Bin There Dump That

Authors: Lauren Tan, Jessica Lew, Tate Arevalo: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—An AI trash can which automatically sorts items into two categories: recyclables or non-recyclables. Users can throw away objects one-by-one into a box, and have them moved into the correct bin by a sliding mechanism. We will embed several sensors underneath the platform of the box and mount a camera on the trash can lid to identify the object's primary material with several sensor classifiers and an image classifier using a neural network model. According to the sensor classifier outputs and the predicted category from the image classifier, the item will be classified as either recyclable or non-recyclable.**

***Index Terms*—Classifier, Jetson Nano, Neural Network, Recycling, Sensor Array, Timing Belt**

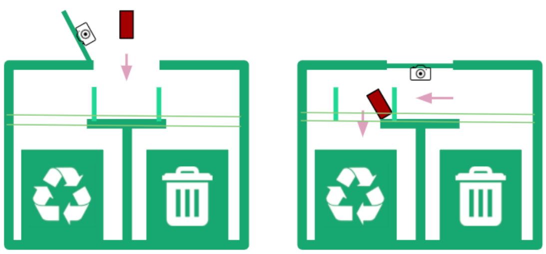

Figure 1: User Flow

1 INTRODUCTION

The city of Pittsburgh's current recycling rate is a staggeringly low 17%[9]. Compare that to CMU's recycling rate of 35%[3], which is still incredibly low considering that ideally, these rates could be above 50% as seen in some other countries. Bottom line, people do not know how to recycle as well as they think they do. In fact, many people engage in "hopeful recycling", meaning that they recycle what they think should be recyclable, rather than what actually is. Only 31%[5] of people always recycle a recyclable item, meaning that the rate of contamination (non-recyclable items in the recycling pipeline) is extremely high, which reduces the percentage of items in recycling that actually become recycled. Our goal is to help increase the recycling rate on CMU's campus.

To do this, we are creating an intelligent waste bin capable of sorting waste one item at a time into either the recyclable bin or the non-recyclables bin in under a second and with an accuracy rate of 90%. There are four main types of materials we consider recyclable: metals, glass, PET and HDPE plastics, and paper/cardboard. In terms of similar products on the market, there are few, and some are priced upwards of $1,000. Thus, our trash bin offers a less expensive solution to recycling. Zero thinking is involved in the use of our product as users simply place a waste item into the bin and can walk away. The trash bin will automatically classify that waste item as recyclable or non-recyclable and dispose of the item into the correct bin housed within our product. Gone are the days of debating whether or not certain trash is recyclable, which ultimately leads to low recycling rates and high contamination, as we welcome this easy-to-use and innovative solution.

2 DESIGN REQUIREMENTS

Our main goal is to improve CMU's recycling rate, so our trash can must make recycling easier, more convenient, and more efficient for the average user. Thus, we have developed several accuracy and latency requirements that are critical to the success of our project.

2.1 Classifier Accuracy

First, the accuracy of the overall classifier used to sort items into recycling and non-recycling categories must be at least 90%. This accuracy rate is significantly higher than both the average recycling rate and contamination rate.

The overall classifier will combine outputs from the image classifier and sensor classifiers. We will test each individual classifier and then test the overall classifier to ensure that the resulting percent of correctly categorized trash is at least 90%. To test the image classifier accuracy, only images will be used. We will be using a pre-labelled data set in addition to our own manually collected and labelled data set. To test the sensor classifiers, we will place objects into the sorting box corresponding to that classifier's target material. For example, metals will be used to test the accuracy of the metal classifier. Once we have tested the individual classifiers, we will place objects into the sorting box for the overall classifier to categorize as recyclable or non-recyclable so that we can verify the overall accuracy rate.

2.2 Mechanism Accuracy

The second requirement refers to the accuracy of the mechanism that will be used to push an item into one of the two bins. This accuracy relates to the likelihood of moving an item into a bin after classification occurs. Therefore, this accuracy level needs to be at least 99% in order for the overall system to work as well as possible.

To test this accuracy rate, objects will be placed into the sorting box. The rate of successfully placed objects by the mechanism into the specified bin will be calculated and compared to 99%.

2.3 Latency

The third requirement for this project is overall latency, which should be less than one second. We define overall latency as the end-to-end processing time of one object. This is measured as the time between an item being dropped into the sorting box of the system and the sliding mechanism returning back to the center of the platform after moving the item into a bin. Thus, this latency includes classifier latency (the time taken to classify an object as recyclable or non-recyclable) and mechanism latency (the time taken to move an object into a bin). This requirement is less than one second, since this is approximately how long it takes for a user to drop a piece of trash into the bin. This also accounts for different scenarios in which users will throw away trash, such as waiting in a line with other people to throw away trash, or simply throwing away multiple pieces of trash by themselves. This timing requirement ensures usability of the overall system with the processing of items one-by-one.

Overall latency will be tested by adding the total classification latency to the mechanism latency. Both of these individual metrics will be measured separately and averaged over multiple inputs.

2.4 Object Scope

The last requirement ensures that our trash can is able to accept a variety of items including common trash such as water bottles and food containers. Given that our trash can is intended for CMU's campus, we aim to handle small to medium sized objects of reasonable weight. We define this as objects less than 10x10x10 (WxLxH) inches in size and less than 2kg (around 4.4lbs in weight). This allows us to accept most items that are able to fit into the sorting box, even ones that wouldn't be common around CMU campus, like a brick.

In addition, we will not be accepting any liquids or sauces poured directly into the sorting box, as this can interfere with our sensor readings and sensors. Therefore, we will only be accepting items that have dry exteriors.

We will test these requirements by placing items of various size, shape, and weight into the sorting box, and then verifying that the mechanism can push that item into the correct bin.

3 ARCHITECTURE OVERVIEW

Please refer to Figure 2 and Figure 9 for our system and software specifications. Figure 2 contains our overall system specification diagram. Figure 9 contains a class diagram which depicts the interfaces that we have defined to facilitate integrating each of our components together.

