GNG1103

Design Project User and Product Manual

Inverse Kinematics Solver and Dual-Prong End Effector

Submitted by:

The Last End-Effector – C10

Samuel Harrison – 300198650

Ethan Salt – 300189165

Hannah Chan – 300176806

Omar Abdul-Ghani – 300070139

Jason Nguyen - 300144344

April 9, 2022

University of Ottawa

Table of Contents

T	able of	f Contents	i
L	ist of I	Figuresi	V
L	ist of T	Γablesv	ii
L	ist of A	Acronyms and Glossaryvi	ii
1	Int	roduction	1
2	Ov	erview	2
	2.1	Cautions & Warnings	6
3	Ge	tting started	7
	3.1	Configuration Considerations	1
	3.2	User Access Considerations	2
	3.3	Accessing/setting-up the System	3
	3.4	System Organization & Navigation	5
	3.5	Exiting the System	5
4	Us	ing the System	7
	4.1	Water Blaster	7
	4.1	.1 Vacuum	8
	4.2	1	9
	4.3	LED Light	9
	4.4	Moving the 3 DOF Arm	1
	4.4	.1 Drawing Tool	2
	4.5	Emergency Stop Button	3

5	Tro	ouble	shooting & Support	24
	5.1	Erro	or Messages or Behaviors	24
	5.2	Spe	cial Considerations	25
	5.3	Mai	intenance	25
	5.4	Sup	pport	25
6	Pro	duct	Documentation	27
	6.1	Inv	erse Kinematics Test Arm	27
	6.1	.1	BOM (Bill of Materials)	27
	6.1	.2	Equipment list	27
	6.1	.3	Instructions	28
	6.1	.4	Testing & Validation	29
	6.2	Enc	l Effector and GUI	32
	6.2	.1	BOM	32
	6.2	2	Equipment List	33
	6.2	3	Instructions	33
	6.2	4	Testing & Validation	38
	6.3	Enc	l Effector Components and Arduino Connection	43
	6.3	.1	BOM	43
	6.3	.2	Equipment List	44
	6.3	.3	Instructions	44
	6.3	.4	Testing & Validation	48
7	Co	nclus	sions and Recommendations for Future Work	49
A	PPEN	DICI	ES	51

List of Figures

Figure 1 – The final prototype of the end effector with the water blaster and LED being	
used	3
Figure 2 – From left to right: the vacuum, drawing tool, LED, and water blaster	4
Figure 3 – The robot arm with our end effector on it. The Arduino board and CNC shield	
are out of frame on the back panel	5
Figure 4 - Screw locations for end-effector	7
Figure 5 - Attachment holder	8
Figure 6 - Example of proper attachment setup	9
Figure 7 - GUI	9
Figure 8 - GUI Coordinate Error	10
Figure 9 - GUI Emergency Stop	11
Figure 10 - Photos of the equipment	12
Figure 11 – The GUI with the Water Blaster circled	18
Figure 12 – The GUI with the Vacuum circled	19
Figure 13 – The two cylindrical pieces with LED lights in either one sticking out of the	
top	20
Figure 14 – The GUI with the LED Light circled	21
Figure 15 – The GUI with inputted coordinates, ready to click send	22
Figure 16 – The GUI after hitting the emergency stop button	23
Figure 17 – Setup used to test inverse kinematics algorithm	28

Figure 18 – Angle output from inverse kinematics algorithm with inputted values for test	
case 1	30
Figure 19 – Aerial view of test case 1 for the inverse kinematics prototype at 64 degrees	30
Figure 20 – Angle output from inverse kinematics algorithm with inputted values for test	
case 2	31
Figure 21 – Aerial view for test case 2 for the inverse kinematics prototype at 145	
degrees	32
Figure 22 – Grid pattern of the PTinker front end to help user replicate the design	34
Figure 23 – Concept of prototype I in OnShape CAD	36
Figure 24 – Concept of prototype II in OnShape CAD	37
Figure 25 – Concept of prototype III in OnShape CAD	38
Figure 26 – Pop-up display when inputted coordinates are not within the robot arm's	
range	39
Figure 27 – Angles given when x=-4, y=3, z=1.5	40
Figure 28 – Prototype I end-effector with and without a pencil attachment	41
Figure 29 – Prototype II end-effector with and without a pencil attachment	42
Figure 30 – Prototype III end-effector with and without an attachment	43
Figure 31 – Wiring for the DC motor of the water pump and water vauum	45
Figure 32 – CAD concept of cylinders for end effector and the printed cylinders for the	
end-effector	46
Figure 33 – Pop-up display when EMERGENCY STOP is pressed	47
Figure 34: https://www.youtube.com/watch?v=k4C8fNMt_x0	48
Figure 35 - https://www.youtube.com/watch?v=Y4jDBAXgk7I	48

List of Tables

Table 1. Acronyms	viii
Table 2. Glossary	viii
Table 3. Referenced Documents	51

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
UPM	User and Product Manual
DOF	Degrees of Freedom
IK	Inverse Kinematics
GUI/UI	Graphical User Interface
PPE	Personal Protective Equipment

Table 2. Glossary

Term	Acronym	Definition
End Effector	EE	The end piece of a robotic arm, typically the piece it uses to interact with the environment

1 Introduction

This User and Product Manual (UPM) provides the information necessary for proper setup and use of the to effectively use inverted kinematics solver and end-effector. This UPM contains all the important information concerning prototype documentation, system overview, maintenance and general setup. This document should be consulted before installing the product, as it will explain the necessary steps and precautions. This document should also be consulted for general maintenance, and error behaviours. By following the steps accurately as described in the UPM, the product will be able to continue functioning properly and accurately. This document will first discuss the product overview explaining its properties and functions. The overview briefly explains the cautions and warnings as well. Next, the document will explain how to set up the IK and EE and how to use the product. These sections will ensure that the IK and EE is correctly installed and operated. The document will also explain the troubleshooting and maintenance. This portion of the document is important to ensure that the IK is properly maintained in order to reduce the risk of malfunction and increase the product longevity. The user is also provided with proper information on how to contact the help team for questions about the IK and EE. This document then explains the product documentation about the prototypes, explaining the necessary tools and materials needed for each prototype. Finally, the document explains the future recommendations for the product.

Introduction 1

2 Overview

On board the Halifax class warships, there is an overwhelming amount of rust and debris that builds up on the side of the ship, not to mention the degrading of the paint. Also, there are plenty of these spaces that are in low oxygen areas, requiring an oxygen tank for a naval officer to enter. To cut down cost and increase efficiency, using a 3 DOF robotic arm capable of water blasting, painting, and viewing areas in low oxygen is vital. This arm also allows work to be done in spaces that humans cannot reach, not to mention being able to work overnight.

