Divide and Conquer 2.0: Trash-E

Group Information

Group: 26

Senior Design 1 EEL 4914-0001

Due: October 1st, 2021

Members

Thomas Greco: Computer Engineering

Christian Mayo: Computer Engineering

Alex Rizk: Computer Engineering

Chrizzell Jay Sanchez: Electrical Engineering

Overview

In this paper, we discuss our design for a robot that detects trash and picks it up. It is meant to help automate the process of cleaning up a venue such as a ballroom, sporting event field, or even a backyard. The purpose of the project is to create an autonomous robot that can reduce the manpower required to do a tedious task such as cleaning up after an event.

Motivation

Litter is an increasingly difficult problem to address as the world progresses. Large amounts of money is spent to pay individuals to pick up leftover objects after many different events such as tailgates, and parties. Due to the chaotic nature of those situations, it is extremely difficult to govern and manage each individual and ensure every piece of litter is brought to a place to be disposed of. Therefore, venues choose to clean up after the event rather than take preventative measures. This leads to another problem that once litter reaches a certain size, it is too large to pick up multiple at one time. One must bend down, grab the litter, then put it in a receptacle for storage until they may properly dispose of it.

A general consumer's options for a device that completes this task autonomously is very limited. Based on our research, there is only a Roomba® style device that can be purchased. One problem with this style of robot is the size of objects it can pick up. Roomba®'s are designed with the intention of replacing vacuum cleaners that companies such as Dyson® and Bissell® manufacture. This means their goal is to pick up very small objects like crumbs, dirt, dust, and other similar things that can be collected without the use of the vacuum wand.

Another problem with the Roomba® is the capacity of the container it stores litter in. Even if a venue only had litter that was capable of being picked up by it, the container would get full too quickly and would require a person to empty it too often.

Customers, Sponsors, Contributors

For this project there are no customers or sponsors. The decision to do this project was unanimously voted by everyone in the group. Each member of the group will be individually contributing to the expenses of the project.

Budget

Split between four group members we've decided that $400 should give us enough wiggle room to complete the project. Some parts already owned by team members may be provided to further reduce costs.

Description

Trash-E needs to be fully autonomous which will require computer vision along with AI/Machine Learning for litter identification and object avoidance. This will utilize a camera and some type of mini-computer that will do the image processing and object detection. It also needs to have a way to move around either with treads or wheels which will also need some form of motor to move them. A container will be fixed to the top with enough space to allow multiple pieces of litter. A sensor will be used to determine the fullness of the container. Some options we have considered is a weight sensor that will be put under the container, or an ultrasonic or infrared sensor that will be used to tell the distance of the object on the top of the pile. The ultrasonic/infrared is a more comprehensive approach due to the weight of objects being different after expanding our scope of litter. The bin will be filled to 85-90% capacity to ensure the litter stays inside the container. An arm will be made to move the litter from the ground into the receptacle that's placed on the robot. This arm might be straight with a rotating "wrist", or a 2-segment piece that more resembles a human arm with a bend being in the "elbow". A claw to pick up the litter will be mounted on the end of the arm. The exact shape of the claw will be determined after analyzing more about the shape of our definition of litter. To move the arm and claw, motors will need to be utilized. To control all the motors, a microcontroller is needed. This will allow us to communicate with the separate motors to move the wheels and arms. A way to communicate between the mini-computer and microcontroller will need to be developed as well otherwise the robot won't be able to move and interact with the litter. Trash-E will be powered by a rechargeable battery. The final robot should be easily portable, relatively lightweight, and able to be transported from location to location.

How do we define litter?

Litter is a broad term, and to create a robot that can pick up all litter would not fit within the scope of the project. For this project, we must refine what litter is. To start off, we plan to use 16 ounce Red SOLO cups. These cups are 3 x 3 x 5 inches in size. We plan to use these cups because they are very popular at large events and often are discarded on the ground at events such as college parties. The cups are also a distinct red color and have a distinct shape, which will aid in easing the process of identifying the cups. Once we are able to distinguish the Red Solo Cups, we plan on moving onto 12 ounce aluminum soda cans. The reasons being similar to the Red Solo Cups, except the aluminum cans have more variety of color and design. If we are able, we may expand our scope of litter later on.

How do we define an obstacle?

In our use case, an obstacle is anything that can impede the pathing of the robot. Anything greater than the size of a Red Solo Cup, 3 x 3 x 5 inches, will be classified as an obstacle to avoid. We plan to use wheels big enough to roll over anything smaller than the size of a Red Solo Cup if necessary. As we run testing for obstacle avoidance, we may need to refine our algorithm for what counts as an obstacle.

Other obstacles we may need to consider are obstacles that are too big for the camera to see. Something like a wall may throw off the obstacle detection, and thus need to be considered when programming.

Chassis

The chassis of our robot is currently still up to debate. We are in the process of researching possible prebuilt options, or even 3D Printing our own custom platform. The advantages for 3D Printing our own chassis and platform is that it will be much cheaper to manufacture than to order one from a prebuilt site. However, a major disadvantage is that creating our own chassis will add in more mechanical work for the team to do, which adds more variables that can go wrong, as we are not mechanical engineers, there may be oversights that we may make. Another disadvantage is that the 3D printing filament that we use may not be strong enough to support the trash and bin and may crack during use.

The chassis of the robot must be big enough to hold a 2 gallon trash bin, and sturdy enough to carry all the trash, but small enough to be easily portable. The overall design can be found in Figure 1 below.