The Jetson Nano receives images from the Raspberry Pi v2 camera, which is connected via MIPI CSI-2 (Camera Serial Interface) and mounted underneath the trash can lid. The inductive, LDR, IR, and capacitive sensors send digital output (1/0) to the Jetson Nano, while the load sensors have analog output. The inductive, IR, LDR, and capacitive sensors at the bottom of the sorting box platform are connected to GPIO pins on the Jetson Nano, and the load sensor is connected via I2C. The Jetson Nano controls the display of an LED strip under the lid, which is connected using GPIO pins.

The Jetson Nano also controls the motor driver and communicates between the motor controller using two GPIO pins and a Python library called `RpiMotorLib`. The motor driver drives the stepper motor for the sliding mechanism. The contact switches for the mechanism connect to the Jetson Nano via GPIO, and will be used to re-calibrate the position of the motor in the motor controller.

After obtaining and resizing images taken from the camera (via the `PiCamera` library), these images will be analyzed to determine whether an object is in the box above the platform. After obtaining images that contain objects, background subtraction will be done on the images to remove the sensors displayed on the bottom of the platform, and any other background noise. These final images will then be used in the image classifier to obtain a predicted category of recyclables or non-recyclables.

The image classifier will then use the trained ResNet model to classify the object as non-recyclable, or one of the recyclable categories (metals, paper/cardboard, plastics, glass).

Our sensor array feeds data through GPIO and I2C ports into the sensor classifier, which we will be creating ourselves. Within the sensor classifier, there are individual material classifiers which will output whether or not the inputted item is of that particular material. Each of these individual sensor classifiers is composed of various combinations of multiple sensors that are able to detect certain materials. For example, in the metal classifier, inductive sensors will be used to determine whether an item is metal or nonmetal because inductive sensors are specifically able to detect the presence of metal.

The image classifier predicted category and the output of the different types of sensor classifiers are used in the overall classifier, which makes the final classification decision of recyclable or non-recyclable. This decision is then sent to the motor controller which prompts the stepper motor of the mechanism to move the sliding belt so that the item is pushed into the corresponding bin. At this point, the mechanism also will re-calibrate itself using two contact switches at the ends of the rails after each object is sorted, in order to re-align correctly over the center of the platform that objects are placed onto.

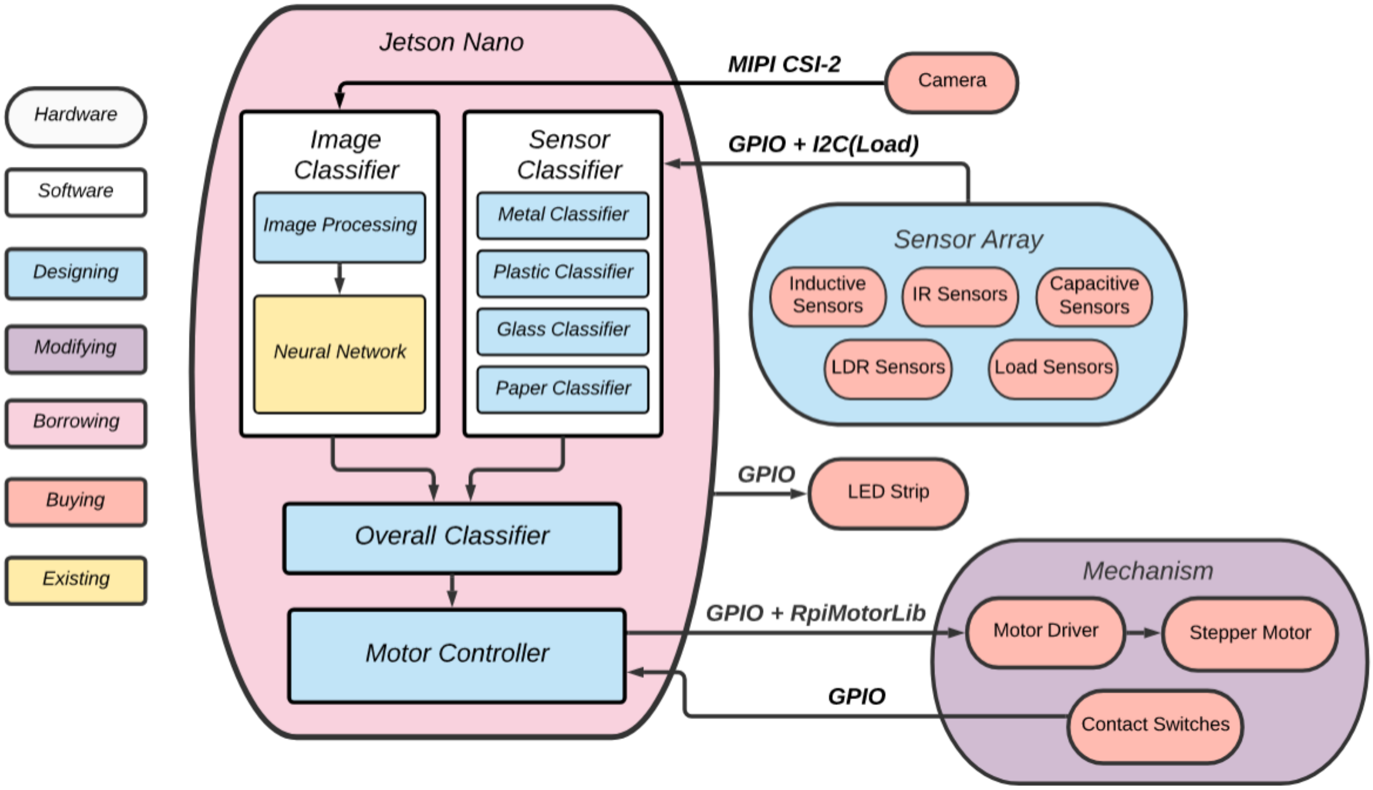

Figure 2: System Specification Diagram

4 DESIGN TRADE STUDIES

Throughout the design process, we discussed several different approaches for our mechanical, software, and hardware components. For each of these decisions, we considered the solution which would be able to meet our design requirements with minimal cost and complexity.