For the arm to be as efficient as possible, it's movement must be performed using inverse kinematics. This method allows the user to give the end coordinates and the arm can determine how to get there with the correct angles. It is also vital that the GUI is very simple and easy to use since some of the naval officers only have high school knowledge. Another key aspect is having an end effector that can perform most if not all the tasks allotted. The end effector must be able to at least perform one or more of the following: water/sand blast an area, paint an area or draw a logo, and have a camera to view specific areas. Finally, the arm must have adequate safety features as an emergency stop button on the GUI, along with a sensor for detecting movement if applicable.

The product that we have created achieves almost all the criteria given by the client. Our inverse kinematics solution was solved using triangular projections in all 3 planes, along with simple trigonometry identities. Our end effector is a unique two-prong design that allows the user to use any two tools at the same time. It also has two universal cylindrical holders, allowing for more attachments to be added later if needed. We developed a functioning water blaster, drawing tool, vacuum, and LED light. Our GUI is very easy to use and has a highly visible emergency stop button

that cuts all the power to the end effector tools, and activity can be resumed from once the emergency has been dealt with. The final reason our design is the optimal choice is because everything including the Arduino code, the GUI, and the inverse kinematics solution were all coded in python. This allows everything to be in one language and easy to adjust if needed.

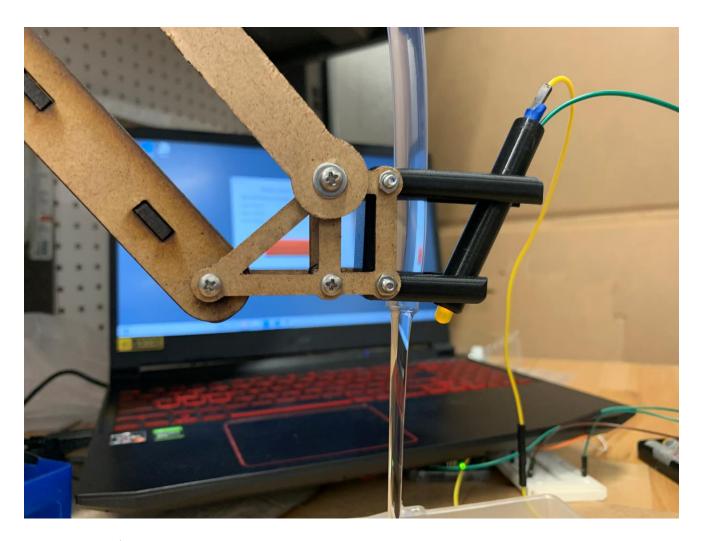


Figure 1 – The final prototype of the end effector with the water blaster and LED being used.

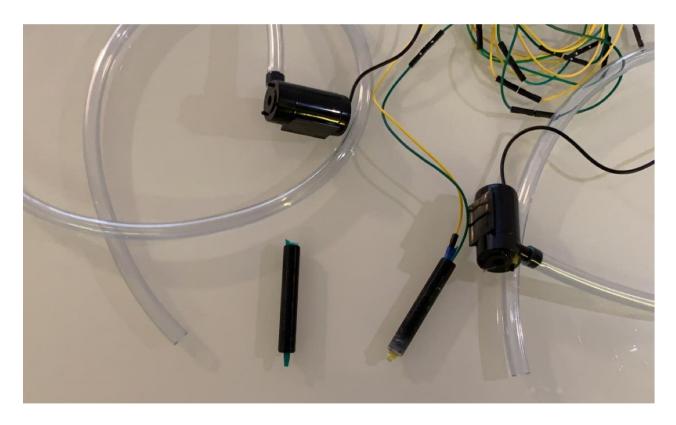


Figure 2 – From left to right: the vacuum, drawing tool, LED, and water blaster.

The key physical features of the product include the 3D printed end effector and cylindrical pieces, the water blaster, LED light, vacuum, drawing tool, along with the Arduino board and components that control them. The product also has many software features such as the GUI, the inverse kinematics solver, and the code for the Arduino. The major functions of the physical pieces include water blasting an area, drawing any shape, lighting up a low-light area with the LED, and vacuuming the used water from the water blaster to avoid pollution into the ocean. There are also many software functions such as the GUI being able to turn on and off whichever tool is on the end effector, inputting coordinates into the GUI to move the arm and also draw, along with an emergency stop button to cut all power to the arm.

The 3 DOF arm consists of many sheets of wood that were laser cut, to become pieces of the arm itself or the gears for the arm to move. There are 3 servo motors attached to the arm, each responsible for 1 DOF each. Controlling the motors are an Arduino board attached to the back with a CNC shield on top of it. All of this can be seen in Figure 3.



Figure 3 – The robot arm with our end effector on it. The Arduino board and CNC shield are out of frame on the back panel.

Our end effector is 3D printed from black PLA, along with the cylindrical pieces. The water blaster is a simple 3V water pump, that was reversed to act as a vacuum. The pump uses simple 9.5 mm tubing to transfer the water. The pumps and the LED are controlled with another Arduino board and breadboard, separate from the one on the arm. The marker is a simple dry erase marker that fits into the cylindrical pieces.

Our software components include the GUI which is generated through VS code, the Arduino IDE, and the inverse kinematics solution which were all coded using python.

2.1 Cautions & Warnings

Before using our robotic system, it is important to know that there is a lot of moving parts that can cause a lot of safety concerns if proper safety protocols are not followed. The first thing the user should be careful of is the end-effector piece itself. If loaded too much in terms of the weight of the items attached to it, you risk the possibility of potential failure at any point during the operation. This can cause cracking, or even permanent deformation, thus compromising its structural integrity. Moreover, the end-effector is attached to a moving robotic arm which contains many gears, tubing and wiring running all around the system. To prevent any possible bodily harm, make sure to tie long hair back, make sure all power is cut off to the robot in the case of making any repairs, and most importantly wear PPE when performing any modifications to the system.

3 Getting started

To get started with the Inverse Kinematic Robot:

You must first ensure that the end-effector is properly attached to the robotic arm itself. One can check by making sure there are two screws on both ends of the end-effector, going through the specified holes as shown in Figure 4 below.

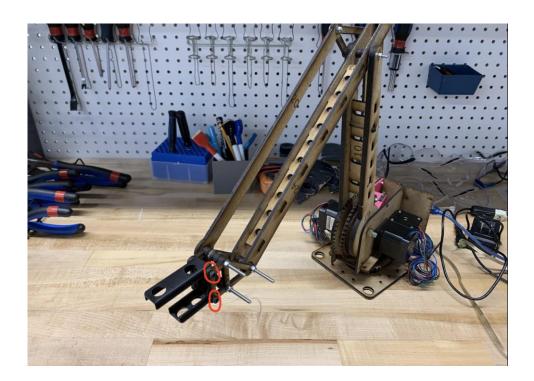


Figure 4 - Screw locations for end-effector

Once the end-effector is securely placed onto the robotic arm, one can now start to place the required attachments onto the end-effector, which can vary depending on the task at hand. The attachments include an LED light, water blaster, marker, camera, or infrared sensor. Before fitting any attachment onto the end-effector, certain pieces such as the LED light and marker must first be placed into the provided holder shaped as a long cylinder, as shown in Figure 5.