Context

Our robot may be useful in any event with a large gathering of people such as a concert, football game, or party. In these events the last thing on attendee’s minds is their trash and it is not uncommon for the venues to be filled with garbage after the event ends. These events have teams of workers who are dedicated to cleaning up all the trash that is left over. With larger venues, cleaning up may require upwards of 100 people. We aim to reduce the manpower necessary with this robot.

Maneuverability

One thing that we have to keep in mind as we build our robot is the terrain that the robot may encounter. If used inside, or on turf of a professional football field, the robot may not encounter much resistance when roaming around. However, the robot may encounter trouble when it is placed somewhere without an ideal flat surface. A beach is an example, as it is composed of different types of sand that is already hard for many vehicles to travel through. Another example would be a random park, as they are often not kept in the best conditions. These scenarios would be classified under Hard Terrain, and will be considered. One solution could be implementing treads similar to tanks, also known as continuous tracks, or using larger and fatter wheels, similar to fat tire bikes. Fat tire bikes are bikes that are designed for all terrain by having tires that are wider than normal bikes which provide more surface area.

Features

The features table is split up between initial, primary, secondary, and stretch goals. The initial and primary goals are what we want to make sure the robot does. The secondary are goals that we hope to accomplish, and the stretch goals are goals that are unnecessary but may be added if we have the time and budget. This table is subject to editing at any point, as we may find new constraints that may limit or even make a goal easier to achieve.

Table 1. Features

Description	Feature Type
Identify Red Solo Cup	Initial Feature
Move Around Freely	Initial Feature
Able To Pick Up Litter	Primary Feature
Move Arm To Litter And Bin Properly	Primary Feature
Identify Aluminum Can	Primary Feature
Identify And Avoid Obstacles	Primary Feature
Store Litter In Bin	Primary Feature
Solar Panels	Secondary Feature
IR Remote Control	Secondary Feature
Sense When Bin Is Full	Secondary Feature
Function For 1 Hour	Secondary Feature
Return Home When Bin Is Full	Stretch Goal
Maneuver Through Hard Terrain	Stretch Goal
Visual And Audio Cue When An Object Is Picked Up	Stretch Goal
Empty Bin Into Large Trash Bag	Stretch Goal

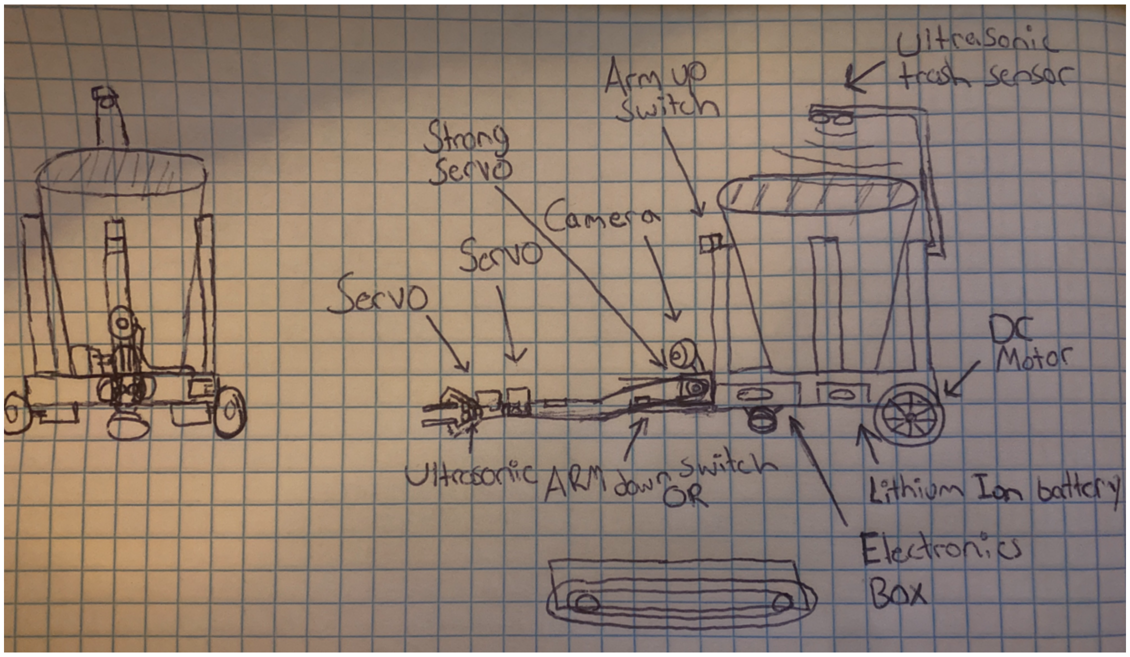


Figure 1. Project Illustration by Alex Rizk

Design Constraints

Economic

As mentioned above, the budget for this project is currently $400. Due to this price we may have to do more research into building things ourselves like the chassis rather than buying pre-built kits online. This can also limit the power of our hardware we choose, like opting for parts that we already own and are slower rather than buying a new, faster part.

Time

For this project we have two semesters to design, build, test, and present. This greatly impacts the sophistication of our robot as we have to make sure we can implement our idea in about 3 months. This can also affect the amount of primary and secondary features we get to. If something goes wrong or takes longer than expected, we will have to re-evaluate what we can get done in the remaining time.

Safety

If this robot is to autonomously operate in areas where human traffic is taking place there must be multiple failsafes in order to prevent harm to bypassers. This issue can be dealt with in several ways: rigorously training the machine learning model to differentiate between humans and litter, having ultrasonic sensors for collision detection, allowing for manual shutdown, operating at slow speeds, and having a small and lightweight design. Another option would be to only have the robot operate at times where human activity is minimal to remove any chance of harm. Further down the development timeline we plan to adhere to safety standards for autonomous robots.Overall, Trash-E would not have the capabilities to cause any sort of destruction to any environment.

