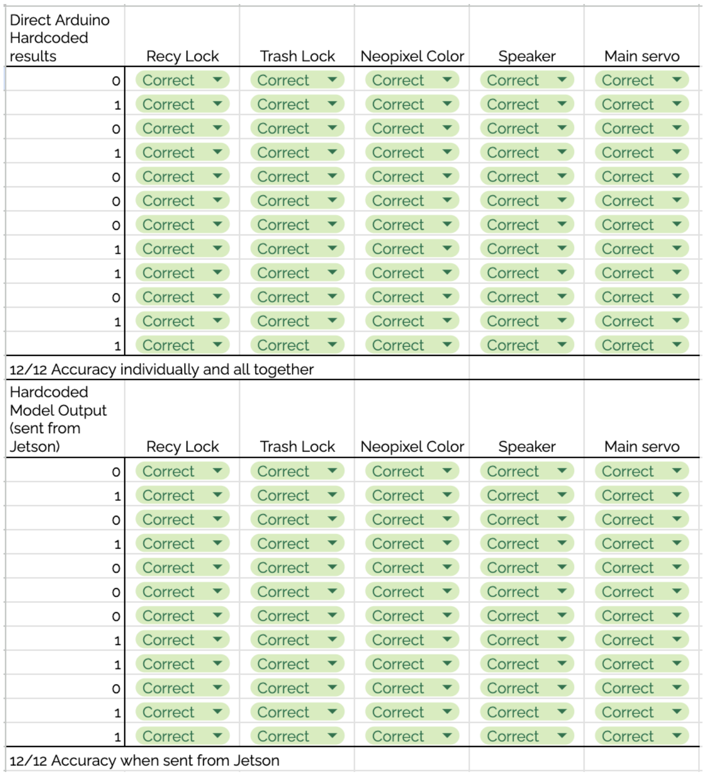
We have achieved 100% accuracy for the operation of the alert component, with all 24 out of 24 initial tests resulting in the correct output, reaching our target. 12 tests were conducted testing the individual components with a hard-coded input in the Arduino code. This was followed by 12 more tests, this time sending the values from the Jetson to be read by the Arduino. The results of the tests can be seen below in figure 7. Given the correct input of 0 for trash and 1 for recyclables, the output color of the neopixel and and sound of the speaker is always correct, with red and buzz for trash and green and jingle for recycling. The outputs also reset after each round. We also test the time for the alert systems, which takes less than 3 seconds to trigger and operate, as targeted.

7.4 Bin Operation

We tested if differently shaped and weighted objects will successfully fall into the smaller bins. We used commonly seen waste objects, such as plastic water bottles, snack wrappers, cans, etc. Given a correct input of 1 for recyclable and 0 for trash, the main servo turns towards the recycling bin (left) when the input is 1 and towards the trash bin (right) if the input is 0. The door turns the correct amount (about 50°) in the correct direction, and the locks block and unblock the correct side accordingly, all actions in the correct order, and then resetting at the end. Accuracy results were 100% correct, as seen in figure 7. The overall operation takes about 6 seconds on average to operate alongside the alert system as seen in figure 8.

7.5 Integration

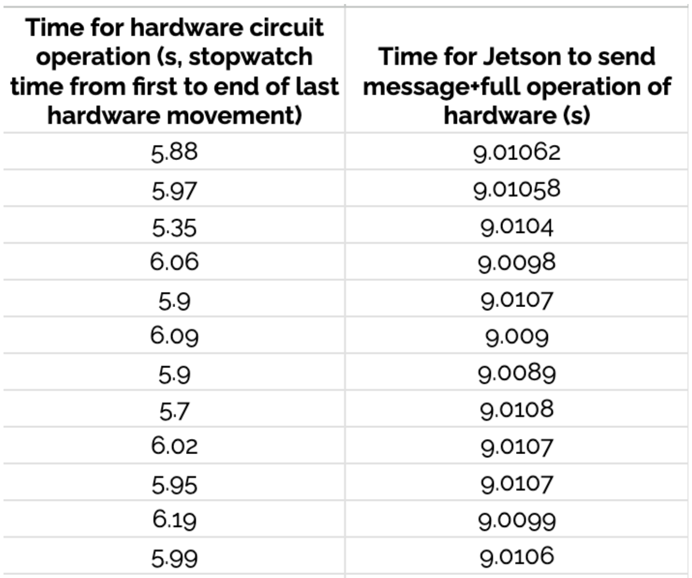
We were able to test integration of the hardware and ML early, and found that the Jetson can send a serial message to the Arduino and set off the hardware cues as shown in figure 7. The time of this operation averaged around 9 seconds as seen in figure 8. During integration, we found that the bootup of the YOLO model takes 12 seconds, which slows down the whole process.



