Ship Recycling Robot

Major Qualifying Project

Written By:

JOSEPH CYBUL
JOSH HOY
NOBLE KALISH
ISABEL MORALES-SIRGO
TYLER REISER

Advisors:

BERK CALLI
WILLIAM MICHALSON

A Major Qualifying Project
WORCESTER POLYTECHNIC INSTITUTE

AUGUST 2020 - MAY 2021

Abstract

The dangerous process of dismantling large ship hulls often involves cutting down and recycling the ship's infrastructure. According to US OSHA standards, ship breaking operations expose workers to a wide range of hazards or conditions likely to cause injury or illness. This project aims to improve worker safety in the ship-breaking industry by showcasing the potential for automation of the metal cutting process. We designed a computer vision system and torch attachment to work in tandem with the Franka Emika Panda arm. Furthermore, we implemented a test bed to quantify the parameters of the cutting process. Our resulting prototype displays the viability of a robotic approach to ship recycling.

ACKNOWLEDGEMENTS

We would like to extend our gratitude to our advisors, Professor Berk Calli and Professor William Michalson, for their guidance in completing our project. We would also like to thank Roger Morton and EMR for allowing us to work with them in designing the ship cutting system.

Table of Contents

List of Tables

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

The global ship breaking and recycling industry is responsible for reclaiming the valuable metal and components in expired ships of all sizes. Ship breaking is necessary for economic and environmental reasons, however, ship breaking comes with its costs. The process to recover these assets often puts workers in dangerous environments. As a result, working at ship breaking facilities is a dangerous occupation all across the world.

Our goal is to develop a prototype model of an easy-to-use autonomous metal cutting robot. This robot will take on the dangerous aspects of the cutting work and thus, significantly increasing onsite safety. In order to achieve this goal, our objectives are as follows:

1. **Research** current limitations and issues with human-driven ship breaking, as well as limitations and criteria to assist our design phase.

2. **Design** the physical cutting mechanism that ensures the safety of the robot, as well as the vision guidance system.

3. **Implement** our design onto a Franka Emika Panda robotic arm.

4. **Test** different cutting conditions to determine optimal cutting parameters for our implementation. Perform demonstrations that showcase how the vision system produces valid solutions on a variety of testing patterns.

Our research identified key points in the process that requires human intervention, but also showcased potential tasks that are prime for automation. Our team determined that we would limit the scope of our project by allowing human technicians to paint outlined markings onto a metal surface that indicates the cutting location on a metal piece, which could then be scanned by our robotic system. Once scanned, our implementation would determine a trajectory along the

outline and perform the cut. The vision for our prototype implementation informed our design criteria, and allowed our team to disassemble the problem into multiple pieces. Our team would need to design a solution that would be capable of cutting metal plating using an oxy-propane cutting torch while conforming to the size and weight limitations of the Panda Arm. Additionally, our vision system would need to be able to translate a non-uniform outline into discrete motor positions that are perceived as robotic motion to allow the robot to cut along the line created. After designing a test bench, and assessing a variety of testing parameters, our final iteration included a mounted oxy-propane torch and was able to accurately trace complex outlines. This affirms the feasibility of a human assisted robotic cutting arm, and details future improvements that would further expand the usability of our work.

CHAPTER 2

BACKGROUND

Sea ships are large metal structures designed to transport goods or passengers, and have an average lifespan of around 10.5 years [9]. As with the goods they transport, these ships too become waste when they reach the end of their operational lives. There are a three main methods for disposing for these ships. The first option is to lay-up or permanently anchor the ship. Lay-up postpones the issue and although limited, creates opportunities to turn ships into museums such as in the case of the U.S.S Midway. The second option is scuttling, which involves cleaning the ship and sinking it in a specific location to re-purpose it as an artificial reef [10]. Finally, there is ship breaking or recycling the process of tearing down the ship and reusing materials. Lay-up and scuttling, however, can only be used to dispose of a very limited number of ships. For the vast majority of ships, recycling is by far the most environmentally friendly and economically sound disposal method. Recycling ensures the reuse of valuable resources such as steel, iron, aluminium and plastics, however, only a fraction of end-of-life ships are handled in a safe and environmentally clean manner. Ship breaking is a heavy and hazardous industry that exposes both workers and the environment to a great number of risks [11].

We conducted research on the ship dismantling industry to understand the need for its automation. We studied the current state of ship breaking and recycling including the issues and hazards associated with the industry. We found that current dismantling methods used in ship breaking are dangerous for the workers, and companies are moving to automation in order to protect workers from unnecessary risks.

We found that for economic and environmental reasons, the vast majority of ships are recycled. “Almost Every part of the ship - the hull, machinery, equipment, fittings, generators, batteries, hydrocarbons, and even furniture - can be reused. Thus ship recycling is considered an integral element in a sustainable development strategy providing jobs, raw materials for construction, and economic benefits”, [12]. According to the NGO Ship Breaking Platform global impact report,

Figure 2.1: *Aliaga Ship Breaking Yard [1]*

a total of 1418 vessels were disposed of in 2019. Of those, 70% were sold to South Asian beaching yards, and created nearly 90% of the world's dismantled ship material, equivalent to 28.9 million gross tonnes [11].

These operations begin by beaching the enormous ships in shallow shores and striping the ships of any amenities or appliances left on the ship. Workers then break up the ship into manageable pieces that are separated into their raw materials or resold as is. Ship hulls and frames are constructed of thick steel plates. In order to break apart the huge ship hulls in a cost effective manner, workers use cutting torches to cut the plating into slabs that can be recycled. The next section will detail some of the differences between various cutting methods used for industrial applications.

2.1 Oxy-Fuel in The Ship Breaking Industry

Oxy-Fuel cutting is a thermal cutting process used to cut metal plating with thicknesses of between 0.5mm and 250mm [13].The Oxy-fuel cutting process is an oxidizing chemical reaction between oxygen and the iron-rich metal being cut. The process begins by burning the fuel to heat the metal surface. Once up to oxidizing temperature, about 1800°Celsius, a mixture of oxygen and fuel is blown out of the torch and through the ferrous metal. This initiates an exothermic chemical reaction between the oxygen and the iron in the metal, creating iron oxide (slag) and blasting the discarded material through in the metal. 2.2

This relatively wide range and low cost, make oxy-fuel cutting a popular choice in a multitude of applications including ship breaking. However, Oxy-fuel cutting does have its limitations. Oxy-fuel cutting works by oxidizing the iron in metal, therefore this process only works when cutting ferrous metals. Furthermore, since the metal is not being melted away, the melting temperature must be higher than the ignition temperature of the material. The slag must also have a lower melting point than the rest of the material. This allows the slag to be blown out

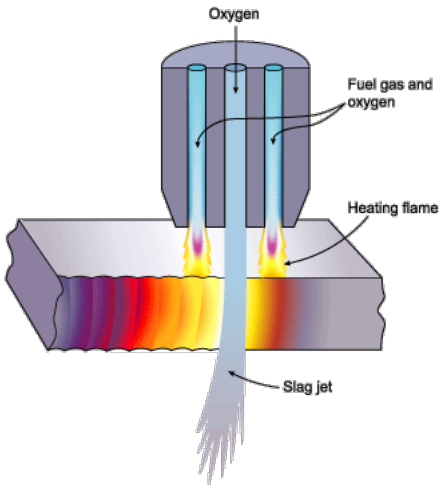

Figure 2.2: *Oxy-fuel cutting process [2]*

through the other end creating the cut.

The cutting process revolves around multiple different factors including the type of fuel, the heat of the torch, the purity of the oxygen, and the composition of the metal. The most important factor being the type of fuel used. We researched the three most prominent fuel gases used. Acetylene, methylacetylene-propadiene (MAPP) & Propylene, and propane are the most commonly used. Each fuel is characterized most by its heat and fuel consumption. The heat of a fuel is defined in two sections, the initial flame temperature and the combustion temperature. These temperatures together determine what thickness the fuel can cut.

Fuel	**Description**
1. Acetylene	Acetylene is the most common of the fuels and the primary flame burns the hottest of the three. For this reason, it excels at making clean cuts on thinner metal.
2. MAPP and Propylene	MAPP and Propylene does not have the ability to cut thick metal plating like the alternative methods because of its lower heat. As such, it is not used very frequently.

3. Propane	Propane produces a lower temperature primary flame making it slower than the acetylene and worse at making precise cuts. However, it has a high combustion temperature making it much better at cutting thicker pieces of metal. Propane is also cheaper; as the amount of fuel required per unit of oxygen is nearly 4 times lower than acetylene.

Table 2.1: *Most Common Oxy-Fuel Types*

2.2 Current Problems of Ship Breaking

The abundance of oxy-fuel cutting in the ship breaking industry presents many dangers. According to the International Labor Organization, ship recycling is one of the most dangerous jobs in the world [14]. According to a study conducted in the ship breaking yards of Bangladesh from 1993 to 2013, 400 workers were killed and over 6000 were injured while performing ship breaking work. 47.5% of these accidents are caused by toxic gas explosions or inadequate safety equipment [15]. Additionally, the toxic fumes from ship breaking and hazardous materials increase the chance of developing cancer [16]. The resulting dangers of ship breaking lower life expectancy of a ship breaking worker to between 40 and 50 years old [12]. Far lower than the global average, which according to the World Health organization is 72 years in 2016 [17]. These issues persist even in locations with strict safety standards and regulations. In reports from the United States Department of Labor, between 2015 and 2019, there were 20 reported accidents while using oxy-fuels that resulted in severe injury or death [18].

Ship breaking can also have negative impacts on the environment. In many unregulated operations, the first step to dismantling a ship is beaching the ship on a shallow shore. As the ship is dismantled, hazardous materials, such as Asbestos and Polychlorinated biphenyl, are discard into the ocean [19]. High levels of ammonia in the sea water were reported in ship breaking sites, which is toxic for fish. Heavy metals found on various components of the ship were also found in shipyards, which affect both human health and the environment. Oil left over in tankers and fuel lines were also found in sea water and the shore [20]. Some companies have begin to cut thousands of mangrove trees along the shoreline to make room for more ships to dismantle. These mangrove trees are important to the ecosystem [19]. All of these issues compound when dismantling a ship and can take a huge toll on the environment.

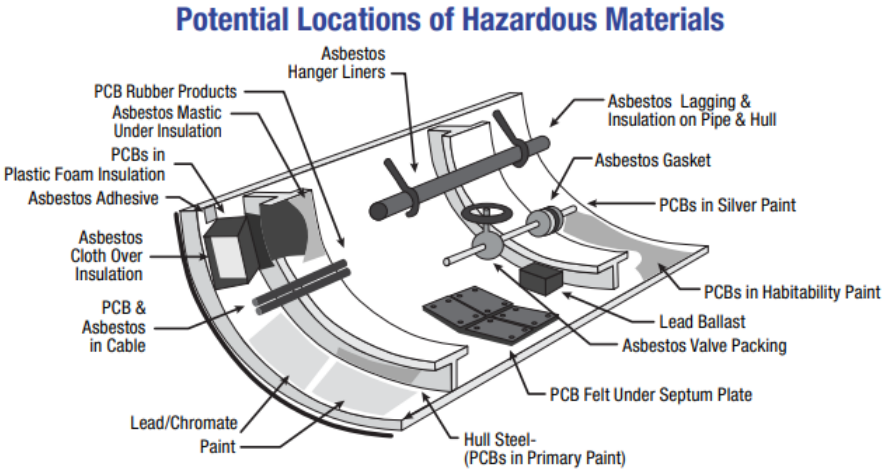

Figure 2.3: *OSHA graphic showcasing locations of hazardous material when ship-breaking [3]*

2.3 EMR Group

Our sponsor is the European Metal Recycling group, or EMR. EMR is a global scrap metal company founded in the United Kingdom, and employs almost 4000 workers across more than 150 locations in the world [21]. Their core business is recycling metal from end-of-life vehicles and products, and has a focus on maritime vessels and structures at their specialized ship breaking facility. Their ship breaking facility, located in Brownsville, TX is the hub for the ship breaking industry in the US, and focuses mainly on safely recycling ex-US Navy vessels and supporting structures [22]. As a business, safety and profits are EMR's main priorities [21]. EMR is constantly looking to increase safety in the work environment, including incorporating robotics into their recycling process.