Manufacturing

Currently we are limited to the following methods for making parts for our robot:

1. 3D Printing: Using the 3D printer in the Innovation Lab, or using the 3D printer that a group member owns.
2. Purchasing wood: Going to a home improvement store and getting the required wood to build into what we need. We also need access to a workshop that is provided by UCF.
3. Purchasing metal: This approach would be potentially most expensive and difficult due to not having the correct tools.

Standards and Protocols

Standards and protocols are typically used to explain and document the guidelines for manufacturers, business owners, engineers, and anything of a similar manner to follow when developing a product that is intended for public or private use. There are a myriad of protocols to look out for when in the developmental stages of a product. Furthermore, these protocols should be followed to maximize the accountability of a project in order to reduce the potential blame for not taking into consideration standards later in a project's lifetime. In the case of our autonomous litter cleaner robot various standards have been created by organizations like IEEE Robotics & Automation Society and IFR (International Federation of Robotics).

Serial Communication Protocols/Standards Considered

Here are our potential options of serial communication protocols that are being considered for use across the project for sensors, motors, peripherals,and communication between the microcontroller and mini-computer (Nvidia Jetson or Raspberry Pi)

1. **USB**
   The USB standard is vast and multi-layered with protocols that have been developed and improved by several major companies; to date, there are six different versions of USB. These versions are USB 1.0, 1.1, 2.0, 3.0, 3.1, 3.2, and 4. Within each of these versions are the connector types such as Type A, B, C, Mini A, B, Micro A, B, and Micro AB. For the scope of this project USB 2.0 or 3.0 with a Type A connector should be necessary in order to connect the camera peripheral to the Nvida Jetson or Raspberry Pi. Shown in Figure 3 is the difference in the pin connections we may encounter when choosing our USB peripherals, both of which are compatible with any USB 2.0 or 3.0 port due to the shared pins. USB allows us to easily implement a camera for computer vision to our design since it handles the configuration of DMA (direct memory access), interrupts, formatting, and selecting input and output addresses.

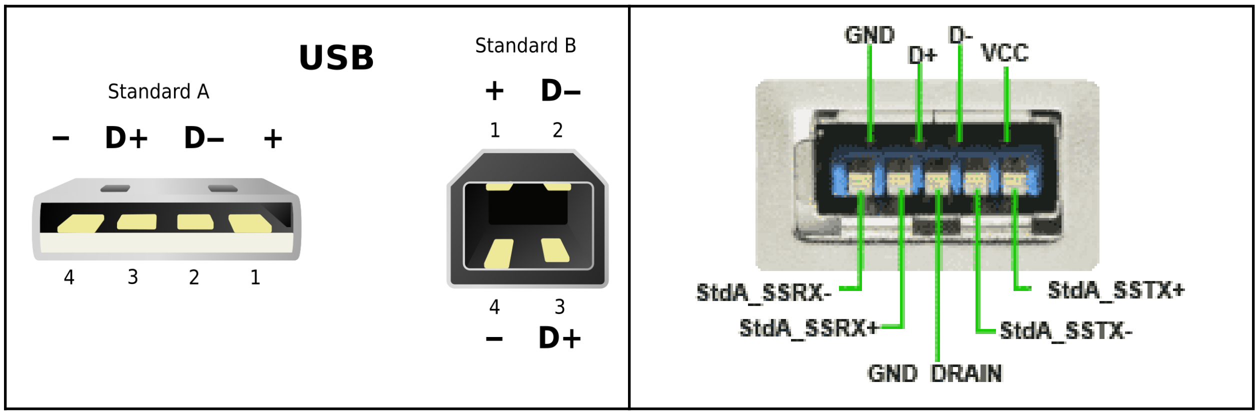


Figure 2. Type A USB 2.0 & 3.0

2. I2C

Can be used for communication between multiple masters and slave devices all through two wires that transmit a clock signal and data shown in Figure 4 below. Depending on whether a device is sending or receiving data an open drain allows for bidirectional communication through the data line. This can be useful in the instance where we want to control several motors through the use of two pins rather than occupying several pins on our microcontroller. On the other hand, I2C can be relatively slow compared to other serial communication protocols such as SPI since it's handling of multiple devices using only two wires causes overhead. The overhead is more specifically due to the fact that you have to send a start condition, then address the slave, send the data, and send a stop condition though depending on clock speeds you can increase the speed of this process.