Direct Arduino Hardcoded results	Recy Lock	Trash Lock	Neopixel Color	Speaker	Main servo
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
12/12 Accuracy individually and all together					

Hardcoded Model Output (sent from Jetson)	Recy Lock	Trash Lock	Neopixel Color	Speaker	Main servo
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
0	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
1	Correct	Correct	Correct	Correct	Correct
12/12 Accuracy when sent from Jetson					

Figure 7: Hardware Circuit Testing Accuracy Results

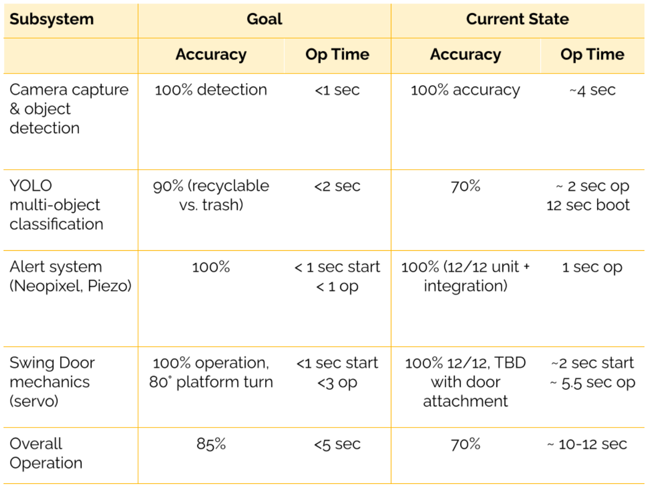


Time for hardware circuit operation (s, stopwatch time from first to end of last hardware movement)	Time for Jetson to send message+full operation of hardware (s)
5.88	9.01062
5.97	9.01058
5.35	9.0104
6.06	9.0098
5.9	9.0107
6.09	9.009
5.9	9.0089
5.7	9.0108
6.02	9.0107
5.95	9.0107
6.19	9.0099
5.99	9.0106

Figure 8: Hardware Circuit Testing Time Results

7.6 Overall Experience

We also tested the overall experience of our device. We judged on whether the device was easy to use, safe, educational, engaging, and did its job properly. As of the moment we still have a significant lag time that could work against the engagement of the user, but the mechanics and hardware of the experience work well together. The swing of the platform is clean, and the lights and sounds react well.



Subsystem	Goal		Current State	
	Accuracy	Op Time	Accuracy	Op Time
Camera capture & object detection	100% detection	<1 sec	100% accuracy	~4 sec
YOLO multi-object classification	90% (recyclable vs. trash)	<2 sec	70%	~ 2 sec op 12 sec boot
Alert system (Neopixel, Piezo)	100%	< 1 sec start < 1 op	100% (12/12 unit + integration)	1 sec op
Swing Door mechanics (servo)	100% operation, 80˚ platform turn	<1 sec start <3 op	100% 12/12, TBD with door attachment	~2 sec start ~ 5.5 sec op
Overall Operation	85%	<5 sec	70%	~ 10-12 sec