4.1 Mechanism

Timing Belt Mechanism: While developing the mechanism, we considered a variety of different linear motor actuators including a conveyor belt, rack and pinion mechanism, and timing belt mechanism. We ultimately chose the timing belt mechanism because we would not be able to embed sensors onto a moving conveyor belt platform and the cost of the gear rack for the rack and pinion mechanism was much more expensive than the timing belt. During the design presentation, we had intended to use an existing CAD model to laser cut most of the major components of our mechanism including the pulley gear and the motor mount. This method was significantly less expensive than ordering parts, but required time for manufacturing parts and would have to be significantly modified to fit our sliding box design. Since no one on our team has significant mechanical design experience, we instead decided to repurpose a timing belt mechanism from one of our teammates' previous projects which was based on this model from Thingiverse [8]. We will still need to make some modifications such as increasing the size of the timing gear and redesigning the moving platform, but these modifications are much more feasible and have currently been finalized.

Motor Control: We chose the Nema 17 Stepper Motor and DRV8825 motor driver because they were among the cheapest models that fit our use case. Based on our design requirements, the mechanism should achieve an approximately 0.5 second latency and be able to move small to medium sized objects. We calculated that the Nema 17 would be able to meet these metrics given its 59 Ncm holding torque and 600 RPM motor speed.

The maximum linear speed of the belt mechanism is dependent on both the motor's speed and the size of the pulley gear which turns the belt. Because we can control the pulley gear size, we only needed to calculate the exact gear diameter necessary to meet our latency requirement.

$$speed = \frac{2 * 10 \text{ in.}}{0.5s} = 40in/s = 1.016m/s \tag{1}$$

$$d = \frac{60 * speed}{\pi * rpm} = \frac{60 * 1.106}{\pi * 600} = 35mm \tag{2}$$

Equation 1 details the linear speed of the belt needed to move the sliding box off of the platform and back in 0.5 seconds. Equation 2 then details the diameter of the gear needed to meet this linear speed given the Nema 17's 600 rpm. From our calculations, we can meet our mechanism latency metric using any pulley gear with a diameter greater than 35mm. As a result, we have selected a 42mm gear.

Then, to verify that the stepper motor would be able to meet our weight requirements, we calculated the minimum torque required to move a mass of 2kg (our maximum object weight).

$$F = m * a = 2kg * 9.8m/s^2 = 19.6N \tag{3}$$

$$torque = F * r = 19.6N * 0.0175m = 34.3Ncm \tag{4}$$

Equation 3 details the force of a 2kg object, and Equation 4 details the torque needed to move that amount of force given the radius of our gear. Since the Nema 17 has a maximum holding torque of 59cm, we should easily be able to support small to medium sized objects.

4.2 Hardware

Jetson Nano: We chose the Jetson Nano over similar platforms like the Raspberry Pi 3 for its powerful GPU and higher performance. The Jetson Nano was designed to run multiple neural networks in parallel and contains a quad-core ARM processor with an 128 core GPU, which will allow us to minimize our image classification latency. Initially, we had considered using an Arduino in addition to the Jetson Nano to interface with the motor driver, but decided that step was redundant since the Jetson Nano contains 22 GPIO pins and supports our motor driver. Thus, an Arduino would add unnecessary complexity and latency to our project.

Camera: We chose the Raspberry Pi v2 camera because of its compatibility with the Jetson Nano. The camera connects directly to the Jetson Nano's CSI port and can be easily controlled using the `PiCamera` library. We had considered other cameras with higher quality, but decided that a standard 1080p resolution and 30fps frame rate would be sufficient for our use case. We do not require a higher resolution because we will decrease the size of our image for image classification anyway to reduce latency in the neural network. Similarly, we do not require a higher frame rate because we assume that the item is stationary within the bin so we do not need to process a smooth video stream.

4.3 Sensors

We had originally intended to segregate garbage using only computer vision. Although this method was simpler and less expensive than adding a sensor array, we realized that a camera would not be sufficient for classifying visually similar materials such as different types of plastic. Thus, we decided to add a variety of sensors, ensuring that at least one sensor was capable of detecting each of our target materials: paper, glass, plastic, and metal. While selecting sensors, we consulted research papers and previous projects whenever possible to roughly estimate the sensor's material detection accuracy.

Capacitive Sensor: Capacitive sensors are commonly used for material detection because they are capable of being fine-tuned to detect different capacitance levels. Using capacitive sensors alone we would be able to distinguish between all of our recyclable materials because each has a specific capacitance range. However, capacitance sensors use more GPIO pins and cost significantly more than any of the other sensors that we were considering such as LDR and IR. This is partly because the sensor output is binary, so we require two capacitive sensors to detect the lower and upper range of capacitance for each target material. In addition to this, we cannot afford capacitive sensors with large sensing ranges, so we would need to purchase many capacitive sensors for full coverage of our platform. Given all of these drawbacks, we decided to limit the number of capacitive sensors wherever possible in our system; however, there were several materials that could not be reliably detected without a capacitive sensor. Thus, for glass and plastic detection, we decided to use capacitive sensors, but still rely primarily on other sensors such as load and IR to further minimize the number of capacitive sensors required.

Inductive: To detect metal, we decided that using only inductive sensors should be sufficient based on a research paper which achieved 98% metal detection accuracy using inductive sensors alone [2]. We also considered using capacitive sensors, but due to its previously mentioned drawbacks, opted for inductive sensors as a less expensive alternative. Inductive sensors are still more expensive than some alternatives such as ultrasonic sensors, but we decided that its high accuracy rate warranted the extra cost.

