Boston University
Electrical & Computer Engineering
EC464 Senior Design Project

Full Functional Testing Report

Ecobin

by

Team 9
Ecobin

Team Members

Aditya Wikara adwikara@bu.edu
Hayato Nakamura hayaton@bu.edu
Kevin Sinaga kfsinaga@bu.edu
Shreenidhi Jayaram shreej@bu.edu
Charles Thao cthao19@bu.edu

Equipment

The required equipment for this prototype testing could be divided hardware, software, and any mechanical components. The ECE team provides the software and hardware while the ME team build any mechanical parts for the prototype.

Hardware

- Raspberry Pi 3 B+ (with 32GB SanDisk SDHC Class 10 card)
- Raspberry Pi Camera Module v2
- Wide Angle Camera Lens
- PIR Motion Sensor
- Ultrasonic Sensor HC-SR04
- LED Strips (12V, 0.9W)
- 5V/12V DC Power Supply Unit
- Transistor (PN2222A BJT)
- TB6612 Stepper Motor Driver
- Phidget Motor Controller
- PCB Breadboards/Protoboards

Software

- Object Classification (Machine Learning)
- Web Scraper (Collect datasets from Google Images)
- MongoDB Atlas (database)
- Python Scripts for sensors and overall Ecobin functionality
- IOS Application (10.2.0+, Swift 4.0+)
- REST API deployed on cloud instance
- Dependencies
    - Pymongo, base64, numpy, PIL, keras/tensorflow,httpauth,requests

Mechanical

- Object Detection Platform
- Ecobin Housing
- Camera Lens Mount (3D printed)
- 2 Stepper Motor (NEMA-17, and NEMA-17 + Gearbox)

Description of Equipment & Set Up

For this prototype, there are several additional equipment for both hardware and software. For hardware, there is a new power supply unit, a Phidget motor driver which controls a stepper motor, and an Arduino Uno board to control another stepper motor. In addition to the new equipment, the PIR motion sensor, ultrasonic sensor, LED strips, and camera lens from the previous prototype test will be tested again. The power supply unit has two DC outputs, 12V and 5V, allowing all the components in the Ecobin device to be powered using a single wall outlet connection. The PIR motion sensor detects motion to activate the Ecobin. The ultrasonic sensor measures distance, which will be used to measure capacity build-up in the receptacles. The LED strips will illuminate the enclosed Ecobin system. Lastly, the camera lens minimizes the distance required to take a picture, hence saving a lot of space. We have migrated parts of the circuitry from solderless breadboards to a protoboard, allowing for more permanent connections.

In terms of software, we applied transfer learning and fine tuning to the Keras pre-trained VGG16 model. VGG16 is a general object classifier, trained on Imagenet and can output 1000 classes. We modified it into an object classifier with 13 classes: Bottle, Carton, Compostable_Containers, Cup, Food, Fruit, General_glass, General_Plastic, General_Recyclable, General_Trash, Leafy_greens, Metal_can, Utensils. Each class consists of roughly 1000 images which we scraped from Google images or collected from ImageNet. This new model will be trained and run on the Shared Computing Cluster at Boston University. The current configuration includes modifying the last 4 layers of VGG16:

1. 1 Convoluted layer with 512 nodes
2. a Flatten layer
3. a Dense layer with 4096 nodes
4. DropOut layer with rate 0.8
5. Output layer of 13 nodes (for 13 classes).

In addition, the Ecobin system is now also interfaced with MongoDB. The RasPi sends d data in to the MongoDB database via the API and the Keras model retrieves this data and runs the sorting algorithm. This allows the RasPi to securely communicate with the cloud, and later also to the mobile application. Additionally, MongoDB is not stored locally, allowing access to the database from any device, creating flexibility.

In terms of the mechanical components, the Mechanical Engineering team constructed the final Ecobin housing made out of thick plastic. This housing has an automated lid which contains gears, shaft, and a stepper motor. This motor has a driver which is controlled by an Arduino. The Arduino will communicate with the RasPi via digital pins. They have also created the platform for the object recognition. This platform is attached to a sweeper which will mechanically push the object to the desired compartment. The sweeper is attached to the stepper motor and a gearbox. This motor has a driver which is controlled by the RasPi. The ecobin system can now mechanically sort the object to the desired compartment.

The general setup will be similar to previous prototype testing. The RasPi functions as the microcontroller of the Ecobin. The motion sensor detects motion and triggers the camera to take a picture of the object inside the enclosed box. However, the RasPi now sends the encoded picture to a RESTful API which has been deployed in a cloud instance. The picture is then retrieved by the backend, which classifies the object and updates the API. The RasPi then gets the class type of the object, either ‘trash’ or ‘recyclable’. The RasPi then outputs this signal to the stepper motor, which mechanically pushes the object to the desired compartment. Additionally, the data processed from backend.py will be pushed to the Mongo Atlas Database and the API which will respectively summarize and supply data to the iOS application . The information will then be displayed by the iOS application for the user to keep track.

Pre-testing Setup Procedure:

Software

1. The RasPi has internet connection.
2. Make sure the model is trained with the desired classes and the HDF5 file is ready.
3. Make sure the computer/SCC has all the dependencies.
4. Mongo Altas database set up to communicate from raspberry pi and SCC ML database
5. The API server 128.31.22.22 is up and running
6. The RasPi is interfaced with the API and the SCC

Hardware

1. Plug in power supply.
2. Make sure all the port and pin connections are correct and secure.
3. Boot up the Raspberry Pi
4. Run the python script, *ecobin.py.*

Testing Procedure:

1. Place an object on the detection platform(cardboard box), and wave hand over PIR motion sensor.
2. LED strip turns on
3. Lid opens
4. Image of trash is captured by the Raspi camera inside the enclosed space
5. LED strip is turned off
6. The image is then encoded to a string format, which is sent to the API
7. The machine learning code then pulls the information from the API that has the encoded string of the image taken by the Raspberry Pi camera.
8. The machine learning code then processes the image and returns a dictionary with key value pairs holding attributes such as the the object name, percentage accuracy, its classification type. This is done in the SCC. The backend also updates the API with the processed information.
9. The backend file connects to the Mongo Atlas database and stores the classification and accuracy information processed from the Machine Learning code
10. The API then receives the summarized information stored in the updated database.
11. The RasPi receives information from the API, whether the object is ‘trash’ or ‘recyclable’.
12. The RasPi outputs this signal to the stepper motor, which sorts the object.
13. The API then sends the information to the iOS application which presents interactive results to the user.
14. This process will be repeated 10 times with equal number of trash and recyclable objects. This data will be recorded in the score sheet, where the accuracy could then be calculated.

Graphical Representation

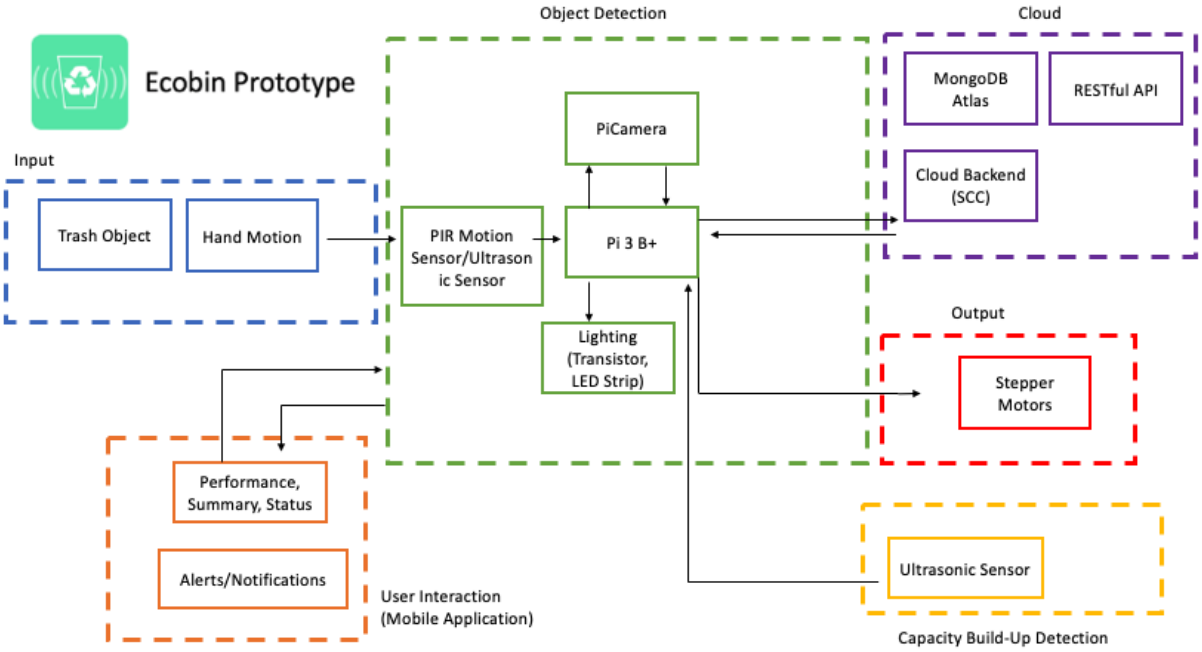

Figure 1: Block diagram of the Ecobin prototype

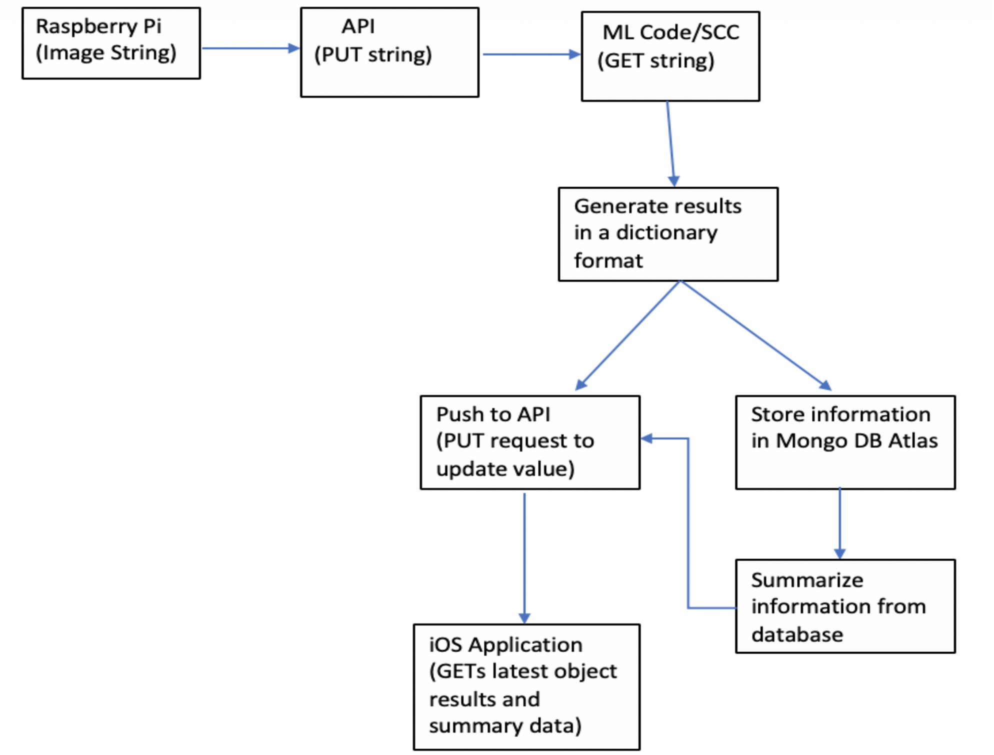
Raspberry Pi
(Image String)
API
(PUT string)
ML Code/SCC
(GET string)
Generate results in a dictionary format
Push to API (PUT request to update value)
Store information in Mongo DB Atlas
Summarize information from database
iOS Application (GETs latest object results and summary data)

Figure 2: Software block process diagram to demonstrate communication between the raspberry Pi , PyMongo Atlas Database, Shared Computing Cluster and iOS application using the API

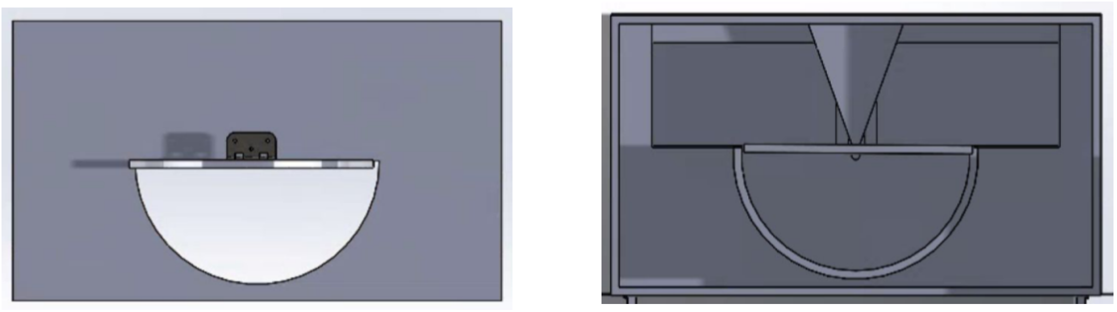

Figure 3: 2D representation of the Ecobin design (the classifier part). This is a top view. The actual design will have the same dimensions with the prototype but with different material.

Measurable Criteria

The criteria for successful running and output is as follows:

I. The Raspberry Pi should successfully capture an image and output whether it is trash or recyclable, as well as the confidence level (probability).

II. The object detection should be done in an enclosed space to make the testing environment ideal. This way, the picture will be taken in the dark where the object is illuminated by the LED strip. This is the type of picture that the Ecobin will receive for the final product.

III. The RasPi must be able to communicate to mongoDB server and the API in order to process the picture in the cloud/SCC. This process is deemed successful when the handshaking is completed in < 2 seconds.

IV. If there's motion above the PIR motion sensor, the Raspberry Pi should successfully take a photo of the object. The LED strip should also turn on when motion is detected, and turned off after the picture has been taken.

V. The Ecobin system should be able to mechanically push the object to the desired compartment.

VI. The Raspberry Pi should successfully classify whether an object is recyclable with **75%** accuracy

VII. Both encoding and decoding the 5Mp raspberry pi image in base64 binary format should be successful, with the resulting image being identical to the original image.

VIII. Our API key should successfully be able to execute GET and PUT requests depending on requests

IX. iOS Application successfully retrieves the latest waste information and displays it under the *Trash* page.

X. The *Statistics* page on the IOS application accurately displays the total waste count, trash waste count, as well as the recyclable waste count in the form of a pie chart.

Score Sheet

Test Object	**ObjectType**	**Category**	**Correct? (Y/N)**

Plastic Bottle	Bottle	**Recyclable**	
Box	Carton	**Recyclable**	
Shampoo bottle	Bottle	**Recyclable**	
Drinking cup	Cup	**Recyclable**	
NyQuil Bottle	Bottle	**Recyclable**	
Apple	Fruit	**Trash**	
Soda Can	Can	**Recyclable**	
Body wash container	General_Plastic	**Recyclable**	
Kiwi	Fruit	**Trash**	
Lemon	Fruit	**Trash**	
Spoon	Utensils	**Recyclable**	
Drink Can	Metal_can	**Recyclable**	
Fork	Utensils	**Recyclable**	
	Result →		%

Hardware Pinout

RasPi Pin #	Usage/Description
2	5V Power -> PIR VCC, HC-SR04 VCC
6	Ground
7	GPIO 04 -> PIR Output
14	GPIO 27 -> HC-SR04 Trigger
16	GPIO 23 -> HC-SR04 Echo
11	GPIO 17 -> LED strips Output
13	GPIO 19 -> Signal to Arduino