Project Deliverable E
Project Schedule and Cost

Ani Preedom
Christy Lau
Paul Shedden
Claire Durand

GNG1103 Project Group 6
February 6, 2022

Abstract

On Halifax-class frigates, the Department of National Defence has a need for a robotic arm that uses inverse kinematics to paint surfaces. The robot must also scan and clean areas to identify and remove defects. To design the robot, a design process with several steps will be followed. Thus far, conceptual designs have been generated using design criteria based on raw data gathered from one of the users. At this stage in the design process, prototype testing must be planned to ensure that the project can be completed in the allotted time. This report presents detailed design drawing, a prototyping test plan needed to create the design, a bill of materials (BOM), a list of materials needed for each prototype test, and some associated risks and contingencies for each test.

Table of Contents

1.0 Introduction

The Department of National Defence expressed the need for a robotic arm with three degrees of freedom to paint areas on Halifax-class frigates. Conceptual designs were created based on the design criteria produced from the interpreted needs. Using the conceptual designs a system was created and analysed to create a bill of materials and a prototype testing plan.

2.0 Detailed Design Drawing

In Deliverable D, many design ideas were created for each subsystem. Three design ideas for the overall system were selected after analysis of the preliminary ideas. Of the three overall system designs, the best design according to the design criteria was chosen to be pursued. Figure 1 shows a detailed design diagram of the three stepper motors for controlling the degrees of freedom of the robotic arm, the ultrasonic sensor to detect possible individuals or other objects in the area, and the kill switch to turn off the arm in case of an emergency.

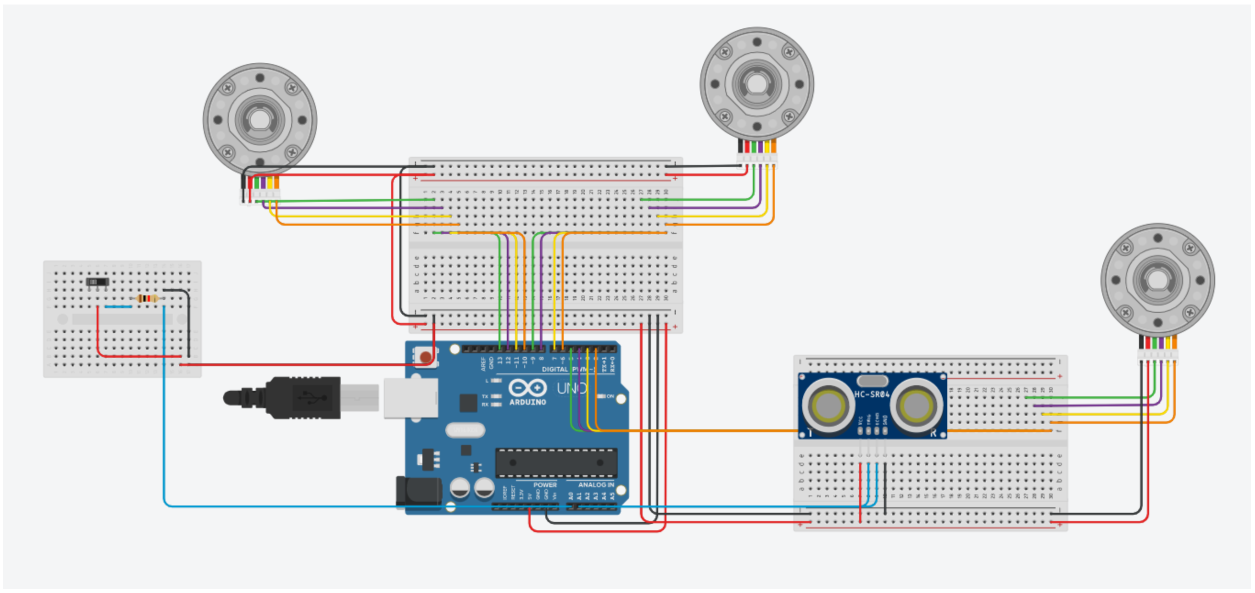

Figure 1. Detailed design diagram of Circuit for Motors and Sensors.