Figure 9: Testing summary

8 PROJECT MANAGEMENT

8.1 Schedule

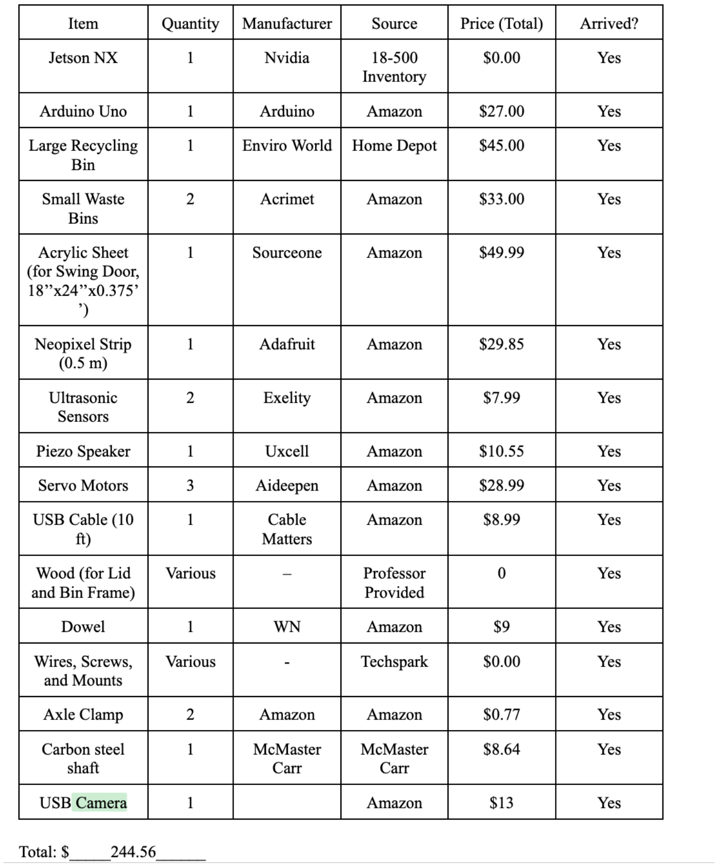
In general, a large portion of our tasks ended up taking longer than expected. Thankfully, we had planned some slack in our schedule and started integrating early on so that even if certain parts of the device were not fully developed, they still somewhat worked together. Specifically, our schedule was revised in terms of the ML model training due to switching of datasets past the MVP. This portion also caused a lot of issues as datasets that we had previously found were either too small, uneven, or not following the rules of Pittsburgh recycling, requiring last minute changes when issues were discovered during testing. Mechanics were also pushed back since the initially ordered parts didn't work as well as we expected, resulting in a wait for new ones and time for re-assembly. The updated schedule is shown in Fig. 15 in Appendix.

8.2 Team Member Responsibilities

Aichen is in charge of writing scripts for organizing the dataset file structure and for image comparison and helping with debugging the ML code. Vasudha is in charge of simulating hardware components, programming the Arduino, researching and designing the mechanics, dimensions, and parts for the physical device. Ting is in charge of setting up and training the model to detect waste, finding the datasets to train the model on, and helping with designing the physical bin.

Aichen and Vasudha set up the Jetson and its connection to the Arduino. Aichen set up camera capture and deployed the detection algorithm. Vasudha implemented the hardware circuit, including controlling the alert system and servos using the Arduino, and worked on the connection of these parts with the mechanics. Ting continued experimenting with different datasets and training the model, as well as deployed it to the Jetson. The whole team worked on the mechanics of the device.

8.3 Bill of Materials and Budget



Item	Quantity	Manufacturer	Source	Price (Total)	Arrived?
Jetson NX	1	Nvidia	18-500 Inventory	$0.00	Yes
Arduino Uno	1	Arduino	Amazon	$27.00	Yes
Large Recycling Bin	1	Enviro World	Home Depot	$45.00	Yes
Small Waste Bins	2	Acrimet	Amazon	$33.00	Yes
Acrylic Sheet (for Swing Door, 18''x24''x0.375'')	1	Sourceone	Amazon	$49.99	Yes
Neopixel Strip (0.5 m)	1	Adafruit	Amazon	$29.85	Yes
Ultrasonic Sensors	2	Exelity	Amazon	$7.99	Yes
Piezo Speaker	1	Uxcell	Amazon	$10.55	Yes
Servo Motors	3	Aideepen	Amazon	$28.99	Yes
USB Cable (10 ft)	1	Cable Matters	Amazon	$8.99	Yes
Wood (for Lid and Bin Frame)	Various	–	Professor Provided	0	Yes
Dowel	1	WN	Amazon	$9	Yes
Wires, Screws, and Mounts	Various	-	Techspark	$0.00	Yes
Axle Clamp	2	Amazon	Amazon	$0.77	Yes
Carbon steel shaft	1	McMaster Carr	McMaster Carr	$8.64	Yes
USB Camera	1		Amazon	$13	Yes

Total: $_____244.56______

Figure 10: List of parts: everything used except ultrasonic distance sensor

8.4 Risk Management

8.4.1 Camera

As we started classifying real-time captured images, it has been found that the CSI camera, supposedly compatible with Jetson, is not reliable in connectivity. Jetson struggled to identify it as a connected external device; even if connected, video and image capturing was unreliable. After problem isolation, it has been found that the camera could not work with Jetson Xavier or Jetson Nano and it was necessary to change to a camera that connects differently. Since there are still available USB ports and most webcams are connected with USB, we have decided to switch to USB cameras. Although USB cameras also have weaknesses in the hardware aspect, for example, vulnerable wires and more complicated setup (an extra wire instead of the existing CSI connector), its overall performance is much better than the CSI camera. Therefore, we have decided to use a USB camera instead.

