TrashPorter – Autonomous Trash Robot

Dhanesh Singh, AnnaBelinda Zhou, Jourdan Callahan, Andy Kuang

Dept. of Computer Science and Electrical Engineering, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — Autonomous robots are a growing asset in people's daily lives to make essential duties more convenient. This paper will focus on the methodology on the mechanics of the components that contribute to the system integration. The robot consists of multiple DC motors that run on a 12 V 2A power source. These components are managed through our printed circuit board which acts as one of many powerful microcontrollers. The intelligent mounted camera on the robot processes the data it visualizes using computer vision that takes in different colors and shapes to identify and learn objects through a machine learning algorithm.**

***Index Terms* — DC motors, Autonomous robots, System Integration, Microcontrollers, computer vision**

I. INTRODUCTION

The goal of the autonomous robot is to move from one stationary area to another when directed so as to collect trash from a desired location. This can be achieved in two ways, the first being manually controlled, and the other through an autonomous mode, both methods are done through a mobile application that connects to the WI-FI module on the microcontroller.

Starting off with the idea of the project, it can be easily inferred that this project was inspired by other devices that are currently popular on the market as within the recent decade there has been an increase of autonomous robots within people's homes. This brought up different variations of that idea and seemed like a robot to collect trash from rooms is quite unique. Combining the building of an autonomous vehicle as well as the collection of disposable items makes this project a good challenge for the knowledge the team has gathered throughout the years.

Determining the correct motor to use for this project is crucial for the success of getting the robot to function and traverse correctly. The load of the robot would have to be calculated and be able to handle the max capacity, as well as having enough power to each motor to operate in all conditions. Next, the motor drivers will have to be picked to handle the amount of current drawn from the power supply as well as managing the heat that is generated from the motors. The motors mounted on the device will control the wheels, which are made from wood that have been laser cut, these wheels will be covered in an insulated material to increase the amount of traction for it. This is all mounted on a base plate that gives the main body more stability as well as more volume to mount other components like motors and circuitry.

For a device as sensitive as this, many precautions need to be installed and monitored so it can know when it has reached its limit on capacity. For this issue there will be a weight sensor to determine when the maximum load has been reached. The scale will be installed at the bottom of the bin where it will take a zeroed-out value, and once the max capacity weight has been reached it will flag that it cannot carry any more items.

For the robot to determine where objects are there will be a mounted camera that reads in the input of its surroundings and learns what targets are considered important. This camera uses a machine learning algorithm that looks for the shape, size, and color to distinct one object from another. The camera can learn and remember multiple objects, and once learned, it can recognize the object from a long distance away. The most important trait that makes these processes as successful as possible is the distinct color, similar colored and sized objects can possibly be mistaken for what it thinks is the learned object, which could confuse the computer vision device. Using the information gathered by the camera, we can have it work with the microcontroller and motor to have the robot start moving towards the desired target being the trash it needs to collect.

The microcontroller is the foundation of the operations for this robot. It processes all the data that it receives from the environment to determine what it should do, then communicates to the other components to execute the task at hand. A printed circuit board was produced to take the place of the microcontroller and has the components that are quite similar to one, it has the motor drivers, heat sink, processor, WI-FI module, and many more connective components to make this run smoothly.

II. STANDARDS AND CONSTRAINTS

There were specific standards and constraints that were needed to be followed to determine that our goal was reached for this project.

A. Standards

The motor drivers that were installed use a different standardized library on our IDE because of the use of multiple motors functioning concurrently. This new library needed to be learned from the standard library for

motors as the logic will be the same, but the method of the functionality is slightly different. The motors will communicate with the circuit board through SPI communication.

B. Constraints

Some of the constraints that are needed to be considered are supply shortages within the current market, this will increase price and delays as parts are not readily available. These delays are difficult on the project because there is a limited amount that this needs to be completed by. Referring to the project, a factor that needs to be considered is the total weight that the robot can carry as there is a threshold it can physically hold for capacity as well as the limitations of how much weight it can hold.

III. HARDWARE IMPLEMENTATION

A. Stepper Motors

Under max load (<= 15 lbs), the Trashporter should move at a speed of 4 inches per sec, which is about 0.10 m/s. To achieve this the necessary calculations were performed.