Figure 2.4: *EMR Aircraft Carrier Recycling [4]*

2.4 Possible Solutions

If a robotic system could be made to perform some parts of the metal cutting process, it would safeguard workers from many of these risks. Robots performing metal cutting can also increase profits, due to the increase in work performance, and reliability inherent to robots. According to RobotWorx, an industrial robot supplier, robots offer: a quicker return of investment, improved throughput, and increased space efficiency [23]. Robots are also great at repetitive work. RoboticsTomorrow, an online robotics magazine, mentions that industrial robots are great for repetitive work, through their precise and accurate programming [24]. We determined that the task of removing flat steel side panels from massive ship hulls is often a large section of the ship recycling process and is well suited for automation [25]. This objective defined the basis of our project.

CHAPTER 3

DESIGN

Applying robotics to the field of ship dismantling and recycling has only recently begun to take shape, in large part due to its complex nature and scale. The focus of our project was to design and build a robotic ship recycling system. This chapter describes our design process, which is broken down into detailed subsections.

3.1 Determining Design Criteria

The potential to apply Industrial Robotics to the field of Ship Recycling has numerous benefits, but poses significant design challenges. These robots require a physical method of interacting with on-site workers that is both intuitive to use, and has the flexibility to produce complex and meticulous cuts. Our design needed to allow on-site workers to directly control the robot arm by marking up the ship hull with soapstone welder's chalk. When the outlining is finished, the worker will prompt the cutting arm to begin and any markings will then be processed by a camera mounted to the robot. Once processed, the cutting arm will trace along the marked lines. This method allows the on-site worker to determine the complexity of the cut, while still allowing for precision if necessary. One additional benefit to this method is that no technical training would be required, and paths can be easily modified by adjusting the outline before cutting.

Our design used a modified Franka Emika Panda Powertool arm (Panda arm). This robot is a 7-DOF(Degrees of Freedom) manipulator, which allows the robotic end effector to reach any discrete position within its reach from multiple orientations. This was ideal for our setup, as it allowed the orientation of the cutter to create angled, or bevel, cuts in the metal plating. These bevel cuts will cause the metal plates to fall away from the cutting arm instead of toward the robot. This will increase safety and predictability for the workers and the robot onsite. Additionally, the Panda arm came with a collection of libraries and network layers called libfranka. This handled

Figure 3.1: *Franka Emika Panda Powertool [5]*

all the network communications with the controller. It also handled all of the sensor feedback from the arm to read the robot state and publish information at a 1kHz rate. This collection of well developed built-in libraries allowed us to focus more on how to apply this technology to the field of ship recycling, instead of developing our own communication system and kinematics interpreter.

One early objective when designing our system was determining what kind of sensors would be required to map a movement strategy for cutting, from a simple outlined mark-up. Our vision system is a central component to the cutting apparatus, therefore our implementation required a high resolution camera. The camera needed to be able to interpret not only color data to track a shape, but also measure distance to determine how far away the shape being traced is from the arm. Our implementation also needed a central communication system to allow the independent vision system and physical controllers to relay data between one another, or receive emergency stop signals from an end user.

Early on in the design phase, we met Roger Morton, the Managing Director of Innovation & Technology at EMR. We asked Roger a series of questions to determine design specifications to create a system that would aid EMR, specifically what the average steel thickness of the ships they cut are, because this was a key piece of information. He stated that one inch A36 steel was the thickest and most common type of ship hull they deal with. Thus, we decided to use a torch that supported a cutting tip for one inch steel.

As our project continued, we narrowed the scope of our project to focus more on the vision system. Initially, we planned on mounting the robot arm onto a mobile base. This would allow the arm to be moved around its work space to allow for larger cuts, but given the time constraints we decided not to include the mobile base. There are many commercial solutions that sell self-enclosed modular mobile bases that our system could be placed directly onto in the future that

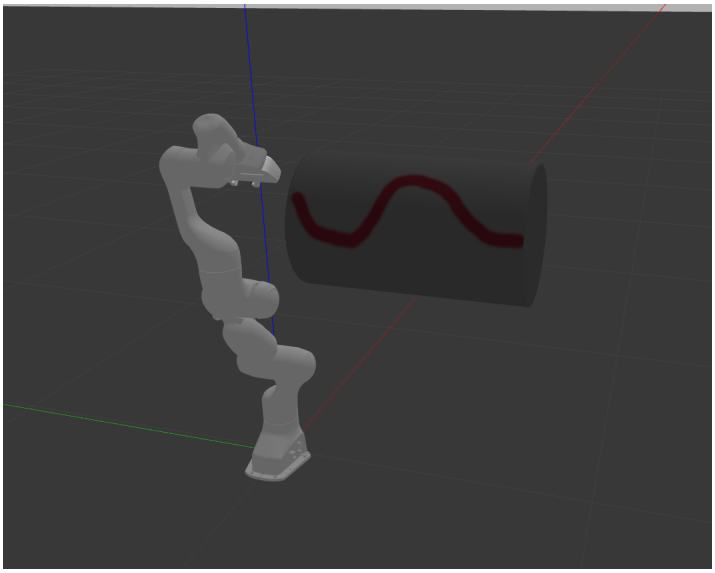

Figure 3.2: *Simulation Environment*

we encourage future teams to investigate. Instead, for our implementation, the Panda arm would be mounted on a hard surface table with wheels to allow our system to be moved around and tested in different environments and spaces. Additionally, we determined that we would stop real-time image processing after initializing the cut. The camera is light sensitive, and the bright sparks of the torch would cause interference with the data it collects. Instead we separated both processes, first we would process the image and then start cutting.

3.2 Software Architecture

We used Robot Operating System (ROS) as the middle-ware, or underlying connective tissue, for our robotics system. ROS is an open source and widely used software architecture in the robotics research community. ROS provides a hardware level abstraction, as well as a messaging system to communicate between different subsystems. This will be very important because the camera system and motion controllers have no inherent way of communicating with one another. It also allows us to have code written in different languages, this is convenient because high level logic is easily implemented in Python, while image processing and hardware control code requires C++. ROS provides a messaging structure that breaks down all the parts of the system into nodes that have specific functions. ROS does this by providing a master node that internally controls the interactions between all the other nodes. Our system uses two forms of communication provided by ROS: topics and services. Topics are named buses that allow message exchange between nodes; a given node can subscribe/publish to a topic to send or receive information. Communication using topics is used when the transfer of messages is unidirectional, otherwise we use services, which upon being called with specific arguments returns a value (bidirectional communication).

Our project utilized Python2.7 and C++11. These specific versions were chosen because ROS Melodic only supports Python2.7, and the PointCloud Library tools are supported in C++11.

One primary focus during the project was ensuring that all systems were tested through simulations, before we built the physical system. Simulations usually have less error and external factors that might affect the behavior of the robot, however they are a good method to determine how our software subsystems will interact. ROS supports integration with Gazebo, a 3D simulation tool, that allows us to create environments to test our robot on, without having a physical model. We can also set up virtual sensors and use the data to model our system, ensuring that when we transition into the physical implementation, our software will still perform as expected.

Our system architecture relied on the Chain-of-Responsibility pipeline shown below in Figure 3.3. In the system we designed, ROS would initialize our vision node, which would receive camera data through a topic subscription to the camera which is constantly publishing sensor data. The Vision System then processed the data and would create a Path message to send to the Trajectory Planner. The Trajectory Planner then uses the incoming path to plan a Cartesian Trajectory to send to the Panda Arm.

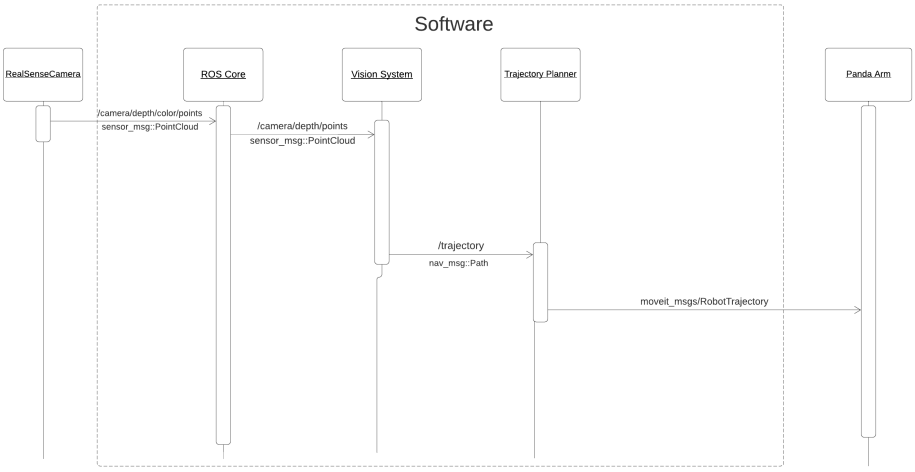

Figure 3.3: *Sequence Diagram*

3.3 Mechanical Design

In order to devise the mechanical design for this project, we had to first decide what equipment best suited our needs, then create an attachment that would allow the Panda arm to manipulate the torch. Our decision in which torch to buy considered two key aspects: the price and capabilities of the torch. As mentioned by our sponsor, we would primarily be required to cut one inch thick steel. However, we had a limited budget and multiple expensive components to buy which meant that given the choice we wanted the more cost effective option. These factors led us to buy the

Victor Cutter ST400C Heavy Duty Torch.

We had to purchase oxygen & propane regulators, hoses, and a cutting tip in addition to the torch in order to have a fully-functional cutting apparatus. As with the torch, we focused our search around finding compatible parts that could perform reliably at the lowest possible price point. We purchased a two-stage regulator for the oxygen, as oxygen flows at a higher pressure than propane and has to be more precise. This was more expensive than a single stage regulator, but was necessary. Since the propane gas flows at a much lower pressure, we were able to use a single-state regulator, which allowed us to minimize expenditures. When buying hoses, we simply chose the least expensive option keeping in mind that we would want enough length for the arm to be able to move, while still keeping the fuel tanks at a safe distance from the cutting process.

After buying the torch, we had to give the Panda arm the ability to manipulate it by creating a custom attachment. The most important design factors for the attachment were weight, strength, and heat resistance. These were chosen to make sure that the arm would be able to successfully and safely operate the torch. We also wanted to be able to add the oxygen to the torch flame electronically. This meant mounting a servo motor onto the attachment with enough support and strength to manipulate the lever/plunger controlling airflow through the torch.

We decided the best option for initial design of the attachment was to have two halves that encompassed the torch and created a solid attachment similar in design to those made to work with the Panda arm. This would allow a solid grip without any overly complicated moving parts. The design was also meant to allow for the knobs on the torch to be manipulated by hand or some other mechanism, meaning they could not be covered.

A proper cutting station was imperative for this project as Oxy-propane cutting is a dangerous process. To create a safe and functioning cutting station for our robot, we planned to mount the Panda arm to a table. Aside from being mounted securely, the process would also require plenty of ventilation.

Along with the attachment to the arm, we also decided we wanted to create a motorized linear slider to test the speeds and distances needed to cut with the torch. Since this was not needed for the final product, we wanted to make it simple and inexpensive. The attachment designed for the arm could be easily adapted to be used on the slider as well.

3.4 Embedded Design

In order to improve the overall portability of the system and achieve integration of all the different components, we decided to port our application into an embedded system. Figure 3.5 showcases a high-level overview of the system.