IR/LDR: Both IR and LDR sensors are much cheaper and have a much larger sensing range than capacitive sensors, but may not be able to achieve the same accuracy level. Thus, when designing our sensor classifiers, we made sure that none of our classifiers relied on IR or LDR outputs alone. Since LDR sensors measure the presence of light, we can use them to detect transparent items such as plastic and glass. However, both of these materials may not be perfectly transparent which would reduce the accuracy of our plastic and glass classifiers. Thus, we will also use capacitive sensors for plastic and IR sensors for glass to increase our accuracy rates. IR sensors can detect nearly all materials other than glass, so we can use them to distinguish between glass and non-glass materials.

Load: We decided to use load sensors for paper detection and for handling certain edge cases. Although the assumption that paper is lighter than other materials may not always hold, we will combine the output of the load and capacitive sensors to increase the accuracy of our paper classifier. By adding the output of the load sensor to our paper classifier, we can decrease the amount of capacitive sensors needed while maintaining a high accuracy rate. Furthermore, load sensors can detect edge cases which are not detectable given our other sensors. In particular, in cases where non-empty items are placed onto the platform, like a full bottle of water, we will be able to detect that there is something inside the water bottle, and classify it as non-recyclable accordingly. By using output from the load sensor, along with the image classifier category, we will be able to handle many edge cases.

4.4 Sensor Array

We decided to place our sensors in the formation of a sensor array because most of our sensors have small detection ranges. In fact, the sensing range of our inductive sensor is so small (8 mm over the sensor's 0.5 inch contact point) that we would need 400 inductive sensors to fully detect any object on our 10x10 inch platform, which is infeasible given our current budget. After measuring common objects such as water bottles and tin cans, we decided to limit our scope to a 2x2 inch minimum detectable object size because we can rely on our image classifier for smaller objects. Based on this assumption, our inductive sensors can be spaced 2 inches apart, reducing the number of sensors needed from 400 to 16 sensors. Because we require two capacitive sensors at each point to detect the lower and upper range of capacitance, we would still require 32 capacitive sensors for each target material to meet this minimum object size. This means that we would need a total of 64 capacitive sensors to detect both paper and plastic, which would exceed our budget. As a result, we decided that we could reduce the total number of capacitive sensors to 12 and rely on using additional sensors, load and IR sensors for detecting paper and plastic, respectively.

In addition to budget constraints, the number of GPIO pins was a major limitation in the design of our sensor array. The Jetson Nano only has 22 GPIO pins, but given the original design of our sensor array, we would need 48 GPIO pins (12 for capacitive, 16 for inductive, 4 for IR, 16 for LDR). We considered using shift registers but realized that the logic of our system could be simplified to avoid this added complexity. For example, if any one of the inductive sensors detects a metal, the classifier should identify that a metal was detected, even if the other inductive sensors did not detect anything. Because we only need the output of one sensor to be high in order to detect the object, we can wire sensors in parallel which uses only 1 GPIO pin. Overall, we will need 11 GPIO pins (4 for capacitive, 1 for inductive, 1 for IR, and 1 for LDR, 2 for motor driver, 2 for contact switches). We cannot further reduce the number of pins used for capacitance because the capacitance output is binary, so we must be able to detect the lower and upper capacitance range of each target material, paper and plastic. Therefore, we need 1 pin per upper bound, and 1 pin per lower bound.

4.5 Software

AWS: We considered using AWS or a GPU in order to train our image classifier. Using AWS was a better choice for our use case because we only need to train the model once, and AWS allows us to pay by usage. There were also budget concerns with using a GPU over AWS because we would need to purchase a GPU (which can get expensive), while AWS was free with AWS credits provided by the class.

Resnet50 Model: The ResNet50 is a built-in model from the `torchvision.models` module in the Pytorch library. It consists of a convolutional neural network made up of 50 layers that has been pre-trained on over a million different images. We decided to use an existing model for our image classifier rather than build a new one from scratch due to the difficulty of determining an object's material from images (i.e. visually similar objects could have different materials), which is necessary in order to correctly classify the object as recyclable or non-recyclable. This model has also been previously used to obtain a very high accuracy rate for garbage classification, 95%, so we are also aiming for a high classifier accuracy rate after training the model on the images from the Kaggle dataset[7] and our own images. Additional considerations we used when choosing the model included the number of layers in the neural network, since using fewer layers will reduce classifier latency.

5 SYSTEM DESCRIPTION

5.1 Image Classifier

The image classifier only uses images as input to classify an item, and has 5 different categories for the output. These categories consist of non-recyclables, and the different types of recyclables: metals, plastics, paper or cardboard, and glass.

Before the image classifier classifies an image, preprocessing of the image will happen. This step includes subtracting the background of the bottom of the platform using openCV, so that only the object itself is visible against a white background. This is necessary due to the way sensors are embedded underneath the platform, as they will be seen from a birds-eye view of the bottom of the platform. Resizing each image to a smaller size for prediction after training will also occur. In addition, we will be detecting objects using `openCV` on the images that the Raspberry Pi camera takes.

An existing dataset[7] from Kaggle, along with our own images, will be used to train the model for the image classifier. The model we are using is ResNet-50, which is a convolutional neural network with 50 layers. ResNet-50 is part of set of pre-trained models from PyTorch's torchvision.models, a machine learning library. The model will be trained on the dataset offline using AWS.

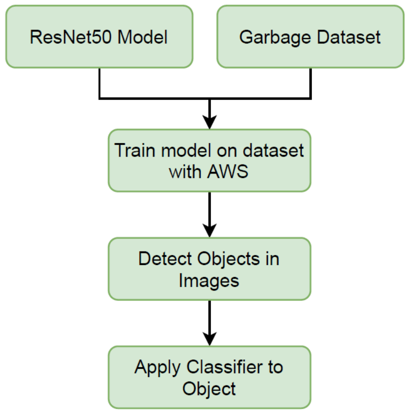

Figure 3: Image Classification Flowchart

5.2 Sensor Classifiers

There are four sensor classifiers that correspond to each of the recyclable categories: metals, paper/cardboard, plastics, and glass. Each of these sensor classifiers uses a specific kind of sensor or combinations of sensors to distinguish between each material.