1. Desired Speed: v = 4 in/sec = 0.10 m/s
2. Max Weight Under Load: M = 15 lbs = 6.8kg
3. Wheel Radius: r = 0.05m
4. Wheel Speed: = vr = 0.10 m/s2(0.05m) = 0.32 rps = 19 rpm
5. Thrust Force: F = M * g = (6.8kg)(9.81 m/s2) = 66 N
6. Torque: T = F * r = (66N)(0.05m) = 3.3Nm
7. Power: P = ωT/9550 = (19 rpm)(3.3Nm)/(9550) = 0.0065KW = 6.5 watts

The wheels radius should be about 0.05m (3) and under full load (2) the speed of the robot should be about 0.10 m/s (1). To achieve this speed, the wheels must complete about 19 rotations per minute (4). To overcome the thrust force/weight of the robot, which is about 66 Newtons (5), a torque of 3.3 Newton-meters must be applied on the drive wheels (6). To apply the 3.3 Nm torque on the wheels at a sufficient speed of 0.10m/s, a stepper motor rated at a minimum of 6.5 watts (7) should be used. The STEPPERONLINE stepper motors used to move the Trashporter is rated for 2 amps running at 12 Volts totaling 24 watts of power, which is enough to move the Trashporter.

B. The Battery

A robot tethered to a wall for power is quite inconvenient, and therefore a portable power source is required. A 12 volt battery source will provide enough power for the Trashporter. The SPARKOLE 12 volt battery pack is a battery pack that utilizes small cylindrical battery cells wired in series to create 12 volts. The SPARKOLE battery pack also comes with a BMS, which manages important features for batteries, such as overcharge protection, over discharge protection, cell balancing, and short circuit protection. Without these features the battery may fail and can cause catastrophic damage. The Trashporter draws about 2.2 amps of current under max load, and with the 5200 mAh capacity the device should have about 2.3 hours of uptime before the battery needs to recharge.

C. Wheels

We chose to use wood for the wheels as they can be expensive to purchase. We laser cut out many prototype versions ranging between three inches to fives and a quarter inches in diameter as we had to see how they fit with the base of our robot and made sure they are in balance with the two free back wheels that are purchased. We also had people cut out thick wheels using a metal split saw. However, their size is not the desired size as they can only be as big as three inches. A few of the prototypes can be seen in the pictures below. They vary in diameter and thickness. Since the laser cutting machine can only work with materials under the thickness of a quarter inch, we decided to laser cut out four of the five and a quarter inch wheels as the final two front wheels of the robot. Two of the wheels are screwed together to act as one wheel. In this way, it will be strong to hold more weight and accelerate in the correct manner rather than having the wheels slant outward with heavy weights.

Figure 1: Laser Cut Wheels

Figure 2: Metal Split Saw Cut Wheels

D. Weight Sensors

Weight sensor is a weight transducer that converts the mechanical input such as load, weight, tension, compression, force, torque, or pressure into electrical output signals (Futek). There are ten types of weight sensors. They are photoelectric, hydraulic, electromagnetic force, capacitive, magnetic pole variation form, vibration, gyro, resistance strain, annular plate, and digital weight sensors (R, 2020). To select the right weight sensor for this robotic design, we had to consider how we are measuring the weight of the trash on the robot, the way of mounting the sensors, the minimum and maximum weight capacity, the size and weight of the sensors, and the output type of the sensors. First of all, the weight sensor has to be mounted at a fixed position on the robot. The weight capacity of the weight sensor should be ranging from zero to approximately fifty pounds. The dimensions and the weight of the weight sensors have to be small so they can be easily placed on the chassis of the robot. In this robotic project, resistance strain weight sensors, also known as load cells, are used. They are selected because they are commonly used, their price is cheap, and the considerations mentioned above are met.

The specific weight sensors we chose are called 50 kilogram half-bridge strain gauge load cells. It is set up like a bathroom weight scale with four sensors on each corner of the scale. Thus, we laser cut a rectangular shape out of an acrylic sheet to use as the platform of the weight scale like shown in Figure 3.