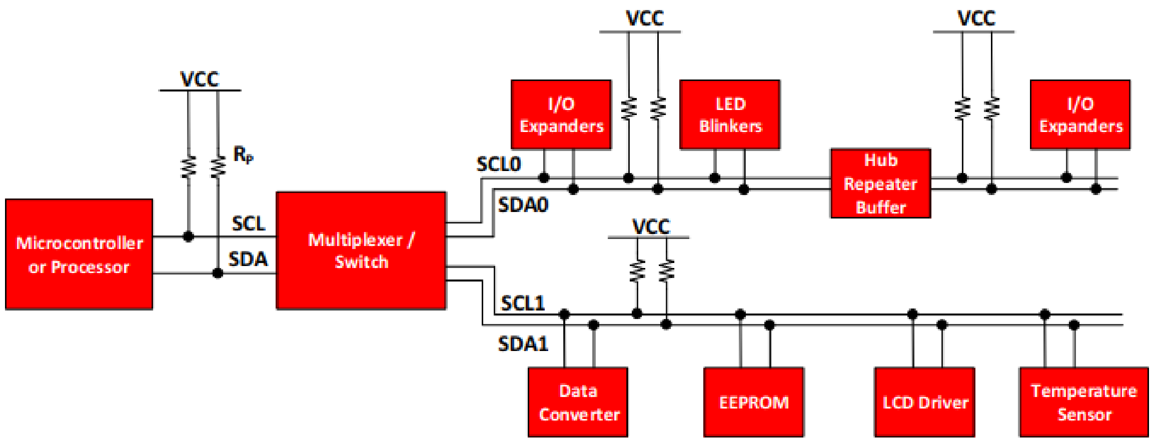


Figure 3. Example I2C Bus

3. SPI

Serial peripheral interface acts similar to I2C where you have a master that can individually select several devices that are connected through three plus the number of devices and the amount of pins. The three essential pins are SCLK (serial clock), MOSI (master out slave in), MISO (master in slave out) while the number of SS (slave select) or CS (chip select) depends on the number of devices you want to connect shown in Figure 5. The largest disadvantage is that as the number of peripheral devices increases the complexity increases, needing more SS pins. Despite the negative previously mentioned, SPI is one of the fastest protocols among the many other choices due to its simplicity and no need for overhead processing.

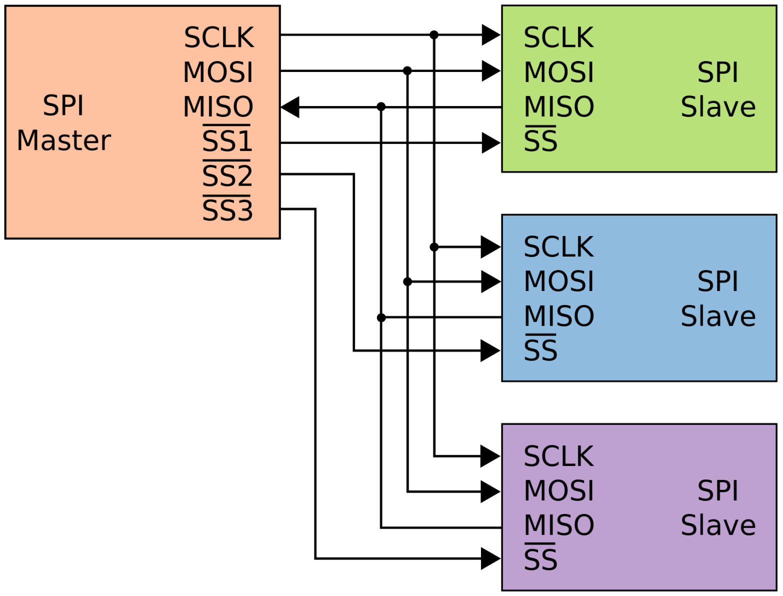


Figure 4. Example I2C Bus

Requirements

Formulated from our initial planning phase, we decided our initial project requirements that may be subject to change down the line due. Due to limitations with price and time we kept performance requirements and specifications relatively low as can be seen on Table 2. Furthermore, taking into consideration the environmental, manufacturing, cost, and time may positively affect our design forcing us to be efficient with our resources.

Table 2. Requirements

Description	Requirement
Maximum objects simultaneously grabbed	1
Maximum weight of object to be picked up	2 lbs
Maximum size of object to be picked up	3in x 3in x 5in
Time to retrieve object (Detection to bin)	30 s
Length of charge.	1 hr
Robot speed	5 Mph maximum
Robot size	Determined by size of garbage can
Success detection	Read from ultrasonic bin sensor
Amount of items in the can	85% bin capacity
Size of holding bin	2 gallons
Microprocessor	5V 2.5Amps
Microcontroller Power	5V 1A

Milestones

For the fall semester we have established milestones for the project. This semester will mainly be focused on research, documentation, and design as shown in table 3.

Fall 2021

Table 3. Fall Milestones

Week	Date	Milestone
1	8/23 - 8/29	☑ ~~Group Formation~~
2	8/30 - 9/5	☑ ~~Brainstorm Projects~~ ☑ ~~Choose Project~~ ☑ ~~Senior Design Bootcamp~~
3	9/6 - 9/12	☑ ~~Determine Project Requirements~~ ☑ ~~Discuss Project Budget~~ ☑ ~~Divide Project Block Diagram~~
4	9/13 - 9/19	☑ ~~Divide and Conquer 1.0~~
5	9/20 - 9/26	☑ ~~Decide Primary and Secondary Features~~ ☑ ~~Revise Project~~
6	9/27 - 10/3	☐ Divide and Conquer 2.0
7-12	10/4 - 11/14	☐ 60 Page Draft ☐ Research and order Microcontrollers and other parts ☐ Research Software ☐ Begin PCB Design ☐ Decide Pre-built or Custom Chassis
13	11/15 - 11/21	☐ 100 Page Draft
14-16	11/22 - 12/12	☐ Final Submission

Spring 2022

As the spring semester approaches and more design specifics are fleshed out, the time frame will be determined. Table 4 below shows a general outline for milestones of the semester.

Table 4. Spring Milestones

Week	Date	Milestone
TBD		Understand component Implementation
		Construct Hardware
		Create Software
		Project Testing
		Project Website
		Final Presentation
		Final Report

Block Diagram

The block diagram in Figure 5 demonstrates the general components we will use, who is responsible for acquiring and researching them, as well as the state of the acquisitions.