The only component that the camera interacts with is the Jetson that runs CV detection and YOLO classification. For the whole system, changing of camera model has almost no effects, especially for the software.

8.4.2 Mechanics

No one on the team has had much experience with mechanical construction, which is a critical risk for us since the construction of the lid structure needs to be able to physically sound while able to swing and also hold up a weight. This requires both woodworking and laser cutting of the acrylic. To manage this risk, we asked the Techspark staff for help with cutting the larger pieces of wood and learned how to use other tools for the rest of the construction. We managed the risk of the platform not performing as expected by remaining flexible in the material that we were using. We swapped out acrylic for plywood quickly when we realized that the servo was not strong enough to turn it.

8.4.3 YOLO Classification

On the software side, we had the risk of the ML algorithm not being accurate enough in classifying. We can mitigate this by using the GPU in the ECE machines to train for many more epochs. We also attempted to train on better, larger, and self merged datasets to get results closer to what we wanted. Additionally, another risk is the ML algorithm operating too slowly, or there not being enough memory space in the Jetson for fast operation. While we didn't get to this, we could potentially mitigate this by optimizing our algorithm and removing unnecessary/unused data from memory to free up space, and already started doing so in the initial implementation deleting unused result images.

9 ETHICAL ISSUES

Dr. Green seeks to solve the problem of contamination in recycling bins in our community as well as rectifying common misunderstandings of recycling rules. The ideal customer for Dr. Green would be a school filled with students who are open to learning about recycling rules and will carry on these learnings to other communities they join as they grow older. The students in the school will be the most vulnerable to failure. If Dr. Green fails to correctly classify, and therefore teaches incorrect recycling practices, then these children will learn false information that they will carry on and may never get corrected. If the product is misused, it could be dangerous as the children could get hurt if the bin malfunctions due to mishandling.

Our decision to have a swinging platform instead of a drop-in hole or a chute is due to our use case of educating about recycling rules. If a child sees a swinging platform swing to one side or another based on an item that they placed, it creates a memory that is more real than if the item just dropped into the void and the sorting was done out of sight. This may create some issue of public health and safety, if some traces of the waste is left on the platform, it could cause contamination. This is the tradeoff between education and public health.

There could also be negative psychological effects if a child hears a buzzer sound and red lights in reaction to something they did. Our original idea was that when they place a piece of trash on, there would be a screen that flashes a big red X telling them that they recycled incorrectly. But this would be bad for public mental health, so we decided to go with neopixels, since that would be a less extreme option. But there needs to be some way to tell the user that what they're trying to recycle belongs in the trash, so this is another trade off between education and public mental health. The product Dr. Green is made to promote public welfare, in the sense that helping the public learn about recycling will help the Earth that we all live on even if it's just a little bit.

10 RELATED WORK

One project that is closely related to ours is Oscar, the world's first AI trash can. It has the same idea as our project, sorting waste that the user places into the device recyclable and non-recyclable trash, and having a big bin that contains two little bins. Instead of using a swinging platform, they use a conveyer belt to transport the waste to the left or right bin. Oscar also uses a camera that captures images to feed through a neural network in order to classify the object. They use Inception Network instead of YOLO. One special feature that Oscar has is buttons to teach him about what is recyclable or not when the system is confused. We do not have this feature, since we want Dr. Green to be educating the students and not the other way around. The ML classification should be accurate enough that there should be no confusions, and the students are able to learn about recycling rules instead of propagating their misconceptions. This is also why our system has an alert system, with lights and sounds.

11 SUMMARY

This past semester, we were able to make significant progress, going through the design and development process to create Dr. Green, an educational waste bin for schools and students that will self sort the user's waste to reduce recycling contamination while teaching them recycling rules. While we were not able to meet all of our design specifications entirely, some examples being classification accuracy due to lack of an ideal dataset, automatic operation of the CV system, and overall operation speed due to back up plans being put in place to prioritize functionality. With more time, we could better train and test our model to work more accurately and set up automation so that the device can start up and operate on its own on boot, especially since we had implementations that were close but unable to reach the intended operation in time. Ideally with the best version of our device, our stakeholders and users, schools and students, will be willing to use and learn the correct recycling rules from Dr. Green, eventually carrying this on to other places that do not yet have Dr. Green to reduce contamination overall.