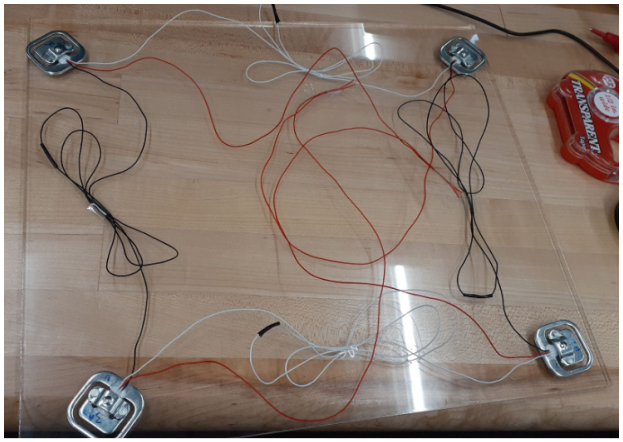


Figure 3: Weight Scale Layout

To mount the four sensors to the acrylic platform with bolts and nuts, we created frames for the four sensors as can be seen in Figure 4. The frames are 3D printed using Ultimaker 2 printer with the Cura software. Each frame used approximately one and a half hours to print. Thus, it took two days to 3D print them.

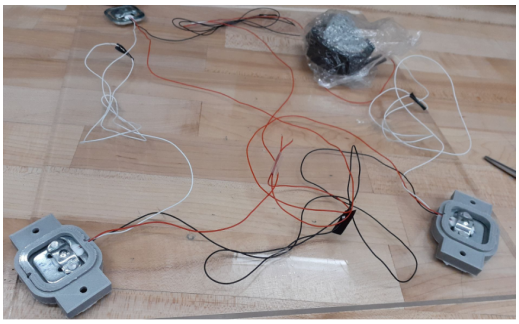


Figure 4: Weight Sensors with Frames

To connect the weight sensors to ESP32 microcontroller, the HX711 amplifier analog-to-digital converter is used. The wiring between the weight sensors and the HX711 amplifier is shown in Figure 5 below. The wiring from the HX711 amplifier to the ESP32 microcontroller is shown in Figure 6 below.

While working with the weight sensors we noticed that when calibrating the scale we would get different readings of the weight in grams. We saved values that gave us the most accurate reading by using an iPhone 12 pro max. Since we knew the weight of the device we were able to use that as a measuring tool to help us with our process of calibrating the scale. We first placed it on one corner then calculated the calibration factor. We then repeated this process with each corner until we got our desired calibration.