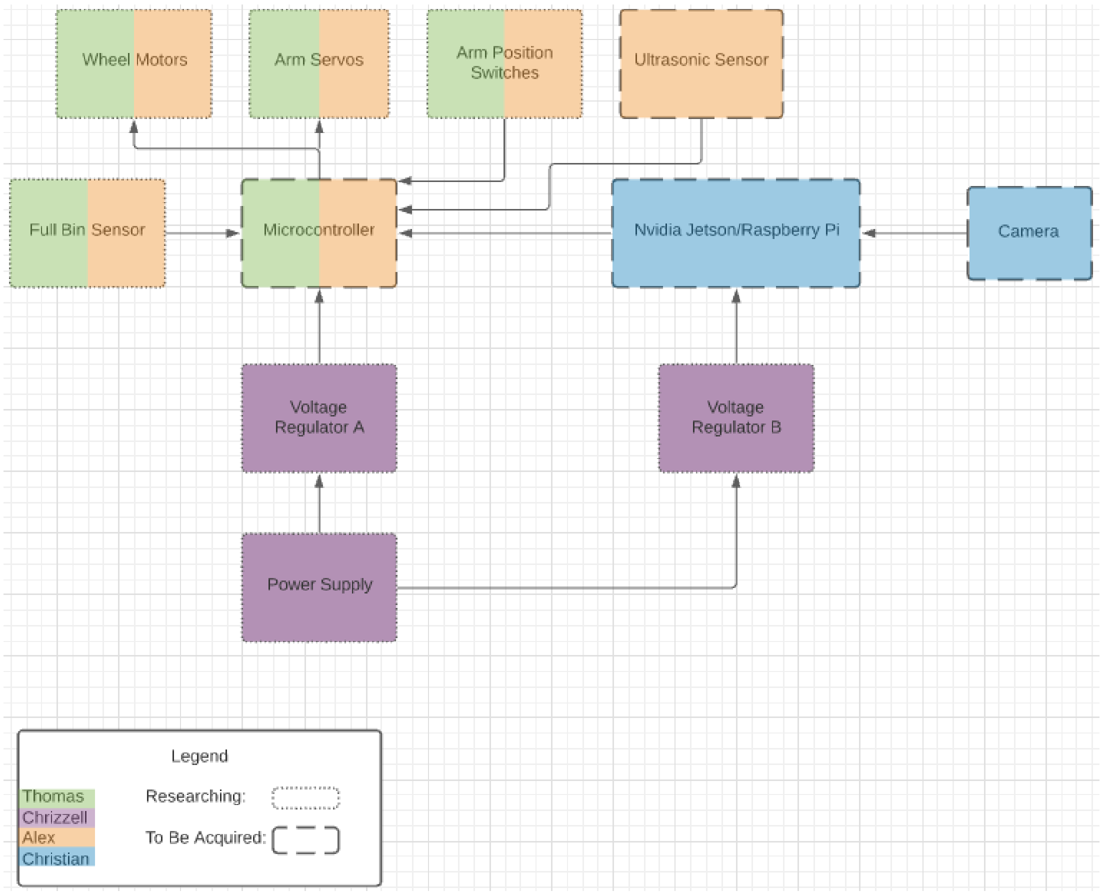


Figure 5. Project Block Diagram

Computer Vision

Why do we need computer vision?

Trash-E should be able to maneuver and spot trash and pick it up on its own. Trash-E will be roaming on its own until it finds a piece of trash. We identified earlier what trash items Trash-E will be identifying. Trash-E must be able to recognize these trash items and maneuver its way towards each object. However, spotting and identifying the correct trash objects is only the first step. Trash-E will have an arm with pincers that it will use to pick up the trash. It will need to be able to decide how it will approach the pickup of the item. It should be able to precisely grab the item and put it into its trash bin. To do this, we are looking to use computer vision with our robot Trash-E. Computer vision is a field of AI that can allow our robot to see. Meaning that it will be able to derive meaningful information just from digital images or video. With that information, Trash-E can identify trash objects and figure out how it will be grabbed.

How will Trash-E use computer vision?

Trash-E will have a camera connected to its main computer. A camera will be used to capture images of what is in front of Trash-E. The camera will be capturing many images. The computer on Trash-E will be powerful enough to process the images so that Trash-E can scan and identify trash objects in its view in real time. Once a trash object is detected it will move towards that object and pick up the trash. Trash-E should be able to figure out where the object is in real space and should move its arm accordingly to be able to pick up the object and move it into its trash bin.

How does computer vision work?

For computer vision to work, a lot needs to get done before it is used for cases like Trash-E identifying trash in real time. Computer vision requires a lot of data and would need to use machine learning techniques to accomplish it. It needs to analyze a lot of data and learn from it until it can make certain distinctions in images and ultimately recognize what it needs to find in an image or video. Figure 6 is an example of what a computer sees after identifying objects via computer vision.