Figure 2 shows a detailed drawing of various viewpoints of the robotic arm, end effector while holding a paintbrush.

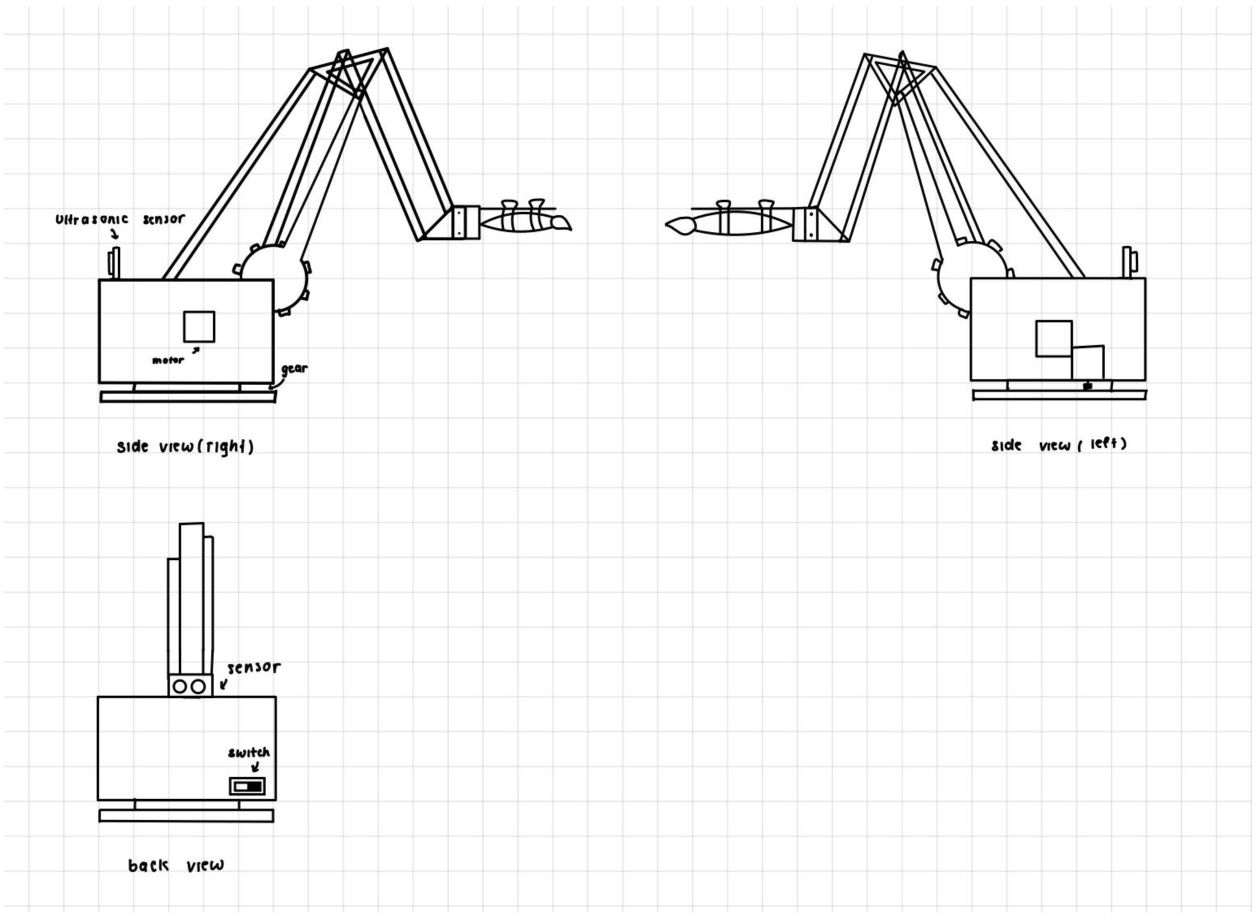

Figure 2. Detailed Drawing of Robotic Arm with End Effector.