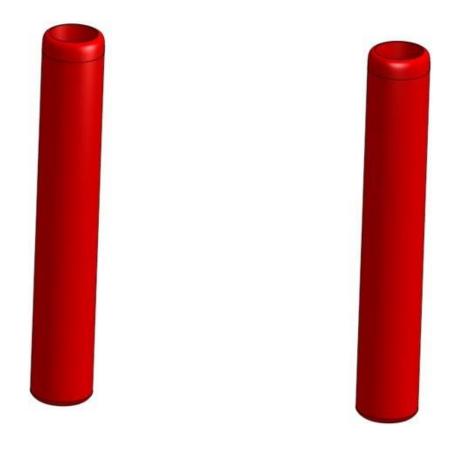


Figure 5 - Attachment holder

Once placed inside the cylindrical holder as shown in the photo above, you can then attach the water blaster or whatever piece that suits the task at hand. Since this is a dual-pronged modular design, you can place up to 2 items on the end-effector at once for multi-purpose applications. A clearer instruction on how to properly attach the attachments to the end-effector is shown below in Figure 6.

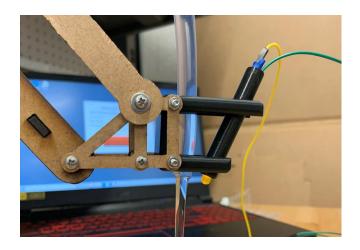


Figure 6 - Example of proper attachment setup

Moving on, you will need to get familiar with the GUI (Graphical User Interphase) that is provided with the robotic end-effector. The GUI allows you to set the coordinates of where you want the robot arm to go to, as well it allows you to toggle on and off which attachment you want to use for the given task at hand, depending on the attachments mounted on the end-effector.

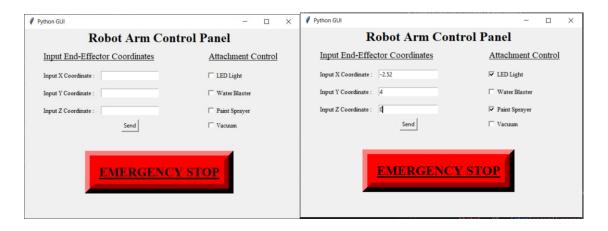


Figure 7 - GUI

Figure 7 shows you the home screen of the GUI. On this screen, you can see the different options pertaining to the Input End-Effector Coordinates, and the Attachment Control. Figure 5 gives an example of possible inputs that you can put into the system. The checkmarks under

Attachment Control in Figure 7 indicate that those attachments are on the end-effector and are 'on'. The boxes without a checkmark mean that those specific attachments are 'off'.

Warning! If you input coordinated outside the reach of the robotic arm, you will be prompted with a warning message as shown in Figure 8 below. If this occurs, you must adjust the coordinates to fit within the range of the robot arm in order to not receive another error message.



Figure 8 - GUI Coordinate Error

In the case of an emergency, there is a visibly red EMERGENCY STOP button, which will cut all power to the robotic end-effector, as well as to any attachments mounted on it as shown in Figure 9. In the case of an emergency, you can press this button to cut the power, and once the situation has been handled, you can simply exit out of the emergency screen by simply pressing 'X' or 'OK'.



Figure 9 - GUI Emergency Stop

3.1 Configuration Considerations

The equipment involved in the Inverse Kinematic Robot are the end-effector piece, the M3 screws required to properly secure the end-effector to the robotic arm, the robotic arm itself, the attachments which include a water blaster, vacuum, LED light, paint sprayer as well as the attachment holder. All these will be depicted in the figures below.

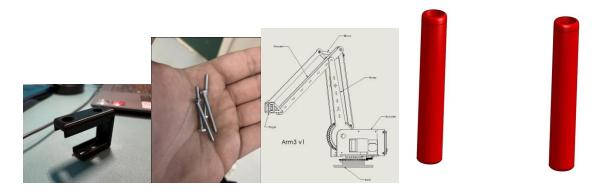


Figure 10 - Photos of the equipment

There are 2 Arduino boards in this system that deal with the communication and configuration throughout. One Arduino contains code that sends a specific signal to the end-effector in order for it to perform its necessary functions like turning the LED light on or turning the water blaster on. The second Arduino controls the movements of the robotic arm itself. It also contains code and sends specific signals to the motors of the robot arm so that it can move/rotate in the x, y and z directions.

3.2 User Access Considerations

The different users and/or user groups that could be using the product are of course engineers. Engineers have the scientific and mathematical knowledge and theories to understand and operate such a system with only a minimal number of restrictions placed on them. However, due to the simplicity of this system and ease-of-use methods involved, engineers are not the only group of people that could use the product! Technicians and even individuals with just a high school diploma can operate this system as well! Parts can be easily interchangeable, and the Graphical User Interphase is simple enough for anybody to use. In terms of restrictions, if the structural integrity of

the end-effector is compromised, or if there is a bug within the software code, an individual with a lot of experience like an engineer or a highly specialized technician might need to be called in to assess the situation. However, due to the simplicity of the design of the end-effector, a new piece can easily be 3D printed instead of having to bring in higher trained personnel.

3.3 Accessing/setting-up the System

The procedures necessary to set-up/turn on the system are as follows.

- 1. The user must first ensure that the end-effector is properly attached to the robotic arm itself using the provided M3 screws and bolts.
- Once attached, the next step is to attach the attachments provided (water blaster, vacuum,
 LED light or paint sprayer) to the attachment holder then onto the end-effector.
 - * Warning! If the water blaster is in use, it is important to ensure that the hose is attached to the proper water pump, which is submerged in a vessel containing water. This ensures that there is a constant supply of water for the task at hand.
- 3. Once the end-effector and attachments are placed onto the robotic arm, the GUI (Graphical User Interphase) must now be set up. To do this, you must open up the GUI software in order for it to prompt you with the system home screen.
- 4. Once at the home screen, the user can now input the required coordinates that they want the robotic arm to move to.

- * Warning! If the coordinates input are out of range of the robotic arm, you will then be prompted with an error message stating that the 'COORDINATES ARE OUT OF REACH'. If this occurs, you must then change your coordinates to make sure they are accepted by the software (within range) before being able to use the system.
- 5. Once the coordinates are input, the user can then select which attachments will be in use which is located on the right side of the GUI. The check mark indicates that it is on and operating at that moment.
- 6. To turn off the system, you must de-select the attachments that are in use. When there is no check mark inside the box, it indicates that that specific attachment is no longer operating, and no power is being sent to it. Do this for all attachments that are involved to completely turn off the end-effector.

This system specifically does not involve any passwords or user ID to long into the system with.

Once the software is installed and all the parts are properly connected to the robotic arm, the user can quickly and efficiently use all the provided functions.

3.4 System Organization & Navigation

The organization of the system is straightforward, and each component is interconnected in some way. The main components include the end-effector, the robotic arm, GUI, the attachments as well as 2 Arduinos. The following will illustrate the organization of the system.

- The end-effector is mechanically connected to the robotic arm using 2 M3 screws
- Attachments are hand-fitted into the attachment holder
 - Once the attachments are fitted into the attachment holder, they are then fitted onto
 the end-effector which was previously connected to the robotic arm
- The GUI of the software allows the user to control the options and motions of the system.
 - The GUI is connected through both Arduinos. Once coordinates are inputted it sends a signal to the motor then gears of the robotic arm to move in the specified direction. Next, once the specific attachment options are selected, it sends a signal to the second Arduino which controls the power being sent to those specific attachments, turning them off or on.

3.5 Exiting the System

The actions necessary to properly exit the system are as follows. The user must first de-select all active attachments that are in use on the provided Graphical User Interphase or the home screen.

Once all have been de-selected, no power will be provided to the system, and it is then shut off. To exit the software, it is as simple as clicking the 'X' button at the top right of the home screen. To put away the device, detach all attachments from the end-effector and place them in the specified storage areas. Next, unscrew the 2 M3 screws holding the end-effector to the robotic arm but make sure not bend or apply too much pressure on the end-effector and robotic arm when doing so, since it can compromise the structural integrity of both. Finally, you can put both the end-effector and robotic arm back in their proper storage areas where they can be easily accessible for use another time.

4 Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions of the End Effector. The five features include water blasting, vacuuming, lighting up a dark area, moving, emergency stopping, and drawing.

4.1 Water Blaster

The water blaster consists of a simply 5V water pump, plastic tubing, and a water reservoir (refer to Figure 2). The pump is connected to the Arduino via a breadboard. In Figure 31 you can see how it is setup on the breadboard. The blaster is fully incorporated into the GUI so that is where it is turned off and on. Make sure the pump itself is in the water reservoir so it can suck up the water and pump it through the tubing. Once the pump is in the water and the other end of the tubing is in the end effector on the robot, simply clicking the box labelled "Water Blaster" on the GUI will turn on the water blaster (refer to Figure 11).



Figure 11 – The GUI with the Water Blaster circled

It will begin sucking up the water through the tubing coming out the other side. To turn it off, simply click the box again and the blaster will turn off. Once turned off the water will stop being sucked up and the remaining water in the tube will fall out. Another way to turn off the pump is to hit the emergency stop which ceases all devices on the end effector.

4.1.1 Vacuum

A sub feature of the water blaster is vacuuming. The physical vacuum is using the exact same components as the water blaster, as seen in Figure 2. By using the exact same set-up as before in Figure 31, the pump can be switched to suck water up from the end effector instead of pouring from it. Once the vacuum is set-up in the end effector, click on the box in the GUI labelled "Vacuum" to beginning vacuuming (refer to Figure 12). This will begin sucking up water from the end effector

into a container at the other end of the tubing. To turn off the vacuum, click the same box again and it will stop vacuuming. Another way to turn it off would be with the emergency stop, just like the water blaster.



Figure 12 – The GUI with the Vacuum circled

4.2 LED Light

The LED light consists of a simple Arduino powered yellow LED light which has been inserted into the cylindrical tube. As seen in Figure 13 the light simply sticks out of the end and can be slotted into the end effector like in Figure 1.



Figure 13 – The two cylindrical pieces with LED lights in either one sticking out of the top

Inside the tubing, the wires have been soldered together for ease of connecting and disconnecting over and over. Since it is a small light, using a resistor is important to not accidentally burn it out. The resistor has been soldered and fits into the cylindrical tube, out of sight. Like the other attachments, the LED light function has been included in the GUI. By clicking the box labelled "LED light", the light will turn on instantly (refer to Figure 14). The light is usually used along side another attachment such as the user can use this function in a low light environment, an example of this can be seen in Figure 1. To turn off the LED light, simply click the box on the GUI again and it will turn off. Like all the other attachments, clicking the emergency stop will also turn off the light.



Figure 14 – The GUI with the LED Light circled

4.3 Moving the 3 DOF Arm

One of the main features of the include moving the robotic arm. Moving the arm consists of running the inverse kinematics code and transferring that to the Arduino IDE to move the motors. Moving the arm can be done alongside using any of the attachments on the end effector to water blast a big area for example. To move the arm, enter the desired coordinates into the GUI and click send (refer to Figure 15). If the coordinates are out of reach for the arm, a message will confront the user saying that what they have entered is out of reach (refer to Figure 8). To try again, simply click ok once the message comes up and input coordinates in range for the arm to relocate.



Figure 15 - The GUI with inputted coordinates, ready to click send

By clicking send, this will run the inverse kinematics algorithm which will retrieve the required angles for each DOF of the arm. Then the Arduino will move the motors to reach those angles and get to the desired coordinates. To stop the arm from moving for any reason, hitting the emergency stop will kill all the power and stop the arm.

4.3.1 Drawing Tool

The drawing tool is a sub feature of moving the arm. The drawing tool consists of a marker which has been placed into the cylindrical piece to fit into the end effector (refer to Figure 2). Once the marker is in the end effector, the arm can they be moved to draw a shape or spell a word. Moving the arm is the same as explained before, but several coordinates need to be put in to move it several times so the drawing tool can draw something. Like every other moving part in this product, the movement can be stopped with the emergency stop button.

4.4 Emergency Stop Button

The emergency stop button is a key feature of the product. As explained briefly in each other section it enables the user to completely shut off all the motors and Arduino boards on the arm. This means that it will instantly stop running whichever attachments are in the end effector, along with stopping the arm from moving around. To activate this feature, simply click the large "EMERGENCY STOP" button on the GUI. This will then cease all the power to the arm. A message will prompt the user saying what has happened and which files to report for whichever emergency (refer to Figure 16); this message can be changed by the user if desired. Once the emergency is over, the user may click ok on the message, and this will instantly resume activity from where it stopped before the button was clicked.



Figure 16 – The GUI after hitting the emergency stop button

5 Troubleshooting & Support

If the user runs into maintenance problems, error messages, or requires any type of support, there are a variety of procedures for different user requirements. Different corrective actions that could be taken include submitting a formal issue complaint or client request. There are also integrated actions within the system in the case of certain pre-conceived possible errors, such as faulty serial connections, or irresponsive components to commands. Outlined below are the guidelines for said procedures for user support.

5.1 Error Messages or Behaviors

When compiling the code and uploading it to the Arduino Uno microcontroller, it is possible that a message confronts the user informing them that the serial connection between the board and the host computer could not be made. This is simply since the COM port must be selected before anything can be uploaded and can be fixed by going into the Arduino IDE and selecting Tools->Port->COM and choosing whatever port is available. Note that the port chosen should match that inside the project source code found under the '#serial command and pin settings' section.

If the attachments on the end-effector are not responding to the updates on the GUI the user must check whether the jumper cables have been set in the correct PWN female pins. If the problem persists, consult the wiring diagram and ensure that all cables going in and out of the transistor have been laid properly and that no damage has been incurred.

5.2 Special Considerations

Given the presence of water around electronics, it is important to check the seal on the inlet of the silicon water tube as well as its integrity throughout so as not to get any components wet. If the microcontroller gets wet, the user must quickly hit the emergency stop button to kill all power and avoid any shorting of the circuits.

5.3 Maintenance

Considering the water pump will be using the sea's salt water, it is important to regularly check the buildup and deposits of minerals to avoid fouling. The pump should be inspected, cleaned, and documented weekly to prevent fouling and damage. It may be circulated with anti-foaming product and cleaning solution then drained to clean the system.

5.4 Support

In case the user requires emergency assistance or system support, there are a few options for contact and aid.

If the user requires assistance with setup or configuration, they can contact setup support (Omar Abdul-Ghani):

- lastendeff.setup@uottawa.ca
- 300-198-650 ext. 300

If the user requires assistance with production and recreation, they can contact production support (Ethan Salt):

• lastendeff.prod@uottawa.ca

• 189-165-300 ext. 176

If the user requires assistance with repairs of performance issues, they can contact repair support (Samuel Harrison):

- lastendeff.rep@uottawa.ca
- 806-300-070 ext. 139

In case the product is malfunctioning and there are security issues, the user may inform our safety support team (Hannah Chan and Jason Nguyen) of the identified problems and report the issue formally through our website:

- lastendeff.safe@uottawa.ca
- 300-144-344 ext. 555
- www.thelastendeffector.com/invrsknmtcs/issue-report

6 Product Documentation

6.1 Inverse Kinematics Test Arm

6.1.1 BOM (Bill of Materials)

Item	quanitity	Unit Price (CAD)	Total (CAD)
Arduino Uno R3	1	26.95	26.95
MtoM Jumper Cable	20	0.07	1.48
From Jumper Cable	20	0.07	1.48
Breadboard	1	6.95	6.95
AA Battery Case	1	3.99	7.98
5V Water Pump	5	5.06	25.29
PLA Filament	5.6g	0.03	0.168
30mm Philips M3 Machine Screw	2	0.30	0.60
M3 Thread Bolt	2	0.30	0.60
Vinyl Tubing	10ft	0.72	7.15
Transistor	1	0.13	0.13
Resistor (1200hm)	1	0.01	0.01

6.1.2 Equipment list

<u>Hardware:</u> Cylindrical wooden dowels, square wooden dowels, MDF board, grid pattern paper, calculator, ruler, pencil, red marker.

27

Software: Python

6.1.3 Instructions

The arm was created using cylindrical and square wooden dowls as well as thin MDF board. Links were attached using a pin dowel in system allowing for full 360-degree rotation, limited only by other links. The grid pattern is for coordinate purposes, lines are in 2in intervals, as such link length and z-offset is divided by 2 for the new length unit. Red lines are remarkable angles (bounds of 15deg) as a reference for the theta 1 angle.

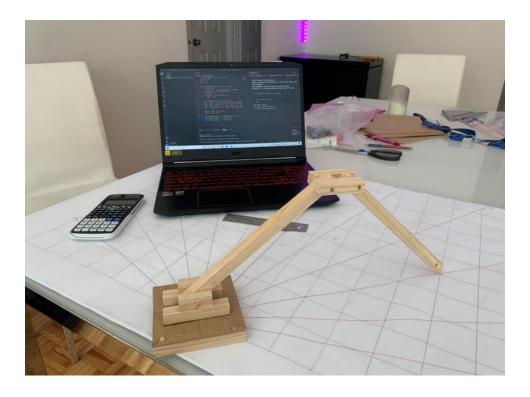


Figure 17 – Setup used to test inverse kinematics algorithm

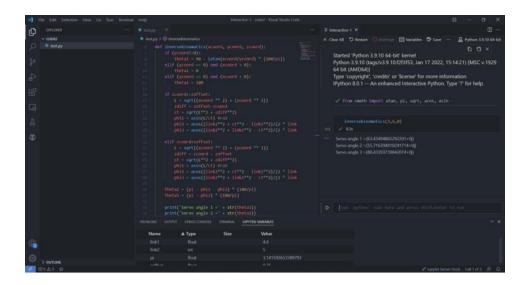
The computer in the background (see Figure 17) contained the code written in Python to determine the joint angles based on the input of end effector coordinated. Code was run in a Kernel to easily interact with and receive data from calculations.

6.1.4 Testing & Validation

The tests that were done on the prototype for validation of the final design was to see if our inverse kinematic algorithm demonstrates high fidelity.

Test case 1:

The first test we did on the code was to see if it can provide the proper angle that the wooden arm was displaying. First, we adjusted the arm to portray 64 degrees, and calculated the x, y, and z coordinates that will give us the exact angle. The values were 63.3 degrees, 55.7 degrees, and 88.4 degrees. Inputting those values into the code produced the angle 63.43 degrees, demonstrating that the code has high fidelity.



 $Figure\ 18-Angle\ output\ from\ inverse\ kinematics\ algorithm\ with\ inputted\ values\ for\ test\ case\ 1$

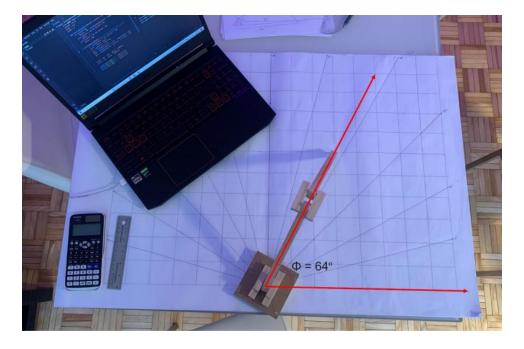


Figure 19 – Aerial view of test case 1 for the inverse kinematics prototype at 64 degrees

Test case 2:

The second test we did on the code was to see if it can provide the proper angle that the wooden arm was displaying once more. First, we adjusted the arm to portray 145 degrees, and calculated the x, y, and z coordinates that will give us the exact angle. The values were 143.13 degrees, 29.6 degrees, and 88.4 degrees. Inputting those values into the code produced the angle 143.13 degrees, demonstrating again that the code has high fidelity.

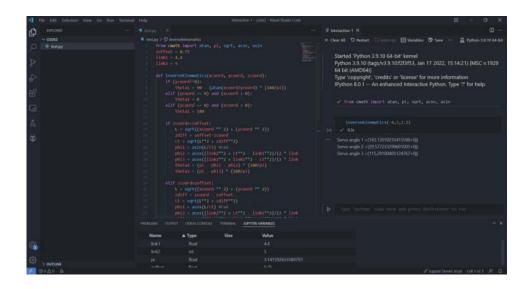


Figure 20 – Angle output from inverse kinematics algorithm with inputted values for test case 2

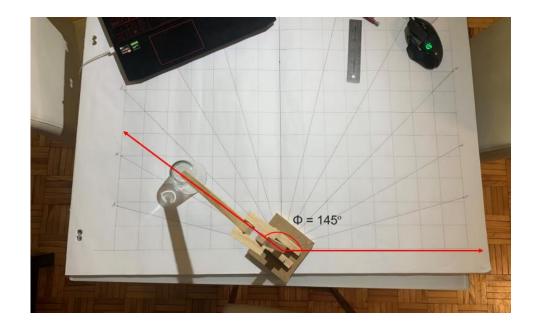


Figure 21 – Aerial view for test case 2 for the inverse kinematics prototype at 145 degrees

6.2 End Effector and GUI

6.2.1 BOM

Item	quanitity	Unit Price (CAD)	Total (CAD)
Arduino Uno R3	1	26.95	26.95
MtoM Jumper Cable	20	0.07	1.48
From Jumper Cable	20	0.07	1.48
Breadboard	1	6.95	6.95
AA Battery Case	1	3.99	7.98
5V Water Pump	5	5.06	25.29
PLA Filament	5.6g	0.03	0.168
30mm Philips M3 Machine Screw	2	0.30	0.60

M3 Thread Bolt	2	0.30	0.60
Vinyl Tubing	10ft	0.72	7.15
Transistor	1	0.13	0.13
Resistor (1200hm)	1	0.01	0.01

6.2.2 Equipment List

<u>Hardware:</u> 3D printer, PLA filament with an 8mm nozzle at 44 mm/s and 20% infill, 2 M3 machine screws, pencil

Software: Python, OnShape CAD

6.2.3 Instructions

GUI:

The client wanted everything in the project to be written in Python. As such, we used the Tkinter Python Library to create a visually appealing and functional GUI. To properly lay everything out, we used grid development similar to CSS. Each item was added to its specific grid where it was

then either centered or set to the far left. The font on everything was Times New Roman with a front size of 12, except for the emergency stop that took a size 48.

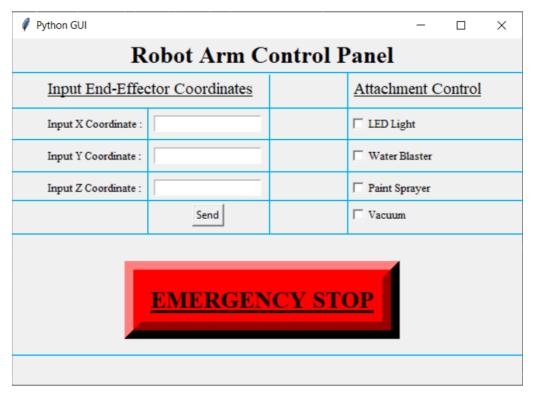


Figure 22 – Grid pattern of the PTinker front end to help user replicate the design

Each instance of text is initialed using the Label() function, input with Entry(), buttons with Button() and checkbox using Checkbox(). In order to get the emergency, stop border and colour one put simply change the bandwidth inside the Label function and set the bg to 'red.

The popups created by the GUI are done using the command function. When a button is pressed you can have it execute a certain function. For the case of a clone, 3 are needed. The inverse kinematics function returns the angles when the send button is clicked, the sendcoords functions retrieves the input of the user and uses showerror in the case of invalid coordinates, and the

emergency stop function uses PyMata to set the digital pins to low and .showwarning to display as warning message.

Attachment Control:

Since the attachment control equally needed to be written in Python, a workaround the native C++ Arduino IDE had to be found. Using the PyMata Firmata Communication Protocol one can communicate with the Arduino via Python. In order to do so, the selected port must be initialized the .port function, then all the pins must be initialized to be used in the board.digital[pin].write() function. Equally, the type of Arduino board must be selected, so the Uno R3 in this case. Next, using some conditional statements regarding the state of the checkboxes and buttons, the attachments can be set up to be turned on when clicked. To get the emergency stop to work, rather than .get(), one must use the button['state]== function as .get() returns None regardless of its state.

End-effector Prototype I:

When 3D printing the prototype, it was printed using PLA with an 8 mm nozzle at 44 mm/s and 20% infill. The end-effector was attached using two M3 machine screws.

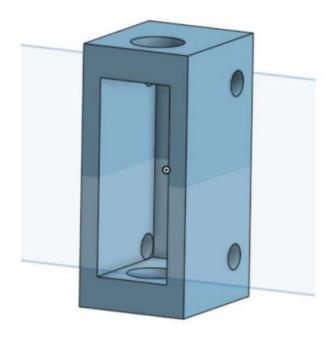


Figure 23 - Concept of prototype I in OnShape CAD

End-Effector Prototype II:

Regarding aesthetics, the end-effector was rounded off all the outside cornering. Next, since the walls didn't need to be as strong, it went from a thickness of 5 mm to 3 mm and created a hole between the top and bottom holders. The overhang, however, still had a 5 mm thickness to act as I beams for a higher shear strength in the Z direction.

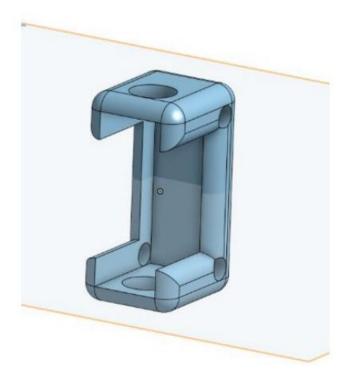


Figure 24 – Concept of prototype II in OnShape CAD

End-Effector Prototype III:

In this prototype, we extended the top holder and bottom holder, allowing a vertical attachment as well as an additional vertical attachment coming in at a 30 degree angle.

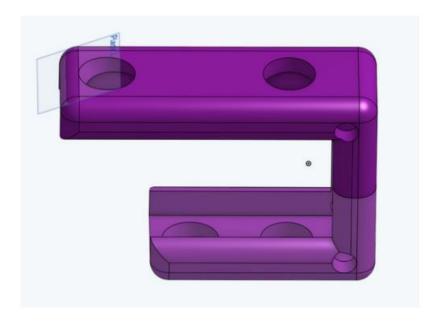


Figure 25 – Concept of prototype III in OnShape CAD

6.2.4 Testing & Validation

We conducted tests on the next important subsystems, the GUI and end-effector. Regarding the GUI, we tested its ability to communicate with the back-end algorithms. Regarding the end-effector, we tested its ability to be used as an attachment for the robot arm.

GUI Test Case 1:

We entered values that were out of range for the arm to see if the inverse kinematics code can communicate to the GUI that the coordinates are out of reach. When inputting x = 14, y = 14, z = 14, our code was able to detect and communicate to the GUI that the coordinates are not valid, because they are out of range of the arm, leading to the GUI displaying the words:

"COORDINATES OUT OF REACH". This test has confirmed that the GUI can properly communicate with the back-end algorithms.



Figure 26 - Pop-up display when inputted coordinates are not within the robot arm's range

GUI Test Case 2:

We entered values that was within the range for the arm to see if the inverse kinematics code can communicate to the GUI the specific servo angles that were produced from the coordinates. When inputting x = -4, y = 3, and z = 1.5, our code was able to detect the inputs and communicate to the GUI that the coordinates are valid, leading to the GUI displaying the window with angles 1, 2, and 3 (1 = 143.13, 2 = 29.577, 3 = 115.29). This test has confirmed that the GUI can properly communicate with the back-end algorithms.



Figure 27 – Angles given when x=-4, y=3, z=1.5

End-Effector Prototype I Test Case:

The first prototype's goal was to create an end-effector that can easily and securely attach to the robot arm as well as make sure that it can hold the flexible tubing and pencil. The aesthetics, weight, and structural integrity were not considered during the making of the design. After creating our prototype on OnShape CAD and 3D printing it, it was able to easily be attached to the arm and hold a pencil with enough strength to hold the pencil at max motor torque. From this test, we concluded that this design is one to continue developing, as it produced a high level of fidelity.

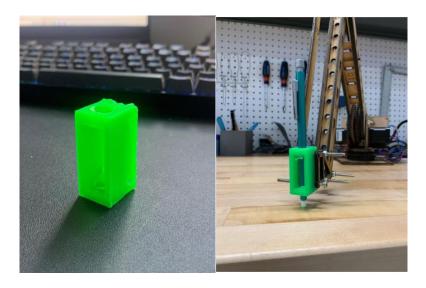


Figure 28 – Prototype I end-effector with and without a pencil attachment

End-Effector Prototype II Test Case:

The second prototype's goal was to build onto the first prototype's short comings as previously stated (aesthetics, weight, and structural integrity), since the first prototype accomplished all of its goals. After making changes previously stated in the second prototype section under "Instructions", it allowed us to reduce the volume from 8.220 cm³ to 4.439 cm³, resulting in a 46% decrease in weight. Despite the decrease in weight, it still maintained a high level of structural integrity and resistance to mechanical stress. From this test, we concluded that this design is one to continue developing, as it produced a high level of fidelity.

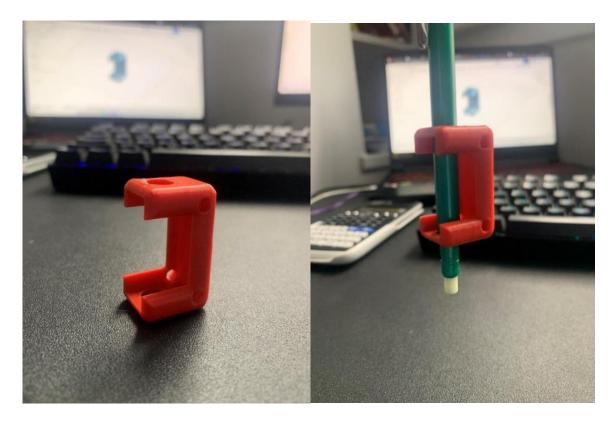


Figure 29 – Prototype II end-effector with and without a pencil attachment

End-Effector Prototyping III Test Case:

The third prototype's goal was to build onto the second prototype, and include an additional prong to develop a dual-prong end effector. The changes discussed in the previous section under "End-Effector Prototype III" were tested on its stress levels, because of its increase in torque as a result of its increase in length. The design did slightly bend, but it only started to bend with full strength of a human being. The motors would never be able to produce the force required to bend or break the design, making this prototype high-fidelity.

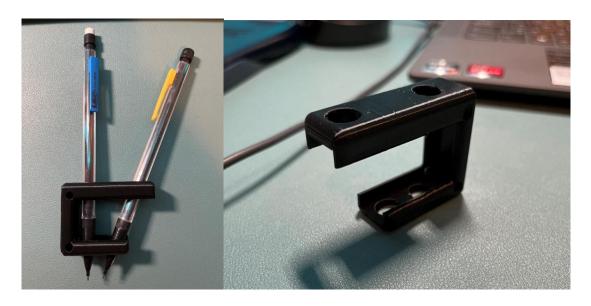


Figure 30 – Prototype III end-effector with and without an attachment

6.3 End Effector Components and Arduino Connection

6.3.1 BOM

Item	quanitity	Unit Price (CAD)	Total (CAD)
Arduino Uno R3	1	26.95	26.95
MtoM Jumper Cable	20	0.07	1.48
From Jumper Cable	20	0.07	1.48
Breadboard	1	6.95	6.95
AA Battery Case	1	3.99	7.98
5V Water Pump	5	5.06	25.29
PLA Filament	5.6g	0.03	0.168
30mm Philips M3 Machine Screw	2	0.30	0.60
M3 Thread Bolt	2	0.30	0.60
Vinyl Tubing	10ft	0.72	7.15

Transistor	1	0.13	0.13
Resistor (1200hm)	1	0.01	0.01

6.3.2 Equipment List

<u>Hardware:</u> PLA, 3D printer, red LED light, blue LED light, water pump, tubing, Arduino R3 Uno, breadboard, BC547 transistor, 4 AA batteries, power supply, DC motor, soldering tool, jumper cables, video camera

Software: TinkerCAD, python, Arduino IDE, OnShape CAD

6.3.3 Instructions

Wiring:

The wiring for the water pump was first prototyped in TinkerCAD to make sure everything would run correctly. See Figure 31 for how the wiring was done. Using actual hardware, the same circuit was then replicated to test its implementation. A BC547 transistor was used as we are running 6V through the breadboard. The power supply is 4 AA batteries in series. Jumper cables attached to the DC motor were soldered.