The hardware components are shown in orange and the software components are shown in blue. We aim to have the control system and software running entirely on our embedded board. This board will be running a custom embedded Linux image, which contains only the

Figure 3.4: *Panda Powertool Arm Mounted on Husky Table*

dependencies required for our project, keeping the system lightweight and efficient. Additionally, the board will be in charge of receiving images from the RealSense camera and controlling the torch's actuator.

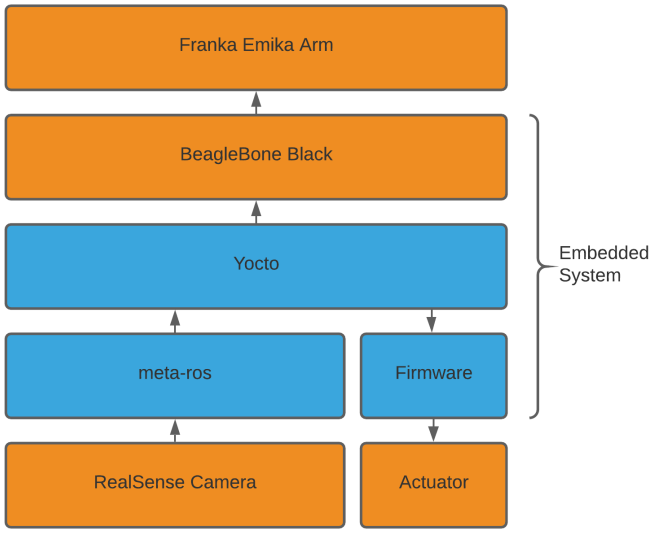

Figure 3.5: *High-level System Overview*

3.4.1 Yocto & Meta-ROS

We used Yocto, an open source collaboration project that provides templates, tools, and methods to create custom Linux-based systems for embedded devices, regardless of their architecture. We chose Yocto since it is widely used in the industry and has a large support community. Additionally, it's flexibility allowed us to initiate development even before the selection of a specific SoC.

Figure 3.6 showcases the Yocto workflow. In order to build our custom image, Yocto requires a configuration file, and in this file we specified the system requirements. Dependencies and packages to include are defined as metadata, made up of recipes (.bb files) and patches. In our case, we had to include the meta-ROS recipe and write a patch to configure it to pull ROS Melodic and our required dependencies such as Python 2.7 and PCL 1.8.

3.4.2 BeagleBone Black

As the main processing unit, we selected the BeagleBone black (Figure 3.7). The BeagleBoard is a low-power, single-board computer. It was designed with open source development in mind, making it ideal for Yocto development. The board uses an Sitara ARM A8 processor.

3.5 Designing Our Testing Framework

In this section we discuss how we designed our final tests for both the Motion System (Panda Arm) and the Oxy-propane cutting (Slider).

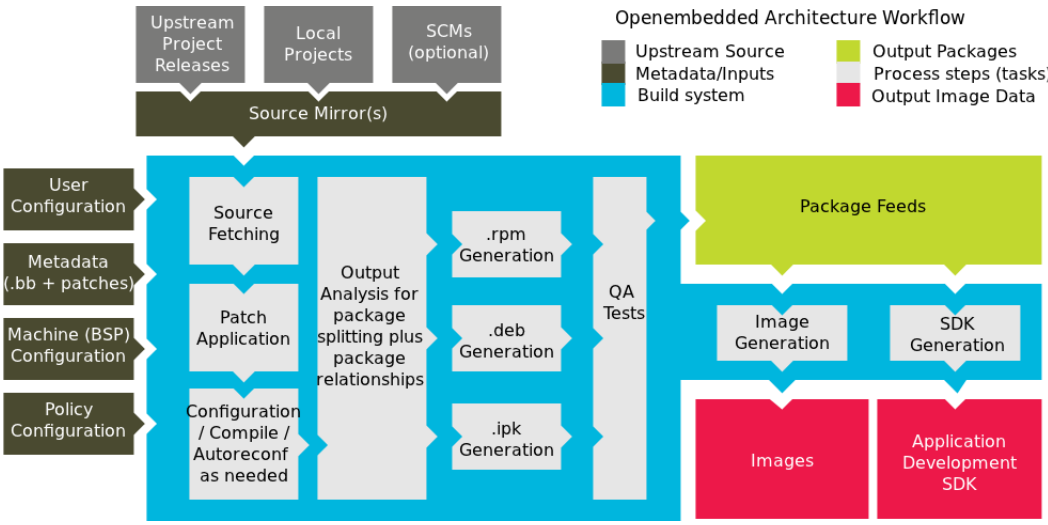

Figure 3.6: *Openembedded Architecture Workflow[6]*

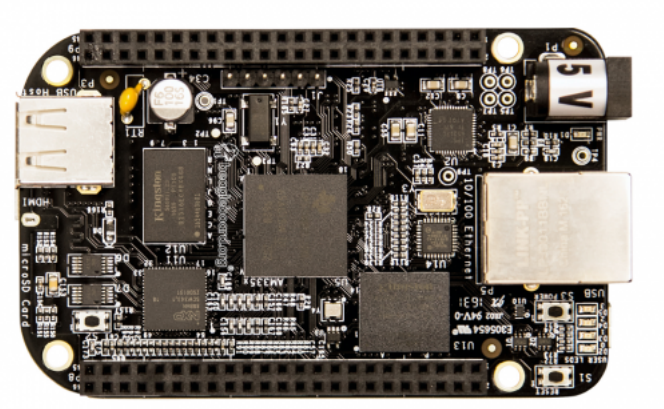

Figure 3.7: *BeagleBone Black [7]*

3.5.1 Oxy-propane cutting test

In order to automate the process of Oxy-propane cutting we designed a series of tests. First, we looked at all the parameters involved such as speed, gas pressures, cutting distance, etc. We then defined which of these parameters we would be keeping constant across test runs and which ones would be our variables.

Based on our research we were able to define the Oxygen and Propane pressures as constant parameters. We defined the following testing parameters, shown in figure 3.8, which set up a series of four tests in which we would try two different cutting speeds at two different distances. In order to quantify the tests, we developed a brief criteria that will grade the different cuts from one to five. The description for each of the scores are shown in table 3.1.

	Distance	Speed	Propane (PSI)	Oxygen (PSI)	Score (1-5)	Observations
Test1	1"	speed 1	8	30		
Test3	1/4"	speed 1	8	30		
Test4	1"	speed 2	8	30		
Test6	1/4"	speed 2	8	30		

Figure 3.8: *Oxy-propane cutting testing sheet*

Score	**Description**
1	There is no sign of cutting and the metal's state is the same as it was before the test started.
2	There is some sign of cutting, the metal's state is clearly changed from the initial state, but there is still a single piece of metal.
3	The cut is successful but there is a lot of excess metal on the edges and the cut is not clean.
4	The cut is successful, and there are some signs of excess on the edges, but there are two pieces at the end.
5	The cut is successful; and there are no signs of excess and the edges on both pieces are very smooth.

Table 3.1: *Oxy-propane cut scoring criteria*

3.5.2 Motion test

In order to test the motion of the Panda arm we aimed to have the system take in an image from the camera, create a path from that image, send that to our robot arm, and have it follow the path. A series of different paths were designed, the one selected for the final test is shown in figure 3.9.

Figure 3.9: *Final motion test path*

In addition to the image interpretation and tracing, we wanted to test the communication between the arm and our chosen embedded board. We decided that the best way to showcase this communication would be to listen for data from the robotic arm using the embedded board to run a subscriber node. This will allow us to record the robot's data as it moves. The information could then be used to interface our two systems.

CHAPTER 4

IMPLEMENTATION & RESULTS

In this section, we discuss how we implemented our designs from the previous section. We detail how we iterated through the process of creating the attachment for the Panda arm and slider. We then report an analysis of the mechanical design. Afterwards, we showcase the results of the cutting test. We then describe how the computer vision system was accomplished, and how the output of the vision system was used to create a Cartesian plan. Then, we talk through the implementation of the embedded system. Finally, we examine the results of the motion test and if the Vision System was successful.

4.1 Iterating Through Mechanical Design and Analysis

The design of the final attachment changed drastically from when we first began planning out the designs of our robot. Our initial intentions included a mobile base which was quickly scrapped when we realized that it would be nearly impossible to create at the scale we needed, considering our time limitations. After this initial downsizing of the project, the design went through a few small but important changes as we began to take more factors into consideration. Most of that design focused around the attachment which connected the oxy-propane cutter to the Panda arm. The arm was built to have interchangeable end effectors, which we created ourselves.

The attachment went through multiple iterations of design using SolidWorks and 3D printed prototypes. Initially, we had to decide what orientation the torch should be mounted to the arm in. In order to find the best orientation, we created mock models of the robot and the torch to visualize the arm's work-space given each orientation as shown in Figures D.1 and D.2. Through this process we decided that the torch should be mounted perpendicular to the robot in the same orientation it would be as if the arm were a human's in Figure D.1. This orientation yields a couple benefits over the other. Firstly it gives the arm a greater range of reachable area in front

of it as tested within Solidworks. Secondly, the second orientation places the point of contact with the torch near its center of mass making it easier for the Panda arm to wield it precisely and for the attachment to hold it steady.

With the orientation of the mount decided, we were able to proceed with implementing the cutting apparatus. This brought up further concerns: most importantly, how would the arm control the lever which activates the torch. Mechanically, this is controlled by a lever which lifts a plunger allowing oxygen to travel through the torch. In order to create a more sleek design, we originally planned to remove the lever in favor of a linear actuator which would lift the plunger with the press of a button. After some testing it was found that the plunger requires about 20-25lbs to lift. An actuator that powerful would be difficult to get in the size and price range we were looking for. After further discussion of the feasibility of this plan, we decided not to remove the lever and instead use it to allow for a smaller actuator. We were able to choose an actuator with a lifting force of only four pounds as a result of keeping the lever. However, this required the attachment's design to change in order to accommodate the lever leading to the attachment shown in Figure D.4. This model was 3D printed and tested on the torch itself, which led to the third and final design of the attachment, shown in Figure D.5. The second iteration fit too tightly onto the torch and did not close properly, creating a safety hazard. The final iteration solved this by creating more room for the torch within the attachment. The final design also shifted the contact point of the torch down which brought it closer to its center of mass whilst also making the lever more accessible to the actuator. Additionally, the final iteration implemented a physical safety feature to manually turn off the oxygen flow. This was done using a key piece which slides into place over the actuator. The actuator itself is mounted on an axle allowing it to swing outwards if not held down. While this will not turn the torch off completely it will lessen the the flame to a propane burn rather than the mixed oxy-propane flame.

Our last design decision was to determine how to fabricate the final iteration of the attachment. Originally, we planned to machine it out of metal, but decided this would be too heavy as the Panda arm has a maximum payload of 3kg. For the sake of ease and weight, we decided 3D printing the attachment mount out of Acrylonitrile Utadiene Styrene (ABS) was our best option. ABS is a rigid and light material that can be easily manufactured through 3D printing. We calculated the bending stress that would be on the attachment to make sure that our current design would be strong enough to safely hold the torch which can be seen below. In these calculations we rounded up the mass of the torch to be 2kg. In order to simplify the calculation we cut the trapezoidal cross sectional area of our attachment into a $5cm \times 2.5cm$ rectangle.

$$W_{torch} = (2kg)(10m/s^2) = 20N$$

$$Where: d_{arm,torch} = 8.5cm = 0.085m$$

$$M_{torch} = (0.085m)(20N) = 1.7N * m$$

$$MomentofInertiaI = \frac{bh^3}{3} = 2.6 \times 10^{-7} m^4$$

$$MaximumBendingStress = \frac{M_{torch} \times y}{I} = (1.7N * m)(0.025m/2)(2.6 \times 10^{-7} m^4) = 81,600N/m^2$$

$$TensileBendingStress_{ABS} = 40,000,000N/m^2$$

$$SafetyFactor = (4 \times 10^7)/(8.16 \times 10^4) = 490.20$$

We found that the current design could easily hold the weight of the torch with a safety factor of over 450. While this is a large safety factor, there are two considerations that make the extra safety worth it. First, ABS is somewhat heat resistant with a glass transition temperature of 105°C but the torch can reach far higher temperatures than that. During field testing we found that this was not a large detriment as the majority of heat the attachment actually came in contact with while cutting were individual harmless sparks from the cut. Still, the extra safety factor means that if something were to happen that damaged the attachment, such as a failed cut sending slag back at it, the attachment would not immediately fail and drop a still operating torch. The second consideration for the high factor of safety is that ABS is cheap and light. Adding the extra width to secure the torch cost little in both funds and payload capacity.