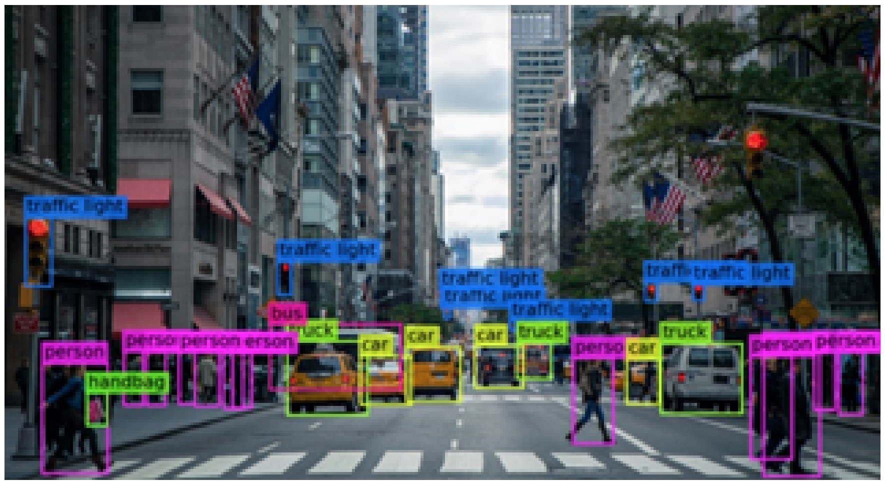


Figure 6. Object Detection Using Computer Vision

To accomplish computer vision, we need certain algorithms that can learn from given inputs and produce an outcome on its own in a way that the human brain would. The best way to accomplish this is to use deep learning. Deep learning is a subset of machine learning that uses neural networks to learn large amounts of data through lots of training. The most tried and true deep learning algorithm for computer vision is a convolutional neural network. A convolutional neural network should be the best algorithm to implement for Trash-E to detect trash objects using image recognition. Convolutional Neural Networks (CNN) are deep learning algorithms that can take an input image and then learn various patterns and features about that image and make decisions about the image. Figure 7 is a diagram of a convolutional neural network showing its general architecture.

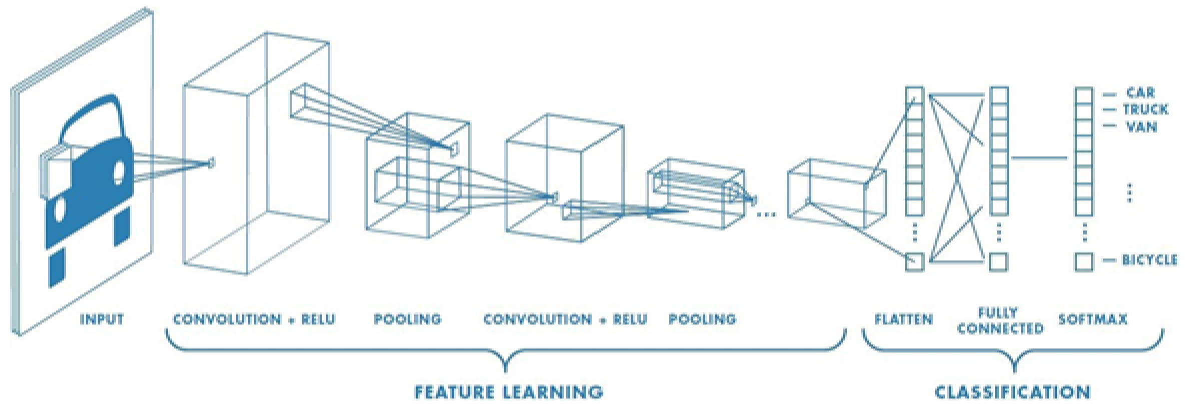


Figure 7. Convolutional Neural Network Architecture

An image is fed into a CNN in the form of a matrix with pixel values. In the convolution layer a kernel/filter, which is a square matrix, is typically used to hover over the original image in a certain number of shifts and strides. Each time the filter is over a new section of the image matrix, a matrix multiplication is performed and produces a new value for the image matrix. Depending on the kernel/filter values, different types of high-level features can be extracted from the input image. Following convolutions, the pooling layer follows which is responsible for further extracting dominant features and reducing the spatial size of the convolved feature. After the convolution and pooling layers, we have enabled the model to understand the features in the input image. Now, the output is flattened and fed into a normal neural network for classification. This layer is referred to as the fully connected layer and is where the network learns the many features we have extracted from the image. The output is then flattened and fed to a feed-forward neural network. After a certain number of epochs, the model can distinguish between features of the input image and classify them using the SoftMax activation function. This function will decide based on multiple probabilities what the image likely contains and outputs the best result(s).

Convolutional Neural Network Architectures

There are many CNN architectures out there. Some of the notable ones are: LeNet, AlexNet, VGGNet, GoogLeNet, ResNet, and ZFNet. Each are different takes and variations on the general CNN architecture shown earlier. For Trash-E, we have many options available to choose for the CNN architecture and would likely involve more experimentation and fine tuning down the line to decide which architecture would return the best results. We could also opt for designing our own CNN architecture for Trash-E if none of the available architectures give us satisfactory results.

Programming Languages, Libraires, and Technologies

We have many options for the programming language we could use for writing Trash-E’s software. Several programming languages are used for AI and machine learning nowadays. Some of the popular ones are Python, C/C++, and Java. The most popular of these languages for AI and Machine Learning is by far Python. For computer vision and robotics, the most popular languages to use are C/C++ and Python.

Python

Python is one of the most supported machine learning languages out there. It is a relatively simple language in comparison to C/C++ and Java. Python has an extensive number of tools and libraries for machine learning such as TensorFlow, scikit-learn, PyTorch, and Keras to name a few. These libraries support computer vision and deep learning allowing the ability to easily create convolutional neural networks and train them.