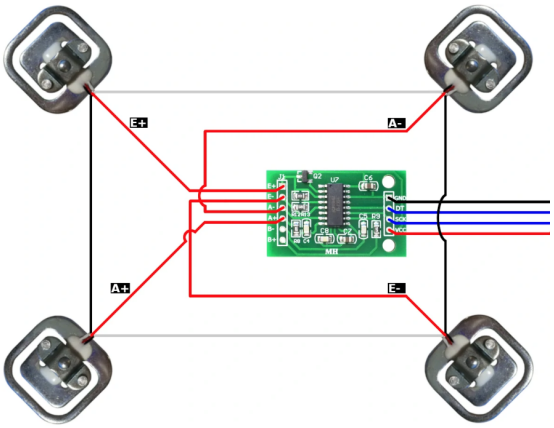


Figure 5: Wiring between Weight Sensors and HX711 Amplifier

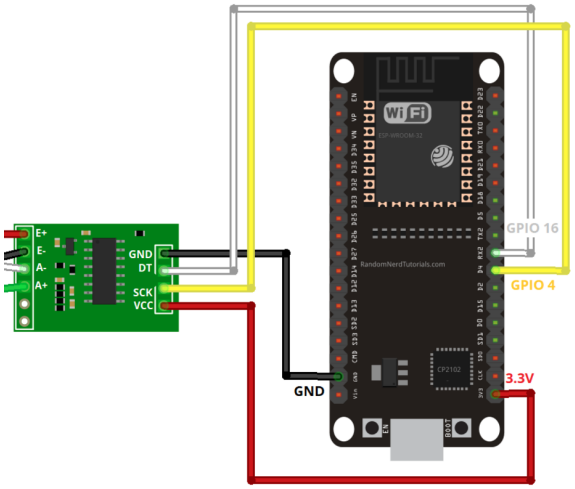


Figure 6: Wiring between HX711 Amplifier and ESP32 Microcontroller

E. Object Detection Sensors

In general, there are seven types of technologies to sense and detect objects. They are electro-mechanical, pneumatic, magnetic, inductive, capacitive, photoelectric, and ultrasonic sensors. Electro-mechanical sensors use mechanical actuators to detect objects by switching the states [1]. We chose to use the HC-SR04 ultrasonic distance sensors due to the fact that we have worked with it before and we already have the sensors. The HC-SR04 distance sensor is used to detect objects or hands less than 50 cm on top of the trash can on the trashporter. Whenever a hand is detected, the lid of the trash can will open to throw trash in. We used Arduino Uno to control the sensor rather than the ESP32 because of the limited pins left on the ESP32. In this way, we can also have it constantly checking for objects. The wiring between the HC-SR04 and Arduino Uno is shown in Figure 7 below.

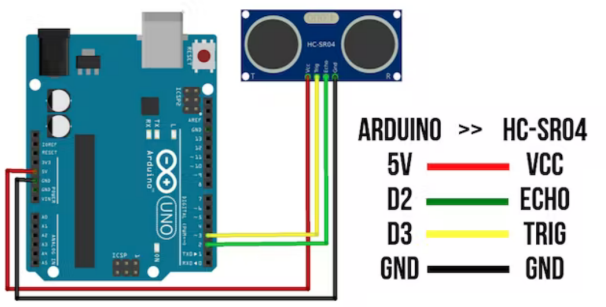


Figure 7: Wiring between HC-SR04 and Arduino Uno

IV. SOFTWARE IMPLEMENTATION

Our trashporter use the pixy 2 camera to be able to identify and follow an object (a smaller bin) and create a mobile application using the Blynk software which will allow us to move the trashporter manually and send it commands within a time frame.

A. Autonomous Functionality

The Pixy2 is a smart camera sensor with an onboard processor that will allow us to use robot vision to identify any object with a solid color on a specific spectrum. Pixy2 uses a color-based filtering algorithm called color-connected components to detect objects. To get a visual of what the pixy cam sees we use a software called pixy mon. The Pixy2 calculates the color and saturation in each pixel to filter out the object from the image. Pixy Cam shows a square around the object which we can use in our code as it gives us information such as the x and y-axis of the upper left corner of the square, the height and width of it, and an identifier. The identifier is used if we have multiple objects we want to track at the same time (up to six). In our demo, we will only be using one.

In our code, we used the x, y, and height to do x-based tracking. We will track if the x position is in the left field of view or the right field of view. To do this we established the range of the field of view of the camera (0-316). Then we broke this into three segments: left, right, and center. If the object is to the left of the field of view, we turn the robot left; if it is right, we turn it right. If the robot has the object in the center of its view, it will move forward towards the object and stop until it reaches a certain height of the square around the object. The x position will control the speed of the left and right stepper motors.

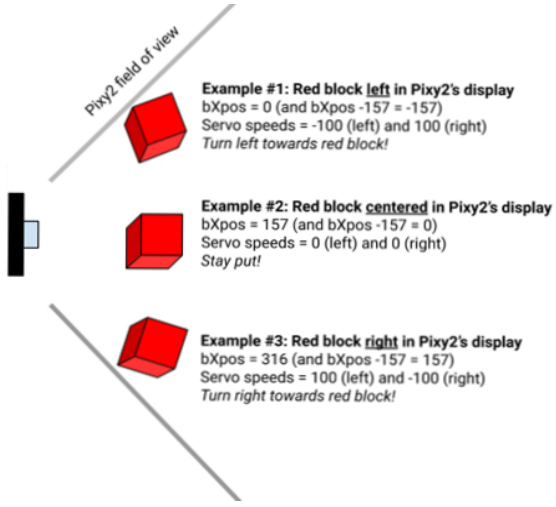


Figure 8: Demonstrates how the Pixy2 camera perceives the X position along with the calculations to determine whenever to turn the device left, right, or stay out.