After the attachment was fully designed we needed to create a design for the slider mechanism we were going to use to test the cutting speeds and distances. Key factors in this design were cost, simplicity, and safety. The slider mechanism was for temporary use, but would still be manipulating and operating a dangerous tool. We decided the best course of action was to use t-slotted framing and an appropriate slider. We then mounted the torch, a stepper motor, and a timing belt using 3D printed parts. The attachment for the torch was a modified version of what we designed to connect the torch to the arm. This way, we were able to use the actuator to turn on the oxygen while cutting.

4.2 Cutting Test Results

The speed and distance used to cut metal with oxy-propane are some key variables that define whether a cut is successful or not. In order to determine the parameters required to properly cut the steel, we performed the series of tests previously mentioned in our design section. We made use of the slider to control the speed of the cut as well as the distance between the torch tip and

the metal. The table below shows the result of the test performed and the parameters used on each one:

Test Number	Distance	Speed	Propane (PSI)	Oxygen (PSI)	Score
Test1	1"	0.12 in/s	8	30	1
Test2	1/4"	0.12 in/s	8	30	2
Test3	1"	0.06 in/s	8	30	3
Test4	1/4"	0.06 in/s	8	30	5

Figure 4.1: *Test Parameters*

We recorded a written description of each of the test results based on the criteria we mentioned in the previous section, and they are as follows:

- Test 1: At one inch distance and a speed of 0.12 in/s we observed that there were no signs of cutting and the metal remained unchanged.
- Test 2: At a ¼ inch distance and a speed of 0.12 in/s we observed some markings on the metal, but the cut was not successful and the pieces remain together.
- Test 3: At one inch distance and a speed of 0.06 in/s we observed that the torch tip was too far, causing the hot metal to bounce back and stick the parts back together, as observed in Figure C.13.
- Test 4: At ¼ inch distance and a speed of 0.06 in/s we observed a successful cut, there were no melted edges or excess slag, this was an ideal cut. In Figure C.14 we can observe the removed piece after cutting was done.

After conducting the cutting tests and analysing the results; we determined that the ideal parameters for a clean and precise automated cut were a speed of 0.06 inches per second at a 1/4" distance. From our results we were able to determine that a larger distance would increase the error and decrease the cut's cleanliness. In addition, a faster speed would result in an incomplete cut for metal pieces of an 1" thickness. Images and additional observations on the cutting tests can be found in Appendix C.

4.3 Vision System

The Vision System node is the central node for computer vision. It has three main objectives:

1. Collect data from the RGBD camera
2. Filter the colors to include only the line

3. Apply a B-Spline to create a path

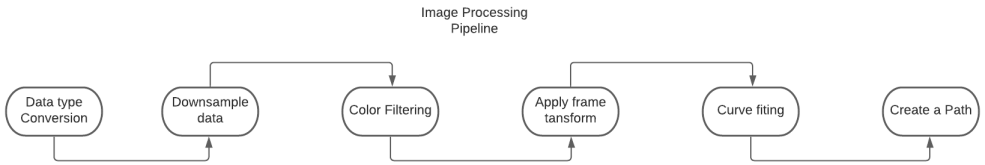

Figure 4.2: *Image Processing Pipeline*

The goal of the Vision System is to identify the line in the metal and create a path from the line. The Vision System pipeline is initiated when the RealSense D435i camera begins to capture data. The D435i interprets its surrounding area as a large collection of individual points in real space. These points each have a red, green, and blue color value. As a collection, this is known as a *PointCloud*. Additionally, we used the PointCloudLibrary, or PCL to manipulate the data from the camera. PCL is a C++ library that has many built-in functions for interpreting, handling, and converting *PointCloud* data types.

Initially, we were planning to use MATLAB for all the image processing requirements because of our previous experience with MATLAB. Unfortunately, there were two main problems when we tried to use MATLAB. The first challenge was being able to load MATLAB onto the embedded system. MATLAB has a complex licensing policy and its dependencies are not available for building the image onto the embedded board. The next issue was that there are no tools for processing *PointCloud* in MATLAB. We choose to use C++ instead of MATLAB to avoid these problems.

After collecting data, the RealSense camera sends data to the Vision System for interpretation. The data is transmitted in a *PointCloud2* data type, as shown in figure 4.3, and must be converted to a *PointCloud* data type. Initially, the data is too large for the system to process so the system first down-samples the data with a PCL random filter.

```
std_msgs/Header header
uint32 height
uint32 width
sensor_msgs/PointField[] fields
bool is_bigendian
uint32 point_step
uint32 row_step
uint8[] data
bool is_dense
```

Figure 4.3: *PointCloud2 ROS Message [8]*

4.3.1 Color Filtering

In order to create a path out of the line, the Vision System must isolate the line. To achieve this objective the Vision System must filter the colors in the *PointCloud*. The Vision System creates a conditional filter that takes in six values: the maximum and minimum for each of the red, blue, and green values. The filter then removes any data point that has a RGB value outside of the six chosen values. To identify the six values needed we chose the closest RGB values of the line and increased the values until only the line was present. After the function is finished, the Vision System is left with just the line as shown in Figure 4.4.

4.3.2 B-Spline NURBS

The last step in the process is to convert the filtered *PointCloud* into a curve representing the line. To create the curve we create a mathematical model that represents the line we want to follow. We then sample very small intervals along the line to create a high resolution trajectory to transmit to the arm. This creates a smooth motion during cutting. To create the modeled line, we could use polynomial piecewise functions. However, without many sections sliced together, polynomial functions cannot react to many adjustments in a curve to achieve a desired shape. Sampling this segmented line would also take significant processing power, as re-sampling the line requires recalculations of the current polynomial function along the curve each time. Considering polynomials would be too computational powerful we used Non-uniform rational Basis-Spline (NURBS) modeling as an alternative method. NURBS curves are defined by a set of weighted control points. These control points are coordinates in 3D-space that the curve will pass through, so if the defining characteristics of a shape are entered as control points, the robot could model complex shapes with a single B-Spline. Our vision pipeline transforms our filtered *PointCloud* into a series of vectors, and creates a default NURBS curve. The data vectors from the *PointCloud* are then fit onto the curve, and refined to produce a smooth curve that represents our desired shape. Originally the NURBS created a closed curve. For our implementation we need an open curve. To achieve this the line is then sampled using triangulation to determine how accurately it reflects the original *PointCloud*. Afterwards, any disputed points are removed and recalculated. This process repeats five times resulting in the final NURBS curve producing an outline of the original shape drafted on the metal plating. The NURBS curve is then converted into a *PointCloud* for more processing.

4.3.3 Creating A Path

The Panda arm does not understand *PointCloud*, and must be converted to a Cartesian plan. After the curve is created, the Vision System creates a path to send to the Trajectory Planner. The path published by the Vision System contains coordinates and orientations that the robot is unable to execute alone. The Trajectory Planner receives data from the Vision System in the form

of a Path message. The Trajectory Planner creates a plan that the Panda arm can execute. The Path message, shown below in Figure 4.5, is a built-in ROS message. It allows us to include a list of points that have a position and orientation to send to the arm. This is useful as the points, known as the ROS message Pose, are needed by the Trajectory Planner to plan a Cartesian plan. This Cartesian plan is the format needed by the Panda arm to move.

The Vision System creates this path by iterating through the newest *PointCloud* and creating a Pose from every point of the *PointCloud* curve. These Poses, the position and orientation we want the Panda arm to be in, are then used to create the Path message that will be sent to the Trajectory Planner.

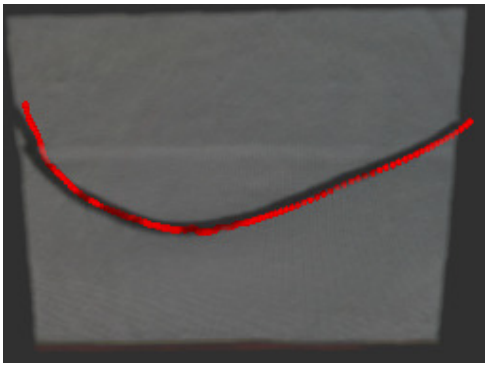

Figure 4.4: *B-Spline in RVIZ*

4.4 Trajectory Planner

The Trajectory Planner is used in our system to create a Cartesian plan for the Panda arm. A Cartesian plan allows the arm to move in one continuous and smooth motion needed to cut metal properly. The planner begins when it receives the Path message from the Vision System. The original path message has points too close together creating a jittery movement if sent to the Panda arm. To solve this, the planner filters through the path to select points that are a certain distance away. This distance was determined through various tests. The planner iterates through the points checking to see if the points are far away enough to previously selected points. Finally, the planner sorts the points so that they are in a continuous line before creating a Cartesian plan. To create the Cartesian plan the planner sends the selected points to the Moveit library. The Moveit library is provided by ROS and is used to create motion plans for the Panda arm to perform.

```
Header header
  uint32 seq
  time stamp
  string frame_id
geometry_msgs/PoseStamped[] poses
  Header header
    uint32 seq
    time stamp
    string frame_id
  geometry_msgs/Pose pose
    geometry_msgs/Point position
      float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64 y
      float64 z
      float64 w
```

Figure 4.5: *Expanded Definition of the Path Message in ROS [8]*

4.5 Embedded Implementation

Due to time constraints, we were unable to transfer the robot's control to our BeagleBone Black. The specialized yocto image including the project's dependencies can be found on the project's github repository linked to in Appendix A.

We decided to prove the image's functionality by implementing a messaging system between the embedded board and the robot's workstation. In order to communicate between the two system's we ran a ROS publisher node on the robot's workstation. To receive and interpret the information being published we ran a subscriber node on the BeagleBone. The success of this test allowed us to monitor the state of the arm in real time from the embedded board.

4.6 Motion Test Results

Figure 4.6: *Image of User Defined Path In Rviz*

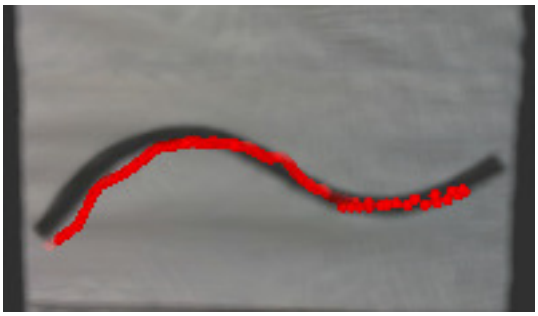

Figure 4.7: *Image with NURBS Curve On top*

The motion tests were able to successfully take in an image from the camera, seen in 4.6, create a path from that image, seen in 4.7, and send that to our robot arm. Our system was then able to send the path to the Trajectory Planner to create a Cartesian plan that the Panda arm followed. After the first successful test, we attached the torch as a new end effector. We then reran the same test while applying a new transformation to account for the torch. The arm was able to complete the first half of the path, but was unable to complete the rest of the path while keeping the torch in the correct orientation.

The results of the motion test reveal that the Vision System and Trajectory planner were successful. The Vision System was able to take an image from the RealSense camera and turn it into a path for the Trajectory Planner. The planner was then able to successfully create a plan and give it to the Panda arm. The arm was not able to fully complete the entire plan because we required the arm to keep the torch and attachment in the same orientation. This orientation was not possible throughout the entire plan because of the lack of movement the arm hand in the X axis.

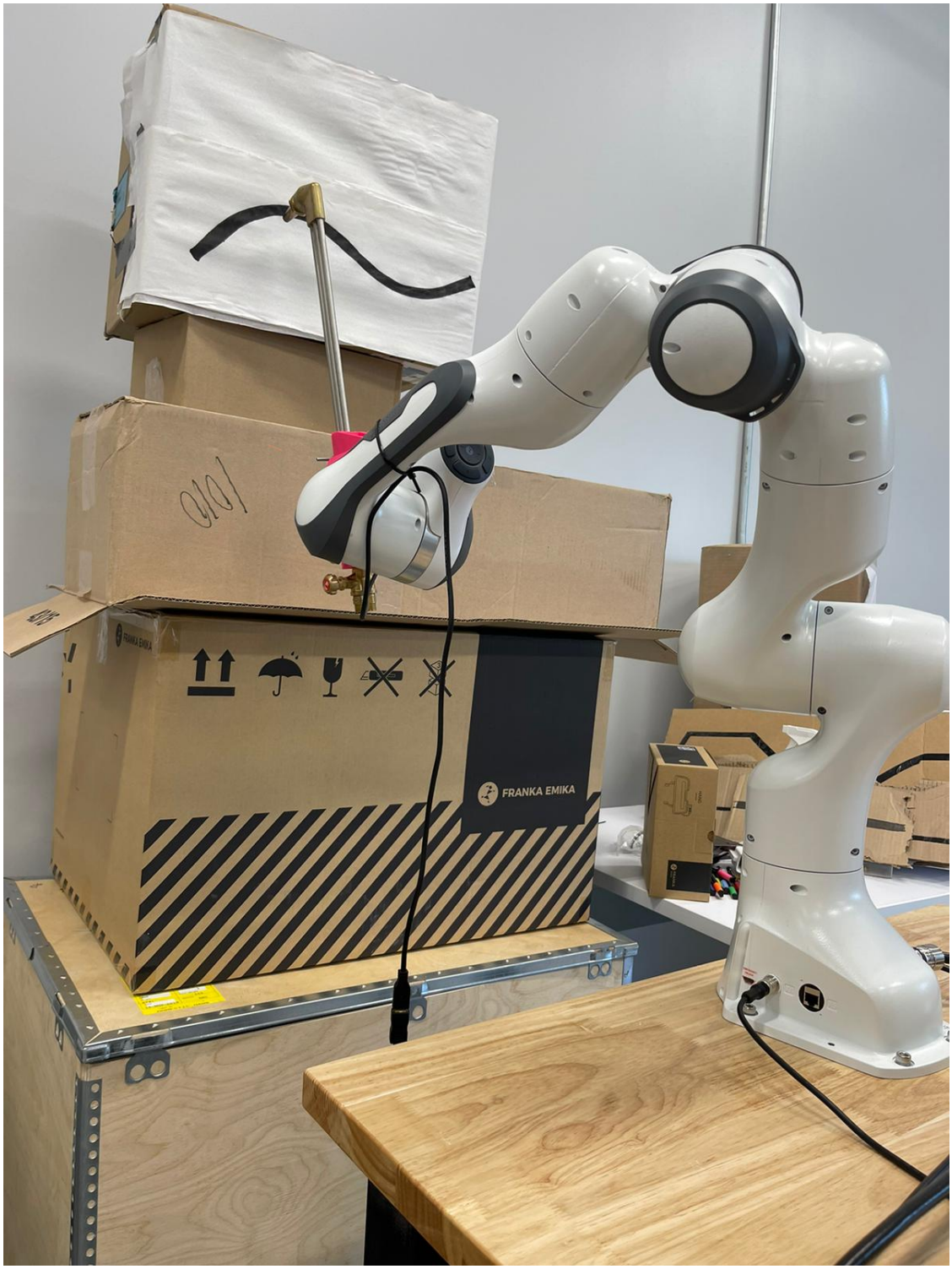

Figure 4.8: *Panda arm In Correct Orientation*

CHAPTER 5

CONCLUSION & NEXT STEPS

In this project, we designed and implemented a system for the Panda arm to serve as a proof of concept that automation of the ship recycling industry is possible. As we mentioned, the ship recycling industry is essential for reclaiming valuable metal and components in expired ships. However, this industry is currently extremely dangerous for human workers. Exposure to high temperatures and flammable gasses create one of the most dangerous work environments of all. [14]

With this project we were able to research the current issues within the industry and design a solution. Our design section covered our initial brainstorming process and areas we chose to focus on when tackling the challenge of autonomous ship recycling within the scope of one year. Our implementation then described how we iterated through that design and made changes based on the data we collected. Moving through those steps we took we also provide the key data that influenced following decisions and final recommendations.

Looking back at the data collected we believe that the automation of the ship recycling industry is possible. Our attachment design proves that there is a way to safely and steadily handle an Oxy-fuel torch. Our cutting tests also proved that the process of metal cutting is possible when using the appropriate parameters. When working with an inch thickness, we recommend a speed of 0.06 in/s at a distance of a quarter inch. These parameters, accompanied with constant gas pressures (in our case Oxygen at 30 PSI and Propane at 8 PSI), provide a clean and precise cut. When working on the vision system, we were able to successfully take in an image, extract a traced line, and generate a path. Using moveit we were able to translate that path into a plan for our robotic arm to follow. Our team was also able to prove that a robotic arm, in our case the Panda arm, with a large range of movement was capable of following a line traced on a flat surface while keeping an Oxy-fuel torch stable.

5.1 Next Steps

After analyzing our results for both of our tests we believe that it is possible to fully automate the process of metal cutting at a large scale. The data obtained also revealed key areas we would need to focus on to achieve a fully functional commercial system.

The testing performed with the slider proved that a cut was possible with an autonomous system once the flame was set up properly. We believe that the next step would be to automate the flame tuning process. The first step would be to implement a mechanism that could automatically manipulate the knobs in the torch, regulating the amount of oxygen and propane being used. In addition, computer vision and machine learning would have to be implemented to determine when the flame has been properly tuned.

We found that the Panda arm has range limitations when restricting the joint movement to account for the torch. We suggest mounting the arm onto a mobile base to take care of movement in the X axis while the arm handles the trajectory in the Y & Z axis. Another option would be to upgrade to a larger arm that has an overall larger range of motion. We believe these two upgrades would be necessary to bring the arm up to standard for commercial use.

Lastly, we believe that a fully functional system would require portability to be effective, this could be achieved by full migration onto an embedded board such as the BeagleBone used during our testing. This embedded board would run the kernel image required to communicate with the arm and process the images provided by the RealSense camera.

APPENDIX A

SOFTWARE USAGE

A.1 Source Repository

All source code for the Ship Recycling software is available online, hosted in a public GitHub repository. This repository is located at: https://github.com/Ship-Breaking-MQP/ship_breaking_ws, and can be cloned or forked.

A.2 Dependencies

The Ship Breaking Robot requires multiple dependencies to work.

- Ubuntu 18.04
- ROS Melodic
  - C++ 11
  - Python 2.7
- PCL 1.8
- All Franka Emika Panda Specific dependencies found in https://frankaemika.github.io/docs/

A.3 Starting the Software

There are two different environments to run the Ship Breaking Robot. The first environment is the simulation environment. This one is best for work without the physical robot being accessible. The second environment is for the real world when the robotic arm is available to use.

To run the simulation environment these steps must be taken after cloning and running "catkin-make" on the ROS work space.

1. Source the "setup.bash" in the "ship_breaking_ws"
2. Run "roslaunch scrap_burning main.launch"
3. Once gazebo is running press play in the bottom of the Gazebo window
4. Run the Vision System by running "rosrun pcl_cpp talker" in a new terminal
5. Run the Trajectory Planner by running "rosrun scrap_burning pcl_subscriber" in a new terminal
6. Both systems require user input to start

To run the real world environment similar steps need to be taken. These steps must be taken on a computer that has been set up with a real time kernel described in the Franka Emika manual.

1. Run "roslaunch panda_moveit_config panda_control_moveit_rviz.launch_robot_ip:=172.16.0.2 launch_rviz:=false load_gripper:=false"
2. Source the "setup.bash" in the "ship_breaking_ws"
3. Run "roslaunch scrap_burning main_mqp.launch"
4. Once gazebo is running press play in the bottom of the Gazebo window
5. Run the Vision System by running "rosrun pcl_cpp talker" in a new terminal
6. Run the Trajectory Planner by running "rosrun scrap_burning pcl_subscriber" in a new terminal
7. Both systems require user input to start

APPENDIX B

SLIDER USAGE

All source code for the Slider software is available online, hosted in a public GitHub repository. This repository is located at: https://github.com/Ship-Breaking-MQP/Slider-Testing.git, and can be cloned or forked. To run the slider these are the steps that need to be taken.

1. Inside the Slider test Repository run the "Steppertest.py" file.
2. The user will be prompted to input the delay between steps, insert 0.
3. The user will be prompted to input the number of steps for the initial movement, to reach the edge of the metal.
4. The user will be prompted to input the number of steps for the second movement, to perform the cut.

APPENDIX C

CUTTING TEST DOCUMENTATION

Our team conducted four cutting tests in the Washburn welding lab to determine the optimal cutting parameters in a safe environment. Our first cutting test showcases a distance between the torch and cutting surface of 1 inch, and a speed of 0.12 in/s. With these parameters, the torch could cut into the metal, but not through the entire one inch thickness.

Figure C.1: *Picture of First Cutting Test, Rough cutting at beginning*

Figure C.2: *Picture of First Cutting Test, Rough cutting at beginning*

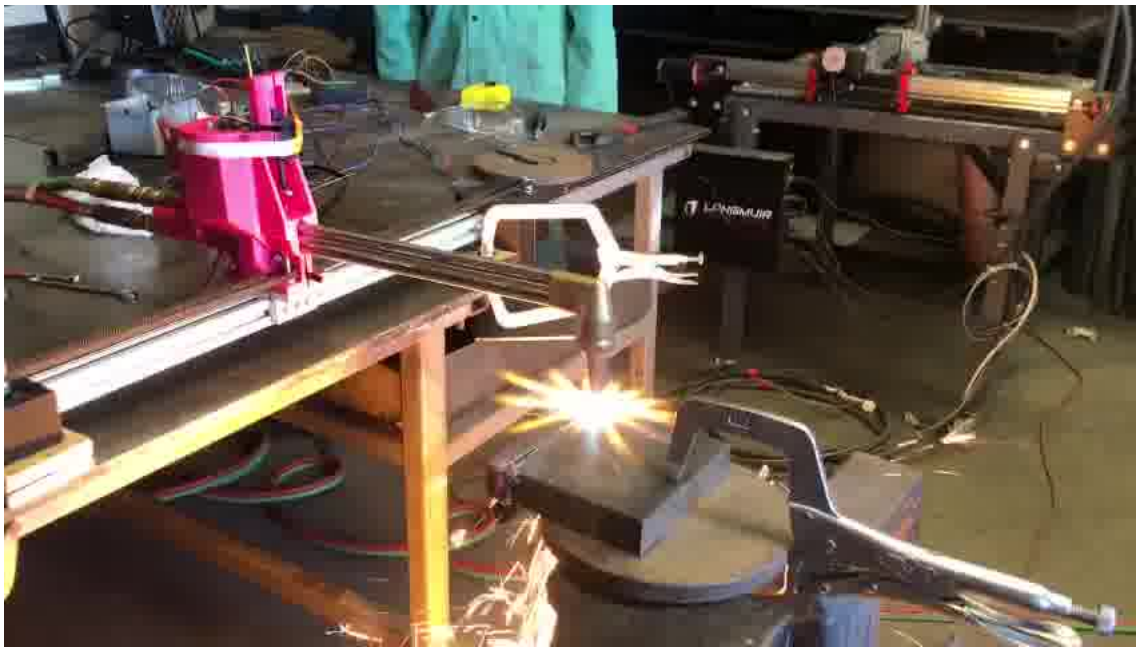

Figure C.3: *Picture of First Cutting Test, Rough cutting at beginning*

Our second cutting test showcases a distance between the torch and cutting surface of a quarter inch, and a speed of 0.12 in/s. With these parameters, the torch was so close to the metal, the flame was easily suffocated and prevented the metal from being cut.

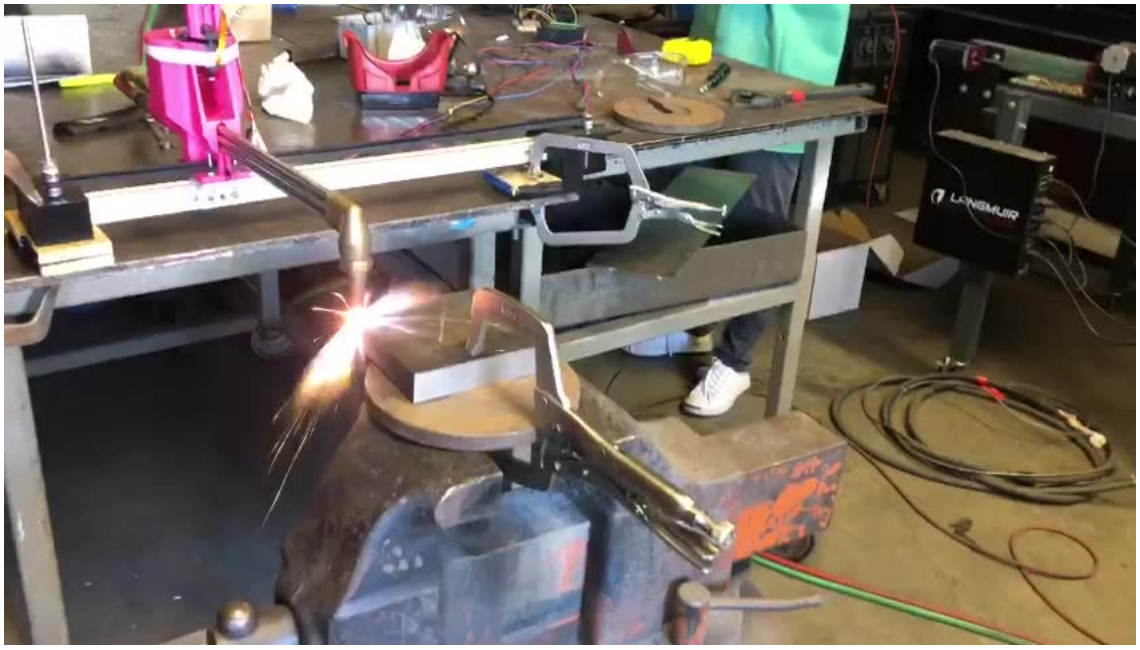

Figure C.4: *Picture of Second Cutting Test, close distance to metal created bad cut quality*

Figure C.5: *Picture of Second Cutting Test, close distance to metal created bad cut quality*

Figure C.6: *Picture of Second Cutting Test, close distance to metal created bad cut quality*

Our third cutting test showcases a distance between the torch and cutting surface of one inch, and a speed of 0.06 in/s. With these parameters, the torch was too far from the metal, this allowed the metal to melt back together after the cut, and prevented the metal from separating.

Figure C.7: *Picture of Third Cutting Test, cutting quality improved*

Figure C.8: *Picture of Third Cutting Test, cutting quality improved*

Our fourth cutting test showcases a distance between the torch and cutting surface of 0.25 inch, and a speed of 0.06 in/s. With these parameters, cutting was optimal and allowed for a smooth cut. Figure C.14 shows the smooth edge that was created after completing the cut.

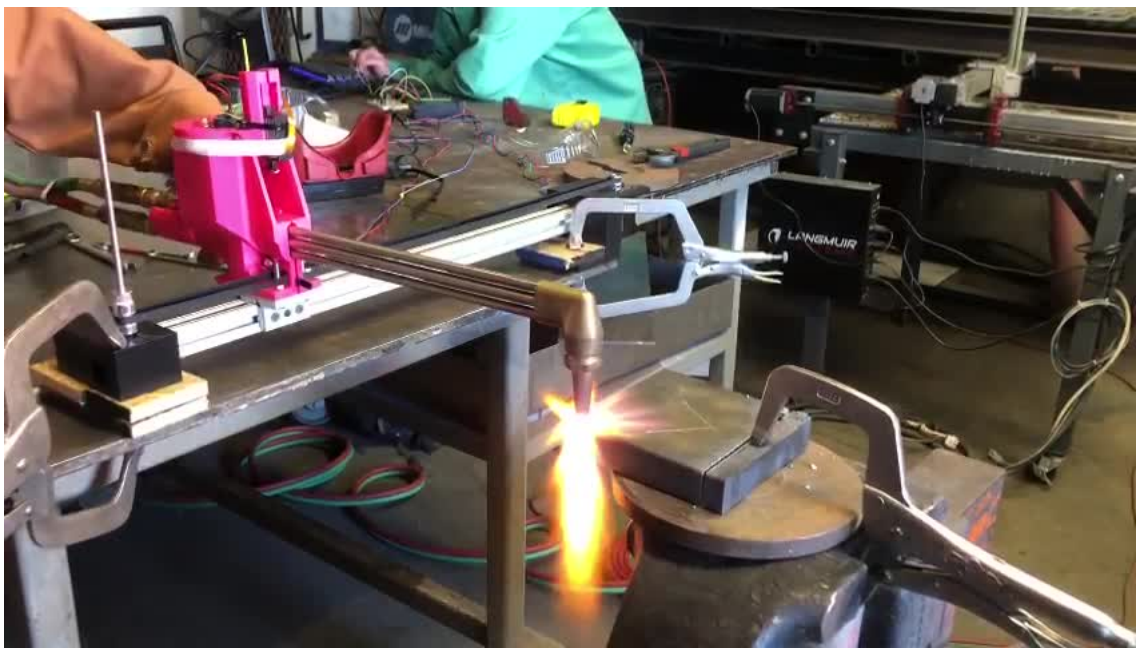

Figure C.9: *Picture of Fourth Cutting Test, notice constant star pattern*

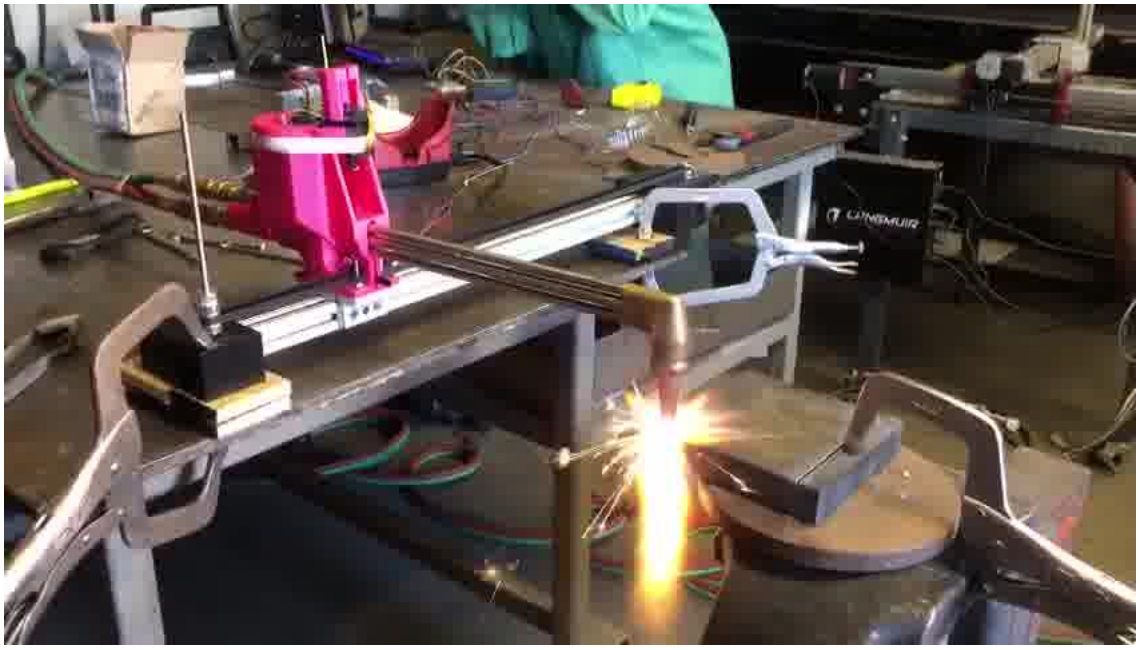

Figure C.10: *Picture of Fourth Cutting Test, notice constant star pattern*

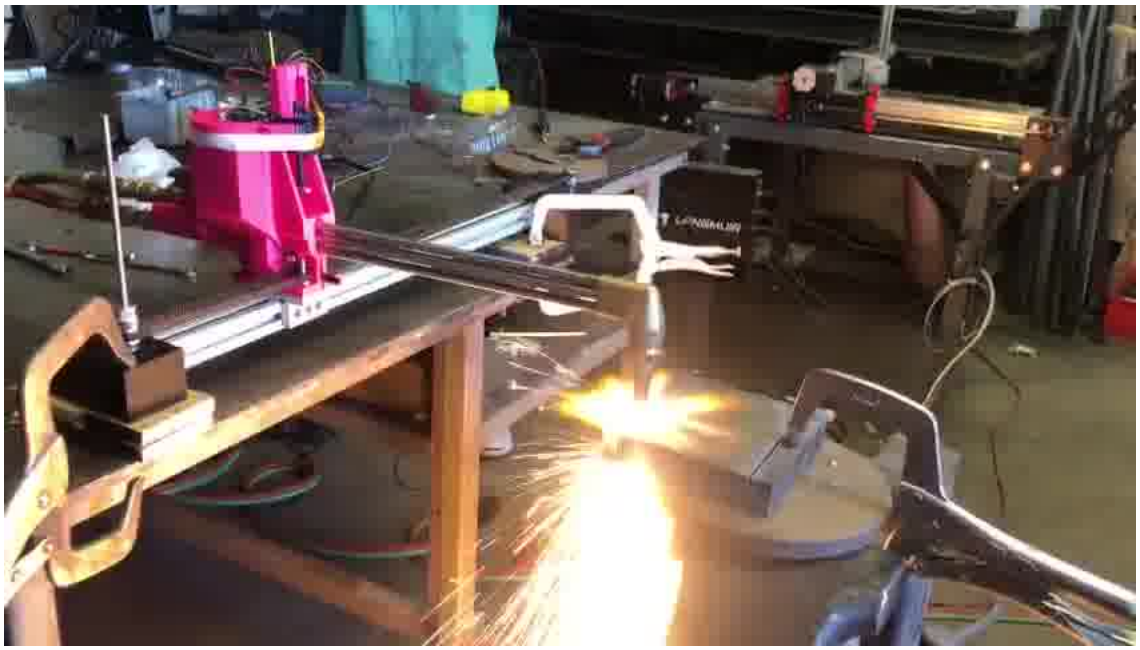

Figure C.11: *Picture of Fourth Cutting Test, notice constant star pattern*

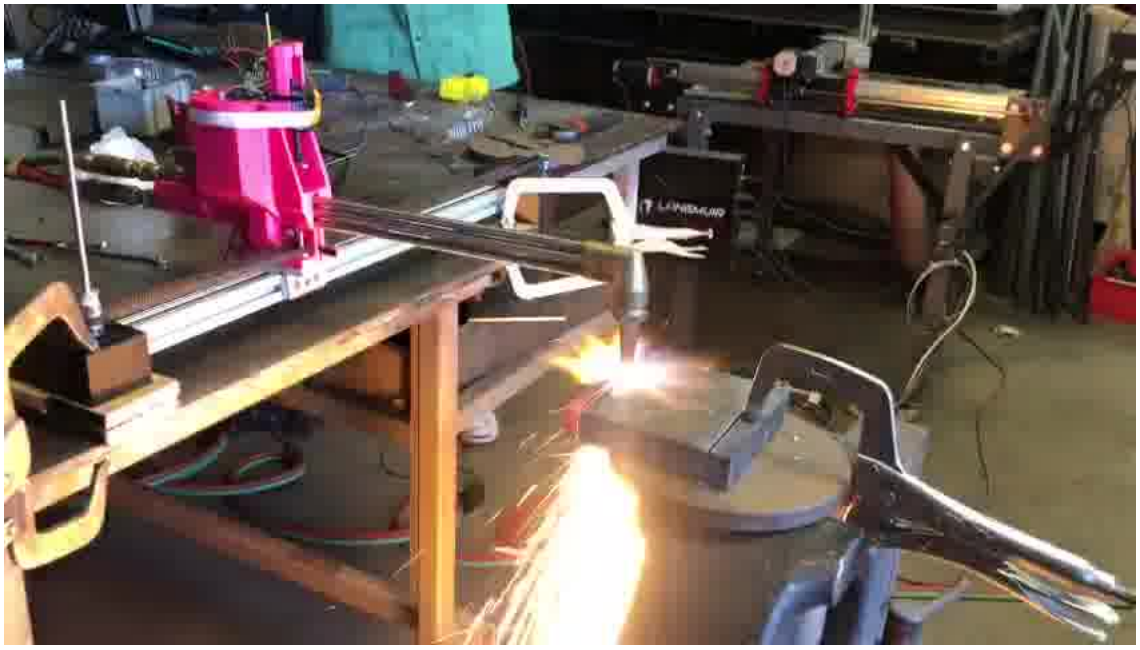

Figure C.12: *Picture of Fourth Cutting Test, notice constant star pattern*

Figure C.13 shows the results of all the cuts performed. This image showcases the first three tests that didn't cut the metal and were messy and unreliable.

Figure C.13: *Picture showcasing all cutting test results*

Figure C.14: *Picture of successful metal cut*

APPENDIX D

TORCH ATTACHMENT DESIGNS

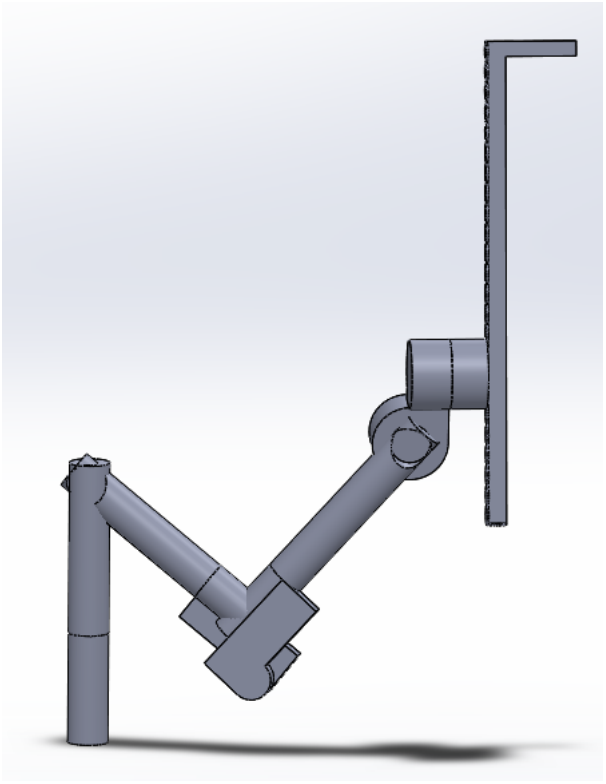

Figure D.1: *Mock arm used to find work-space (1st orientation)*

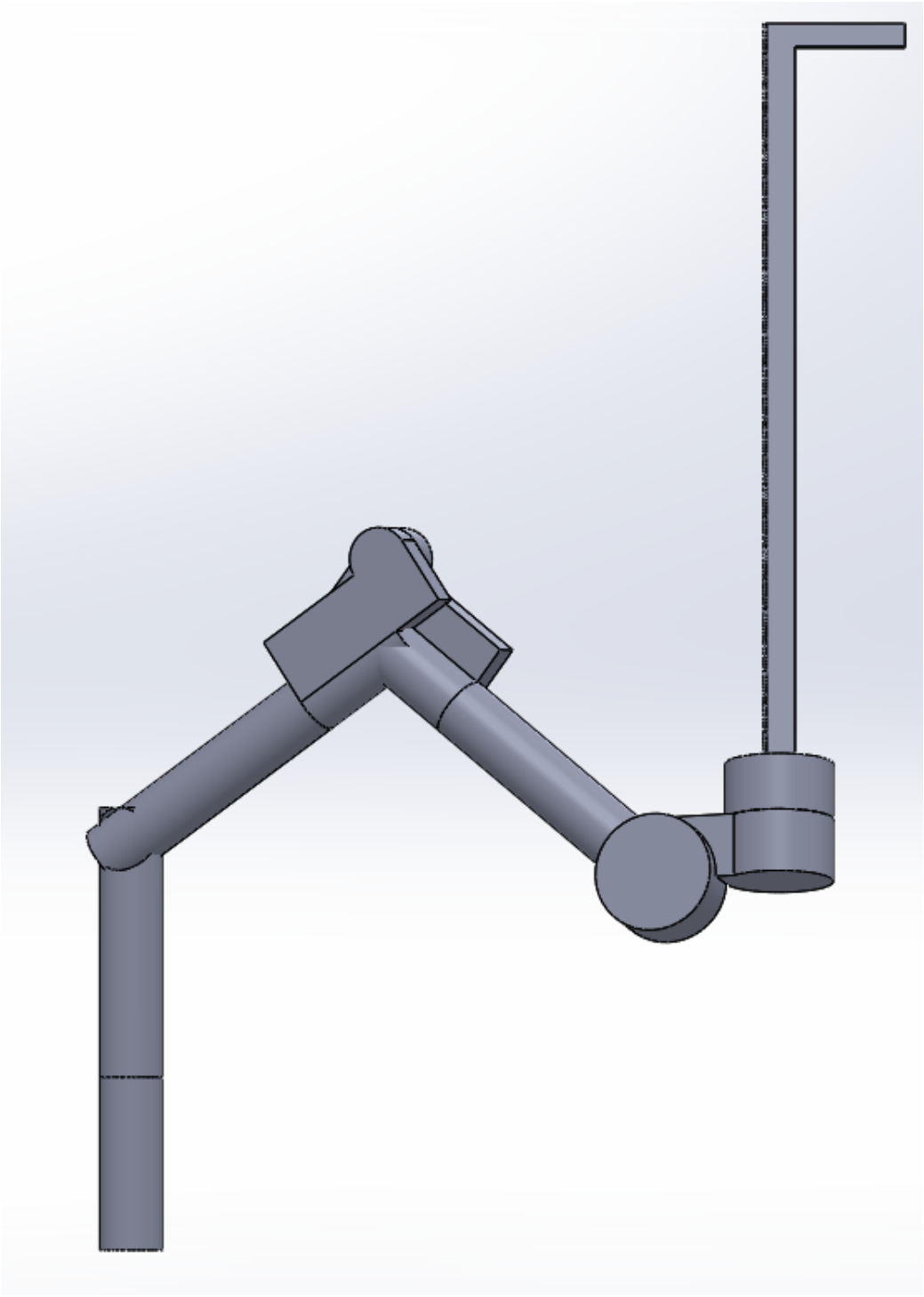

Figure D.2: *Mock arm used to find work-space (2nd orientation)*

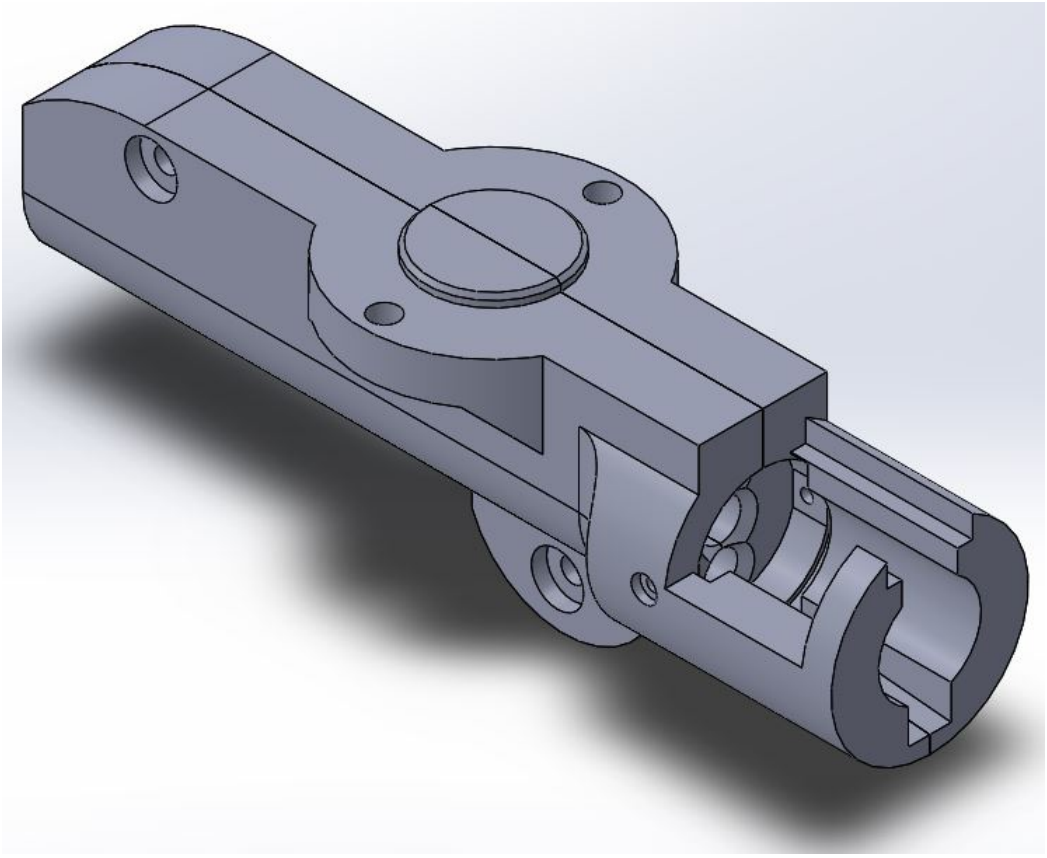

Figure D.3: *First iteration arm attachment*

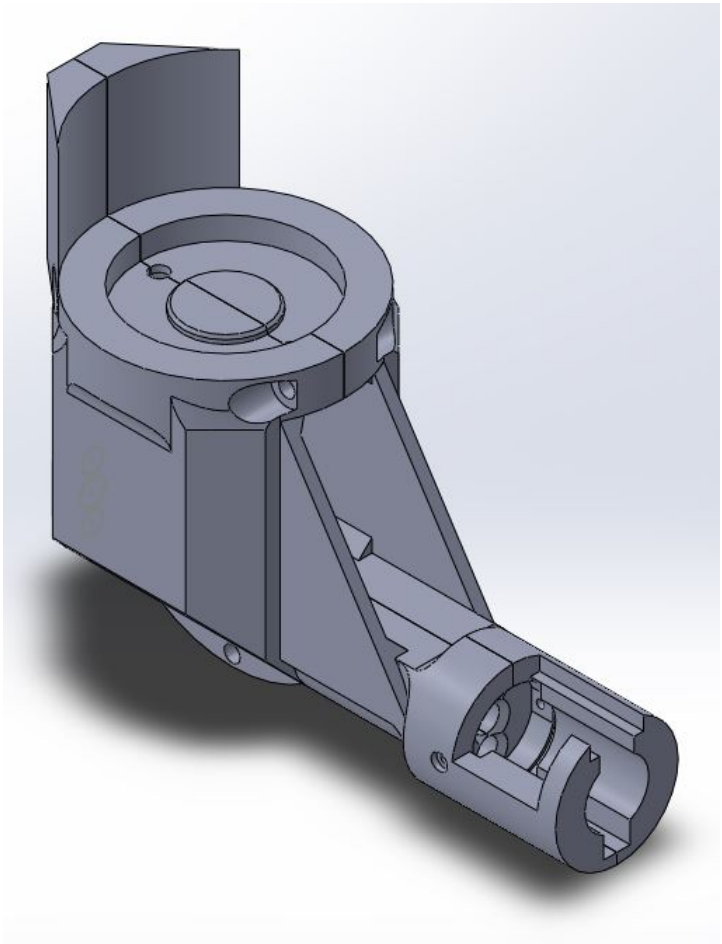

Figure D.4: *Second iteration arm attachment*

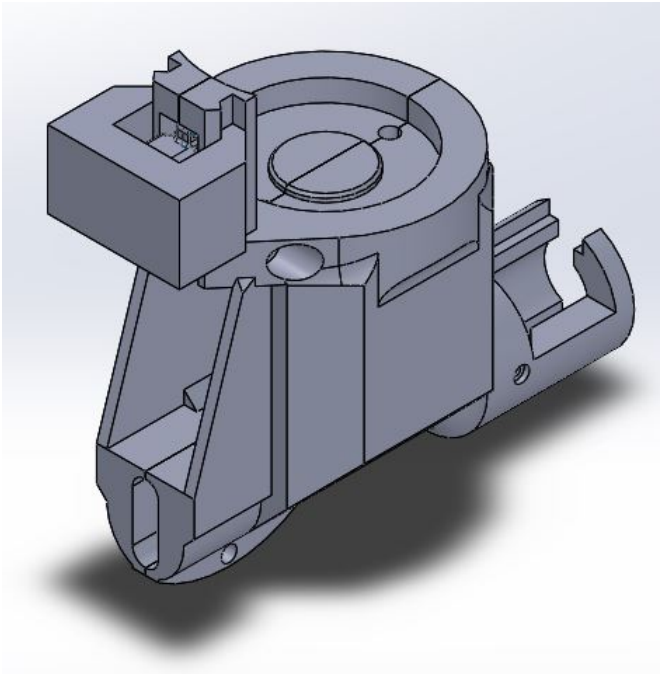

Figure D.5: *Third and final iteration arm attachment*

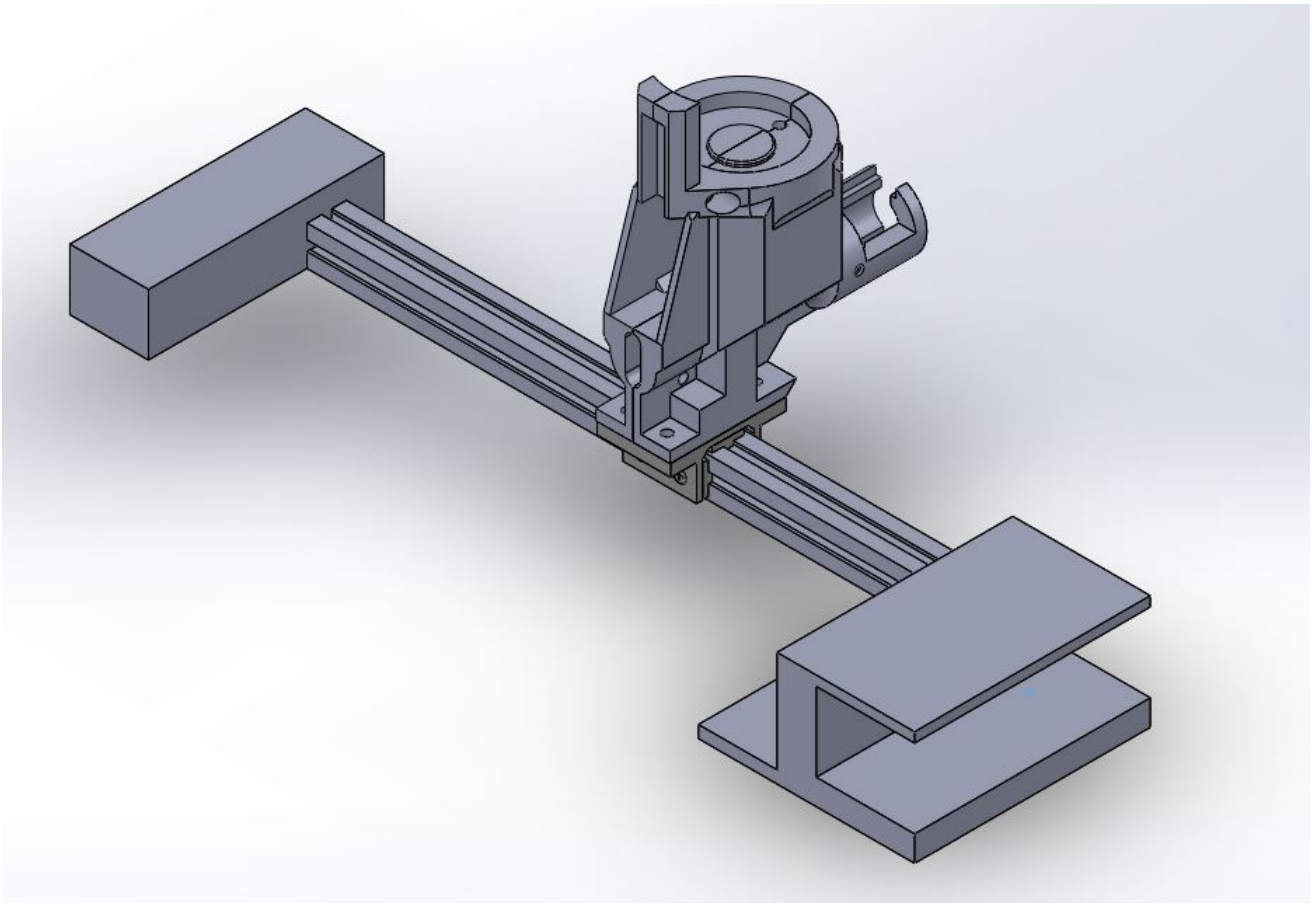

Figure D.6: *Simple CAD model of slider*

BIBLIOGRAPHY

[1] C. McGrath. Photos: Retired cruise ships dismantled in the wake of covid-19. [Online]. Available: https://www.cnn.com/travel/gallery/cruise-ship-breaking-yard-aliaga-turkey/index.html

[2] T. Ltd. Oxyfuel cutting - process and fuel gases. [Online]. Available: https://www.twi-global.com/technical-knowledge/job-knowledge/oxyfuel-cutting-process-and-fuel-gases-049

[3] O. U.S. Dept. of Labor. Hazards associated with shipbreaking. [Online]. Available: https://www.osha.gov/sites/default/files/publications/shipbreaking-factsheet.pdf

[4] E. Group. Marine structures and vessels. [Online]. Available: https://us.emrgroup.com/what-we-do/our-specialist-areas/marine-structures-and-vessels

[5] A. Robots. Franka emika panda arm. [Online]. Available: https://www.active8robots.com/robots/the-franka-emika-panda-robot/

[6] Y. Proyect. Open embedded architecture. [Online]. Available: https://www.yoctoproject.org/docs/2.1/yocto-project-qs/yocto-project-qs.html

[7] Element14. Beaglebone black. [Online]. Available: https://www.element14.com/community/docs/DOC-84108/l/beaglebone-black-development-board-with-1ghz-am335x-arm-cortex-a8-processor

[8] O. S. R. Foundation. Ros wiki documentation. [Online]. Available: http://wiki.ros.org/Documentation

[9] "the development of container ships," mccontainers, 05-may-2021. [Online]. Available: https://mccontainers.com/blog/development-of-containerships/.

[10] G. S.C., *Jutland to Junkyard: The Raising of the German High Seas Fleet from Scapa Flow*. Paul Harris Publishing, 1981.

[11] Ngosbp impact-report 2018/2019. [Online]. Available: https://shipbreakingplatform.org/wp-content/uploads/2020/06/NGOSBP-Bi-Annual-Report-18-19.pdf

[12] T. G. Puthucherri, *From shipbreaking to sustainable ship recycling: evolution of a legal regime*. Martinus Nijhoff Publishers, 2010.

[13] "oxyfuel cutting - process and fuel gases," twi. [Online]. Available: https://www.twi-global.com/technical-knowledge/job-knowledge/oxyfuel-cutting-process-and-fuel-gases-049.

[14] (23-Mar-2015) Ship-breaking: a hazardous work. [Online]. Available: https://www.ilo.org/safework/areasofwork/hazardous-work/WCMS_356543/lang--en/index.htm.

[15] Shipbreakingplatform.org. [Online]. Available: https://www.shipbreakingplatform.org/wp-content/uploads/2018/08/Health-hazards-and-risks-vulnerability-of-ship-breaking-works-Muhibbullah-Nov-2013.pdf

[16] W.-T. Wu, Y.-J. Lin, C.-Y. Li, P.-J. Tsai, C.-Y. Yang, S.-H. Liou, and T.-N. Wu, "Cancer attributable to asbestos exposure in shipbreaking workers: A matched-cohort study," *PLOS ONE*, vol. 10, no. 7, pp. 1–12, 07 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0133128

[17] (15-May-2018) "life expectancy," world health organization. [Online]. Available: https://www.who.int/gho/mortality_burden_disease/life_tables/situation_trends_text/en/.

[18] "united states department of labor," accident search results page | occupational safety and health administration. [Online]. Available: https://www.osha.gov/pls/imis/AccidentSearch.search?acc_keyword=22Acetylene

[19] "the environmental costs," ngo shipbreaking platform, 16-may-2019. [Online]. Available: https://shipbreakingplatform.org/our-work/the-problem/environmental-costs/.

[20] M. S. Hossain, A. N. M. Fakhruddin, M. A. Z. Chowdhury, and S. H. Gan, "Impact of ship-breaking activities on the coastal environment of bangladesh and a management system for its sustainability," *Environmental Science Policy*, vol. 60, pp. 84 – 94, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1462901116300569

[21] EMR. About emr. [Online]. Available: http://us.emrgroup.com/about

[22] ——. Vessels recycled by emr. [Online]. Available: https://uk.emrgroup.com/what-we-do/our-specialist-areas/marine-structures-and-vessels/vessels-recycled-by-emr

[23] RobotWorx. How can industrial robots improve my profits? [Online]. Available: https://www.robots.com/faq/how-can-industrial-robots-improve-my-profits

[24] RoboticsTomorrow. (2018, 07) 7 advantages of robots in the workplace. [Online]. Available: https://www.roboticstomorrow.com/story/2018/08/7-advantages-of-robots-in-the-workplace/12342/

[25] J. C. J. H. T. R. Noble Kalish, Isabel Morales-Sirgo, private interview, September. 28 2020.