The metals classifier distinguishes between metals and non-metals. This will be done using output from a set of 16 connected inductive sensors. The output of each inductive sensors is binary, so each inductive sensor will act similar to a switch and turn on when metal is within the sensing range, and will remain off otherwise. By connecting all of these inductive sensors together, the final output of the set of these sensors will be "on" if any of the 16 sensors detects a metal, or "off" otherwise. Thus, this sensor classifier for metals will have a binary output to distinguish metals and non-metals.

The plastics classifier distinguishes between certain types of plastics (PET and HDPE) that are considered recyclable, and other kinds of plastics that are not. The sensors used to detect these types of plastics are capacitive sensors and LDR sensors. The output of each capacitive sensor is also binary, but not all of the capacitive sensors can be connected in the same way as the inductive sensors. For every position on the platform we want to detect plastics, there needs to be two adjacent capacitive sensors. Every pair of capacitive sensors will then be connected to the Jetson Nano. Materials like plastics have a range of capacitance values, so we need to know if an object lies within the lower and upper bound of the capacitance ranges for PET and HDPE plastics. Each of the two adjacent capacitive sensors will be for these lower and upper bounds, respectively. In addition, since the capacitive sensors can only determine if an object has a capacitance value above a certain threshold, successful plastic detection for objects will mean that one capacitive sensor had output 1 (for lower bound), and the adjacent capacitive sensor had output 0 (for upper bound). Furthermore, we will use LDR sensors to detect the transparency of the object.

The glass classifier will distinguish between glass and non-glass objects. We will be using output from IR and LDR sensors. Since the output of both of these kinds of sensors is binary, we will be similarly using a binary output for the glass classifier.

For the paper classifier, we will be using a combination of capacitive sensors and load sensors to distinguish between between paper and non-paper items. We will be spreading out these sensors across our platform to achieve higher area coverage and meet our minimum object size of 2x2 inches, so the final decision of paper vs. non-paper will be made individually by each type of sensor. For areas of the platform that we have capacitive sensors on, we will be able to use the capacitive sensor's binary output to determine whether an object is made of paper or cardboard. The load sensor, however, has analog output, so we will need to do some testing with paper and other objects to determine the exact threshold we will use for distinguishing between paper and non-paper objects in areas that the load sensor is detecting.

5.3 Overall Classifier

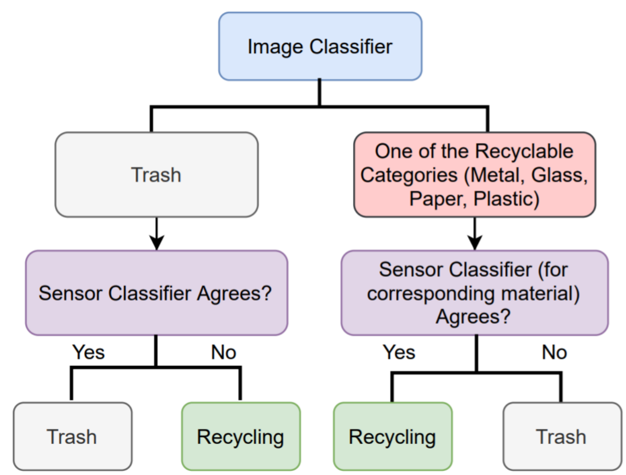

Figure 4: Classifier Flowchart

Combining the output from the image classifier, and the output of the four sensor classifiers, this final classifier will determine the final category of recyclable or non-recyclable for an item.

The image classifier output is used for most items when the confidence level of the output category is high. This confidence level threshold will be determined through later testing of the overall model.

However, when the image classifier is unsure of the classification of an item (low confidence level), the sensor classifier outputs will be used to corroborate or reject the image classifier decision. This depends on whether the image classifier output is part of recyclables or non-recyclables.

If the image classifier output is one of the recyclable categories (metals, paper/cardboard, plastics, or glass), the sensor classifier for that recyclable category will be used. The output category from the image classifier and corresponding sensor classifier must agree in order to classify an object as recyclable. This agreement between the image classifier and the appropriate sensor classifier is necessary to reduce the chance of false positives for recyclables, which is more important than false negatives. This is because items that are non-recyclable and sorted into the recycling bin increase the contamination rate, but recyclable items sorted into the non-recyclable bin do not have a similar negative consequence.

Otherwise, if the image classifier output is non-recyclable, the sensor classifier output will be used to make the final decision. For example, if the image classifier determines an object to be a kind of metal, but the metals classifier output doesn't detect any metal, the final classification of the item will be non-recyclable. In the other case where the sensor classifier does classify an item as one of the recyclable categories, the final classification of the item will be recyclable.

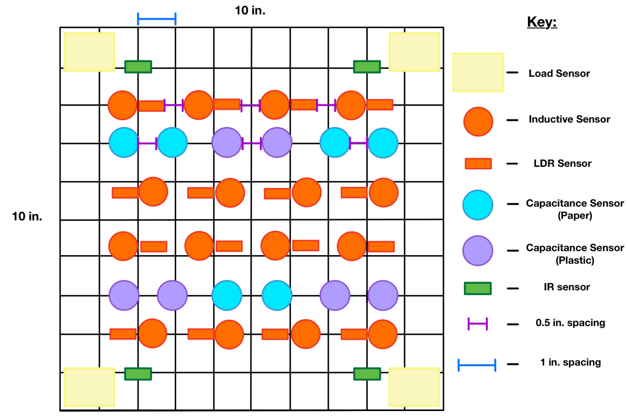

Figure 5: Sensor Array Diagram

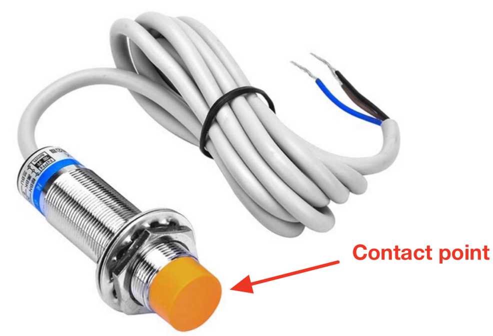

Figure 6: Inductive Sensor

5.4 Sensor Placement

We will be creating a sensor array, which is a configuration of various sensors in a strategic pattern, because most of our sensors have a very small sensing range and thus would make our sensor classifier extremely unreliable if we just used one sensor.