11.1 Future work

11.1.1 Ultrasonic Distance Sensor

One of the features we wanted to add to our system after MVP was sensing the bin fill level. After some consideration, we decided to go with an ultrasonic distance sensor as opposed to other sensors such as a weight sensor, since it would be difficult to set a threshold as the weight of different trash combinations can result in different total weights. With the ultrasonic proximity sensor, we can place the sensor at the top of the bin and simply measure the distance between the waste inside and the top.

11.2 Lessons Learned

We learned a lot about the engineering design process and more general development related lessons such as how implementation can very often turn out to be different than the imagined and planning ahead with slack time is crucial as things always take longer than expected. In particular, integration of hardware and software are prone to problems that are difficult to reproduce and locate the root cause. Overall, being patient and flexible throughout the whole development process is crucial.

Glossary of Acronyms

- CV - Computer Vision
- FPS - frames per second, measurement for GPU capability
- ML - Machine Learning
- PWM - Pulse Width Modulation
- YOLO - You Only Look Once

References

[1] Arie, L. G. The practical guide for Object Detection with YOLOv5 algorithm. towardsdatascience (2022). Available:https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843.

[2] Solawetz, J. Deploy YOLOv5 to Jetson Xavier NX at 30FPS. roboflow (2020). Available: https://blog.roboflow.com/deploy-yolov5-to-jetson-nx/.

[3] Cho, R. Recycling in the U.S. Is Broken. How Do We Fix It? new.climate.columbia.edu (2020). Available at: https://news.climate.columbia.edu/2020/03/13/fix-recycling-america/.

[4] Jetson Benchmarks. developers.nvidia Available at: https://developer.nvidia.com/embedded/jetson-benchmarks.

[5] Jetson Xavier NX Series. developers.nvidia Available at: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-nx/.

[6] Letmaik, “Letmaik/pyvirtualcam: send frames to a virtual camera from python,” GitHub. [Online]. Available: https://github.com/letmaik/pyvirtualcam. [Accessed: 05-May-2023].

[7] “Adafruit NeoPixel Library,” Adafruit Neopixel Library: Adafruit_neopixel class reference. [Online]. Available: https://adafruit.github.io/Adafruit_NeoPixel/html/class_adafruit___neo_pixel.html. [Accessed: 05-May-2023].

[8] T. A. Team, “Play a melody using the tone() function,” Arduino Documentation. [Online]. Available: https://docs.arduino.cc/built-in-examples/digital/toneMelody. [Accessed: 05-May-2023].

[9] SurferTim, Donbindner, ChrisFromFreeMind, Mikatti, Jannick_v, Rsliva, gian21391, MatzeIsar, Lucberger, Dmnc, Kprims, Tigger, Cossoft, Digs, Jdgramp, Dkebler, and system Closed, “The linux ttyacm0 drama - more details after a lot of experimenting,” Arduino Forum, 30-Aug-2017. [Online]. Available: https://forum.arduino.cc/t/the-linux-ttyacm0-drama-more-details-after-a-lot-of-experimenting/126927/27?page=2. [Accessed: 05-May-2023].

[10] Ozkefe, Feyza. “Trashnet.” Kaggle, 19 Nov. 2021, https://www.kaggle.com/datasets/feyzazkefe/trashnet. [Accessed: 05-May-2023].

[11] “TACO/Data at Master · Pedropro/TACO.” GitHub, github.com/pedropro/TACO/tree/master/data. Accessed 5 May 2023.