B. Mobile Application

We also created a mobile application that will be used to control the trashporter robot. The app was created in Blynk, which gives us a platform to create a mobile application and communicate to our microcontroller via wifi by using digital or virtual pins. We used this to control our robot and send commands. We also can track the capacity of our bin using our weight sensors.

To set up the Blynk application, we had to set up our wifi connection with the microcontroller. In our case, we decided to use the esp32, a microcontroller with wifi capability. On our computer, we established this connection by using an authentication key generated by the Blynk website and inputting our wifi credentials. Once this was done we were able to set up our "data stream". The data stream is data that we will be sending back and forth to our microcontroller. Using these streams, we are able to create widgets such as buttons, sliders, displays, and more to give us complete control of our robot. In our case, we utilized multiple buttons, a joystick widget, and a "gauge" widget similar to a fuel gauge to monitor the capacity of our bin.

Once we establish these ports, we define the virtual pins we established on the back end in the code. To utilize the digital pins we set the variable by using a defined function in Blynk. This function will be different depending on the data type (int, string, Boolean). Then we can use these variables to send data to our components and receive data from our sensors. However, for receiving data we simply have to send a variable back by setting it to a Blynk function. We used this function to be able to use the gauge widget in our application. For the joystick, the output of the widget would be a string that we parsed in the code to get an X-ray and y-axis. The x and y axis are the coordinates of the position of the joystick based on where the user moves it. With this information, we were able to move our robot forward, backward, left, and right. To move the robot forward, we had both stepped motors move forward, to move the trashporter backward we moved both stopper motors backward by having a negative speed. To move left and right we would have one motor have a negative speed and one motor have a positive speed. These Blynk functions come from an Arduino library made by the company.

For the overall app, we used a state machine in our code in order to have different "modes" for the trashporter. The modes for the trashporter will be manual mode, autonomous mode using the pixy camera, and sleep/off mode. The manual mode can be activated by pressing a button on the application and the same is true for the autonomous mode. For the "sleep" mode, there will be two options. One will simply press a button so it stops moving, and another will detect if the bin capacity is full. We decided to do this as a way for the user to tell them that the trash bin is full and needs to be changed. You will have to do this to collect more trash as there is a weight limit for the trashporter device.

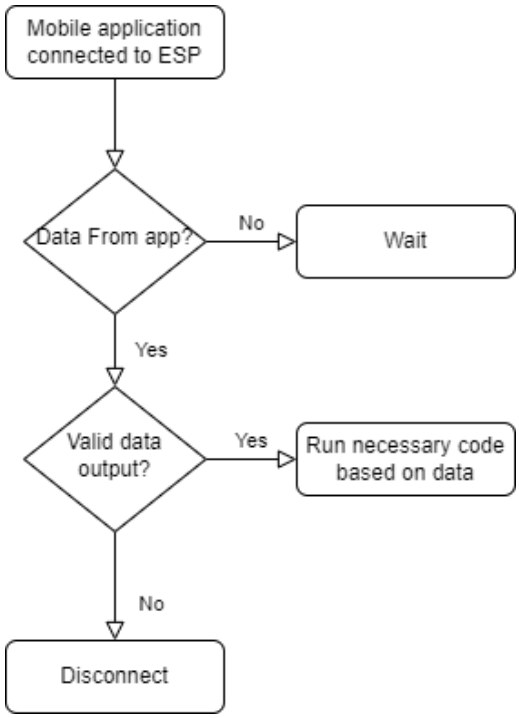


Figure 9: Graphic to show how the trashporter interprets data from the Blynk cloud. This includes data from the widgets being sent to the ESP and data that we will send on the back end via the cloud.

C. Stepper Motor Development

For our device we will be using 2 Bipolar Stepper motors to be able to drive the movement of our robot. We decided to use a stepper motor as it is a motor that has a very high torque, and we can move into positions with high accuracy.

A stepper motor moves in steps. After each step the motor holds itself in position. Internally a stepper motor consists of a magnetized geared shaft that is surrounded by electromagnets. Controlling the current of the electromagnets allows us to step the motor. We will control the stepper motor by applying current to the coils. Since we are using a bipolar stepper motor it consists of two coils which each has two wires attached to it.

Stepper motors are commonly used in robotics but are also used in a wide variety of areas. Most commonly stepper motors are used in 3-D printers, zooming a camera, and even can be used in an analog clock. In this case we wanted to have the stepper motors continuously spin so we can move our robotic device around on the floor. This became an issue for use as we would need the stepper motors to rotate continuously like a motor moving

a wheel on a vehicle. The stepper motor moves in steps and stops multiple times before it completes a full rotation which is great for getting high torque and accuracy but not so much for our desired application.

We originally tried micro stepping. Most stepper motors you can easily get online or on amazon are called 200 steps per revolution steppers. This means that the shaft takes 200 full steps for one revolution. Micro stepping lets us multiply those number of steps and makes the rotation a lot more precise and smoother. However, there is a tradeoff of using this method. After trying to resolve our issue and testing we discovered the big tradeoff is that the overall max speed of the stepper motor reduces significantly, and the torque of the stepper motor also dramatically decreases as well. Although we found that off the floor the motors moved much smoother than before with less steps, the motor was much slower. Also, we tried moving the device on the floor and the motor was not strong enough to move itself even without any bin or load.

However, we did find a solution. In the IDE, since we are using stepper motors, we will include the stepper motor library in the code. For stepper motors, it is required to define a variable which is the number of steps per revolution. Then in the loop function, we simply have the stepper motor rotate for each one of the stepper motors. For our particular use, we decided to use a library called Continuous Stepper. This library was created for anyone who wanted to use the stepper motor in a continuous rotation. This is a third-party library created by an Arduino forum user as they wanted to rotate a stepper motor continuously for a personal project that they were working on and wanted to share with other users. To create the desired effect of running the stepper motor as a “wheel”, we simplified the innate programmed features and functions in our stepper motor libraries. To achieve this he tweaked with the number of steps and also got rid of delays that existed in many stepper motor driver libraries. When we tried this new library, we found that the wheels moved continuously and smoothly as we wanted it to be in the first place. For our application, this was perfect as we were able to use our stepper motors as “DC motors' ' to be able to move our trashporter robotic device with no steps or “stuttering”.

V. MOTHERBOARD

The motherboard for the Trashporter has several components. The motherboard contains a 12V to 5V buck converter, a 3.3V linear voltage regulator, the ESP32, 4 motor drivers, a UART-USB communications chip, and pinouts for every pin on the ESP32. Each component on the motherboard works in harmony to provide power and communication between each other as well as the external components of the Trashporter such as the motors, sonar sensor, PixyCam, and weight sensor.

A. Buck Converter

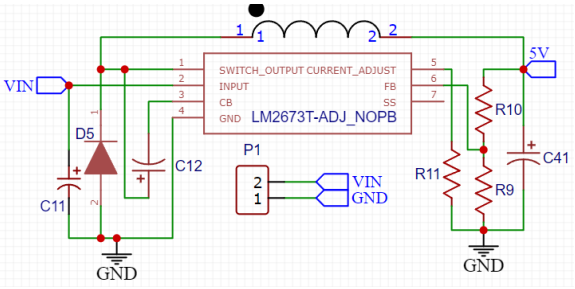


Figure 10: Schematic design for the LM2673 Buck Converter

The buck converter was designed using the LM2673 by Texas Instruments, and following the recommended design on the datasheet [4], and shown above is the buck converter designed for the Trashporter. Since the stepper motors need 12 volts to function, the power source needs to have a minimum of 12 volts. The sonar sensor, the pixy cam, and the weight sensors all require 5 volts to operate; anything over 5 volts will damage the components. To get the required 5 volts, a buck converter was used. The buck converter was used over other DC to DC topologies because a buck converter converts DC to DC voltages more efficiently, and it also increases the output current which can be beneficial to the Trashporter.

The figures below demonstrate that the buck converter works as intended even under load. Figure ?? demonstrates that underload, with an input voltage of 9 volts (blue), the output of the buck converter holds a stable 5 volts. Figure ?? demonstrates that underload, with an input voltage of 12 volts (blue), the output of the buck converter stabilizes at 5 volts as well.

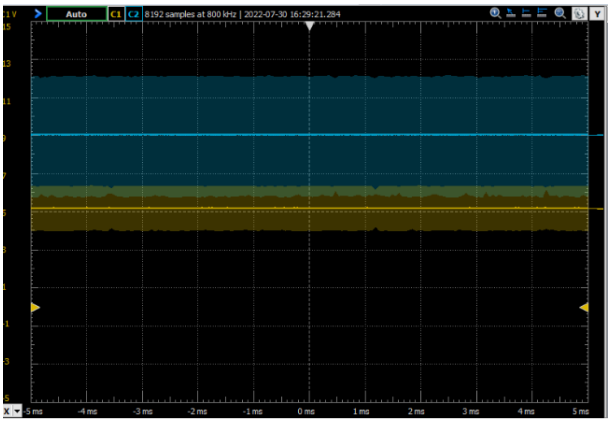


Figure 11: volt input voltage (Blue) and 5 volt output voltage (Yellow) of the buck converter.

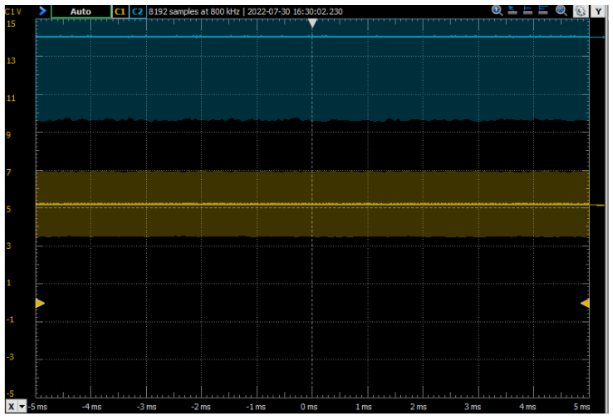


Figure 12: volt input voltage (Blue) and 5 volt output voltage (Yellow) of the buck converter.

B. 3.3 Volt Linear Voltage Regulator

The ESP32 operates at 3.3 volts and has an absolute max voltage rating of 3.9 volts. The UART-USB communications chip also operates using 3.3 volts, therefore a 3.3 voltage source is required. The TLV76733DGNR by Texas Instruments is used as the 3.3V voltage regulator for the Trashporter. The TLV76733DGNR is a linear voltage regulator, and a linear voltage regulator was chosen because the conversion from 5 volts to 3.3 volts produces minimal excess heat. If a buck converter was used, more parts would be required which would increase the overall cost of the Trashporter. Also, converting from 12V to 3.3V would produce a lot of excess heat with a linear voltage regulator, therefore the regulator receives its input voltage from the 5 volt source. Using the recommended design given by the TLV76733DGNR datasheet [5], a 3.3V regulator was designed below.

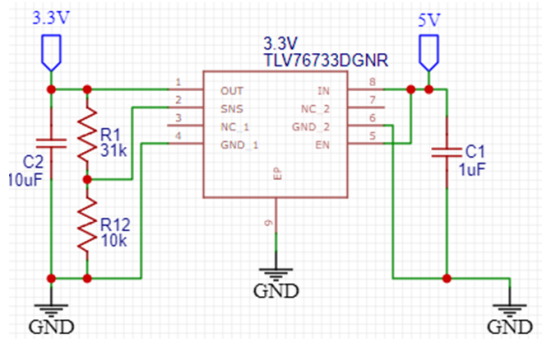


Figure 13: Design schematic of the 3.3V linear voltage regulator

C. The ESP32

The micro processing unit (MCU) is the heart of the Trashporter. It handles the inputs coming from the sensors and PixyCam and then processes this information to send the output signals to the motors. The ESP32 was developed by Espressif based in Shanghai. Although the ESP32 isn't an Arduino product, the open source and open documentation availability of the ESP32 allowed the Arduino community to program the ESP32 with the Arduino IDE. This version of the ESP32 is the ESP32-WROOM-32E that is equipped with 448 KB of ROM, 520 KB of SRAM, 16 KB of SRAM for the Real-Time Clock. This board contains the dual-core Xtensa 32-bit LX6 microcontroller, which should run up to 240 Mhz. The team also decided to work with the ESP32 because several team members already have experience with the Arduino IDE. The Trashporter can utilize the ESP32 dual core processor to handle the IoT specifications while processing the inputs and outputs of the robot. The ESP32 has been soldered onto the board and the necessary connections have been designed to connect to all the other components on the motherboard.

D. The Motor Drivers

The motherboard contains four motor drivers, one for each stepper motor. The DRV8434APWPR manufactured by Texas Instruments was used in the design of the Trashporter. This stepper motor driver comes with integrated sense current, 1/256 microstepping, smart tune and stall detection. For the purpose of this project, the integrated sense current, and stall detection wasn't utilized for the Trashporter as it wasn't necessary features for the robot. Due to chip shortage, this was one of the few stepper motor drivers available and was the only one in stock that had microstepping. It was important for the Trashporter to have microstepping integrated within its design because there will be times in which precise movements may be needed when maneuvering towards objects. Microstepping decreases the step angle for a stepper motor, which increases accuracy in its movements. The current sense was not an original design specification, however, it was an advantageous feature to implement in the Trashporter as it adjusted the current of the stepper motor according to what it needed to operate properly, which saved on power draw. Using the datasheet, the motor drivers were designed below, and implemented on the motherboard.

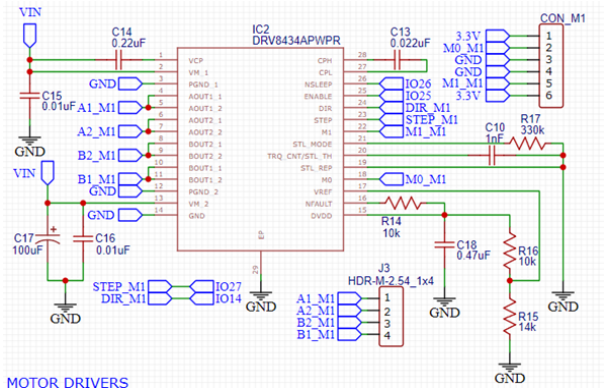
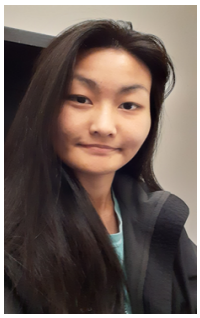


Figure 14: Design schematic of the Motor Drivers for the Trashporter.

VI. THE ENGINEERS

Jourdan Callahan is a 23-year old graduating Computer Engineering student who is taking a job with SAIC in Orlando, FL as a Software Engineer, specializing in Automation and RPA.



AnnaBelinda Zhou is a 22 year-old studying electrical and computer engineering. She is planning to get her Master's in electrical engineering.

Dhanesh Singh is a 22 year-old graduating Computer Engineering student. Dhanesh's career goals are to work for a tech company focused on designing, testing, prototyping, and implementing electric circuits that will enhance society.

Andy Kuang is a 23 year-old studying computer engineering. He is interested in learning more about electronics as well as programming these devices. Andy has worked on personal projects that focused on these concepts and wants to work for companies that contribute greatly to society through new innovative technologies.

REFERENCES

[1] polepositionmarketing@kellertechnology.com. (2021, December 6). *7 Types of Sensors for Object Detection.* Keller Technology Corporation. Retrieved from https://www.kellertechnology.com/blog/7-types-of-sensors-for-object-detection/

[2] R. (2020, July 27). *What are Weight Sensors?* Utmel. Retrieved from https://www.utmel.com/blog/categories/sensors/what-are-weight-sensors

[3] *What is a weight sensor, what are the different types of sensors and how do they work?* Futek. Retrieved from https://www.futek.com/weight-sensor

[4] *(June, 2016). LM2673 Simple Switcher 3-A Step-Down Voltage Regulator with Adjustable Current Limit.* Texas Instruments. Retrieved from https://www.ti.com/lit/ds/symlink/lm2673.pdf?ts=1668097706525&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM2673

[5] (July, 2021). *TLV767 1-A, 16-V Precision Linear Voltage Regulator.* Texas Instruments. Retrieved from https://www.ti.com/general/docs/suppproductinfo.tsp?distId=26&gotoUrl=https://www.ti.com/lit/gpn/tlv767