C/C++

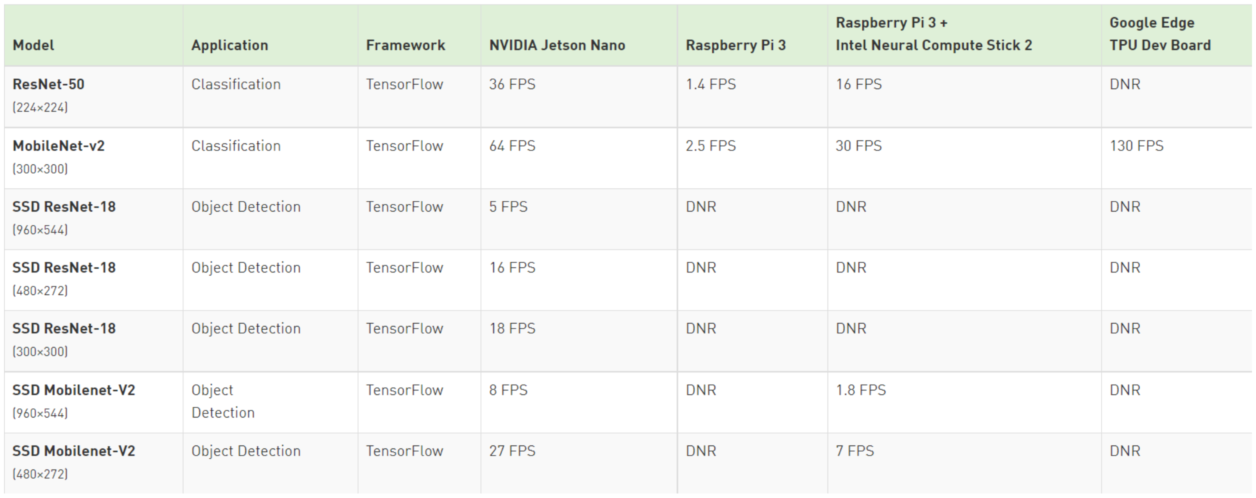
These languages are very much used in embedded and robotics programming. The areas they’re used most for in AI are gaming and robot locomotion. They’re not as simple as python when it comes to building new machine learning applications and getting what you want quickly. However, they are favored when control, high performance and efficiency is needed. C/C++ have some libraries for machine learning and computer vision such as MLPACK, SHARK, and OpenCV.

Java

This language is less popular for embedded and robotics and is used more for desktop and enterprise applications. Java does come with a decent amount of machine learning libraries such as TensorFlow, Deep Java Library, Kubeflow, and Java-ML.

Hardware Options for Machine Learning

Our robot will be using computer vision in real-time. To process the data and algorithms we need for computer vision and deep learning we would need a minicomputer on our robot capable of handling these tasks. These tasks include object detection, and classification. There are several out on the market such as the Nvidia Jetson Nano, Raspberry Pi 3, and Google Edge TPU. Figure 8 shows benchmark comparisons for these products. The Nvidia Jetson Nano appears to drastically outperform the other two boards in this comparison.



Model	Application	Framework	NVIDIA Jetson Nano	Raspberry Pi 3	Raspberry Pi 3 + Intel Neural Compute Stick 2	Google Edge TPU Dev Board
ResNet-50 (224×224)	Classification	TensorFlow	36 FPS	1.4 FPS	16 FPS	DNR
MobileNet-v2 (300×300)	Classification	TensorFlow	64 FPS	2.5 FPS	30 FPS	130 FPS
SSD ResNet-18 (960×544)	Object Detection	TensorFlow	5 FPS	DNR	DNR	DNR
SSD ResNet-18 (480×272)	Object Detection	TensorFlow	16 FPS	DNR	DNR	DNR
SSD ResNet-18 (300×300)	Object Detection	TensorFlow	18 FPS	DNR	DNR	DNR
SSD Mobilenet-V2 (960×544)	Object Detection	TensorFlow	8 FPS	DNR	1.8 FPS	DNR
SSD Mobilenet-V2 (480×272)	Object Detection	TensorFlow	27 FPS	DNR	7 FPS	DNR