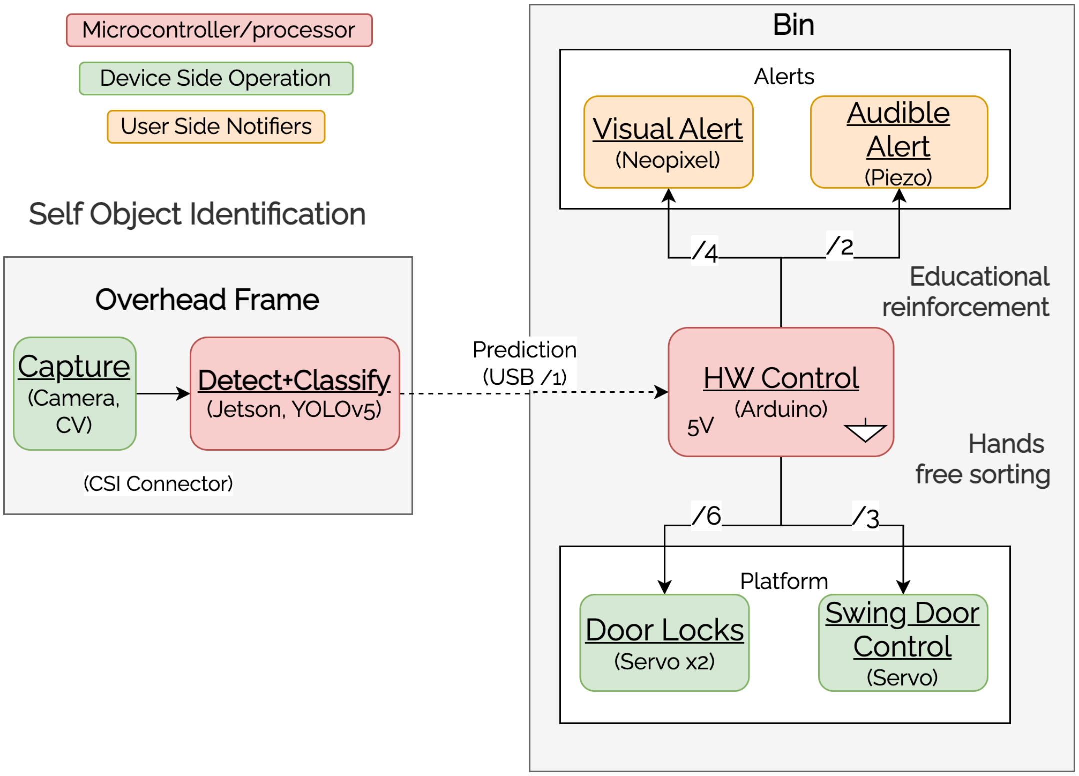


Figure 11: Components of system relating to use-case requirements

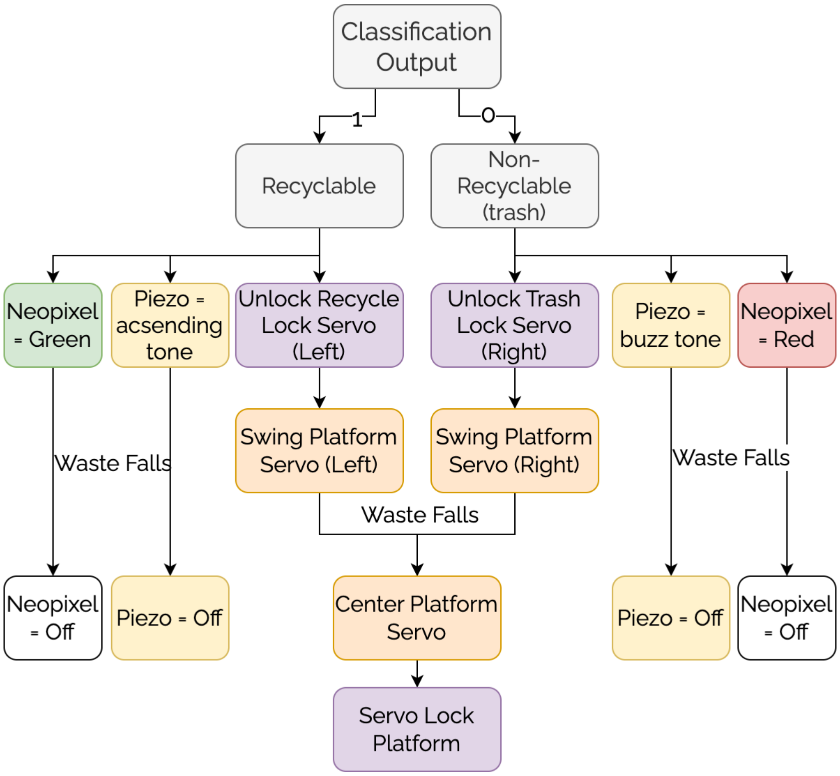


Figure 12: General flow of the system

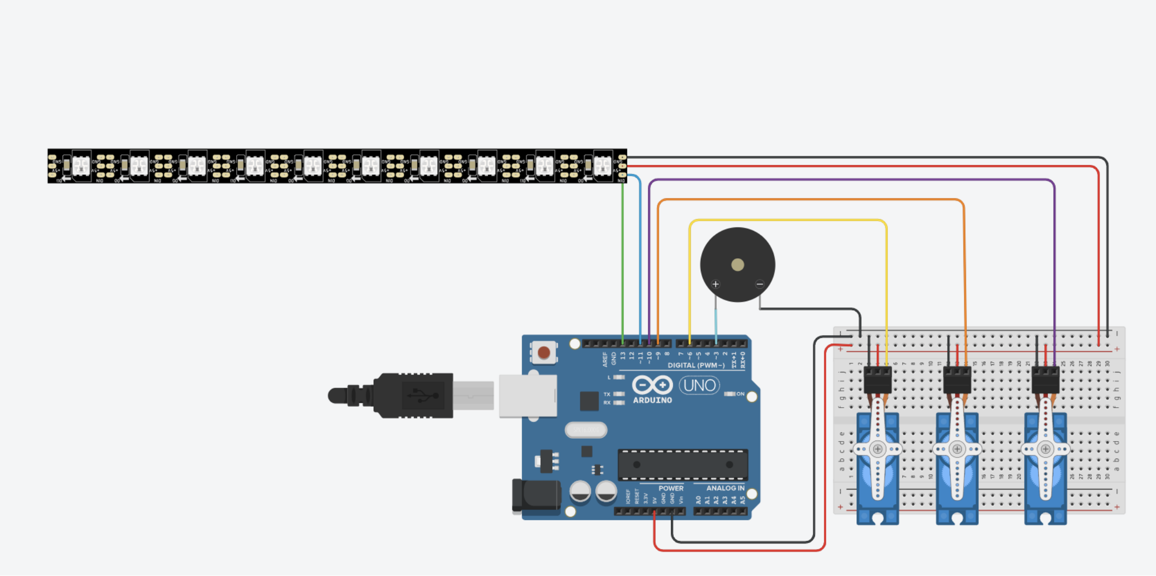


Figure 13: Simulation of Arduino pin connections