Figure 3 shows a close up of the end effector and details the basic mechanics of how the spring loaded end effector will function.

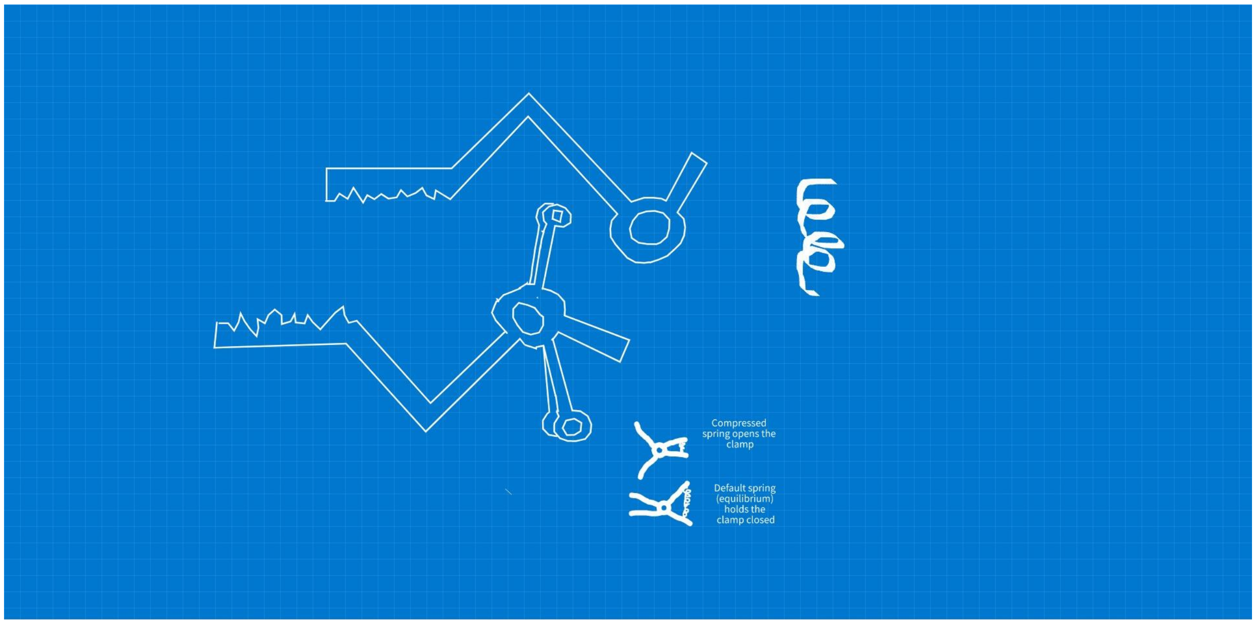
Compressed spring opens the clamp
Default spring (equilibrium) holds the clamp closed

Figure 3. Detailed Drawing of End Effector

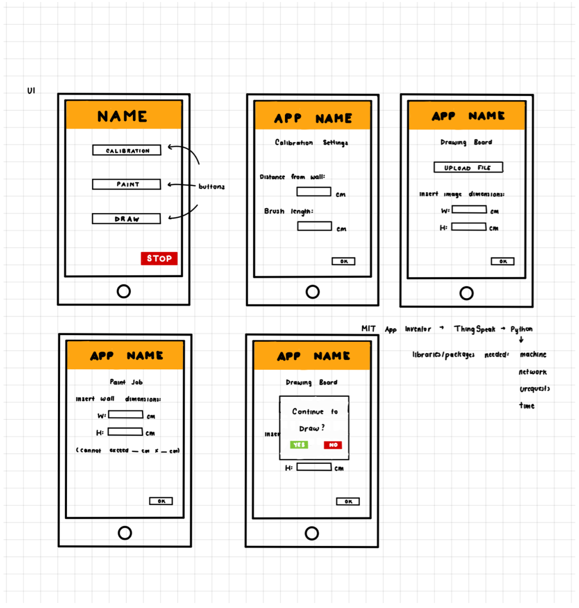

Figure 4. Detailed Drawing of User Interface and Mechanics of Interface

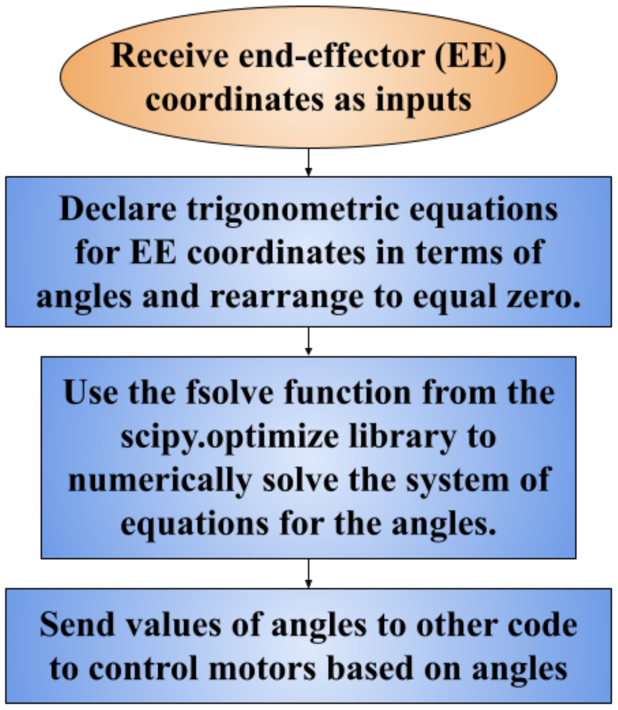
Receive end-effector (EE) coordinates as inputs
Declare trigonometric equations for EE coordinates in terms of angles and rearrange to equal zero.
Use the fsolve function from the scipy.optimize library to numerically solve the system of equations for the angles.
Send values of angles to other code to control motors based on angles

Figure 5. Flowchart of code for the inverse kinematics calculations in Python.

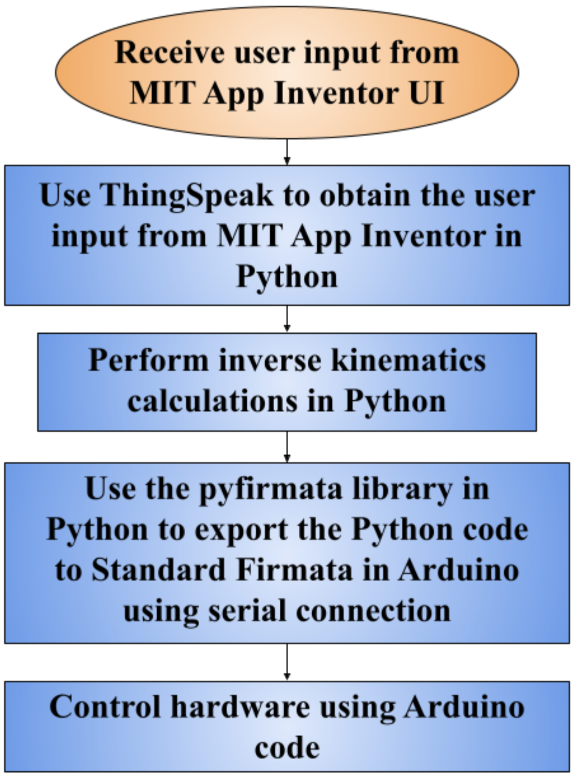
Receive user input from MIT App Inventor UI
Use ThingSpeak to obtain the user input from MIT App Inventor in Python
Perform inverse kinematics calculations in Python
Use the pyfirmata library in Python to export the Python code to Standard Firmata in Arduino using serial connection
Control hardware using Arduino code

Figure 6. Flowchart of overall software subsystem from the user interface to the Arduino.