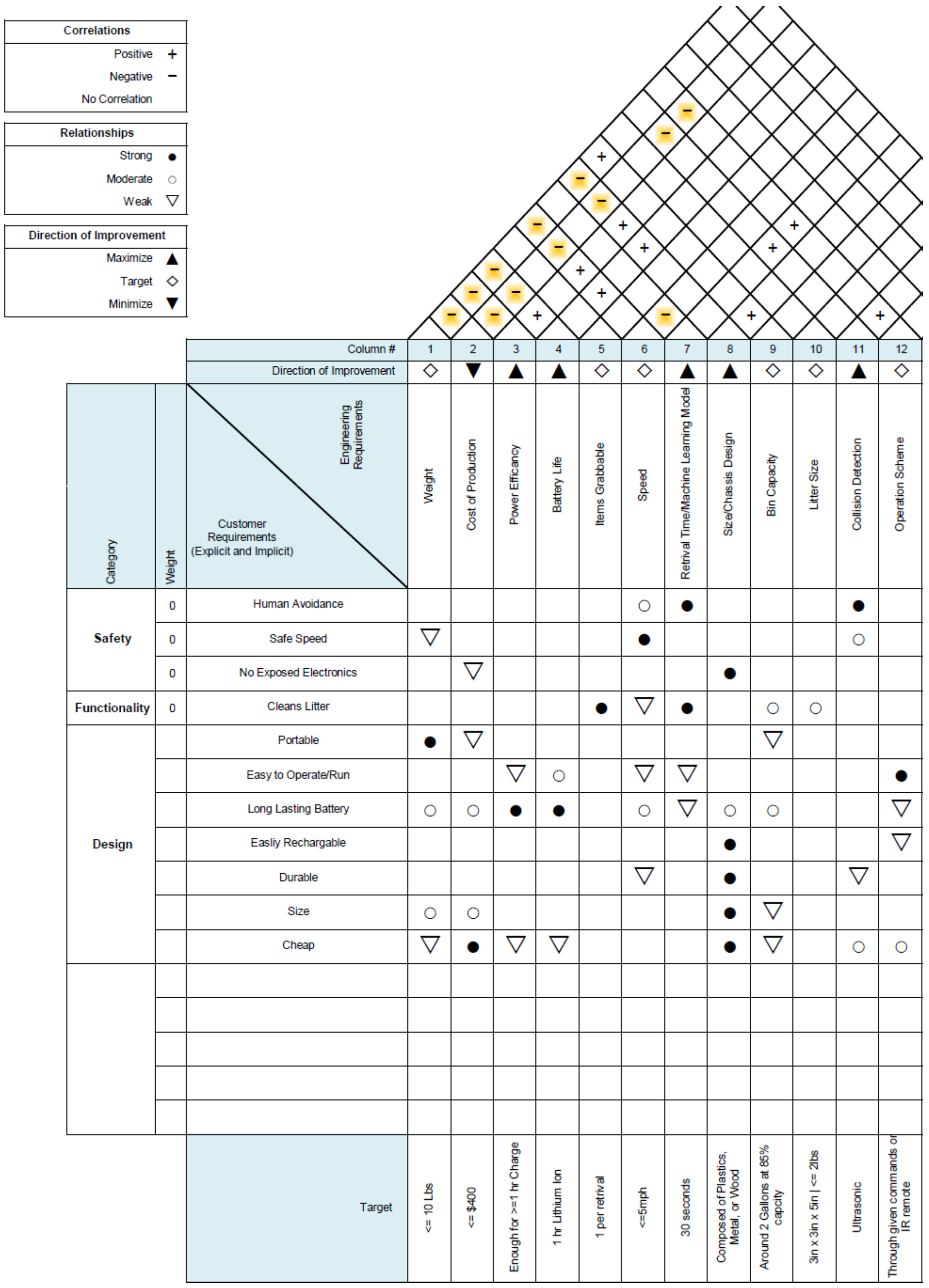
Figure 8. Minicomputer Deep Learning Benchmarks by Nvidia

Communication Between Mini-Computer and Microcontroller

With computer vision being done on the mini-computer and the microcontroller being in charge of the motors, a way to communicate between the two is required. Both have GPIO pins that will be utilized to send bits from the mini-computer to the microcontroller. We will need to determine a system for decoding the information sent onto the microcontroller. On the other hand, given a mini-computer with a large amount of processing power, such as the 4GB Nvidia Jetson nano it would be entirely possible to perform all embedded operations as well as autonomous functionality just from the mini computer. Taking into consideration the capability of the GPIO pins on the Jetson, only two pins can create a PWM signal, but with external servo and motor drivers we can generate more than enough PWM signals to drive hundreds of motors. If we decide to go with the single mini computer design it would reduce the complexity of electrical components (removing the need for a microcontroller), save space, and reduce weight on the chassis. The largest downside to having just a Jetson may be the increased power consumption and the need to come up with a system to manage heat. Overall, I believe that this will be an important deciding factor in the design of the robot.

House of Quality

Shown in Figure 9 is the house of quality that displays the customer requirements and its relation to the engineering requirements. The top of the figure represents the positive/negative correlations between requirements, whether their effects on other requirements hinder or improve them. In the bottom part of the figure are additional target descriptions about the engineering requirements.



Correlations	
Positive	+
Negative	−
No Correlation	

Relationships	
Strong	●
Moderate	○
Weak	▽

Direction of Improvement	
Maximize	▲
Target	◇
Minimize	▼

		Column #	1	2	3	4	5	6	7	8	9	10	11	12
		Direction of Improvement	◇	▼	▲	▲	◇	◇	▲	▲	◇	◇	▲	◇
Category	Weight	Customer Requirements (Explicit and Implicit) / Engineering Requirements	Weight	Cost of Production	Power Efficancy	Battery Life	Items Grabbable	Speed	Retrival Time/Machine Learning Model	Size/Chassis Design	Bin Capacity	Litter Size	Collision Detection	Operation Scheme
Safety	0	Human Avoidance						○	●				●	
	0	Safe Speed	▽					●					○	
	0	No Exposed Electronics		▽						●				
Functionality	0	Cleans Litter					●	▽	●		○	○		
Design		Portable	●	▽							▽			
		Easy to Operate/Run			▽	○		▽	▽					●
		Long Lasting Battery	○	○	●	●		○	▽	○	○			▽
		Easliy Rechargable								●				▽
		Durable						▽		●			▽	
		Size	○	○						●	▽			
		Cheap	▽	●	▽	▽				●	▽		○	○
		Target	<= 10 Lbs	<= $400	Enough for >=1 hr Charge	1 hr Lithium Ion	1 per retrival	<=5mph	30 seconds	Composed of Plastics, Metal, or Wood	Around 2 Gallons at 85% capcity	3in x 3in x 5in | <= 2lbs	Ultrasonic	Through given commands or IR remote

Figure 9. House of Quality

Sources

- Figure 2:
  - USB 2.0 https://commons.wikimedia.org/wiki/File:USB.svg
  - USB 3.0 https://www.aggsoft.com/usb-pinout-cable/usb3.htm
- Figure 3: https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1633022682208&ref_url=https%253A%252F%252Fwww.google.com%252F
- Figure 4: https://commons.wikimedia.org/w/index.php?curid=1476503
- Figure 6: https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e
- Figure 7: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- Figure 8: https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks