Robotic Waste Sorting

Major Qualifying Project

Written By:

MIKAYLA FISCHLER
KYLE HEAVEY
ARIANNA KAN

Advisor:

BERK CALLI

A Major Qualifying Project
WORCESTER POLYTECHNIC INSTITUTE

Submitted to the Faculty of the Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Robotics Engineering and Computer Science.

AUGUST 2019 - MAY 2020

Abstract

The recycling industry is struggling under tight profit margins, changing waste policies, and fast evolving waste market regulations. One notable issue is processing the highly contaminated single-stream recycling waste generated by the world's growing population. This project contributes to the long-term goal of developing a waste-sorting robot to efficiently sort single-stream recyclables. We designed a robotic recycling test bed that consists of a robotic arm, an actuated X-Z Cartesian platform, and a modular control system. We also developed a three-jaw gripper designed for effective cardboard picking from a mixed waste stream. Such outcomes provide the main architecture for establishing a robotic waste sorting experimental setup at WPI.

Acknowledgements

We would like to extend our gratitude to our advisor, Professor Berk Calli, for his guidance in completing our project. We would also like to thank Casella Waste Systems for allowing us to tour their facility and WPI's Office of Sustainability for providing insight on WPI's recycling operations.

Table of Contents

List of Tables

LIST OF FIGURES

Chapter 1

Introduction

Single stream recycling refers to a system in which consumers place all varieties of recyclables into a single bin, which is then collected and sorted at a materials recovery facility (MRF). These recyclables include paper, plastics, glass, cardboard, and metals. The concept of "single-stream recycling" took hold in the US around 1990 and saw over a five-fold increase in the US recycling rate within two decades. This form of recycling grew to widespread popularity in a short period of time due to its inherently simple strategy–consumers are encouraged to put everything that is not trash into a single curbside bin, foregoing the complicated nature of separating and sorting the recyclables themselves. This uptrend also equates to a rapid increase in the volume of recycled waste collected. Unfortunately, single-stream recycling leads to a decrease in the quality of materials recovered, caused by an increase in the level of contamination by other materials.

For decades, the US has sent the majority of its recycling to China for processing. However, starting January 1st, 2018, China initiated a new policy that includes waste import regulations, specifically enforcing a strict limit on the amount of contamination they will allow when receiving recycled imports. Effectively, this caused China to ban almost all of its recycling imports–taking in less than 1% of its 2016 imports that same year [1]. In the wake of China's tightening regulations, the US faces waste processing backups, resulting in many recycling centers sending more and more recyclables to landfills.

Increasingly, recycling centers are relying on automated systems to separate these recyclables. Unfortunately, these systems are still developing and necessitate human workers to monitor the waste stream for items that could damage the equipment, such as hoses, plastic bags and hazardous materials. This puts the human workers in dangerous environments, where they are routinely exposed to intense stress and hazardous waste. To help provide a solution, this project contributes to the long-term goal of developing a waste-sorting robot to efficiently sort single-stream recyclables, allowing for faster and more precise sorting with an increased influence of the role of human workers as supervisors rather than sorters.

Our goal is to develop a test bed, simulating a recycling plant environment, and the base robot capable of providing efficient sorting to a recycling center. In order to successfully execute this goal, our objectives are as follows:

1. **Research** the current state of recycling and current robotic solutions for waste sorting
2. **Design** the four main components of our project: the test bed, the platform/rail system (the frame), the linkage arm/gripper, and the control system
3. **Implement** our design into a physical model
4. **Test** our design: connect the control system to the physical robot and test cardboard removal

We began our project with research–understanding the current state of recycling in the US and how China's recent import regulations affect the US recycling industry. With the conclusion that the contamination levels of the "sorted" batches were too high for recyclers to accept, we focused our project on alleviating this contamination through the development of a robotic solution. We researched current robotic solutions for automated sorting and visited the Casella Waste Systems facility in Auburn, MA to gain a better understanding of the role of a robotic system in the waste sorting process. From here, we determined a set of design criteria for each component that would take in the constraints of the facility and improve upon current robotic designs. Unfortunately, given the unforeseen events of the COVID-19 pandemic, we were unable to complete our project. We successfully built each component separately, but the entire assembly of the robot was interrupted by WPI's prompt response to the growing pandemic. Given the incomplete state of our system, we were unable to test our design for specific contaminant removal. As a result, we provide guidelines for completing the rest of this project, as well as a direction for future work.

Chapter 2

Background

Our team conducted research on the global recycling industry to understand the present-day problems it is facing. We studied the current state of the industry, including the disruptions caused by China's policy change on recyclable waste imports in January of 2018. We found that the industry as a whole is struggling to process the vast quantities of contaminated single-stream recyclable waste generated by modern-day society. Not only is the industry struggling to keep up, the systems in place often result in portions of potentially recyclable waste being sent to landfills. On top of these systematic issues, the workers themselves are often put in harms way, as human sorters have to sift through and remove the sharp, heavy, and sometimes toxic contamination in recycling streams.

During our research, we found potential robotic solutions to these challenges facing the industry. They consisted of various different robot designs, with trade-offs for either speed or accuracy. We used these robots for direction and as inspiration for our robot designs.

2.1 Current State of Recycling

The mass adoption of single-stream recycling has led to higher volumes of recyclable waste being collected due to the inherent simplicity encouraging more people to contribute [2]. Unfortunately, this led to much more trouble actually recycling the materials. Single-stream has the disadvantage of mixing vastly different materials together, often with contamination such as food, oils, and non-recyclable materials, mostly due to lack of awareness [2].

This problem has been further exacerbated by China's recent rejection of low-quality and contaminated recyclable materials [3]. This has shined a spotlight on the figurative and literal mess that is the global handling of recyclable materials. Many often poorly sorted recyclables no longer bound for China are exported by North America and Europe to other, usually less-developed nations, to cheaply attempt to reclaim materials such as metals and glass. The change in China's policy on recyclable imports has shaken up this international network, resulting in significant amounts of recyclables ending up in landfills and these other nations that are less prepared to properly process the immense amount of waste [4].

2.1.1 China's National Sword Policy

Beginning January 1st, 2018, China began enforcing their "National Sword Policy", which regulates the imports of waste [3]. This policy bans 24 types of waste, consisting of many types of plastics, paper, textiles, and metals. Additionally, there is a cutoff of a maximum of 0.5% contamination in imported sorted recyclables [5].

This policy came after China's Green Fence Policy (2013), which was the start of higher standards for contamination tolerances. This was put into effect because approximately 25% of the recyclables they imported needed to be sent to landfills instead [6].

These policies have been the acting arm of Chinese leadership's desire to reduce damage to their environment. This was initiated due to pressure by citizens and media to reduce the prevalence of environmental hazards that have been plaguing the country [5]. One of the main sources of pollution is actually the plastics recycling industry, and in 2017, it was revealed by China's Ministry of Environmental Protection that 590 of the 888 inspected licensed factories were discovered to have violated regulations [5].

2.1.1.1 Why this is a Problem

These limitations on imports have major consequences for the processing of recyclable materials world-wide [7]. This naturally has blocked a significant amount of recyclables from entering the country due to their contamination levels. Such rejections result in nations instead shipping these unprocessed recyclables to other countries, usually third-world or generally less-developed countries. Redirection to these countries raises concerns about how these materials are being processed and recycled, if it is done safely, and if recycling is occurring at all [4].

Polices such as these serve as a wake-up call for those who are unaware of the endemic contamination in the industry and its largely ignored environmental consequences, as few associate recycling with environmental damage. If China, known environmentally for its smog problem, is rejecting the 'recyclable' waste of other nations due to environmental concerns, maybe something needs to change?

2.1.1.2 Why Send it Out

Sorting and processing recyclables is a difficult and labor-intensive task. This is exacerbated by poor recycling habits of most consumers, which arose from unclear or intricate rules on what can and cannot be recycled. Many developed countries opt to export their recyclables because the task of domestically processing them can be disproportionately expensive when compared to the money that can be made by selling the reclaimed materials. In these cases where it is too expensive, waste is either sent to landfills or exported [8]. As previously mentioned, most exports sent to less developed countries end up in the hands of both licensed and unlicensed recyclers.

2.1.2 Waste Management as it is Now

The process of recycling materials, put briefly, consists of collection, sorting, and processing. Once recyclables are collected, they are brought to a sorting facility, and once sorted, they are sent either domestically or internationally to a processing facility. Garbage sorted from the collected recyclables is sent to a landfill, but more and more contaminated recyclables are meeting the same fate. As with almost every industry, the recycling industry is profit driven. If waste processors do not want to purchase poorly-sorted recyclables due to contamination, the sorting facilities have nowhere else to send it [7] since convenient exporting options are no longer available. In many places across the United States, these impure recyclables are being sent to landfills, as it is a more economically viable option. Due to the high cost both of time and money required for sorting and cleaning recyclables by hand with human workers, contaminated waste is simply more expensive for processing plants to handle [1]. Even if a plant succeeds, the reclaimed materials often are more expensive for manufacturers to purchase when compared to virgin materials [9]. This was more feasible when done in unregulated, cost-cutting environments in China, but that is no longer an option.

Contamination is largely caused by a concept referred to as “wishful recycling”. In short, this is where people ‘feel’ as though something is *probably* recyclable or *should be* recyclable, so they proceed to toss it into their recycling bin even though it is either not recyclable in their area or not recyclable at all (and sometimes even may belong in hazardous waste disposal). This results in contamination that becomes disruptive to the recycling systems and can cause whole batches of recyclables to be sent to landfills, not just individual items [4].

In the wake of China’s tightening of waste import regulations, many recycling centers in the U.S. have been sending more and more recyclables to landfills (for example, many hundreds of tons from a single facility), and in areas of some western states, recycling of some materials has stopped altogether. However, as implied, this is not true for all states, as some of those that are unable to process enough of their output domestically turn to shipping it to other nations who continue to accept what China now rejects, including India, Vietnam, and Indonesia [4]. The demand for international importers of waste has risen, as China previously imported approximately 45% of global waste exports. In 2015, $300 million of plastic exports from the U.S. went to China, which dropped to $7.6 million in the first quarter of 2018 [4].

Internationally, countries are facing similar issues. Some, including the UK, Canada, Ireland, and Germany, are being forced to store large amounts of these recyclables until there is an opportunity to eventually send it out for processing. This is a symptom of the major global upset in the flow of recyclable waste [10]. Some parts of Canada had previously been sending over 50% of their recycling to China, for example. To pick up the slack left by China, countries such as Malaysia, Vietnam, and Thailand have been drastically increasing their imports of recyclable waste. This has come at a severe environmental cost to these countries, due to not having facilities that are well-equipped to deal with high volumes of contaminated recyclables. Unlicensed recyclers in these countries are not being held to any regulations whatsoever and therefore unsurprisingly emit significant pollution. This is primarily caused by the burning (or attempted burning) of recyclable waste that is too contaminated to be processed, which can of course include plastics which release toxic chemicals when burned. This situation is very similar to that of the unlicensed recyclers that were previously operating in areas of China of whose actions led to China tightening their import regulations. This causes significant health implications for neighboring communities. In the end, a significant amount of the recyclables across the globe either end up in landfills or are burned due to the inability to properly process the immense volumes of waste produced by modern society [8].

2.2 Single Stream Recycling

Single stream recycling (SSR) refers to a system in which consumers place all variety of recyclables into a single bin, which is then collected and transported to a material recovery facility (MRF) where they are sorted and processed. In general, “single stream” means you can put paper, cardboard, plastic, glass, and metal objects together. The concept of “single-stream” recycling took hold in the US around 1990, resulting in a nearly 50% increase in the US recycling rate, from less than 15% in 1990 to close to 36% in 2017, as is shown in Figure 2.1 [11]. It was also within this time that the US began to see a large growth in material recovery processing facilities capable of handling the single-stream.

SSR provides the potential for streamlining collection and processing to reduce overall operating costs of recycling programs. As mentioned earlier, one of the most notable benefits of SSR is increased recycling rates, due to the inherently simple process of placing everything not considered trash into a single bin and abandoning the complicated nature of sorting the various recyclables themselves. Additional benefits to single-stream recycling are lower collection costs as the need for a single pickup replaces the need for separate pickups for different recycling streams, such as with dual-stream recycling in which fiber components are kept separate from containers at curbside pickup. This then leads to one tipping floor, in which all recyclables can be deposited together, and thus one residue stream to manage [12]. Furthermore, there is an increase in diversion rates due to easier participation by the homeowner.

The most notable criticism of SSR is that it leads to a decrease in the quality of materials recovered, increasing the levels of contamination (such as broken glass and non-approved materials) which cause significant problems for MRF operators [13]. Skeptics of the adoption of single-stream

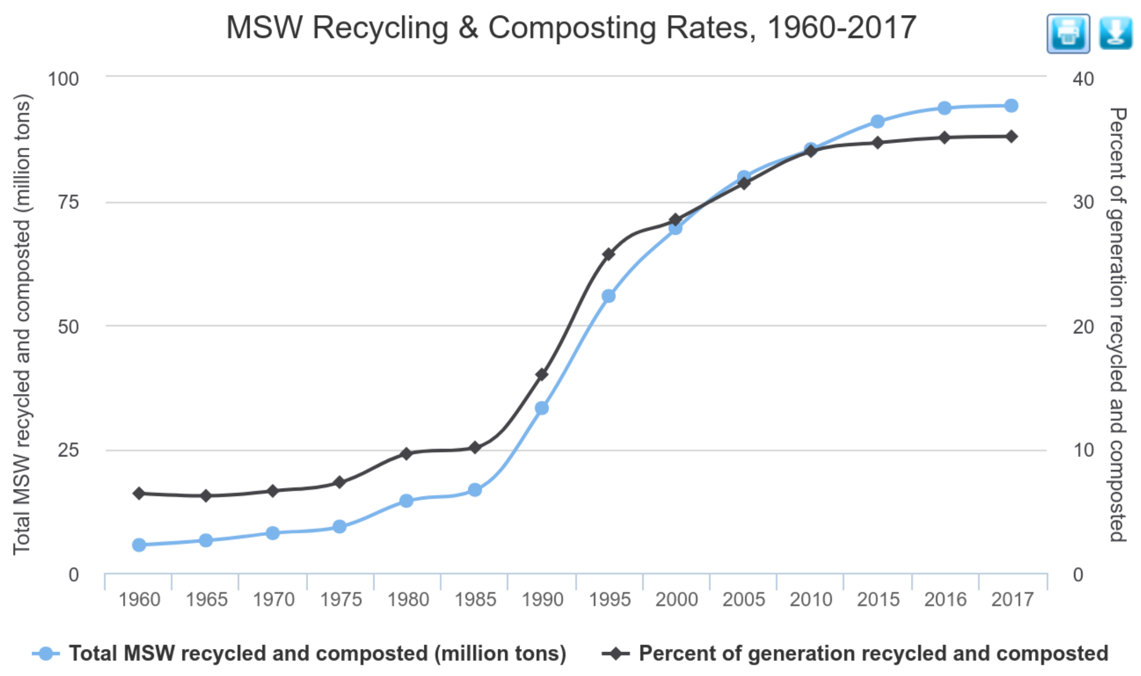

Figure 2.1: *MSW Recycling & Composting Rates, 1960 - 2017*

recycling point out that this is a fundamental flaw of the US single-stream system, including the contamination issue that begins at curbside collection. Additional skepticism relates to paper mill buyer reluctance, increased residue, and additional capital costs [12]. Furthermore, misidentified items can damage expensive equipment or temporarily shut down recycling operations.

2.2.1 The Recycling Process

According to the Solid and Hazardous Waste Education Center, residual rates (the items that should not have been put in the bin and the items that could be recycled but were not) are roughly 1% for multi-stream systems, 2-3% in dual-stream systems, and an enormous increase of 15-27% for single-stream systems [14]. The large amount of residuals coming from the single-stream systems is a consequence of the current technologies available to the facility. Due to the rather recent uptrend of single-stream recycling, the majority of MRFs are built around a combination of machines and human workers.

The current sorting process varies with respect to the automation employed in the system, involving technologies utilizing conveyors, screens, forced air, magnets, optical material identification and eddy currents. Initially, front-end loaders dump large amounts of recyclables onto conveyor belts. Non-recyclable items are then identified and manually removed by trained workers in a pre-sort area. The materials then move to a triple-deck screen where cardboard, containers and paper (items too heavy or too light for the next level of SSR processing) are removed. Cardboard can be removed automatically with an OCC (old corrugated containers) screen. Heavier containers drop to the bottom level while lighter items continue to the second level. This screen also breaks the glass containers for the safety and convenience of workers and removes most of the glass from the single-stream load so that it can be sent to cullet suppliers. The remaining materials pass under a powerful magnet to remove tin and steel cans. Lastly, a reverse magnet (eddy current) causes aluminum cans to lift off the conveyor into a bin. Throughout this process, MRF staff are constantly watching and removing specific materials that have accidentally made it down the line. At the end of the line, the staff separate cardboard, newsprint, and office paper into a bunker below. Once separated, the material is baled and shipped to manufacturers for processing into new material [13].

2.2.2 Contamination

The quality of the materials received from single-stream recyclers tends to be considered inferior to those received from dual- or multi- stream recyclers, especially because MRFs cannot separate recyclables as well as a system that does not mix them to begin with. This forces manufacturers to down-cycle the material–in other words, use it in a cheaper product. Unfortunately, many recyclers are unable to handle the large (sometimes 10% - 20%) amount of contamination and the entire batch is sent to landfill, resulting in a wasted recycling effort altogether [14].

The most problematic of contaminants can be grouped into five common categories: tanglers, film plastics, bagged objects, hazardous material, and what can be summed up as “yuck”–objects or substances that downgrade other materials and clog the system [15]. Table 2.1 [16] provides more details on major contaminants that result from single-stream recycling.

Category	**Description**
1. Plastic Bags	Plastic bags and items made from similar plastic material (i.e. shrink wrap, bubble wrap, ziploc bags, newspaper bags, trash bags, etc) are one of the most problematic contaminants. When placed curbside, they get wet and dirty and cannot be recycled (Grocery store plastic bags can be recycled if clean, dry and empty). In addition, plastic bags are commonly used as recyclable waste containers, which causes a significant inefficiency, since the workers have to slow the conveyor belts to rip them to get the recyclables.

2. Shredded Paper	Shredded paper is too small to sort and can cause recycling issues. It not only slips through the machinery, causing contamination to non-paper recyclables, but it also destroys its potential for reprocessing. This is because the most valuable trait of recyclable paper is its long paper fiber because these long fibers can stand up to multiple recycling cycles.
3. Broken Glass	Much of the glass that enters an MRF is typically broken into pieces of widely varying sizes, which complicate the sorting process. Due to the inherent limitations with MRF's current sorting techniques, glass cullets cannot be completely removed from other recyclable streams, thus increasing the contamination in these batches. This causes quality and safety issues in the finished glass products, such as reducing their structural integrity.[17]
4. Hazardous Waste	Items such as paint, automotive fluids, car batteries, propane tanks, and pesticides are not only unrecyclable in the current recycling process, but also cause work safety issues.
5. Bio-Hazardous Waste	Items such as syringes, needles, diapers and other sanitary products are potentially dangerous for workers to handle.
6. Non-Recyclable Plastics	Plastic lids, foam (Styrofoam™), etc. The most commonly recycled plastics are #1 (PET) and #2 (HDPE). Plastics #3 - #7 are rarely recyclable and get mixed up with the #1-2.
7. Scrap Metal	These items cause excessive damage to the recycling equipment.

8. Flattened Containers	Single-stream sorting equipment separates “flats” (paper) from “rounds” (containers). When containers are flattened, the equipment mistakenly sends them to the paper side of the facility, significantly contaminating the paper stream.
9. Residue	Recyclables that contain residues such as food waste, oil and grease are considered contaminants.
10. Ceramics / Non-Recyclable Glass	Ceramic, china, dishes, mirrors, light bulbs, Pyrex, porcelain, window glass – different melting points and chemical compositions will ruin new glass bottles. If a buyer sees just one of these, the entire load of glass could be rejected.
11. Frozen Food Containers	Paperboard boxes have a plastic polymer sprayed on them to protect against freezer burn. That same coating prevents the box from breaking up in the recycling process

Table 2.1: *Major Contaminants in SSR*

2.2.3 Workplace Hazards

While most MRFs follow the above mentioned steps, the exact process may change depending on the technology they adopt. They do, however, employ human workers as sorters for removing unwanted materials. At older MRFs, they are responsible for separating all the different types of recyclables by material type. Sorters are necessary to ensure that no stray recyclables fall into the wrong group. They also monitor the waste stream before it reaches the automated equipment to pull items that could damage the machinery such as hoses, plastic bags, and hazardous material [18]. Unfortunately, these workers are exposed to intense stress, dangerous machinery, and hazardous materials. They routinely encounter used syringes, glass shards, and biochemical waste, sometimes suffering injury, illness, or musculoskeletal disorders from their constant strenuous positions along the conveyor belt [19].

Older single-stream MRFs require higher staffing levels to work the sorting lines, however MRF technology today has improved drastically due to advanced automated equipment, allowing for faster sorting and lower labor costs while reducing workplace hazards for the human staff. Still workplace hazards occur due to the inherently challenging and dangerous conditions in the plants. The waste management industry has twice the injury rates compared to the average of other industries [19].

2.3 Potential Solutions

There are various approaches for current robotics implementations for recycling sorting, although many have similarities. Most approaches aim to have the robot added on to existing recycling plant lines for easier integration, and to have the robot sort much faster than a human worker can. Depending on the targeted needs in the recycling line they are designed with different end of arm tooling (EOAT) attachments. This section will cover multiple different robots that we researched to inspire ideas for our own sorting robot.

2.3.1 SamurAI

The SamurAI, as shown in Figure 2.2 [20], is made by machinex recycling. The SamurAI is built for speed. It has a four articulation point arm, a delta robot, as shown in Figure 2.3 [21]. This allows it to maintain precise yet quick control over the positioning of the EOAT. It detects the contaminants using a vision system as they enter the robot's range of motion. The arm uses a suction cup, allowing it to quickly attach to and move items off of the conveyor belt. Based on the type and volume of material going through the conveyors the robot can either pick out items that the facility wants, or it can pick out items that are unwanted, switching to achieve the most efficient sorting. It is designed to either work on its own or in a pair of two allowing the first to communicate to the second if it missed a contaminant that the other should pick out.

2.3.2 Bulk Handling Systems Max-AI

The Max-AI is very similar to the SamurAI. It also uses a suction cup EOAT and a delta robot configuration, but with three arms instead of four. However, the Max-AI also features a partner robot, the Recycling Co-bot, as shown in Figure 2.4 [22]. This is a unique addition that goes on the conveyor usually after a Max-AI system. It uses two multi-degree of freedom arms that sits on one side of the conveyor, each with a suction cup to pick items off the conveyor. Using cameras on both arms it can track items and pick them up . It can place in chutes either in next to it or across the conveyor by "ejecting" items from its suction cups, mimicking how workers currently work on one side of the conveyor and take items they remove from the conveyor and toss them into the chutes.

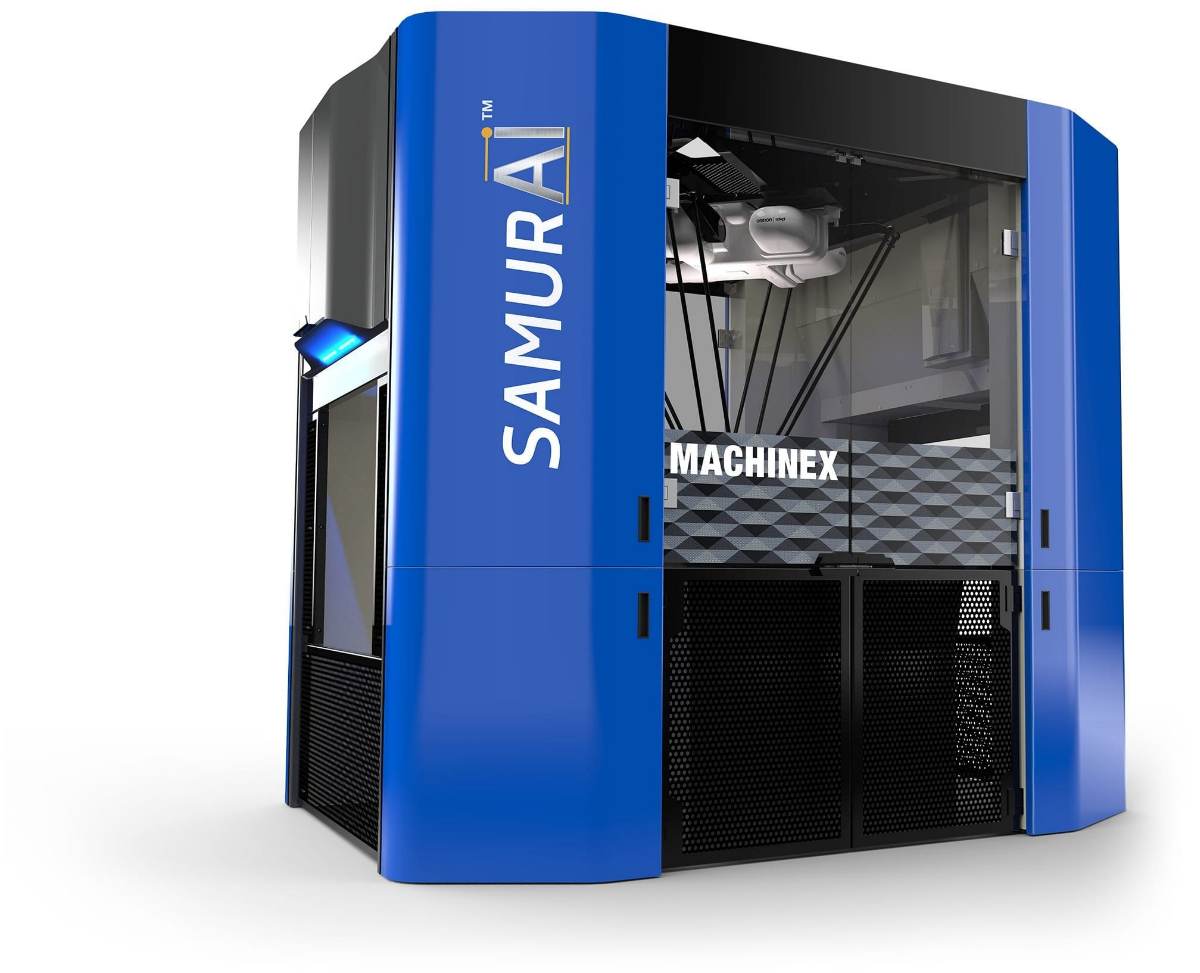

Figure 2.2: *Machinex Recycling's SamurAI*

2.3.3 Zen Robotics Heavy Picker

The Zen Robot's Heavy Picker is the most robust and simple of the designs we looked at. It uses a simple clawed gripper, shown in Figure 2.5 [23]. The Heavy Picker is made to deal with more robust waste such as metals and wood. Its gripper is attached to a linear actuator, and moves on a Cartesian rail system to move the claw to where it picks up the objects.

2.3.4 MIT RoCycle

The MIT RoCycle, as shown in Figure 2.6, [24], is designed to use a soft touch gripper to identify objects and material. It can only pick up a single object at a time in slow speeds. The developers are planning on combining it with a vision system so the touch identification only needs to be used on objects the vision fails to identify with certainty. By using touch, it can determine the interior of an object as well, which can be the deciding factor between the attempted recycling of a TV

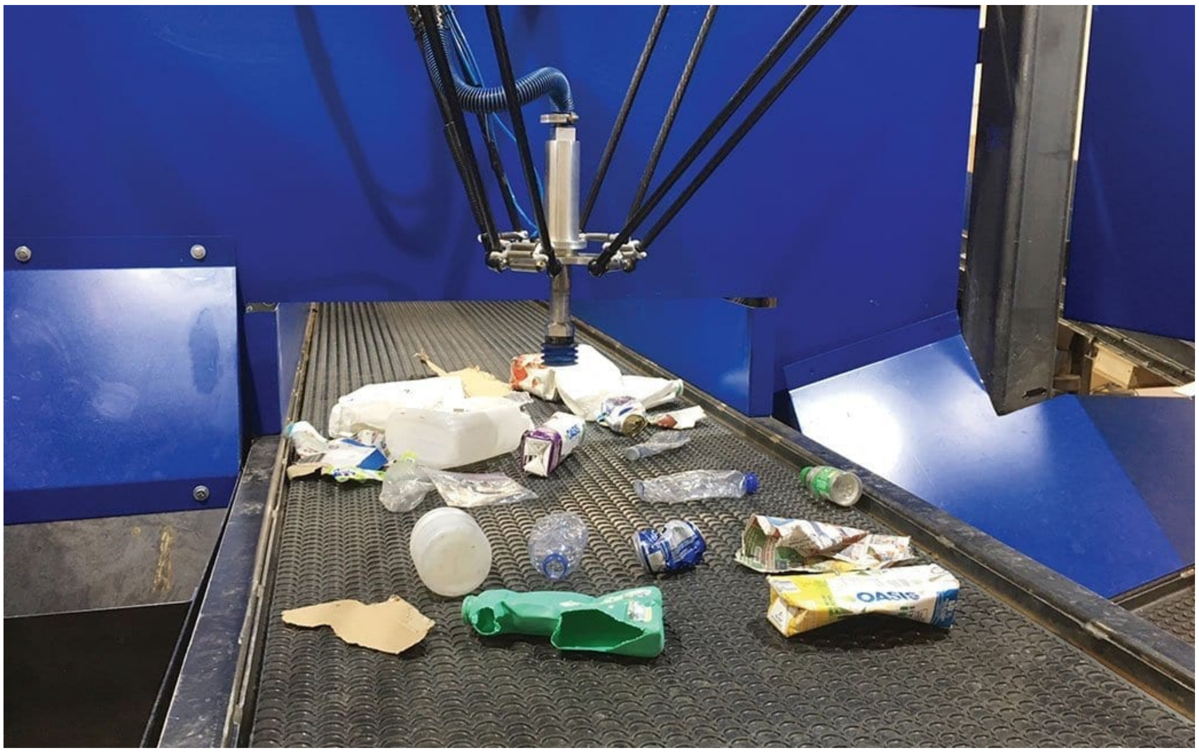

Figure 2.3: *SamurAI's delta Arm*

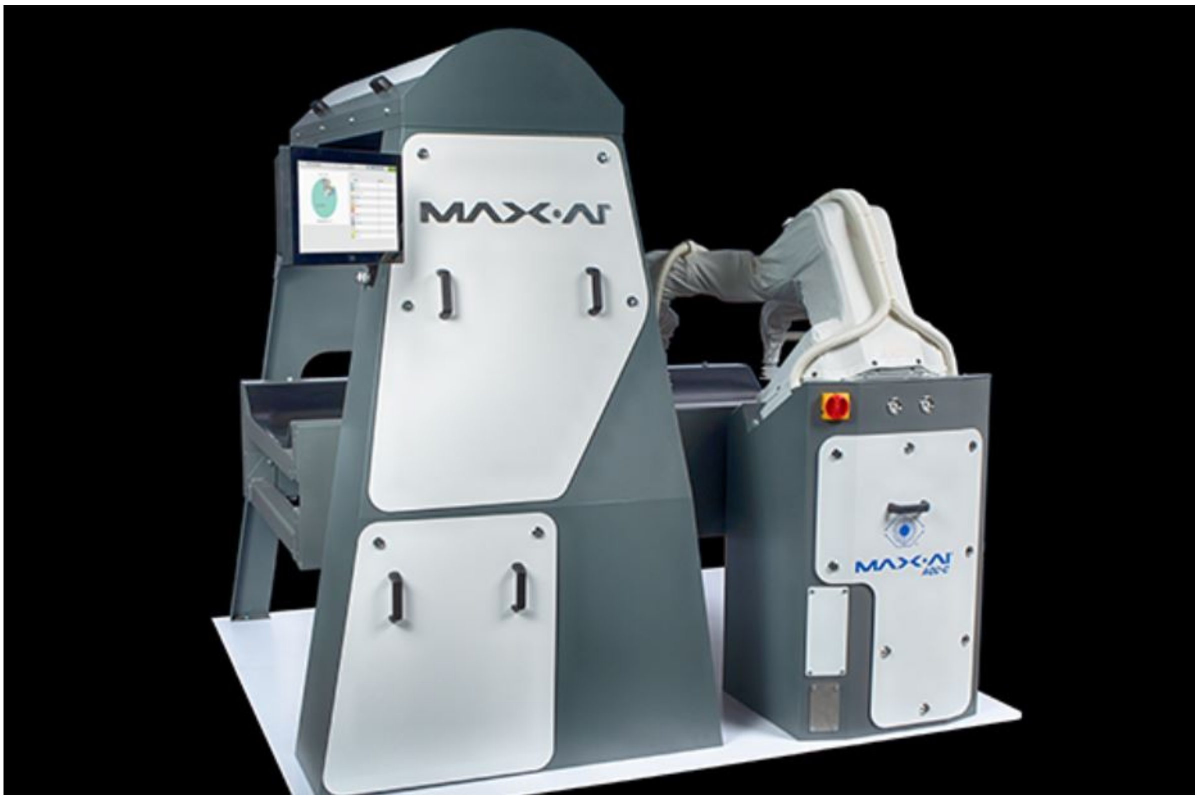

Figure 2.4: *BulkHandling System's Max-AI Co-Bot*

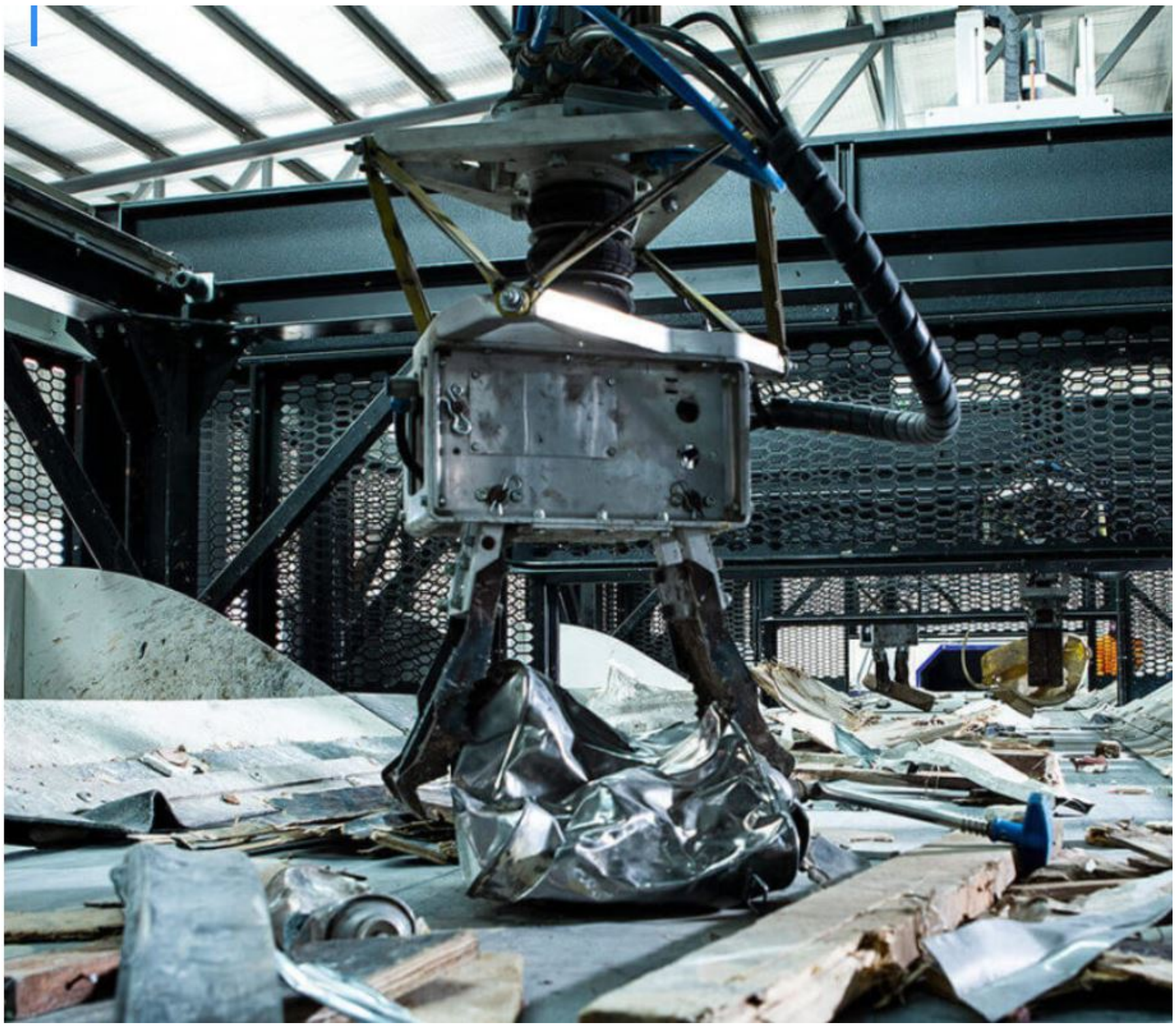

Figure 2.5: *Zen Robotics's Heavy Picker*

remote versus a plastic bottle. Although they both are plastic, the remote requires disassembling and can not be recycled as is. The robot's main goal is to sort accurately but not necessarily efficiently. This can be effective though, since manufacturers want as pure recycled material as possible, which means accurately sorted with a minimal concentration of contaminants.

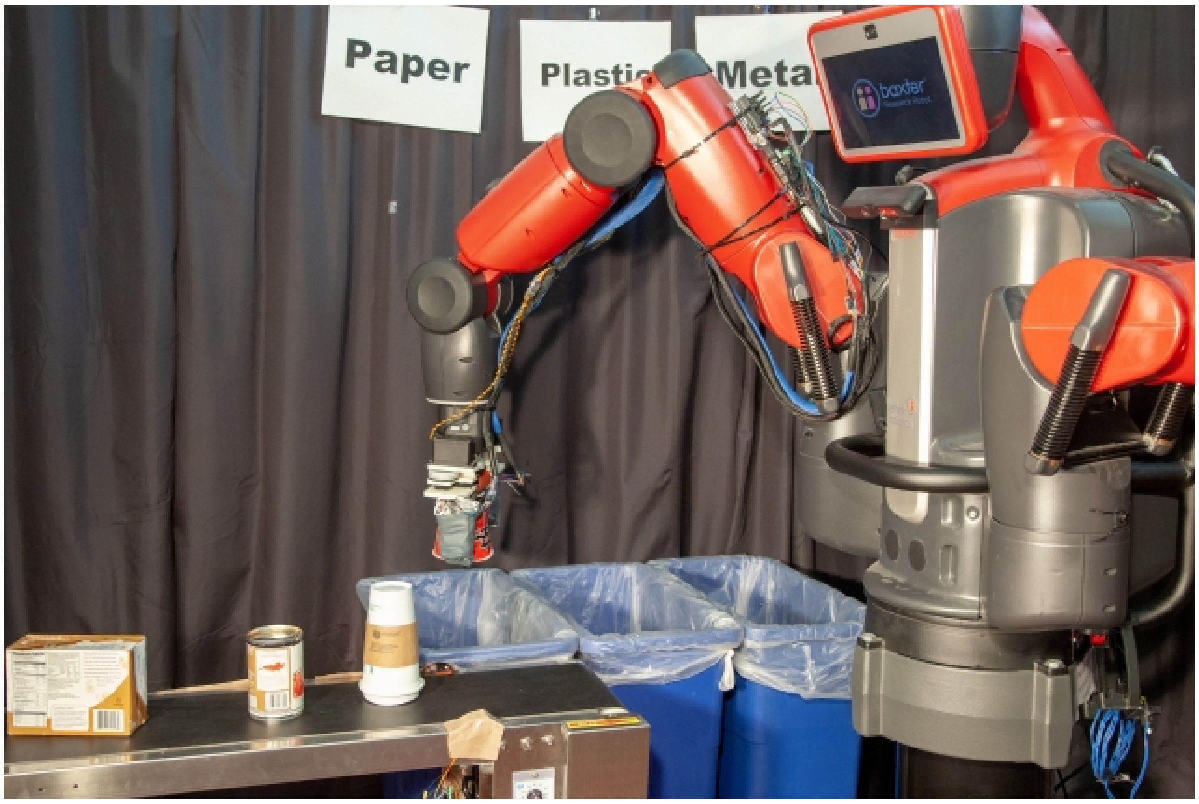

Figure 2.6: *MIT's RoCycle*

Chapter 3
Design

The task of sorting recyclables requires complex, fast, and precise motion to effectively process waste at typical plant conveyor belt speeds. Our project is focused on establishing a framework to go above and beyond current systems in the long term. To achieve this goal, we designed a waste sorting setup, emulating the typical process in a Materials Recovery Facility (MRFs), that will facilitate further research and development in related robotics technologies.

In this chapter, we begin by explaining our design selection process. Following that, we present details about our robot hardware, controller, and software architecture designs.

3.1 Determining Design Criteria

Waste sorting for recycling is a very under-researched, but emerging field in robotics. To the best of our knowledge, there is no open-source design available in the literature regarding the development of a waste sorting setup. Moreover, there is no comprehensive analysis presented for design metrics of such systems. In this project, we needed to develop our design methodology and criteria.

3.1.1 Visit to the Casella MRF

This project is partnered with Casella Waste Systems, a waste management company that provides a wide selection of services, including garbage pickup and environmentally-conscious recycling in Northeastern US. They provide the main recycling services in Worcester, MA, USA.

In order to determine a set of design criteria for our project, we began by visiting the Casella MRF in Auburn, MA and toured their facility to gain a better understanding of their setup and recycling processes. From this trip, we took note of the placements of human sorters along the conveyor belts and observed both the conveyor speeds and amount of clutter being processed by the sorters. At the time of our visit, the conveyor speeds moved at a relatively moderate pace, with a low amount of clutter given the large width of the conveyor belt (roughly estimated to be around five feet in length). However, both the conveyor belt speeds and clutter levels change day-by-day depending on both the conditions, such as the amount of waste to be processed and the target stream purity, and their location within the facility. We aimed to design our system to represent all these conditions, including the conveyor belt design metrics. As such, our conveyor design parameters (explained in Section 3.2) are based on these observations. Figure 3.1 shows an image of the amount of clutter we anticipate on replicating with our test bed. Our aim is to imitate the circumstances of the Casella facility as much as possible in order to test real-world applications.

3.1.2 Analysing the Current Robot Designs

Inspired by prior robotic systems for waste sorting (presented in 2.3), two high-level robot designs were compared, namely a Cartesian style robot versus a delta robot. When taking into account complexity, time limitations, and short-term goals, the Cartesian style was selected. Here, simplicity was favored over complexity because for the first phase of this project, the high speed manipulation of a delta robot would not be an essential advantage in the early research and development stages. Once a powerful control system and object classifier enable the robot to accurately process waste, then high speed should become a high priority focus.

3.1.3 Determining the Role of Robotics

Another important design parameter is the role of the robotic system within the waste sorting process. Robotic waste sorting is a very complicated and convoluted problem. The amount of clutter and the variations in waste types, shapes and appearances make it very hard to design a single solution for all the robotic picking scenarios. Therefore, we needed to identify a subset of the robotic waste sorting problem that would be achievable within the timeline of our project while still targeting a significant problem for the industry.

We decided to focus on identifying and removing cardboard from the stream of mixed paper. We chose to target cardboard because of its continuous growth in volume over the years due to the increasing trend in online shopping. In addition, cardboard contamination is a significant problem especially on rainy days, as the machinery is designed for dry items. Water weakens the paper fibers of cardboard, preventing it from separating properly from other material and thus mixing into other waste streams [25].

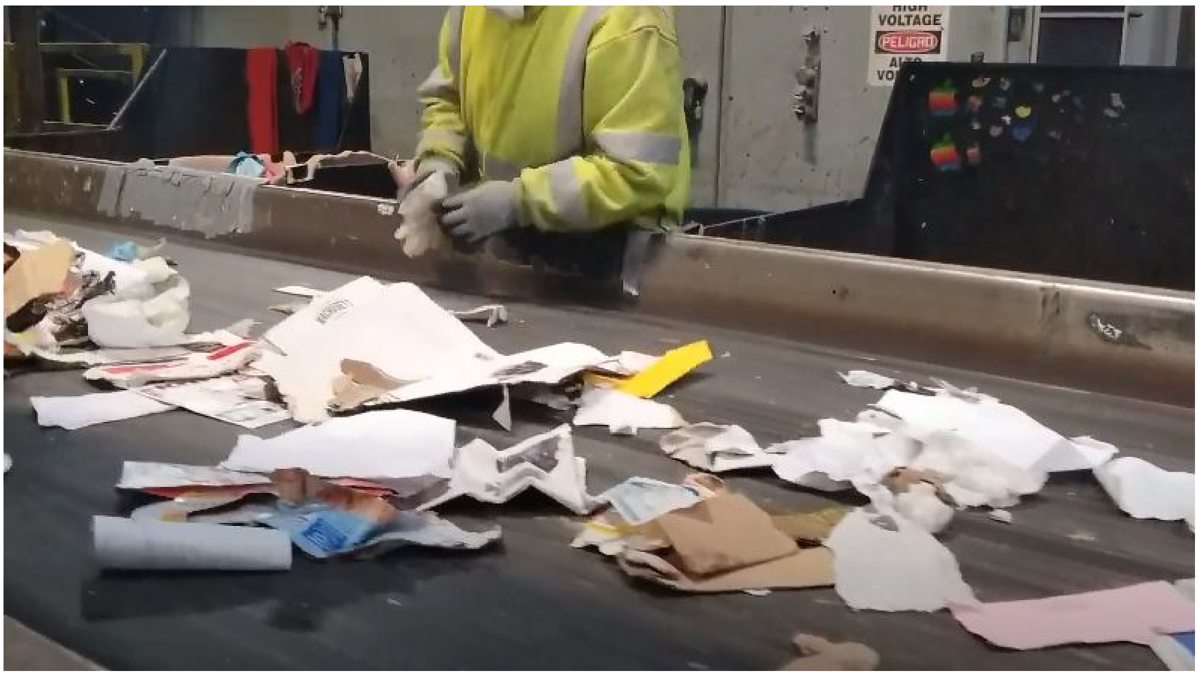

Figure 3.1: *Screenshot from Casella facility video*

3.2 Test Bed Design

In order to provide an environment similar to that of a recycling plant, it would be necessary to simulate the setup that a robot would have when placed in a recycling plant. From both our background research and understanding of Casella's recycling process, we decided to affix our test bed upon a conveyor system, where human sorters are frequently placed for sorting in many recycling plants. With the desire to utilize the extent of our lab space, we established several requirements for our conveyor system. The first is that we should purchase the conveyor belt rather than manufacture one ourselves. We understood the time constraints for this project, and we were aware that designing a robot would already use most of that time. As such, we intended to purchase a conveyor belt that fulfilled these qualifications: (1) reversible, (2) variable speed, (3) overall width between 20 and 30 inches, and (4) overall length between 8 and 10 feet. For testing purposes, having a reversible conveyor belt would allow us to perform continuous testing with our recyclable material rather than expend extra time in replacing the recyclables at the beginning of the belt after every test iteration. A variable speed conveyor belt would provide flexibility during testing, allowing us to slow the belt speed in order to focus on identification and precision, and then speed it up as we begin testing for real-world applications.

3.3 Robot Design

Even though our main goal was to design a robotic system for cardboard picking, we wanted our system to be scalable for sorting other waste types, and be useful for research and development projects for many years. Therefore, we focused on designing a modular system that can support an assortment of grippers designed for the recovery of other types of materials. The conveyor speeds at recycling plants often move at a fast pace, and as such, our robot should have high reliability and precision in a 3-dimensional space, while maintaining efficiency in horizontal travel. Movement along the vertical axis must therefore consist of precise control, with our end-of-arm tooling capable of cleanly picking up cardboard objects. After our research, we observed that a delta robot would be the best option. That style of robot is capable of moving at high speeds and maintaining precise control over its end-effector. However, due to budget, time, and complexity we could not purchase or build a delta robot, so instead we chose to emulate that style of control with more basic functions in multiple parts. We aim to keep the heavy components along the superstructure frame, allowing our articulated vertical-axis arm component, designed for the pickup and drop-off of material, to be very light and thus retain an increased speed. Our overall robot design consists of two main components: the X-Z Cartesian frame, which consists of the actuated platform and rail system, and the articulated arm component, which consists of the multi-degree of freedom arm and our end-effector.

3.3.1 Frame

We chose to use an X-Z Cartesian frame for several reasons. First, Cartesian designs provide a greater work envelope over other designs, such as the SCARA robot and other 6-axis robotic designs that have circular or oval work envelopes that result in a larger region of inaccessible space. Cartesian robots have a rectangular work envelope, which is ideal for application on a conveyor belt, and can be constructed from a variety of linear actuators and drive mechanisms. In addition, Cartesian-style designs have an ease-of-use advantage due to the simpler three-axis kinematics, along with high reliability and precision when operating in 3-dimensional space. Given the dimensions of our conveyor system, we designed a 4-legged frame that would mount over the belt. With an overall length of 6.5 feet, and width of 2 feet, we assembled our frame using steel weld-together rails, as seen in Figure 3.2. For the frame's rail system, we chose to use several timing belt linear actuators to control movement in both the X and Y directions, since the teeth efficiently transfer torque and prevent slipping. Our rail system utilizes these belt drives to position our arm component, with two belt drives working in parallel to guide our arm component along the X direction, and one to move it along the Z direction.

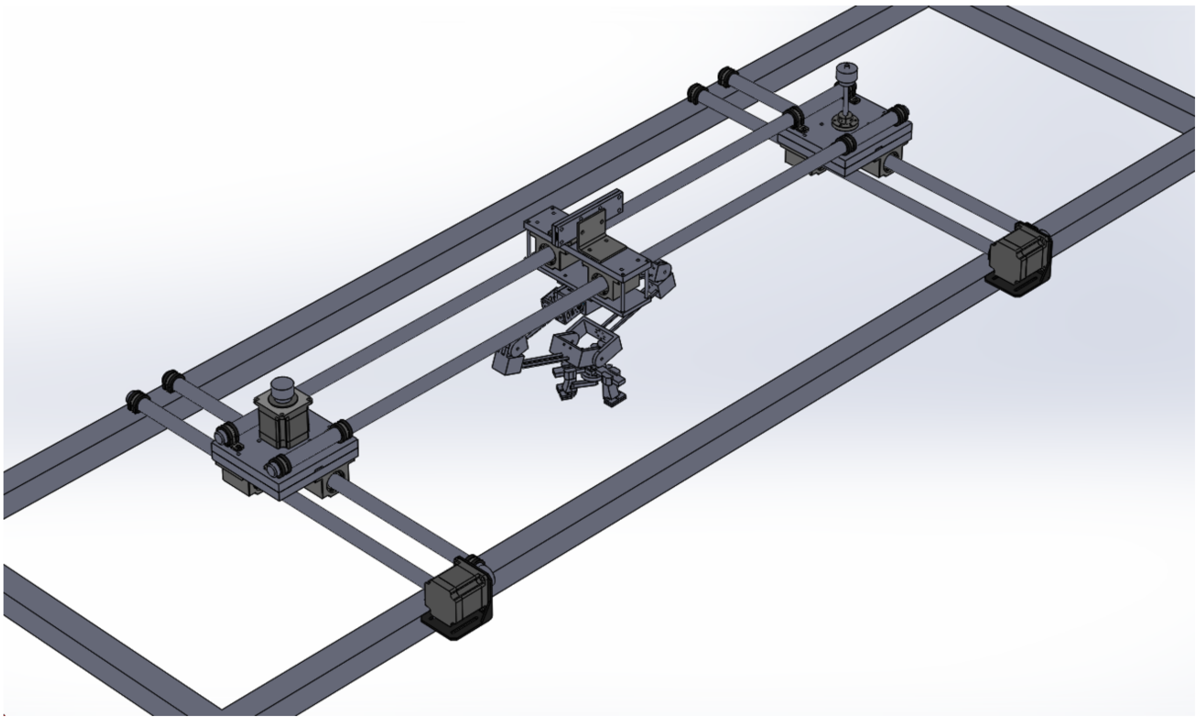

Figure 3.2: *CAD of Frame*

3.3.2 Arm

The arm, as shown in Figure 4.5 went through multiple different designs as we determined our parameters. We started by looking at linear actuators, as our goal for the arm is to have linear up/down motion to be able to quickly grab items and move them off the conveyor. Although linear actuators offered the motion and control we wanted, many were too slow. We decided that having an actuator that responded slowly would require too much travel time and more complex control systems to place the actuator in the correct position ahead of where the item would be in order to catch it before it can no longer be reached. As we planned to use computer vision to detect objects before they entered the robot's range of motion, the actuator's response time would have to be fast enough to pick up the object at a specific predicted point as the object actively changes position on the conveyor belt. A similar option we explored was pneumatics. A pneumatic actuator would be able to move quickly to the object, but unlike linear actuators, they only have two points at maximum and minimum length so they do not offer precise control. We wanted to have the ability to grab items both flat to the conveyor and raised, so we needed control over how far the arm went down to not crush raised objects or break the arm. This eliminated pneumatics as an option.

We ended up deciding that we would need to use a multi-degree of freedom arm. We wanted to minimize the amount of motors within the arm for easier control and to minimize the weight the first actuator would be moving, so we decided on a closed loop, five link arm. Two links off of each arm, joining at a center point, with the fifth "link" between the two actuators. By moving the actuators at the same time in opposite directions, the center point would move in a straight line. This design also gave us the possibility of moving the end effector slightly left and right if needed. At the joining point the end effector would be attached. Our original design used a motor to point the direction of the end effector. For simplicity the motor was removed and the final design instead relies on gravity to keep it pointing down.

3.3.3 End of Arm Tooling

Our end of arm tooling went through multiple design changes as well. The Final Design is shown in Figure 4.6. We explored different options such as a basic claw and a suction cup. Originally, as we decided our robot would be focusing on removing cardboard from a conveyor line, we believed a suction cup would be a good option for handling them. However, cardboard waste can come in different shapes; some can be too deformed and some can be too small to pick with a pneumatic system . Moreover, none of the other parts of our design require pressurized air, so we decided adding that requirement just for the tooling would add unnecessary complexity. Instead, we opted to design a novel gripper, meant specifically for paper and cardboard picking. The gripper was designed similar to a parallel gripper, but with three prongs instead of two, in a triangle shape.Each was pulled towards the center by a disk with links to each arm. Each arm featured small spikes on the bottom. Our goal was to be able to quickly pierce the cardboard to grab it, and then move it from the conveyor. We did not need to be concerned about slightly damaging the cardboard, so piercing was our best option to be able to handle the different sizes of material with a single gripper.

3.4 Control System

Our control system was designed with modularity and future extensions in mind. To successfully identify and sort through recyclable waste, many coordinated operations would need to take place. Due to the wide range of tasks available in the sorting process of recycling facilities, developing an adaptable and safe system is important.

3.4.1 Software

With the goal of maximizing support and modularity of the robot, “Robot Operating System”, known as ROS, was selected as the base of the software component of the control system. ROS provides a software infrastructure that allows component-wise programming of the system. Each system component is developed as a ROS ’node’ which are easily connected by sending each other data packets, using formats defined with ’messages’ and ’services’. ROS makes this communication seamless, as a single device runs the ROS Master and all the clients connect to it. Then, all communication is routed automatically. In addition, ROS provides smooth integration with various hardware components including the sensors and actuators used in this project.

Python was the preferred language of our team due to prior experience with ROS in WPI RBE courses. Python3 was initially investigated as an option, due to python2.7 reaching end-of-life in January 2020, but due to dependencies and support, python2.7 was selected. The specific version of ROS selected also influenced this decision, as no version of ROS (excluding ROS 2) natively supported python3 at the start of this project. Fortunately, all components did support ROS Melodic, the newest version of ROS at the time.

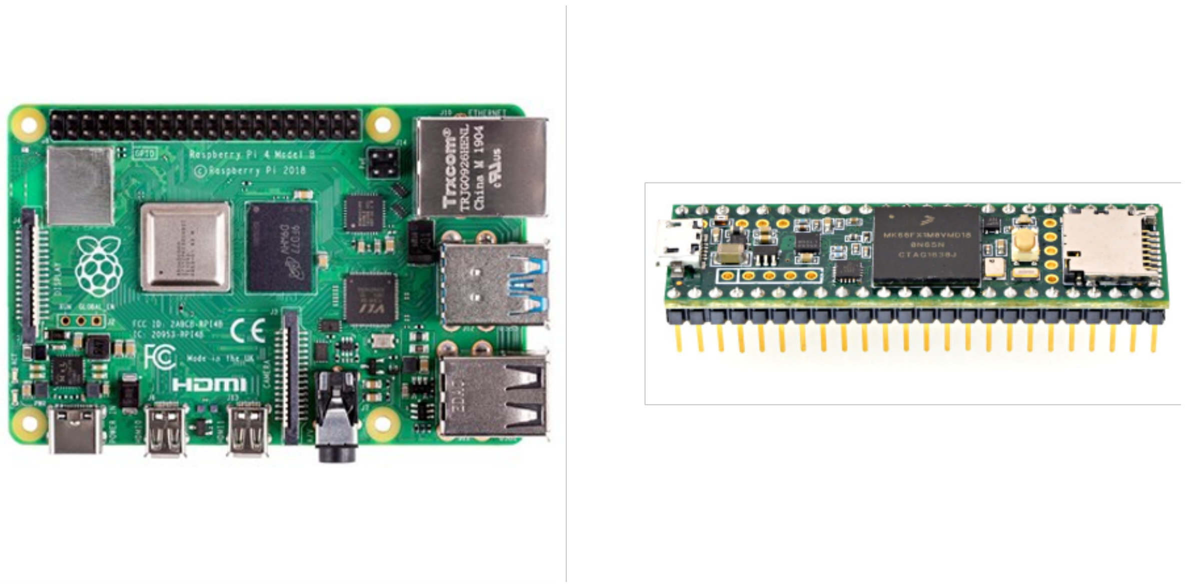

Figure 3.3: *Raspberry Pi 4B (left) and Teensy 3.6 (right).*

The primary controller in our system was a desktop PC, which would run the ROS master to coordinate the rest of the system. It would also be responsible for the inverse kinematics calculations, vision processing, and arm and gripper control through the Dynamixels. To allow for the future addition of sensors, a Raspberry Pi running ROS as well would be connected over Ethernet to the desktop. This would allow the addition of sensors that have Raspberry Pi oriented or compatible libraries available. And finally, to allow access to lower-level digital I/O without the overhead of an operating system, a Teensy 3.6 was selected to communicate with additional systems. This would allow the use of interrupts and real-time sensing of limit switches, safety states, and high-precision control of the stepper motors that control the X/Z position of the arm base. The Raspberry Pi and Teensy are shown in Figure 3.3 [26, 27].

3.4.2 Actuators

The Cartesian design of our robot requires precise linear control of the base of the manipulator. To accomplish this, stepper motors were selected to provide fast, precise, and powerful motion. Initially, ROBOTIS Dynamixels were considered due to their easy integration with ROS and its abstracted control libraries, but these actuators lacked the strength to be able to quickly start

Motor	Configuration	Amps/Phase	Holding Torque (kg-cm)	Est. Max rev/s
Pololu 1473	Bipolar	2	9	12
Pololu 1476	Bipolar	1	4	11.11
HT23-597	Bipolar Series	1.41	8.4	3.94
HT23-597	Bipolar Parallel	2.83	8.4	7.85
HT23-598	Bipolar Series	2.12	11.4	4.97
HT23-598	Bipolar Parallel	4.24	11.4	10.11

Table 3.1: *Subset of Motor Options, HT23-*** are Applied Motion NEMA 23 Steppers*

and halt motion of the system. This was not a worthwhile trade off as it would have significantly inhibited the speed of the pick-and-place motions that are required for efficient waste sorting. Therefore, in order to achieve our desired performance, our focus was put on selecting appropriate stepper motors at the expense of simple ROS integration.

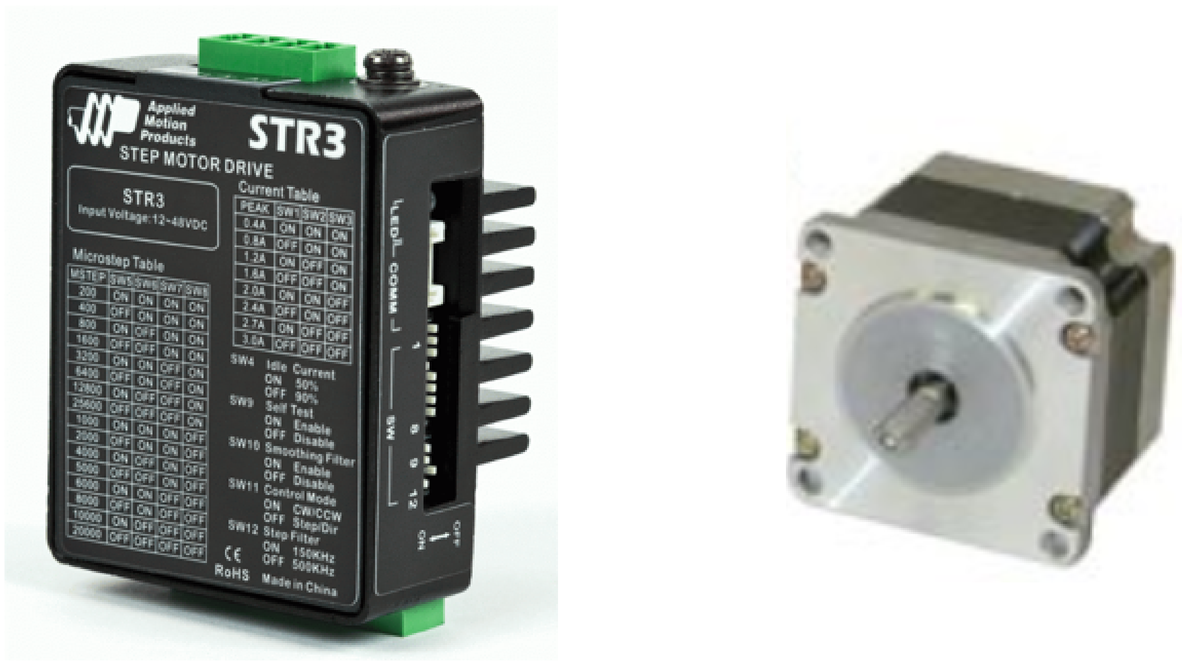

Figure 3.4: *STR3 controller (left) and HT23-597 stepper motor (right).*

Applied Motion offers a wide range of high-quality stepper motors compared to other alternatives available at Pololu, CircuitSpecialists.com, and other manufacturers and distributors. Various performance metrics were compared, including holding torque, speed ratings, and current draw. A subset of the considered stepper motors is shown in Table 3.1. We selected the Applied Motion HT23-597 motor, which is an 8-wire stepper motor and therefore allows for wiring the motor in a unipolar configuration, bipolar parallel configuration, or bipolar series configuration. Bipolar configurations are preferable due to their increased performance and efficiency, so our design planned for the bipolar series configuration but would allow for changing it to parallel if more torque was needed. Doing so would double the current draw but increase the holding torque and halve the resistance and inductance. The wiring design of the motors is shown in Figure 3.5, generated using EasyEDA [1]. These stepper motors would be driven by Applied Motion STR3 motor drivers, also shown in that figure. These allow for multiple different types of control, but we are most interested in simple digital STEP/DIR control, which is straight-forward to use on a micro-controller such as the Teensy. Images of the stepper motor and controller (not to scale) are shown in Figure 3.4 [28, 29].

The HT23-597 motor is available in a few variances, including with a dual shaft or an encoder. We chose to purchase the dual shaft model, as it would allow for the optional addition of an encoder if doing so was desired in the future. This dual shaft option has a shaft accessible both from the front of the stepper and the back of the stepper. This allows for easier mounting of encoders, as one side can have an encoder while the other side is connected to a belt or chain.

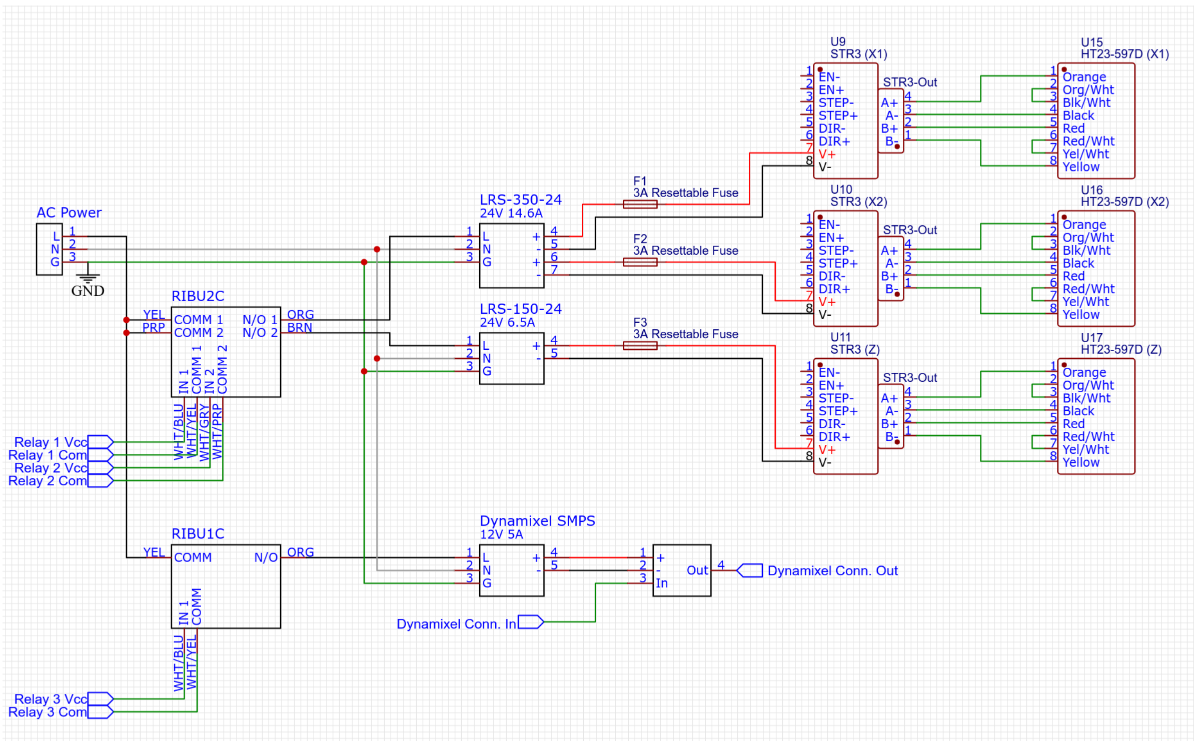

Figure 3.5: *Motor Power Distribution. Ground is common to all devices.*

To control the waste gripper component of the robot, ROBOTIS Dynamixel servos were selected, as it would not require continuous motion per the design. The Dynamixels provided built-in closed-loop control and can easily interface with ROS directly through a computer using the ROBOTIS U2D2 board. Additionally, the Dynamixels can be connected in series to allow for a single combined data/power wire to connect each Dynamixel to the next, resulting in only one wire connecting back to the controller.

[1]https://easyeda.com/editor

3.4.3 Sensors and Safety

A robot of this size and strength can be damaging to both people, objects, and/or itself if something goes awry. Therefore, various software and hardware safety measures must be taken to establish safe and well regulated operation.

To ensure safe, accurate actuation of the robot, proper sensing is required. Regulating the action of the motors was a simple task, as the stepper motors needed no feedback. All that was required were limit switches to prevent damage in the case of attempted movement past hard stops. Additionally, the limit switches would allow the robot to safely home itself by automatically checking all of its limits. The Dynamixel servos will not require any additional sensors as they already contain a closed-loop controller and the ability to set software limits, which are based on the motor's absolute position.

For safety intervention, the ability to manually stop or emergency stop the robot is also required. To accomplish this, research was done to find quality switches and appropriate relays. The safety system is designed with the assumption that it will be used purely in a testing environment by students who are fully aware of the robot's design parameters, behavior, and workspace. Therefore, the relays are standard AC relays and there are no additional sensors such as laser break beam sensors to detect a person entering the robot's frame perimeter. However, the E-Stop and other switches are designed and manufactured for safety systems and rated for use in industrial environments. The relays are shown initially in Figure 3.5 and shown wired in Figure 3.6.

Safe power distribution was also a concern during the design phase. Care was taken in specifying appropriate wire gauges for the various wires needed for the system. Additionally, resettable fuses were included in the design to prevent the motors from drawing too much current in the case of a malfunction or unforeseen issue. These fuses are shown in Figure 3.5.

Of concern while planning wire gauges and cabling was the actual routing of these cables. The cart where the arm is mounted has a varying position, so simply tightly securing cables to mounting points was not an option. Instead, cable drag chains are required to allow parts of the robot to move without tugging on or pinching cables. Cables are routed through these chains, which fold and unfold over themselves as one of our carts moves closer to and further from the static point where cables are routed to and from.

During the design phase of the safety system, it was apparent that there was also risk of incorrect and dangerous behavior if any of the controllers lost communication with each other. This led to the inclusion of a watchdog timer in the software design for the Teensy controller that would halt motion if it lost communication with the Raspberry Pi. This is key for a system safety because the operator's start and stop buttons are connected only to the Teensy, so the rest of the system would not know their state if the Teensy disconnected.

These operator buttons consist of physical start and stop buttons that would be connected to the Teensy on interrupt pins to ensure the fastest possible response to a press without the risk

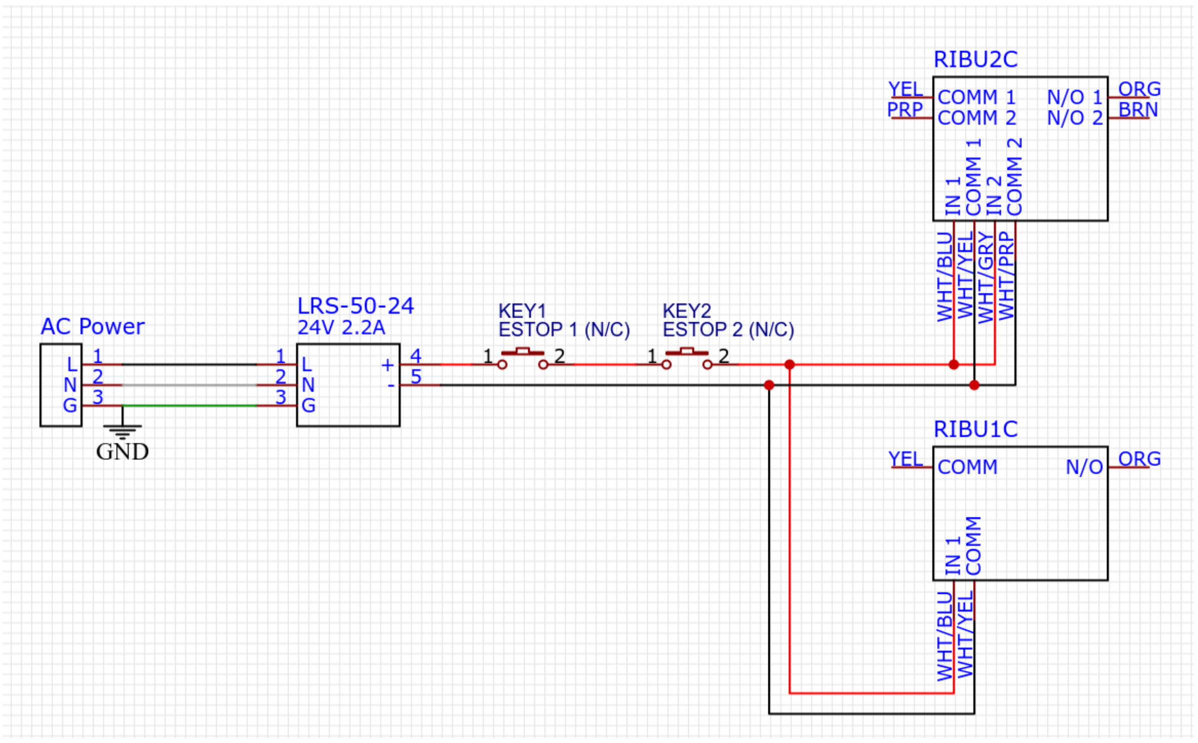

Figure 3.6: *Relay and E-Stop Wiring*

of missing a signal. The wiring of these buttons is shown below in Figure 3.7. No de-bouncing circuit is included in the design, as duplicate presses should have no negative side effects, as receiving a stop signal eight times will be functionally identical to receiving a single stop signal.

Unlike the operator buttons, the E-Stop would not be passed through any controller as that would defeat its purpose: to immediately interrupt all robot motion. Passing it through a software device would risk the chance of controller failure dropping the E-Stop signal. This is especially dangerous because misbehaving software on the controller is one of the most likely causes of unpredictable, damaging behavior. Instead, the E-Stop button will be wired normally closed (N-C) and then connected to the relays so that they are normally open (N-O). This design results in any break in the wires between those components or loss of power in the safety circuit to cause the relays to open and cut motor power. The only case where the motors would be able to operate is while the button is unpressed and power is applied to the separate, isolated safety power circuit connecting the button and relay. Additionally, the E-Stop button locks when pressed until it is manually disengaged by twisting it. Cutting power to the steppers may not always be the preferred option, as simply requesting the Teensy to halt motion would cause the steppers to lock up due to their nature as stepper motors. If the controller was still responsive, this would freeze the robot, while the E-Stop would likely result in a short glide. The combination of these features and options should allow the user to rapidly halt the robot and minimize damage both to the robot itself and its surroundings.