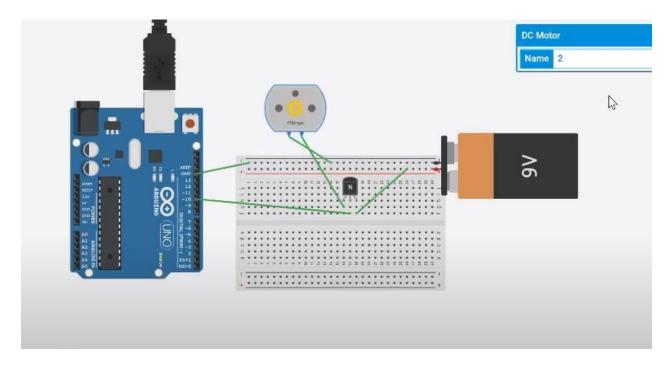


Figure 31 – Wiring for the DC motor of the water pump and water vauum

End-effector:

Two cylindrical pieces were made to fit into our dual-prong end-effector using OnShape CAD. It was 3D printed using PLA to perfectly fit into the holes of the end effector. They were also made to fit an LED light since normally an LED light is too small for the hole of the dual-prong end-effector and will fall through the hole.



Figure 32 - CAD concept of cylinders for end effector and the printed cylinders for the end-effector

GUI:

The same GUI interface created in prototype II will be used for this prototype. However, this time the GUI's "Attachment Control" and its "EMEREGENCY STOP" button will be coded to function. Regarding the "Attachment Control" button, when a box is checked, it will turn on the certain device. When a box is unchecked, the certain device will be off. Regarding the "EMERGENCY STOP" button, clicking it will cease all function of the robot arm, displaying a pop-up window stating: "ALL ATTACHMENTS AND MOTOR FUNCTIONS CEASED FOR FIRST AID REPORT TO C215 INCIDENT REPORT MUCH BE FILED" When "OK" is clicked, the robot arm will resume from where it left off.

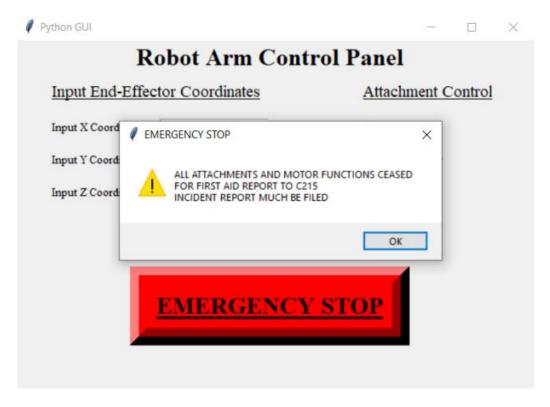


Figure 33 – Pop-up display when EMERGENCY STOP is pressed

Python Code for GUI:

The first part initializes all the pins and what they do, as well as open the serial port. The while true loop is used to keep it running indefinitely. Window.update () is to detect changes done by the user. Multiple if statements were used to check the status of the checkboxes. The last if statement checks the state of the emergency button; if it is clicked, it will turn off all components. See Figure 33.

6.3.4 Testing & Validation

Wiring Test Case:

To test our wiring, we first used TinkerCAD to determine if our wiring would theoretically work. We used a DC motor, NPN transistor, 9V battery, and a breadboard to simulate our water pump. Applying what we did on TinkerCAD to actual hardware, we got the wiring to function, demonstrating that our theoretical test on TinkerCAD produced high-fidelity. See Figure 34.

Figure 34: https://www.youtube.com/watch?v=k4C8fNMt x0

GUI and Python Code Test Case:

To test if our "Attachment Control" and "EMERGENCY STOP" button, we created a code in python that would make the two components functional. Regarding "Attachment Control", we validated that our code was successful because when checking a box, the attachment will turn on (see Figure 35), and when unchecking the same box, the attachment will turn off, making the test produce high-fidelity. Regarding the "EMERGENCY STOP" button, we validated that our code was successful because when clicking the button, the robot arm will cease all function (see Figure 33), producing a pop-up display stating: "ALL ATTACHMENTS AND MOTOR FUNCTIONS CEASED FOR FIRST AID REPORT TO C215 INCIDENT REPORT MUCH BE FILED". This makes the test produce high-fidelity. See Figure 35.

Figure 35 - https://www.youtube.com/watch?v=Y4jDBAXgk71

7 Conclusions and Recommendations for Future Work

The first lesson learned would be that everything always takes longer than anticipated. With our prototypes, we learned that it takes a lot of time to iron out all the issues and physically make the prototype. We had planned to make them much earlier but did not realize how long they would take and ended up making them much later. Along with that, we did not order the materials we needed in time such as the water blaster and tubing, so we had to wait even longer. Another issue was making the Arduino communicate with our LED light and water blaster took more time as well since there were unforeseen errors that we ran into such as the LED lights burning out and the wiring set up not working for an unknown reason. Another lesson that we learned is that hope for the best, expect the worst. The client changed his mind halfway through the project which caused many groups to have to change their design. So, it is important that we come up with extra designs and extra ideas in the event the client changes his mind about the product, so we are not stuck with having to completely restart or completely change our design. The third lesson is related to the previous in that contingency plans are invaluable. As previously mentioned, the client can very easily change his mind or not like your idea. So being able to easily adapt and have a contingency plan is very important. Finally, treat your group members like your friends. We noticed in doing this that everyone got along better, and we all wanted to work on the project. We are more motivated to come to the meetings and work as a team.

If we had more time to work on this project, we might have been able to make a better end effector design. Currently we have a two-prong design and designing a possible 4 prong design might have been possible. With this addition, more items could be attached to the end effector at

one time, majorly increasing the variability of the design. Along with the more prongs, we also might could have made an IR sensor and a camera. These are the two main attachments that we did not have time to make and would be a great addition to the product. An idea we had to abandon was to use SVG files for the drawing tool. Using SVG files would have allowed the user to input an image and the drawing tool would then draw that image by moving the motors correctly, using inverse kinematics. Another abandoned idea was to add a paint sprayer attachment to the end effector. This would allow an easier time of paint spraying a large area like on the side of a boat, instead of using a marker. The paint sprayer is set up in the GUI, but the idea could not come to light because of the lack of time. Ultimately, all our ideas were to increase the variability of the design, really focusing on the ability of the user to use multiple tools at once.

APPENDICES

8 APPENDIX I: Design Files

Makerepo Link: https://makerepo.com/sharr125/1116.gng1103-team10c

Table 3. Referenced Documents

Document	Document Location and/or URL	Issuance
Name		Date
Deliverable F	https://uottawa-	March 7
	my.sharepoint.com/personal/esalt070_uottawa_ca/_layouts/15	
	/guestaccess.aspx?docid=18b61b2b7d3b04ab19fec038674511	
	5fd&authkey=AZ9pqeDnPa1ReCQ3iHqLVLM&e=gm8HUi	
Deliverable G	GNG 1103 - Deliverable G.docx	March 13
Deliverable H	GNG1103 Deliverable H.docx	March 27