The grid depicted in Figure 5 shows a bird's eye view of our platform. Our platform is 10x10 inches and 2x2 inches will be our minimum detectable object size. So, based on our sensor configuration, we will be able to detect an item with some part of the sensor array no matter its placement on the platform. Items smaller than 2x2 inches will be primarily handled by our image classifier.

In total, 16 inductive sensors, 12 capacitance sensors, 4 IR sensors, and 1 load sensor will be placed throughout our sensor array. Again, 2 capacitance sensors will be used in pairs to detect specific materials since one is needed for the lower bound of capacitance for that material and the other for the upper bound. The necessary spacing to avoid interference between sensors has also been taken into account.

Each sensor will be mounted underneath the platform through cut-out holes so that the sensors are flush with the platform. Furthermore, as shown in Figure 6, our capacitive and inductive sensors have a built-in contact point as depicted by the orange cap.

5.5 Mechanism

For our mechanism, we will be using a sliding mechanism that pulls a box structure across the platform, swiping the item into either the recyclable or non-recyclable bin. As previously discussed, we will be using a Nema 17 Motor and DRV8825 motor driver to control our mechanism. Given the Nema 17's 600 rpm motor speed and 59 Ncm holding torque, our mechanism should meet our latency and object scope requirements.

In regard to the design of our sliding mechanism, please refer to Figure 7 and Figure 8. Linear ball bearings attached to metal linear rods will make up the fundamental sliding portion of our mechanism. The linear rods are 31 inches in length to allow the box to push trash entirely off of the platform, and they are spaced approximately 11 inches apart to allow room for the box to move in between. The four mounts located on each side of the rods support the linear rods, pulley gears, and stepper motor. These mounts have already been 3D printed and can be drilled into the sides of our trash can exterior. To turn the belt, we have two 42mm pulley gears, one fitted around the shaft of the stepper motor and the other held in the opposite mount as shown in Figure 7. The timing belt wraps around both of these gears and is also fastened to the box. The box is constructed from four wooden walls, each the size of our platform (10x10 in). The box is then attached to small shelves protruding from the linear ball bearings as seen in Figure 8. These shelves can be directly drilled into the linear bearings.

Once the overall classifier makes its decision, the motor driver will drive our stepper motor to turn the gear around its shaft and move the timing belt. Because the belt is fastened to the box, the box will be pulled in the same direction as the belt, pushing the item off of the platform and into the correct bin. Because the stepper motor may occasionally miss steps, the mechanism may become miscalibrated over time. To account for this, we will mount a contact switch on either side of the rail so that the box's position can be re-calibrated when the box makes contact with one of the switches.

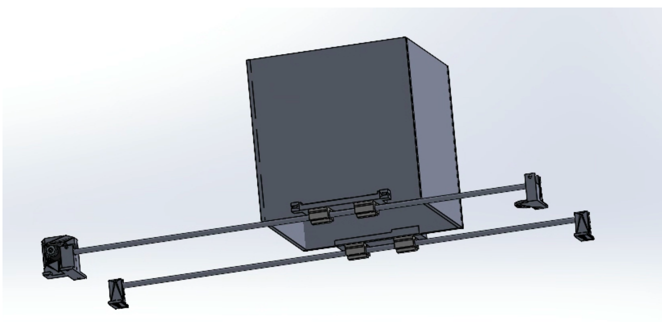

Figure 7: Mechanism CAD Model

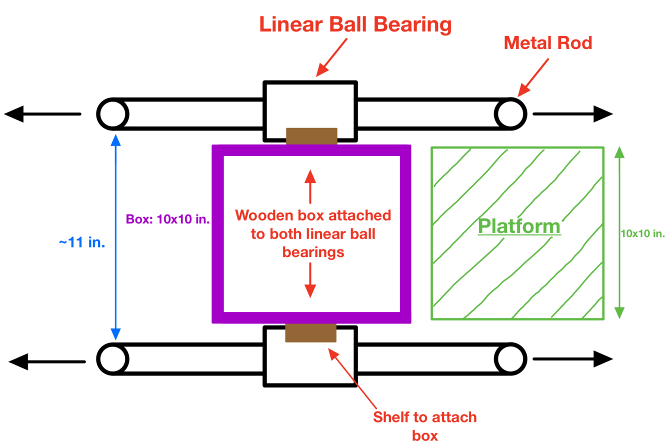

Figure 8: Mechanism Diagram

6 PROJECT MANAGEMENT

6.1 Schedule

In our schedule, we prioritized developing the image and sensor classifiers so that we can better mitigate against unforeseen risks. We have already purchased and tested the majority of our sensors, and we are currently on-track to begin building our image classifier next week. By the interim demo, we plan to have our overall classifier and mechanism fully operational, but not necessarily integrated together. Also, the overall classifier should be able to combine camera and sensor inputs to achieve a moderate accuracy rate. The full Gantt Chart can be found in Appendix B.

6.2 Team Member Responsibilities

Our project is divided into two main categories: mechanism/hardware and classifiers. Lauren's primary responsibility is developing the image, sensor, and overall classifiers. This includes building each of the classifiers, training the models on AWS, integrating each of the classifiers, and testing the classifiers for accuracy and latency. Tate's primary responsibility is building the mechanism and the sensor array as well as the trash can exterior. Jessica's primary responsibilities are spread across the hardware and classifier categories, focusing on helping Tate and Lauren with the tasks that are the most time consuming. Specifically, she will be helping with building and training the image classifier as well as integrating all of the components with the Jetson Nano. All of us are also responsible for collecting sensor data and testing the overall trash can.

6.3 Budget

The budget can be found in Appendix C. Because we are reusing parts for our mechanism, there is a large discrepancy between the cost used by our budget and the total cost needed to recreate the project from scratch. We have separated the cost from our budget and the cost of recreating the project into the columns "Total Spent" and "Total Cost," respectively.

6.4 Risk Management

Our main risk factor is classifying edge case objects. In particular, items placed at different angles or items composed of multiple materials are more difficult to classify. This risk can be mitigated against by adding more training data and adding different types of sensors. To account for items placed at different angles, for instance, we will include images of items at various angles in our image classifier training set. To account for items composed of multiple materials, we assume that the item has a primary material and we will classify based on that primary material. For example, although water bottle lids are not recyclable, we will still categorize the entire water bottle as recyclable. If the item is not empty such as a filled water bottle, we can use a load sensor to detect the liquid inside the bottle and categorize the overall item as non-recyclable. Other edge cases with regards to our classifier may appear throughout the semester, so we have prioritized classifier testing in our schedule. This will give us enough time to order more sensors and collect more training data as needed.

With respect to our sensors, our largest risks are their accuracies and small sensing ranges. Although we have researched each sensor's approximate accuracy at detecting its target material, it is difficult to estimate these accuracy levels without further testing. To mitigate against this, we have already begun testing most of our sensors so that we can quickly determining which sensors we will use and which sensors we will be replacing. For example, we had expected the IR sensor to not detect glass because infrared rays do not pass through glass. However, while testing, we found that the results were more erratic and were able to order ultrasonic sensors as a potential replacement.

In terms of small sensing ranges, our inductive and capacitive sensors have around an 8mm and a 10mm sensing range, respectively. As a result, we have designed a sensor array such that a normally sized object should have contact with at least one of each type of sensor. If the object is abnormally small, then we will rely on our image classifier to make the final classification decision instead. We have also left room in our budget so that we can order more sensors for our sensor array if necessary.

Furthermore, because we intend to spread our trash can across Carnegie Mellon's campus, our trash can must be able to handle a range of environmental factors. Inclement weather such as rain can be mitigated against by adding a lid to our trash can and waterproofing the exterior. We will also mount an LED strip inside the trash can so that our image classifier can still function in low light conditions.

7 RELATED WORK

There are several projects related to automatic trash sorting. These projects tend to use either computer vision or sensors for classification and are able to achieve reasonable accuracy. Researchers from MIT have used soft robotics to create a trash sorting robot based on touch [4]. The robot can detect size and hardness of the material by squeezing the object. On the other hand, Oscar the AI trash can uses solely computer vision for classification[1]. Similar to our project, Oscar is intended for consumer use and can only sort one item at a time.

While initially developing our project, there were several projects that heavily influenced our decisions. For the sensors, we used the results of "Design and Development of the Trash Splitter with Three Different Sensors" to narrow down which sensors to consider for material detection[2]. We were also inspired by the mechanism in the "Sorter Bin" project which used a stationary platform embedded with sensors and a sliding box to push the trash off of the platform[6]. Unlike the "Sort Bin" project, however, we will be using more sensors and computer vision to aid our classification. We are also using a belt driven mechanism rather than a ball screw mechanism to decrease mechanism latency.

7.1 AWS Credit Usage

A special thanks to Amazon, which provided AWS credits that helped make this project possible. The AWS credits were used for offline training of the image classifier.

References

[1] Aiman Adil. *AI-Generated Oscar: Trash-Sorting Robot to Achieve Zero Waste.* URL: `https://medium.com/illumination/ai-generated-oscar-trash-sorting-robot-to-achieve-zero-waste-72531ebacb58`.

[2] Despaleri Angin. "Design and Development of the Trash Spliter with Three Different Sensors". In: *IOP Journal of Physics: Conference Series* (2018).

[3] CMU. *Recycling and Waste Management.* URL: `https://www.cmu.edu/environment/recycling-waste/index.html`.

[4] Adam Conner-Simons. *Robots that can sort recycling.* URL: `https://news.mit.edu/2019/mit-robots-can-sort-recycling-0416`.

[5] Covanta. *Covanta Survey: Americans Don't Know How to Recycle.* URL: `https://www.waste360.com/recycling/covanta-survey-americans-don-t-know-how-recycle`.

[6] Maxime Decleire. *Sorter Bin - Detect and Sort Your Trash.* URL: `https://www.instructables.com/Sorter-Bin-Detect-and-Sort-Your-Trash/`.

[7] Kaggle. "Garbage Classification Dataset". In: (2018). URL: `https://www.kaggle.com/asdasdasasdas/garbage-classification`.

[8] DIY Machines. *Robotic Bartender.* URL: `https://www.thingiverse.com/thing:2478890`.

[9] Ashley Murray. *Single-stream recycling: A system that does not work.* URL: `https://newsinteractive.post-gazette.com/recycling/`.

Appendix A

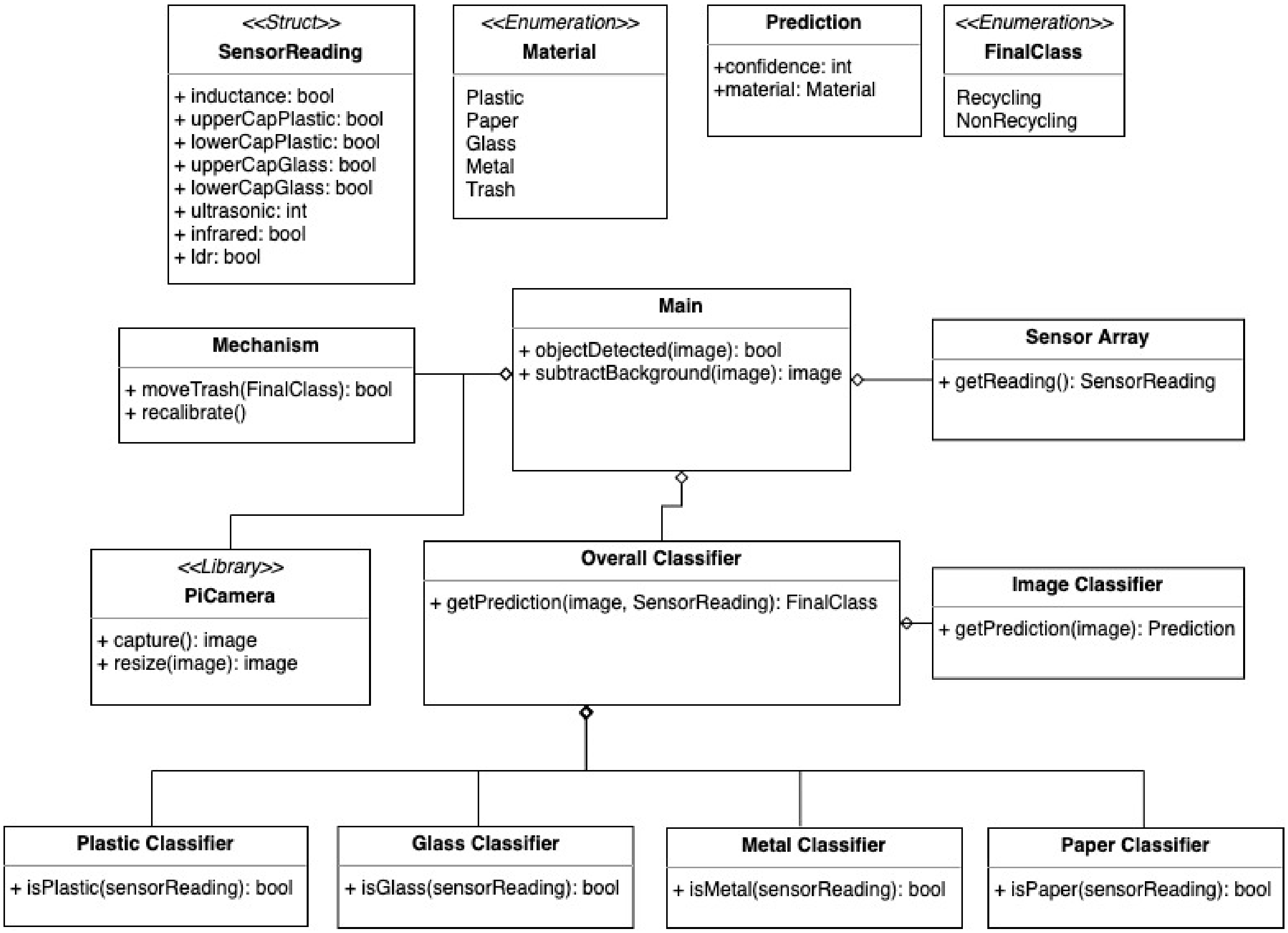

Figure 9: Class Diagram

Appendix B

	TASKS	February				March					April				May	
		Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Finals
		2/1	2/8	2/15	2/22	3/1	3/8	3/15	3/22	3/29	4/5	4/12	4/19	4/26	5/3	5/10
	Milestones				Proposal Presentation		Design Presentation	Design Report				Interim Demo			Final Presentation	Final Report/Demo
1	**Mechanism**															
	Finalize Mechanism Design				JT	JT										
	Order Mechanism Parts					T	T									
	Build Mechanism							T	T	T						
	Connect motors to Jetson Nano								J							
	Build sorting box							J								
	Build infrastructure								T	T						
	Test Mechanism										T	T				
2	**Sensor Classifier**															
	Order Parts				T			L								
	Connect sensors to Jetson Nano						J									
	Collect Sensor Data						All	All	All							
	Build Sensor Array							J	J	J						
	Build Model for Sensor Classifiers								L	LJ						
	Test Sensor classifiers								L	L	L					
3	**Image Classifier**															
	Find datasets					LJ										
	Research Models					LJ	LJ									
	Build Model							LJ								
	Train model for images								LJ							
	Collect Image Data									All						
	Manually train model									LJ						
	Test classifier (images)										LJ					
	Manually test classifer (images)										LJ	LJ				
4	**Integration**															
	Integrate camera with box								T							
	Integrate motion sensor								J							
	Integrate sensor with image classifier								LT	L	L					
	Test overall classifier (images + sensors)										L	L				
	Test overall (classifier + mechanism)											All				
	Slack time												All	All		
5	**Milestones**															
	Design Presentation					All										
	Design Report						All	All								
	Final presentation														All	
	Final report / demo															All

Figure 10: Gantt Chart

Appendix C

Component	Quantity	Cost	Total Cost	Total Spent	Source
Jetson Nano	1	99	99	0	Personal
Nema 17	1	13.99	13.99	13.99	Amazon
DRV8825 Motor Driver	1	8.95	8.95	8.95	Pololu
Inductive Sensor	16	9.99 for 2	79.92	79.92	Amazon
Capacitive Sensor	12	1	9.99	119.88	Amazon
LDR Sensor	16	5.88 for 5	17.64	17.64	Amazon
IR Sensor	4	8.68 for 5	8.68	8.68	Amazon
Load Sensor	4	7.95	7.95	7.95	Amazon
Timing Belt	1	11.99	11.99	0	Personal
Timing Gear	2	13.99 for 2	13.99	13.99	Amazon
Linear Rod	2	15	30	0	Personal
Linear Ball Bearing	4	13.74 for 4	13.74	0	Personal
LED strip	1	11.89	11.89	11.89	Amazon
Raspberry Pi v2 Camera	1	24.96	24.96	24.96	Amazon
Camera Extension Cable + Adapter	1	8.9	8.9	8.9	Adafruit
12V Power Supply	4	11.99 for 2	23.98	23.98	Amazon
Wood (4x8ft)	2	15.38	30.76	30.76	Home Depot
AWS			20	0	
		Total	**436.33**	**371.49**	

Figure 11: Budget