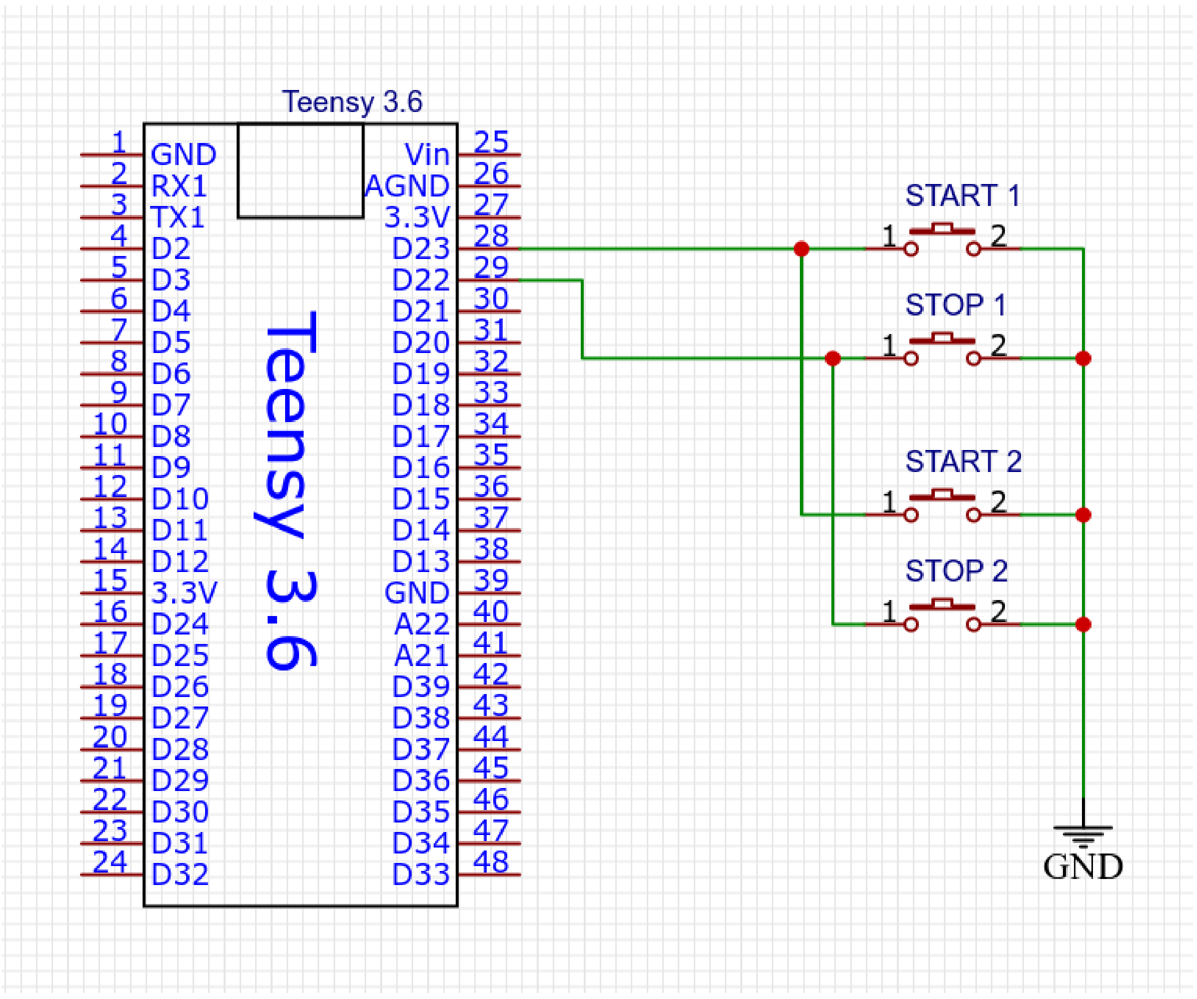

Figure 3.7: *Initial Start / Stop Operator Button Wiring*

Chapter 4

Implementation

This section describes what our team was able to build and program during the implementation phase of the project. Due to the effects of the COVID-19 pandemic, the implementation phase of this project was cut short and testing was not possible. Therefore, this section substitutes for a results and discussion section, and will describe what was able to be accomplished in the limited time frame. Following the explanation of the implementation will be a section describing recommended steps for continuing this project in the future (Chapter 5).

4.1 Robot Construction

Our robot comprises three main components: the attachment frame with the movable Cartesian platform, the robotic arm, and the gripper. We designed these components in SolidWorks and built the following assembly in Figure 4.1.

4.1.1 Frame

We constructed our frame using McMaster Carr steel weld-together rails to create a strong, immobile structure, capable of withstanding a lot of force from our motors. Prior to assembling our frame, we completed a welding tutorial provided by a staff member from WPI's Washburn Labs. We then welded the overall frame together and drilled holes into the frame in order to mount the linear-motion shafts onto the rails. Using five 2.5" x 7.5" x 0.5" aluminum stock, we machined a total of three carts that function as the plates for the linear motion shafts, attached through mounted linear ball-bearings. Two of the carts are identical and sit parallel to each other, holding two perpendicular linear motion shafts between them, upon which the third (mid-)cart is situated, as seen in Figure 4.2. The first two carts consist of a base plate and a support plate,

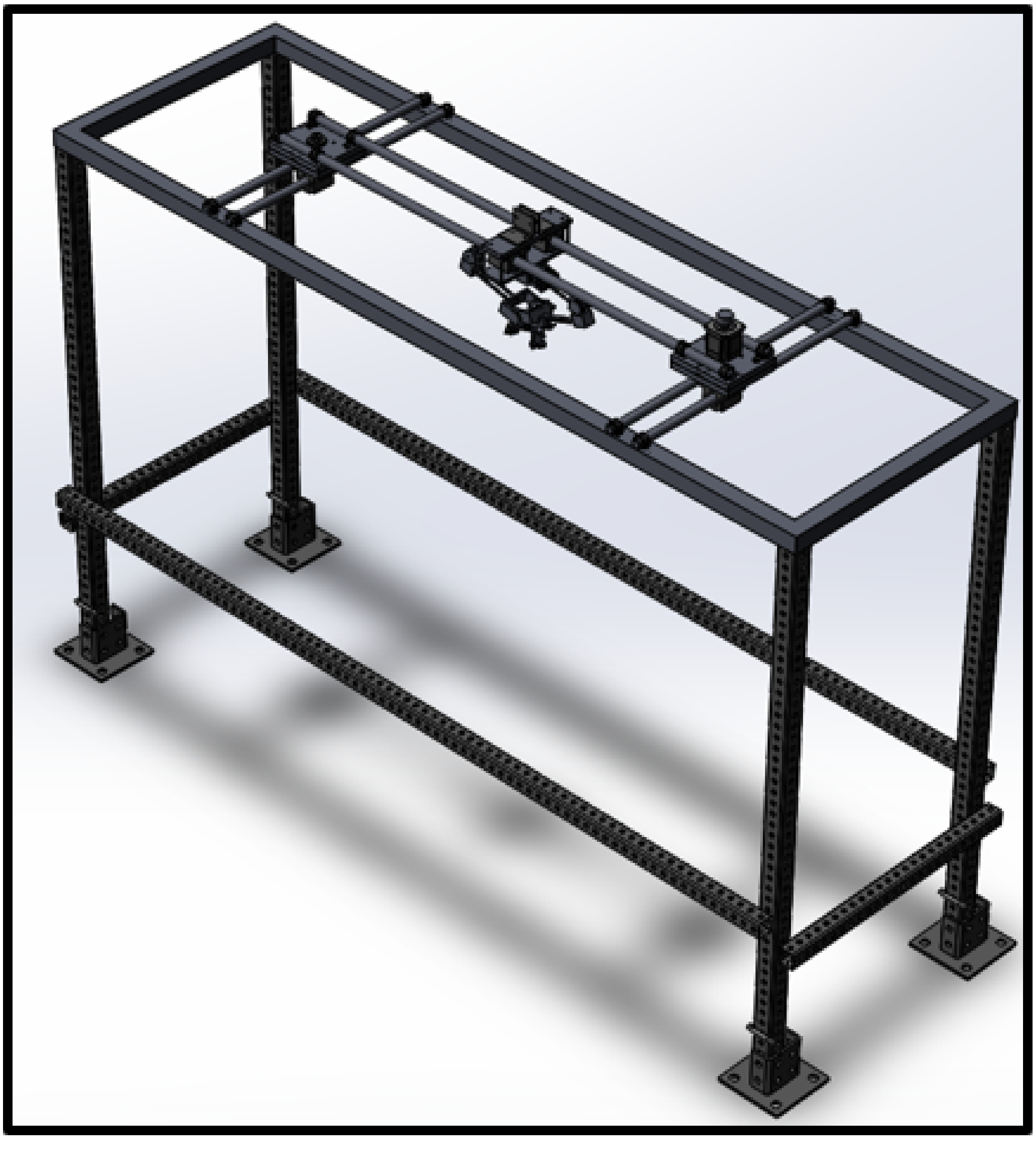

Figure 4.1: *Full Assembly of Robotic Structure*

each measuring 14cm x 13cm x 1.35cm. We used Esprit CAM to convert our SolidWorks CAD models into machinable entities, and sent it to the Washburn minimills to cut out the shape and drill each hole. Our last step was to hand-tap the holes for each cart, thereby creating threads to enable screwing or bolting our pieces together.

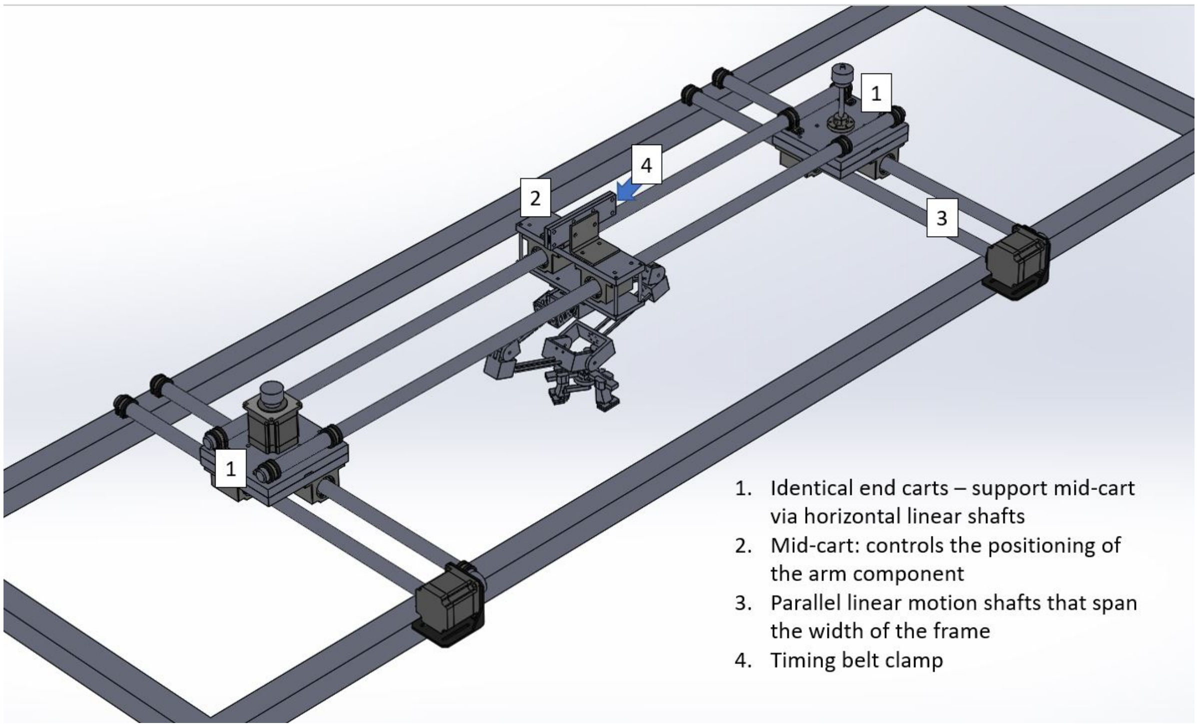
1
2
3
4
1
1. Identical end carts – support mid-cart via horizontal linear shafts
2. Mid-cart: controls the positioning of the arm component
3. Parallel linear motion shafts that span the width of the frame
4. Timing belt clamp

Figure 4.2: *CAD of Frame with Part Identification*

Upon each cart we attached a set of accessories necessary for the Cartesian movement. Figure 4.3 shows a close-up of one of the end carts, with each major component identified. Beneath each of the two identical carts are two mounted linear ball-bearings, offset to maintain balance upon the linear shafts. Situated on top of the end carts are the horizontal linear shafts that connect the two, held in place with vibration damping loop clamps. As can be seen in Figure 3.2, one of the end carts has a vertical NEMA 23 stepper motor held in place by standoffs (not shown), while the opposite end cart has a flange-mounted shaft collar upon which a vertical D-profile rotary shaft is mounted in order to hold an aluminum-insert, nylon timing pulley. The pulley system is connected through a 0.5 inch fiberglass-reinforced neoprene timing belt (not shown), ideal for quiet movement and light drive. Attached to the middle of this particular pulley system is a timing belt clamp, shown in Figure 4.4, connected to the third cart (the mid-cart). This third cart is situated on top of the horizontal linear shafts and serves to provide movement in the Z-direction. The stepper motor drives the system, controlling the positioning of the mid-cart, and thus the arm base.

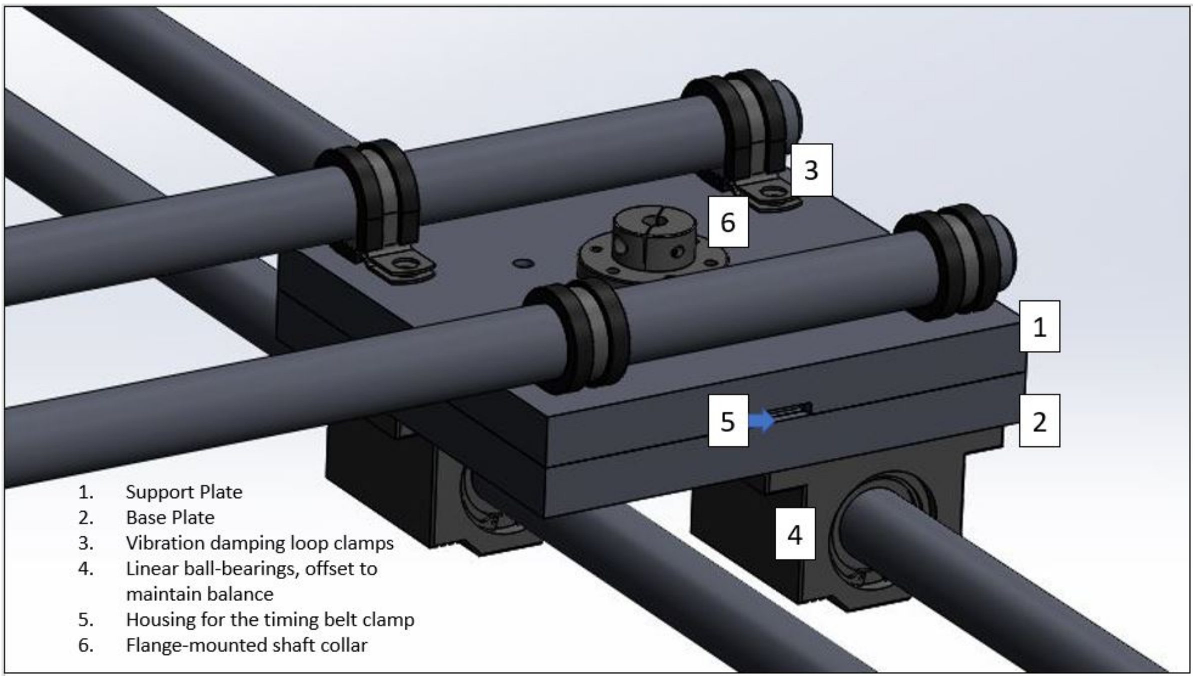
3
6
1
5
2
4
1. Support Plate
2. Base Plate
3. Vibration damping loop clamps
4. Linear ball-bearings, offset to maintain balance
5. Housing for the timing belt clamp
6. Flange-mounted shaft collar

Figure 4.3: *Close-Up CAD of End Cart with Part Identification*

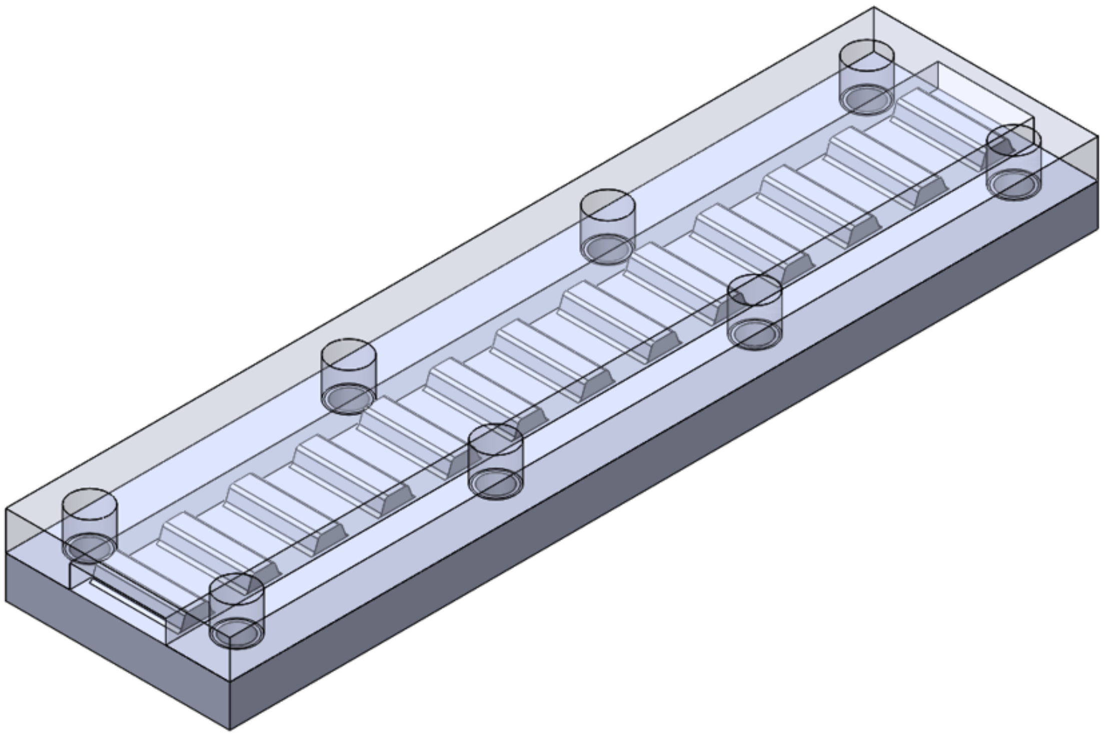

Figure 4.4: *CAD of timing belt clamp*

Nested within each of the end carts is another set of timing belt clamps that connect the end carts to a pair of stepper motors situated on the outer rails of the frame, as seen in Figure 3.2. We utilize two NEMA 23 stepper motors to move each end cart using a timing belt pulley system, with the two identical carts moving in parallel. Overall, we utilized three belt drives for our frame – two across the width of the frame, and one along the length. The belt along the length of the frame serves to distinguish the work space of our system and controls the positioning of our arm component.

4.1.2 Arm

The arm was constructed with a combination of 3D printed parts and parts ordered from McMaster-Carr and ROBOTIS. The arm uses three motors, one on each side of the arm and one at the center point that would control the rotation of the end-effector. Six custom parts were designed in SolidWorks. Four of the pieces are hinges and their connector pieces, to serve as joints at the corner of each side of the arm. A special joint was designed to go around a center motor, allowing the motor to have the end-effector pointed straight down while also allowing the center joint to fold up. Between the joints were 5 and 6 inch standoffs that connected them to give the arm a length of around 10 inches from the base to the tip when both motors are fully extended. The base has the two arm controlling motors connected together and attached to the middle cart by a single 3D printed piece that holds both motors.

After the initial mounting of the center joint motor, it stopped working due to technical issues. Because of time constraints, the center motor was removed and two more custom parts were designed, one to replace the special joint around the motor with a much simpler joint, and another piece to attach the end of arm tooling to the center joint.

4.1.3 Gripper

The gripper was assembled from 3D printed parts and acrylic designed in SolidWorks. Any parts that could be simplified were designed to be the same width and laser cut out of the acrylic, such as the Y rail. Any other parts were printed with ABS plastic. The spikes on each arm are separate printed pieces and were reprinted out of carbon fiber after the original ABS plastic pieces broke during testing of the piercing mechanic. The Y rail, where the motor was mounted and the three arms slid over, broke twice due to accidental impacts. After the second incident, it was redesigned to strengthen the rail that had broken.

4.2 Control System

The control system we have implemented aims to utilize all the available sensing information to execute optimal, precise, and safe movements. The two main components of this system are the software and the electronics. These two components combine to allow the multiple controllers

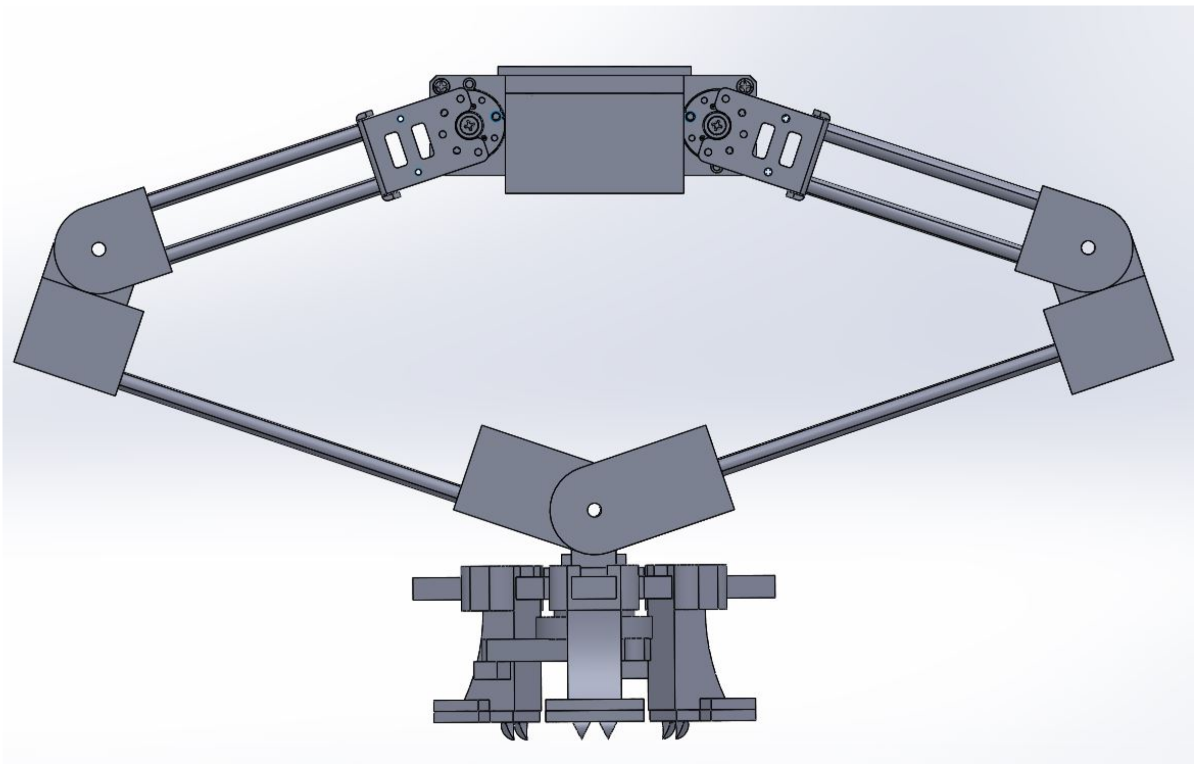

Figure 4.5: *The Five Link Arm, with the Claw Attached*

to share status information, sensor data, and command requests, then utilize their connected actuators to execute various movements.

4.2.1 Software

The software component of this project consists of three sub-components. The first runs on a desktop PC, the second runs on a Raspberry Pi 4b, and the third runs on a Teensy 3.6 micro-controller. The first two components are written in Python and the devices host ROS nodes for the various services necessary to run the robot. The micro-controller runs C/C++ Arduino code that controls the stepper motors and interfaces with the lower level sensors.

4.2.1.1 ROS Components

Robot Operating System (ROS) acts as the core of the control system, facilitating all operations and collecting all the input signals. Figure 4.7 shows a complete overview of the currently implemented ROS node layout, including the use of the written messages and services. This figure illustrates the paths of communication and available functionality provided by our ROS nodes.

The ROS Master and the Dynamixel control node reside on the desktop PC. The current implementation solely conducts communication with the servos using the `dynamixel_control`

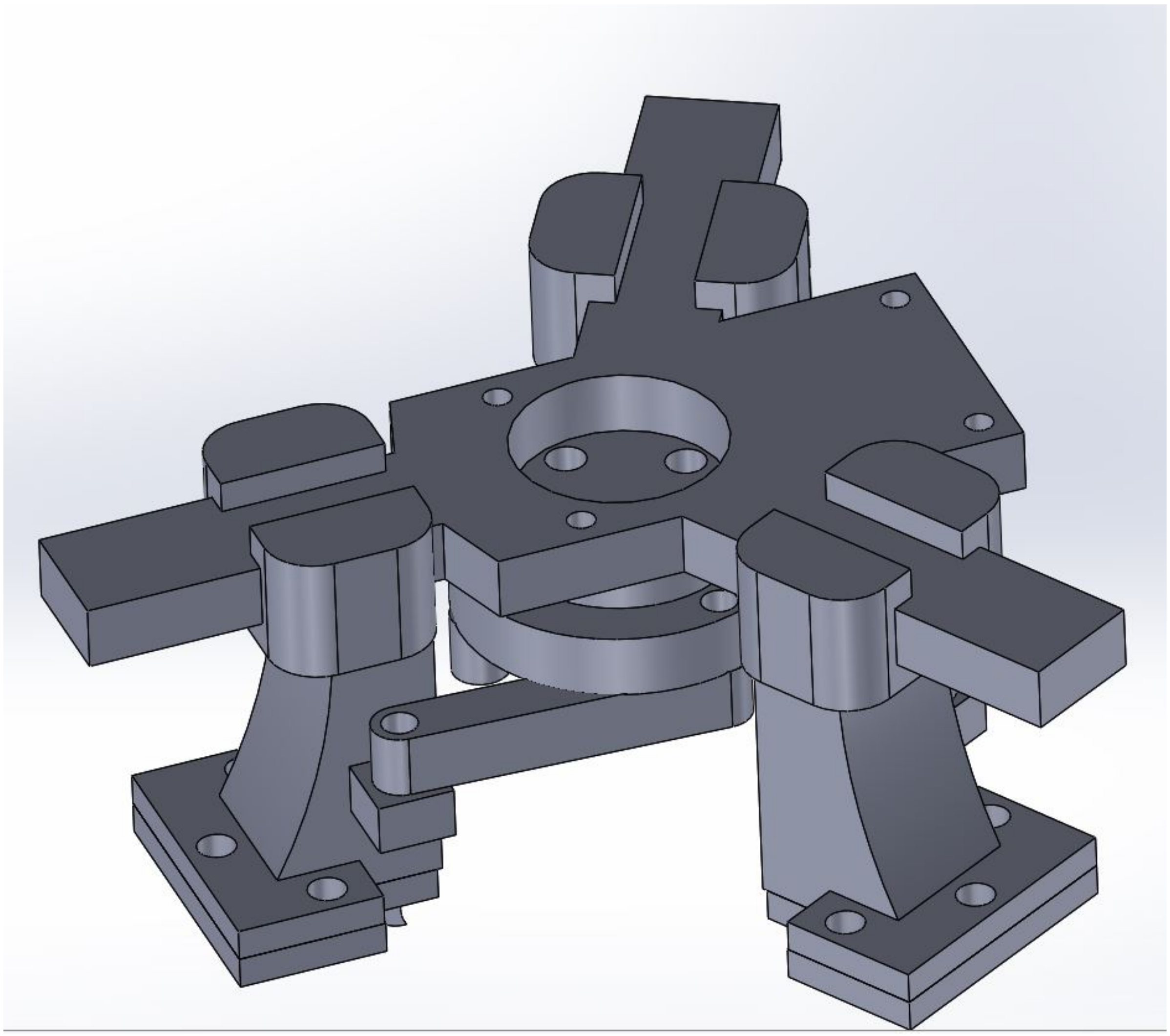

Figure 4.6: *The End of Arm Tooling, the "Cardboard Claw"*

node. This node creates two services, `arm_controller` and `gripper_controller` which control the two arm Dynamixels and the gripper Dynamixel respectively. This is done by sending requests to these ROS services of the types `ArmControl` and `GripperControl`. These ROS service types, shown in Figure 4.8, contain the arm position or the gripper state (open/closed). Each service request calls a respective function to set a new goal position for the arm or set the gripper to a predefined position. To provide feedback, this node also publishes messages (`ArmStatus` and `GripperStatus`) for the arm and gripper states which include positions and whether or not the motor is moving, as shown in Figure 4.9. For the gripper, it also includes whether it is considered open or not to prevent the reader of the message having to calculate that.

In order to more easily control and read from the Dynamixels themselves, a wrapper class was written to abstract the lower level communication with the servos. This class, `DynamixelMotor` provides functions to get and set the goal position, enable and disable both torque and the

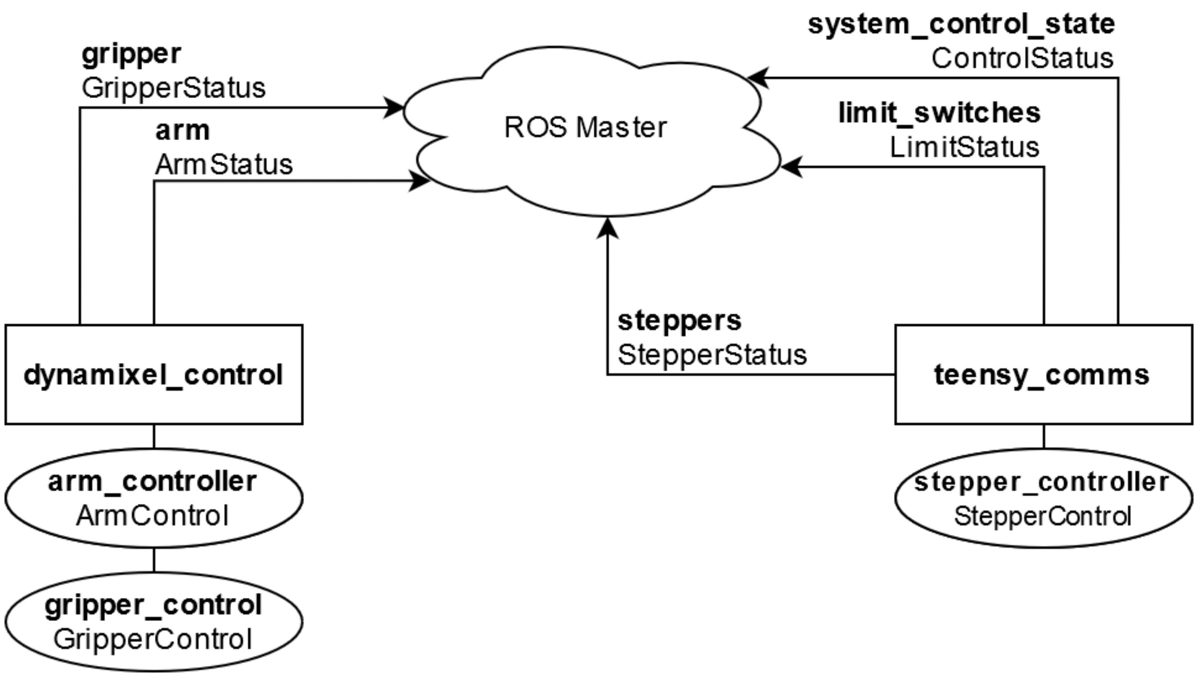

Figure 4.7: *ROS Nodes shown as rectangles. Messages are illustrated using arrows indicating publishing, labeled with the bolded name and message type. Services are illustrated using ellipses, labeled with the bolded name and service type.*

`ArmControl.srv`

```
uint32 position
---
uint8 success
```

`GripperControl.srv`

```
uint8 open
---
uint8 success
```

`StepperControl.srv`

```
int32 position_x
int32 position_z
---
uint8 success
```

Figure 4.8: *ROS Services*

`ArmStatus.msg`

```
uint32 arm_a
uint32 arm_b
uint8 arm_a_moving
uint8 arm_b_moving
```

`GripperStatus.msg`

```
uint32 gripper
uint8 open
uint8 moving
```

Figure 4.9: *Arm and Gripper Status Messages*

StepperStatus.msg	LimitStatus.msg	ControlStatus.msg
`uint8 enabled`	`uint8 limit_x_min`	`uint8 system_state`
`uint8 x_aligned`	`uint8 limit_x_max`	`string state_message`
`int32 stepper_x_pos`	`uint8 limit_z_min`	
`int32 stepper_z_pos`	`uint8 limit_z_max`	
`int8 stepper_x_dir`		
`int8 stepper_z_dir`		

Figure 4.10: *Teensy Data Messages*

LED, get the current position of the servo, and check if it is currently moving. Additionally, a control table file is provided, which has the custom `XC530W150` class containing all of the addresses needed to get and set select settings and states on the controller. Addresses that were foreseen as useful are included in the file, allowing for simpler addition to the `DynamixelMotor` class. To facilitate additions, general read/write wrapper functions are also contained in the `DynamixelMotor` class.

To bridge communication between the Teensy micro-controller and ROS, the other node in the current implementation runs on the Raspberry Pi 4b. This node, `teensy_comms`, is responsible for that functionality. It utilizes basic serial communication over USB at 115,200 Baud with data transmission every 20 milliseconds. The ROS node sleeps for only 15 milliseconds after each transmission to prevent missing the start of the next transmission. The data is transmitted using raw bytes, not string representations, so Python's `struct.pack` and `struct.unpack` functions are required to encode and decode status information and commands to and from the Teensy. When data is received, the node publishes the status of the stepper motors and limit switches to make the data available to other nodes. These message types are shown below in Figure 4.10.

First, the status of the stepper is published with the type defined in `StepperStatus.msg`, containing positions, directions, x axis alignment, and the enable state. The z-axis alignment value indicates if the two steppers controlling the x-axis position are at the same step or not, which could be used to alert the user to potential misalignments or step skipping. The current implementation logged error messages to ROS to indicate either position de-synchronization or opposing reported directions. These messages were throttled to prevent errors from flooding the console due to the higher speed of this node. All other log events in the node are either throttled by time or throttled if identical.

For the limit switches, the eight states, one per sensor, are reduced to four by marking them as triggered if at least one of the redundant pairs reads as triggered, which is published as the type specified in `LimitStatus.msg`. All eight states are transmitted to allow the system to detect if there are inconsistencies between them, alerting the software to possible problems. If an impossible case, such as both minimum and maximum range limits being triggered, an error is logged through ROS. Lastly, the control state of the system is published, containing the

```c
typedef struct packet_send {
    uint8_t control_state;
    uint8_t enabled;
    uint8_t limit;
    int32_t stepper_positions[3];
    int8_t stepper_directions[3];
} packet_send_t;
```

Figure 4.11: *Teensy/Pi Data Packet C Struct*

system state and a string representation of the state. This would be dictated by the use of the start/stop/e-stop buttons on the control boxes if they had been completed. However, a disconnect handler was implemented, where the state is updated to "Stopped (Disconnected)" to prevent continued motion if the connection is lost.

While unimplemented, it is worth mentioning the third written service that is intended for use to control the stepper. This service is registered by the `teensy_comms` node and would handle sending positional instructions to the Teensy. The service consists of an x position field and a z position field, is are shown in Figure 4.8.

4.2.1.2 Teensy Controller

The Teensy 3.6 Arduino-compatible micro-controller is responsible for all digital I/O and control of the stepper motors. The `AccelStepper` and `MultiStepper` classes are used to control the stepper motors, which were developed and made open source by Mike McCauley[1]. The `MultiStepper` simply allows for control of multiple `AccelSteppers` together, which is used to control the two steppers providing x-axis motion. Importantly, for compatibility with the stepper motor controllers used, all the control pins had to be inverted. All that was completed in our limited time was rudimentary testing control of the stepper motors, which was successful, but only tested with the z-axis motor.

As previously mentioned, communication with the Raspberry Pi is done using raw serial over USB. Below, the breakdown of the "packet" is shown in Figure 4.12 and the C struct can be found in Figure 4.11. This packet is preceded by the string "<status>" which is intended to allow for detection of lost bytes. Following this string is a 20-byte sequence containing status information. Currently, this communication is one-way; it only reports status information to the Raspberry Pi and does not accept any incoming commands.

The first three bytes, shown in green, contain the integer control state, boolean enable state, and limit switch byte. Left to right, the limit switch byte contains the state of every limit switch, shown in Figure 4.13. Each switch is represented as a single bit, 0 or 1, indicating unpressed or pressed respectively. Padding bytes are created automatically due to the C struct, and are present

[1]http://www.airspayce.com/mikem/arduino/AccelStepper/

1	2	3	4	5-8	9-12	13-16	17	18	19	20
ctrl state	enabled	limit	pad	stp1 pos	stp2 pos	stp3 pos	stp1 dir	stp2 dir	stp3 dir	pad

Figure 4.12: *Teensy/Pi Data Packet*

8	7	6	5	4	3	2	1
z_max_b	z_max_a	z_min_b	z_min_a	x_max_b	x_max_a	x_min_b	x_min_a

Figure 4.13: *Limit Switch Byte Format*

as byte 4 and 20. Bytes 5 through 16 contain the positions of the three stepper motors as 32-bit integers. This is immediately followed by bytes 17 through 19, which contain 1-byte directions of the stepper motors. These directions are either -1, 0, or 1, indicating reverse, stopped, or forward, respectively.

4.2.2 Actuators

The robot consists of two types of actuators, the stepper motors and the Dynamixels. The set of Dynamixels is made up of three XC430-W150-T servos. The arrangement of these motors was discussed in Section 4.1.2, and electrically the setup is very intuitive. They are powered using a 12V 5A power supply connected through a ROBOTIS SMPS2Dynamixel power injector. The injector supplies current to the motors using the same cable bundle that carries control data. In order to control the motors, a small controller board, the ROBOTIS U2D2, is used to interface between the computer and the motors.

For the X and Z axis motion, three Applied Motion HT23-597D stepper motors are used. The first two of these are moved together to provide even motion along the X axis. The third is used to move the arm along the Z axis. These are powered using a pair of 24V power supplies; one 14.6A for the pair of X axis motors and one 6.5A for the Z axis motor. In order to control these motors, three Applied Motion STR3 controllers are used. These allow for step/direction style control using digital signals from the Teensy.

4.2.3 Electronics

Additional electronics will be needed to complete the system in the future. Most control components were acquired prior to the halt of our project, but the system as a whole is physically incomplete and untested. To review the overall design schematics, refer back to Figures 3.5, 3.6, and 3.7.

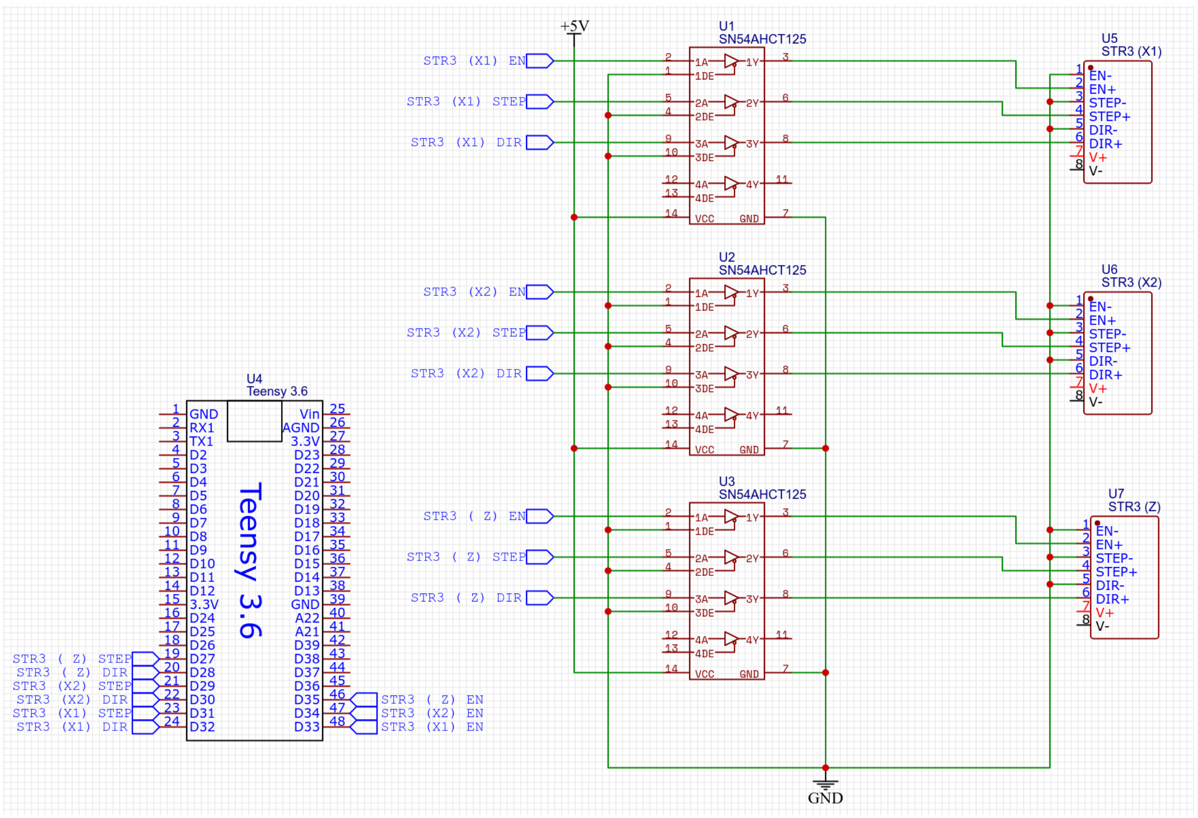

Figure 4.14: *Teensy to STR3 Wiring. Ground is common to all devices.*

4.2.3.1 Controllers

As previously discussed in the software section, a desktop PC, a Raspberry Pi 4b, and a Teensy 3.6 are used for control. As of the end of our project, the desktop was unused as code was run for testing off of a laptop. The Raspberry Pi communicates to the computer over an Ethernet network connection and communicates to the Teensy over USB. The Raspberry Pi is currently only responsible for transmitting data between the computer and the Teensy.

The Teensy controls the steppers using step, direction, and enable digital signals for each STR3 controller. These controllers expect a 5 volt signal, but the Teensy uses 3.3 volts for its logic. To remedy this, three 3.3V to 5V digital logic level converters are used (74LVC245). Each of the three sets of digital lines from the Teensy go through three of the four available pins on each of these integrated circuits before reaching the STR3 controllers. The circuit design for this is shown in Figure 4.14. The actual connections to the SN54AHCT125 ICs may be switched around on the actual assembly, but the relationships of EN to EN, STEP to STEP, and DIR to DIR are maintained. In contrast, the limit switch circuits do not require the logic level shift, as they operate solely on the Teensy's logic supply. When open, an internal pull-up resistor in the Teensy pulls the signal high, and when closed, it is grounded. This is shown in Figure 4.15.

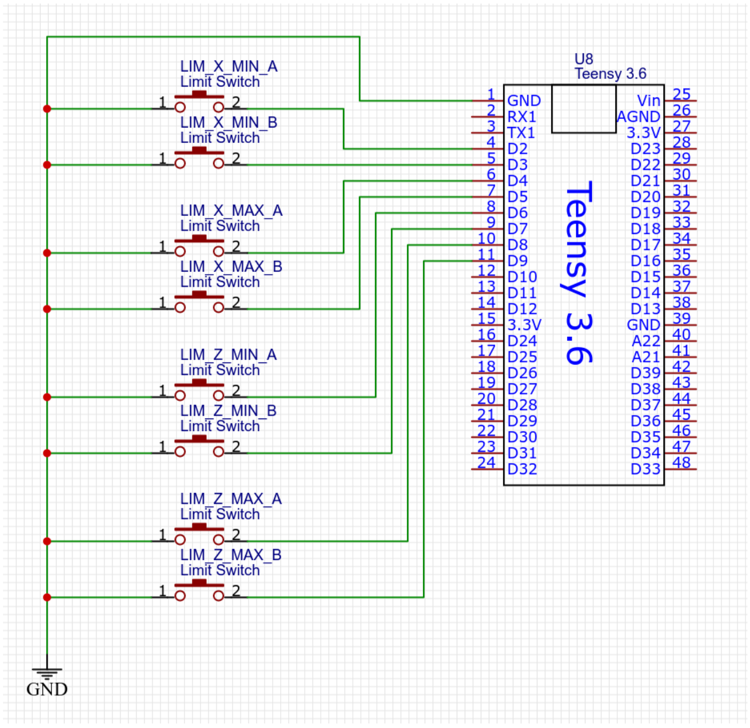

Figure 4.15: *Teensy to Limit Switch Wiring*

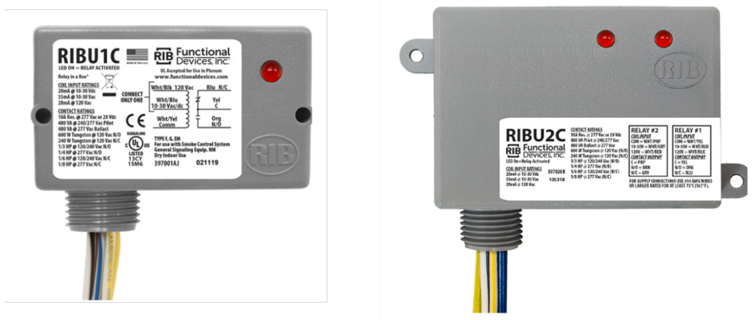

Figure 4.16: *RIBU1C and RIBU2C Functional Devices Relays*

4.2.3.2 Safety

The majority of the necessary safety components were purchased for implementation. We chose to purchase the control and E-Stop buttons from Newark, as they provide quality 24VDC buttons at a price appropriate for a research project. Pairs of each type of button were to be divided across two laser cut housings, providing the ability to control the robot from either side of it and also allow a second user to be able to have access to a separate E-Stop button. The E-Stop buttons would be wired in series to the AC relays we purchased, made by Functional Devices, to safely cut off motor power as described in Section 3.4.3. These relays are shown in Figure 4.16 [30, 31]. To prevent the stepper motors from drawing too much current, power supplied to the them were

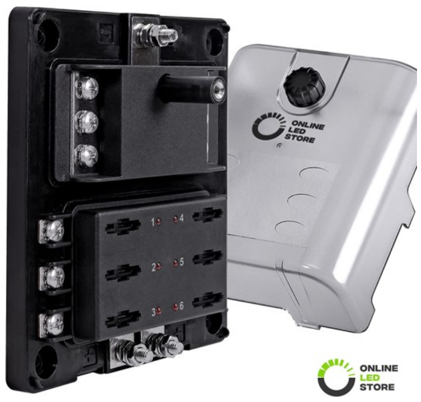

Figure 4.17: *6 Way Fuse Box by ONLINE LED STORE*

to be regulated by a small fuse box we purchased and populated with three 3A resettable blade fuses, one for each stepper motor. This fuse box is shown in Figure 4.17 [32].

Chapter 5

Next Steps

The unforeseen time constraints placed on our project limited the assembly of the robot and its control system and prevented a full code base from being written, as previously discussed. To continue with this task, the following section explains the next steps that we would have taken to complete the assembly and control system to the planned specifications. Future directions to pursue, such as utilizing machine vision, are included as well.

5.1 Assembly

Most of the robot has completed initial assembly. The frame and arm were in sections that were mostly completed. Electronic parts were the least completed, and are mostly parts that have been ordered and arrived but assembling was not started.

5.1.1 Frame Assembly

The frame is finished and standing, with the side carts and rails are assembled along with the stepper motors. The center cart and rail is assembled as its own piece but not connected to the side carts. The arm and it's plate needs to be mounted to the carts. The center cart and rail need to be connected to the side carts, and the motors need their belts to be properly attached. The claw needs a small bit of assembly and mounting, mainly with connected the disk to the motor and the spike plates attached to the three arms.

5.1.2 Cable Drag Chains

The cable drag chains were planned but never added. There are two chains, one for the side carts and one that will go from the side cart to the center cart. The wires for the stepper motor that

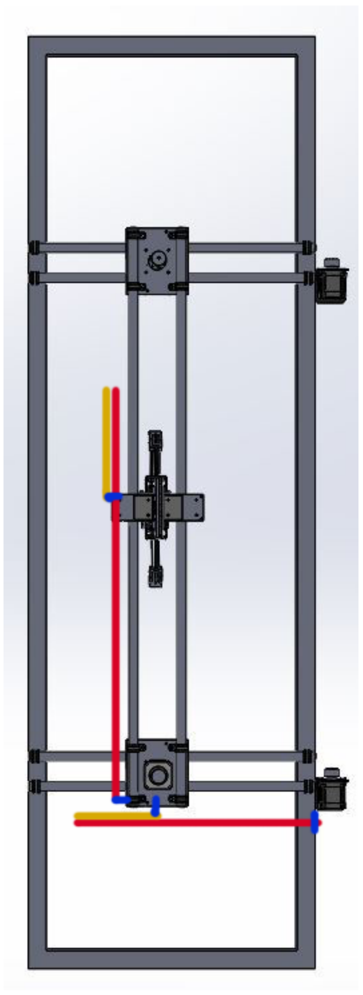

Figure 5.1: *Cable Chain Design Top View*

moves the center cart were to go through the first chain, and all of the wires for the arm should go through both chains. The chains only need mounting at 2 points. The first chain to the side cart should first be mounted to the frame, and then the other end to the side cart. It should for a sideways U shaped loop, with the end that goes to the frame longer, and then "rollover" back to the card. The second is mounted similarly, to the side cart and then to the second cart. As shown in Figure 5.1, where the red indicates the chain, the yellow indicates the chain rollover, and the blue indicates the mounting points. Figure 5.2 shows the cables from another angle, red is the full chain and the blue is the mounting points.

5.1.3 Control Boxes

The button boxes were designed and a first cut was done with excess acrylic from the claw rails. The original cut was slightly off for sizing, and a new button box was designed but not cut. These were designed to hold the EStop button and two other buttons for easiest access by the user.

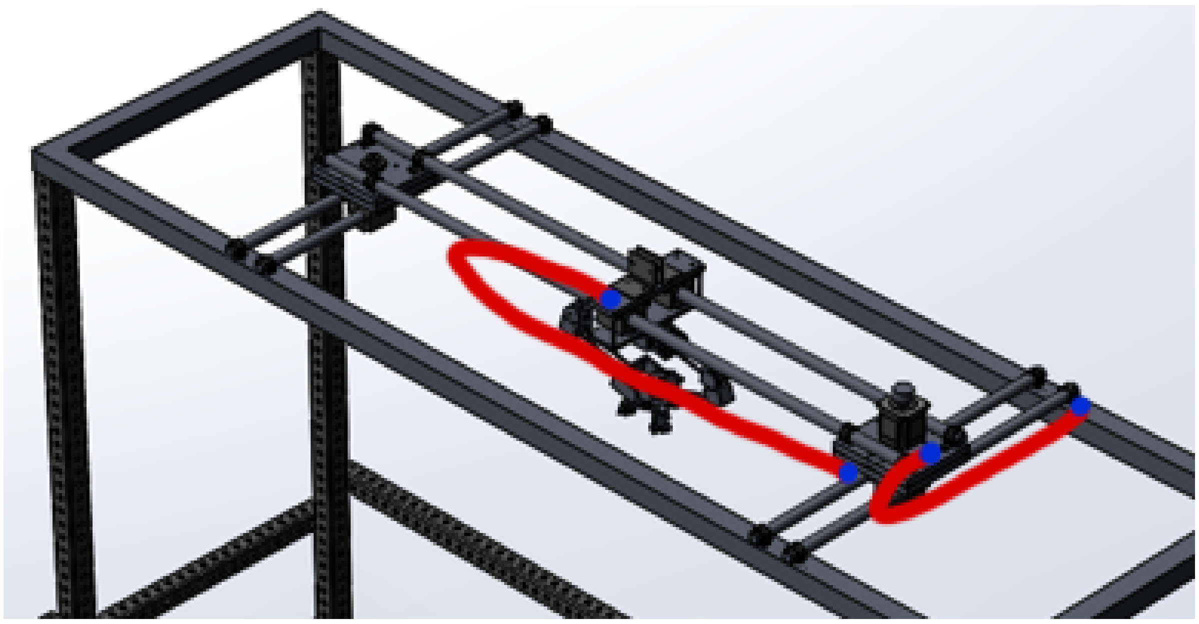

Figure 5.2: *Cable Chain Design Isometric View*

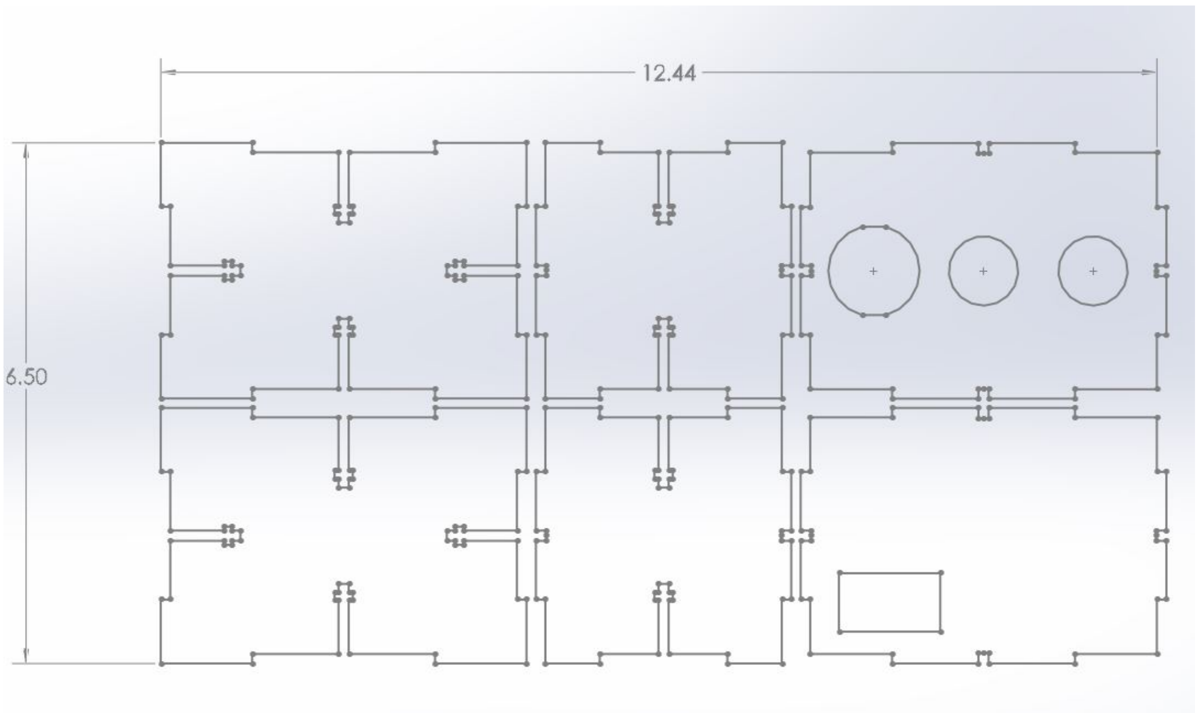

Figure 5.3: *The E-Stop Box Design*

5.1.4 Conveyor Belt

The conveyor belt should be an early focus. The conveyor requirements as outlined in Section 3.2 do not need to be met exactly if decided otherwise, but features like reversability and variable speed will make it easier for testing various system components such as object detection, identification and picking. Programming the robot to follow and then pick up the object as it is moving will be easier to start with the conveyor on a slower speed, and then increase it to show the robot's viability in a real MRF situation.

5.2 Electronics

Time limitations prevented the vast majority of wiring from being completed. Essential wiring for functional testing is provided, but cable lengths are not sufficient for the use of components while mounted. In addition, future work should consist of designing a layout for the control and power components and then designating a surface to mount them on.

5.2.1 Power and Common Grounds

The following paragraphs provide an overview of the voltages and grounds used across the various components, and should be considered when any additions or changes are made to the wiring. When wiring components, ensure all wire gauges are appropriate for the given operating conditions.

The various electronics of this robot run at three different voltages. Firstly, the Raspberry Pi supplies 5VDC to the Teensy 3.6 over USB. It is important to ensure that anything attached to the Teensy does not draw large amounts of current that exceed either an individual pin's tolerances, the board's tolerances, or the Raspberry Pi's USB current supply limits. Secondly, the Teensy and its digital I/O have an operating V_{CC} of 3.3 volts. For the control of the STR3 drivers, the digital signals from the Teensy 3.6 are converted up to 5 volts as discussed in Section 4.2.3.1. The STR3s support differential control for the Enable, Step, and Direction pins, but in the case of our wiring, the negative side is simply grounded to the Teensy's ground. It is important that these devices share a common ground, including between the 3.3V to 5V logic level converter (`74LVC245`) and both the Teensy and STR3s to ensure the signals are meaningful.

The Dynamixels run on their own circuit with a V_{CC} of 5VDC. The only external interaction with this circuit is for the safety relays as shown in Figure 3.5. Lastly, for the stepper motor power, two separate 24VDC circuits are used, as also shown in Figure 3.5. Additionally, the voltage used to control the relays through the E-Stop circuit is also a separate 24VDC system.

5.2.2 Wiring

The majority of the wiring of the robot needs to be completed to progress with testing and usage of the robot. Connectors on short cables have been crimped, allowing for usage of the stepper motors for testing. These short cables do not need to be replaced, as longer wires can be crimped and connected to the existing cables to extend their range. Once all components are mounted properly, the necessary length of the cables will be known and can thereafter be cut to length. Motor wiring will simply consist of adding these extension cables to the existing connectors on the drivers and steppers. For the Dynamixels, only a single combined data and power cable linked from one device to another in a daisy chain setup is required. Longer Dynamixels wires were purchased for extending from the arm to the outer superstructure. Testing of the usage of these long wires was not done, therefore it is important to verify that there is not significant voltage drop over the longer lengths of wiring. If this becomes a problem, the SMPS2Dynamixel should be placed closer to the motors.

The limit switches need to be wired directly to the breadboard containing the Teensy controller. The digital pins are configured to use the internal pull-up resistors, so one wire will extend from the digital pin to the switch, and another will return from the switch back to common ground on the breadboard. Similarly, the two pairs of start and stop buttons should be connected this way. However, the two start buttons and two stop buttons can each share a respective pin if wired in parallel, as shown in Figure 3.7. Alternatively, they can each be wired to their own digital pin. Pin choice is currently undefined in software, so any available digital pin is an option, as all support interrupts. It is important to configure these pins to trigger interrupts, instead of simply polling them, to ensure the signal is not missed, which is most important for the stop buttons.

5.2.2.1 Power Distribution

Power is supplied to the robot across four key power supplies. The first, is the enclosed switching power supply for the Dynamixels, which has a simple connection to the SMPS2Dynamixel board to provide power to the Dynamixel servos. It, along with the other motor supplies, can have its AC power cut off when the robot is emergency stopped. The details of this process are described in Section 5.2.3. No additional wiring should be required for this power supply, except for carefully routing the cabling to the mounting location of the SMPS2Dynamixel.

To provide power to the stepper motors, two 24VDC MEAN WELL switching power supplies are used. The first is a 14.6A supply for the two X-axis stepper motors. The two positive leads and one negative lead already connected to the power supplies should first connect to the fuse box. Only one negative lead is needed to connect to the common ground. The respective leads from the two X-axis STR3 drivers should then connect on the other ends of the fuses and to the common ground. All of these wires should be of at least 16 AWG. Similar wiring should be done for the second, 6.5A supply for the Z-axis stepper motor.

The final power supply is a small 2.2A 24VDC MEAN WELL switching power supply. This supply provides current through the E-Stop circuit, as that circuit interrupts AC power for all the other supplies if it is interrupted. This prevents utilizing an existing supply for this circuit. The details regarding wiring this circuit is mentioned below in Section 5.2.3.

5.2.3 Safety Components

Emergency stop functionality and status indication is critical to any robot capable of injuring users. A starting point for continuing development of this subsystem is purchasing a stack light to indicate robot status, clearly warning if the robot is active. This may be selected to run at 24VDC off of the safety power supply, mentioned at the end of Section 5.2.2.1. If so, DC relays will be required to control the lights using the Teensy, as using the Teensy for controlling the stack light is recommended. If this is wired off of the 24VDC safety supply, ensure it is not wired in series with the E-Stop circuit. If it is wired in series and the E-Stop is triggered, it would shut off the lights, preventing them from showing robot state.

The relays used for interrupting motor power during an emergency stop must be wired by a future team. This will consist of wiring the relays by inserting them between wall AC power and the DC power supplies, as shown in Figures 3.5 and 3.6. The latter figure explains where to wire the E-Stop buttons in the control circuit for the relays. The two E-Stop buttons must be wired in series, otherwise both would need to be pressed to cut motor power.

When wiring the safety system, it is important to understand how motor power regulation operates. Firstly, if the safety power supply is off, the relays will be open, preventing power from flowing to the motor power supplies. Once powered, as long as power is active in the safety circuit, the relays will be closed. Once one of the E-Stop buttons is pressed, this circuit will be opened, bringing the relays back open due to loss of power. It is critical that this behavior is implemented, as it is desired that if any problem happens with the safety circuit, it prevents the motors from having power. In summary, if any wire in the main safety circuit is cut, it will cause the relays to open, stopping the motors. If one of the offshoots to the particular connectors of the relays is cut, that specific relay will open, but the rest of the safety system will continue to operate normally. This configuration will maximize the safety provided by this safety subsystem.

Future teams are encouraged to extend upon this safety system. For a robot used in a real-world environment, relays marketed specifically as safety relays should be used in the place of these standard relays. In addition, more thought should be put into the overall safety of the robot, including the consideration of adding components such as laser break sensors to establish an invisible barrier around the robot that causes it to halt if entered.

5.3 Core Software and Controllers

The source code for this project provides a base for building more advanced applications and control sequences. The majority of the foundation of the system is implemented or outlined, and the process for completing this base is discussed in the following section. The development of higher-order control logic is left to future teams to conduct.

5.3.1 ROS and Python

One caveat to the provided code is that it is designed specific to Python 2.7, not Python 3. If possible, it is encouraged to change to using Python 3 along with ROS Noetic Ninjemys, which will natively support Python 3. This switch should only be undertaken if it is known that all system dependencies will be compatible with these new versions, namely the Dynamixel library.

5.3.1.1 Dynamixels

A control abstraction class, `DynamixelMotor`, has already been written to simplify usage of the Dynamixels. It can be easily extended by reviewing the documentation from ROBOTIS regarding the python Dynamixel library and by referring to the motor-specific control code file, `control_table.py` provided along with the abstraction class.

Careful testing will need to be preformed to verify the movement direction of and safe range for the Dynamixels. The results of this testing should be used to adjust tuning parameters, which could include the settings for the internal PID controller. Additionally, a proper relationship between the two arm motors must be determined to accurately control them simultaneously to achieve the desired positions. See Section 5.4.1 for more details about coordinated control.

5.3.2 Teensy

The Teensy 3.6 micro-controller is the core component of the control system responsible both for controlling the stepper motors and for transmitting signals from the control buttons. The Teensy utilized during testing was lent from a member of the team, so a new one will need to be purchased to continue the project. The breadboard used to seat the controller has been wired up to match the diagrams shown in Figures 4.14 and 4.15, and installation of a new Teensy simply requires properly inserting it into the breadboard. To interface with the control buttons, to read E-stop state, or for any other sensors, additional connections will need to be made. If the Teensy does not have pre-soldered pins, they must be carefully soldered on. The necessary pins have previously been purchased and are available for this purpose.

5.3.2.1 Stepper Control

Applied Motion STR3 stepper drivers are used to control the stepper motors. These support EN/STEP/DIR control, allowing for a micro-controller to only need to control three digital signals without working with more complex protocols. To abstract the actual signals sent to the drivers, the aforementioned `AccelStepper` library is used in driver mode. The configurations for utilizing this library are already implemented, including example usage. It is important to control the motor pair used for X-axis control using the `MultiStepper` class to avoid the motors from de-synchronizing their positions. To complete the base control, the steppers need to be moved according to information received from ROS through the Raspberry Pi. Additionally, acceleration profiles can be selected to tune the response of the stepper motors.

5.3.3 Communication

The currently implemented communication code between the Teensy 3.6 and Raspberry Pi is one way, only sharing status information from the former to the latter. This provides status information and stepper positional information to the Raspberry Pi to distribute into the ROS node network using publishers. To provide control of the steppers, the `teensy_comms` node needs to have its `stepper_controller` service implemented to receive commands of the format specified in `StepperControl.srv` (see Figure 4.8). This will likely need to be extended to provide more functionality. The packet structure for these commands has been left undefined for a future team to implement, along with the associated `encode()` function in the Raspberry Pi python code.

Communication between these two devices is done using raw bytes over USB serial to reduce the complexity of the communication code on the Teensy. Modifying and extending upon this will require knowledge and understanding of Python's `struct.pack` and `struct.unpack` functions. The bytes are sent using little-endian, so the `struct` function format strings need to start with the < symbol. Analysis will also be required to determine the padding of uneven byte sequences to ensure data is interpreted properly by the receiving device. It is unlikely that the communication will de'sync, but to detect any issues, a short header string is sent from the Teensy to the Raspberry Pi. If the start of a packet does not match the expected string, an error will be emitted. This string can be adjusted or optionally removed if it is decided doing so would be of no consequence.

5.4 Advanced Control

The entirety of the advanced control necessary to execute the high-level motions required to successfully identify, navigate to, pick up, and deposit waste items must later be implemented once the software base of the control system is completed and the wiring is in place. A brief overview of recommended topics to pursue are provided in this section.

5.4.1 Inverse Kinematics

The inverse kinematics calculations will be required to achieve proper control over the arm and the rail system. The inverse kinematics along with the vision detection will allow the system to follow the object and pick it up correctly even as it moves on the conveyor.

5.4.2 Machine Vision

Machine vision is currently the only reasonable option to quickly and accurately classifying a variety of recyclable waste out of a mixed waste stream. Our team purchased the Intel RealSense Depth Camera D435 to supply stereoscopic vision with depth sensing. This camera should supply more than what is required in terms of vision features. For the machine vision software itself, we recommend utilizing a powerful Python library such as TensorFlow[1], Keras[2], or PyTorch[3]. Utilizing Python will allow easier integration with ROS and the other scripts used in the software project.

Convolutional Neural Networks (CNNs) are one of the best available option to classify recyclables in this application. Extensive training will be required to successfully classify items in the streams, consisting of a variety of pictures under different lighting sources. Labeling data will pose a problem, due to time cost, so research should be done attempting to locate any available data sets that could prove helpful. For the unlabeled data that is collected specific to our application, one solution to the labeling problem would be outsourcing the process to student volunteers.

This training will require significant computational power, so the utilization of an available GPU compute cluster is encouraged. Optionally, to augment the training data, a Generative Adversarial Network (GAN) can be used to generate more training data. Once trained, the execution must be fast enough to classify these moving objects in time to move the gripper to pick them up. This may require purchasing a more powerful computer. Before proceeding with this task, extensive research should be conducted to understand the design constraints of this machine learning problem in order to identify an effective, efficient solution.

[1]https://www.tensorflow.org/
[2]https://keras.io/
[3]https://pytorch.org/

Chapter 6

Conclusion

Our team successfully conducted in-depth research on the current state of the global recycling industry to obtain a comprehensive understanding of the problems that are being faced. This included a visit to a local waste sorting facility to observe the processes in place. These investigations helped us to identify the needs of the industry, guiding us to design a robot fulfilling those requirements. We proceeded to design and begin the assembly of an X-Z Cartesian robot to precisely navigate a 3D workspace with the capability of identifying, moving to, and picking up cardboard out of a mixed recyclable stream. We developed and purchased components for a multifaceted control system consisting of three controllers, stepper motors, Dynamixel servos, a safe power distribution system, and a carefully planned safety system. We selected ROS as the software core of this control system and developed control abstraction functions and the basis of inter-controller communication. By providing direction for future work, in combination with our accomplishments, we have built the foundation of a robotic waste sorting project at WPI.

6.1 Future Work

Upon completing the tasks outlined in Chapter 5, efforts should be focused on tackling the problems of sorting other items (other than cardboard) out of recyclable waste streams. In doing so, the development and use of a system with modular grippers is encouraged. One good future focus is sorting through plastics, which could be done utilizing technology such as Near Infrared (NIR) reflectance spectroscopy to differentiate between different types of plastics [33]. In tandem to development of the robot and grippers, work should be done pursuing solutions to prevent job loss caused by the implementation of such automated systems in the recycling industry.

Testing this robot requires emulating a real-world recycling plant environment. While a single conveyor belt does seem to provide this, especially with a bi-directional belt, extended testing is still a challenge. In the case that the system is designed only to work in one direction (either due to choice or sensor limitations), the bi-directional movement would then only be beneficial for resetting the test bed. Along with this problem is yet another problem; the fact that this type of setup will keep all waste in the same relative position. To account for this, someone must manually move and distribute waste onto the belt.

A solution to this single-belt problem is using a cyclical system by combining four or more belts into a loop. This would allow for continual, one-way operation without a need for human intervention. To augment this system, some sort of mixer could be included in the loop, such as dropping waste into a funnel above another belt. This would provide a continuous, non-static stream of recyclable waste to route into the robot's workspace.

6.2 The Importance of Improving the Recycling Industry

As a final note, it is important to understand that the ever increasing rate of global consumption is continuing to produce more and more waste, overwhelming the systems in place to recycle all the mixed materials that go into modern products. Drastic changes need to be made to the global recycling industry in order to dampen environmental damage, and robotics offers an opportunity to make these critical improvements a reality.

The implementation of robotic automation in the recycling industry paired with high-performance, high-accuracy machine vision can significantly improve the speed and quality of the recycling process. However, as with any form of automation, it is important to consider the impact on the current human workers. An optimal solution to this problem should not only focus on the robot, but on maintaining the employment of the current human waste sorters by providing them new, safer assignments.

Bibliography

[1] C. Joyce, “Where will your plastic trash go now that china doesn’t want it?” *NPR*, 2019. [Online]. Available: https://www.npr.org/sections/goatsandsoda/2019/03/13/702501726/where-will-your-plastic-trash-go-now-that-china-doesnt-want-it

[2] S. Laskow, “Single-stream recycling is easier for consumers, but is it better?” *The Atlantic*, 2014. [Online]. Available: https://www.theatlantic.com/technology/archive/2014/09/single-stream-recycling-is-easier-for-consumers-but-is-it-better/380368/

[3] J. Margolis, “Mountains of us recycling pile up as china restricts imports,” *Public Radio International*, 2018. [Online]. Available: https://www.pri.org/stories/2018-01-01/mountains-us-recycling-pile-china-restricts-imports

[4] L. Albeck-Ripka, “Your recycling gets recycled, right? maybe, or maybe not,” *New York Times*, 2018. [Online]. Available: https://www.nytimes.com/2018/05/29/climate/recycling-landfills-plastic-papers.html

[5] S. Wong, “New world order,” *Recycling Today*, 2017. [Online]. Available: https://www.recyclingtoday.com/article/national-sword-china-plastics-recycling/

[6] J. Margolis, “China’s ’green fence’ is cleaning up america’s dirty recycling,” *Public Radio International*, 2014. [Online]. Available: https://www.pri.org/stories/2014-02-18/chinas-green-fence-cleaning-americas-dirty-recycling

[7] C. Katz, “Piling up: How china’s ban on importing waste has stalled global recycling,” *Yale E360*, 2019. [Online]. Available: https://e360.yale.edu/features/piling-up-how-chinas-ban-on-importing-waste-has-stalled-global-recycling

[8] D. Mosbergen, “Here’s why america is dumping its trash in poorer countries,” *Mother Jones*, 2019. [Online]. Available: https://www.motherjones.com/environment/2019/03/heres-why-america-is-dumping-its-trash-in-poorer-countries/

[9] A. Semuels, “Is this the end of recycling?” *The Atlantic*, 2019. [Online]. Available: https://www.theatlantic.com/technology/archive/2019/03/china-has-stopped-accepting-our-trash/584131/

[10] K. de Freytas-Tamura, “Plastics pile up as china refuses to take the west’s recycling,” *New York Times*, 2018. [Online]. Available: https://www.nytimes.com/2018/01/11/world/china-recyclables-ban.html

[11] “National overview: Facts and figures on materials, wastes and recycling,” *Environmental Protection Agency*, 2020. [Online]. Available: www.epa.gov/facts-and-figures-about-materials-waste-and-recycling/national-overview-facts-and-figures-materials

[12] B. Marshall, “Separating and sorting supplement – single stream success,” *Recycling Today*, 2001. [Online]. Available: https://www.recyclingtoday.com/article/separating-and-sorting-supplement----single-stream-success/

[13] R. LeBlanc, “Single-stream recycling offers benefits, creates challenges,” *The Balance Small Business*, 2020. [Online]. Available: www.thebalancesmb.com/an-overview-of-single-stream-recycling-2877728

[14] “Single-stream recycling,” *Scientific American*, 2013. [Online]. Available: www.scientificamerican.com/article/single-stream-recycling/

[15] C. Marshall and K. Bandhauer, “The heavy toll of contamination,” *Recycling Today*, 2017. [Online]. Available: www.recyclingtoday.com/article/the-heavy-toll-of-contamination/

[16] D. Rachelson, “What is recycling contamination, and why does it matter?” *Rubicon*, 2020. [Online]. Available: www.rubicon.com/blog/recycling-contamination/

[17] S. D. James Bohlig, “Systems and methods for sorting recyclables at a material recovery facility,” Patent US 20 140 131 488A1, 2005-06-16.

[18] D. Liming, “How recycling works,” *U.S. Bureau of Labor Statistics*. [Online]. Available: www.bls.gov/green/recycling/

[19] “Report: Recycling workers exposed to safety failures, needles, high injury rates.” *Massachusetts Coalition For Occupational Safety And Health*, 2015. [Online]. Available: www.masscosh.org/publications/featured-articles/report-recycling-workers-exposed-safety-failures-needles-high-injury.

[20] Machinex Recycling. Self-aware Sorting Robot. [Online]. Available: https://www.machinexrecycling.com/samurai/

[21] ——. Delta Robot inside of the SamurAI. [Online]. Available: https://www.machinexrecycling.com/samurai/

[22] Bulk Handling Systems. BHS Launches the Max-AI® AQC-C. [Online]. Available: https://www.bulkhandlingsystems.com/bhs-launches-the-max-ai-aqc-c/

[23] Zen Robotics. Zen Robotics's Heavy Picker. [Online]. Available: https://zenrobotics.com/solutions/heavy-picker/

[24] Conner-Simons, A. and Dorfman, J. Robots that can sort Recycling. [Online]. Available: http://news.mit.edu/2019/mit-robots-can-sort-recycling-0416

[25] "Why can't i recycle wet cardboard?" *GreenAndGrumpy*, 2019. [Online]. Available: https://greenandgrumpy.com/why-cant-i-recycle-wet-cardboard/

[26] CanaKit. Raspberry Pi 4B. [Online]. Available: https://www.canakit.com/raspberry-pi-4-2gb.html

[27] PJRC. Teensy 3.6. [Online]. Available: https://www.pjrc.com/store/teensy36_pins.html

[28] Applied Motion. HT23-597. [Online]. Available: https://www.applied-motion.com/products/stepper-motors/ht23-597

[29] ——. STR3. [Online]. Available: https://www.applied-motion.com/products/stepper-drives/str3

[30] Functional Devices. RIBU1C. [Online]. Available: https://www.functionaldevices.com/support/downloads/datasheet-generator/?SKU=RIBU1C

[31] ——. RIBU2C. [Online]. Available: https://www.functionaldevices.com/support/downloads/datasheet-generator/?SKU=RIBU2C

[32] Online LED Store. 6 Way Fuse Box. [Online]. Available: https://www.amazon.com/12-Circuit-Thumbscrew-LED-Independent-Connections/dp/B07CLWBSNJ

[33] S. Zhu, H. Chen, M. Wang, X. Guo, Y. Lei, and G. Jin, "Plastic solid waste identification system based on near infrared spectroscopy in combination with support vector machine," *Advanced Industrial and Engineering Polymer Research*, vol. 2, no. 2, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542504818300113