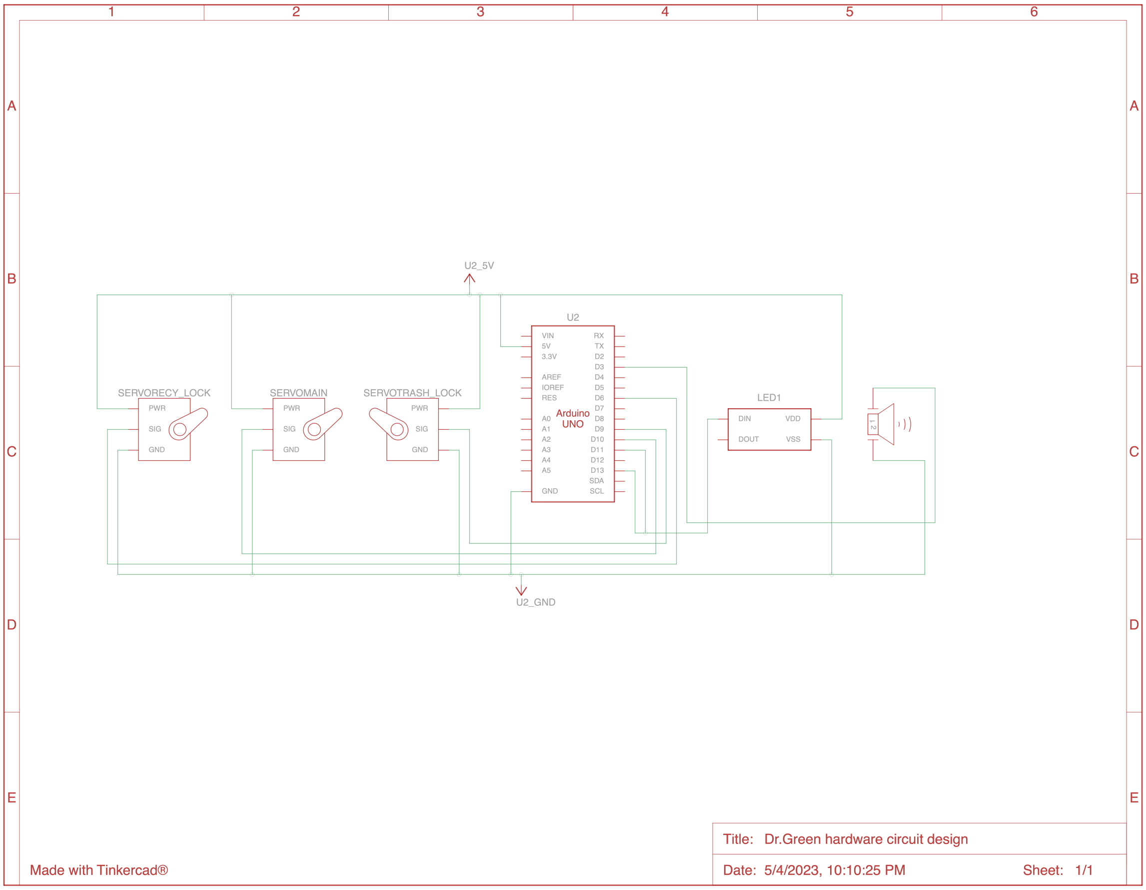


Figure 14: Hardware circuit design

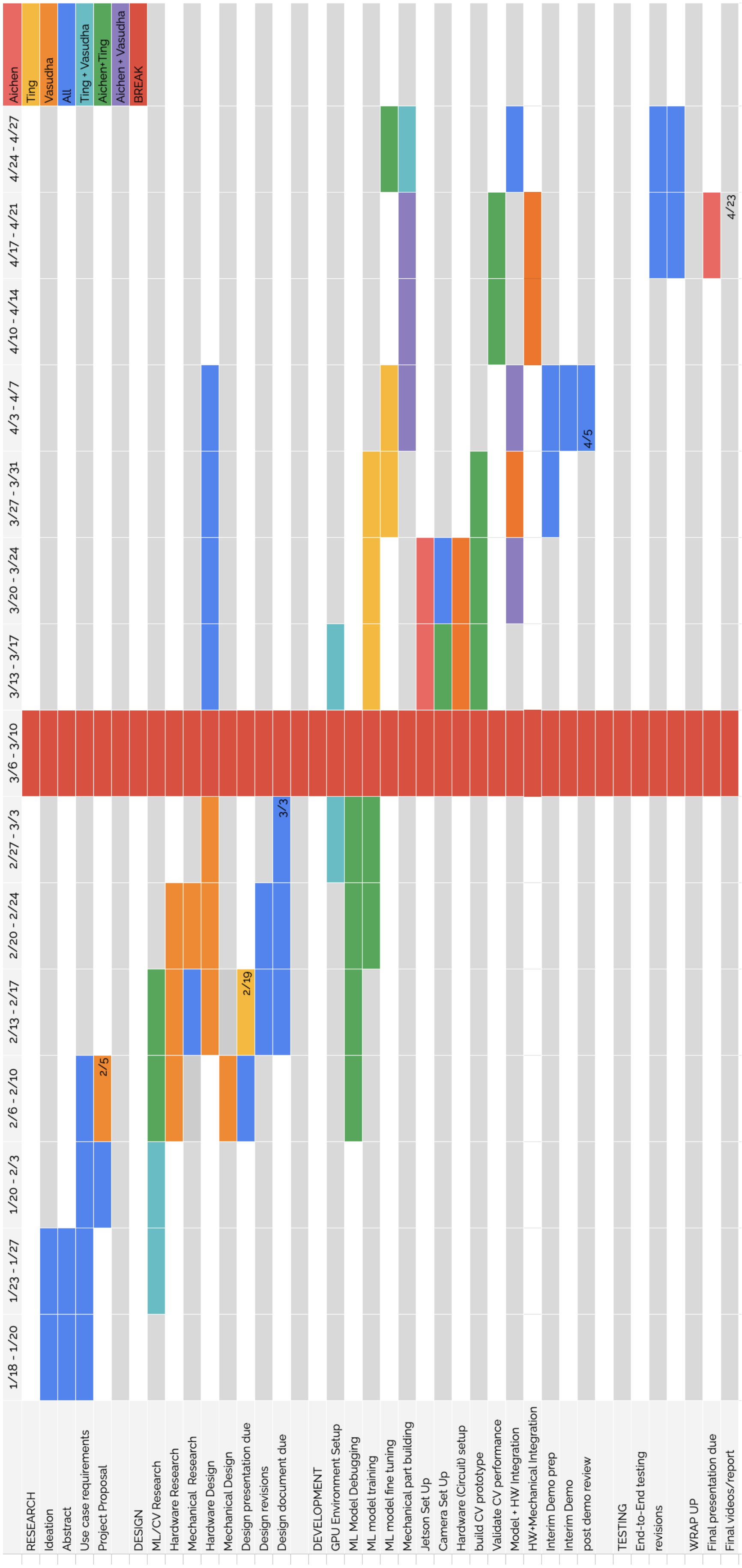


Figure 15: Gantt Chart