3.0 Bill of Materials

See spreadsheet for details.

4.0 Prototyping Test Plan

Prototyping is important because it allows specific design elements to be isolated and tested. The approach chosen for this project is to conduct many tests that are smaller. This approach will allow the team to ensure that each part of the design works before building a more complicated part. To ensure that all tests can be completed on time, the following table will provide a structured plan that outlines the details of the prototype testing. Since there are several subsystems, preliminary tests are not necessarily dependent on each other.

Table 1. Prototype Test Plan

Test ID	Test Objective	Description of Prototype used and of Basic Test Method (What)	Description of Results to be Recorded and how these results will be used	Estimated Test duration and planned start

			(How)	**date (When)**
1	Compute correct values for angles	Python code will numerically compute a solution of a 2D inverse kinematics problem of which the analytical solution is known. The end-effector will be static.	Values of angles will be compared to the correct (known values). A match will indicate that the inverse kinematics code was successful.	Duration: 1 minute Test date: February 21
2	Compute correct values for angles	Python code will numerically compute a solution of a 3D inverse kinematics problem of which the analytical solution is known. The end-effector will be static.	Values of angles will be compared to the correct (known) values. A match will indicate that the inverse kinematics code was successful.	Duration: 1 minute Test date: February 21
3	Compute the correct values for angles.	Python code will numerically compute a solution of a 2D inverse kinematics problem where the end-effector must trace a line to simulate painting of a line.	-Compute an analytical solution of a static end-effector at multiple nodes (points on the line) and check if the angles are correct when the end-effector passes through each point -If possible, make an animation of the arm drawing and check whether the line is correct and the robot movement is feasible	Duration: 15 minutes Test date: February 21
4	Compute the correct values for angles.	Python code will numerically compute a solution of a 2D inverse kinematics problem where the end-effector must trace a circle to simulate painting of a circle.	-Compute an analytical solution of a static end-effector at multiple nodes (points on the circle) and check if the angles are correct when the end-effector passes through each point -If possible, make an	Duration: 1 minute Test date: February 21

			animation of the arm drawing and check whether the circle is correct and the robot movement is feasible	
5	Compute the correct values for angles.	Python code will numerically compute a solution of a 3D inverse kinematics problem where the end-effector must trace a line on a 2D surface to simulate painting of a line.	-Compute an analytical solution of a static end-effector at multiple nodes (points on the line) and check if the angles are correct when the end-effector passes through each point -If possible, make an animation of the arm drawing and check whether the line is correct and the robot movement is feasible	Duration: 1 minute Test date: February 22
6	Compute the correct values for angles.	Python code will numerically compute a solution of a 3D inverse kinematics problem where the end-effector must trace a circle on a 2D surface to simulate painting of a line.	-Compute an analytical solution of a static end-effector at multiple nodes (points on the circle) and check if the angles are correct when the end-effector passes through each point -If possible, make an animation of the arm drawing and check whether the circle is correct and the robot movement is feasible	Duration: 1 minute Test date: February 22
7	Ensure that the Ardunio can interpret and use Python code to perform calculations.	Python code will be interpreted by Arduino and cause a motor to rotate. The pyfirmata library will be imported to allow Python and Arduino to communicate.	If the motor moves, the test is successful because it is evidence that the python code is able to be used by the Arduino. If no movement occurs, the code did not work and debugging of code or investigation of other communication methods	Test duration: 10 minutes Test date: February 27

			will be required.	
8	Ensure that the Arduino can communicate with the motors.	Motors and the arduino will be connected to a breadboard. Arduino code will instruct the motors to rotate.	If the hardware components can communicate, the motor will move. Once this test is passed, Test 9 can take place and aim for a more precise result.	Test date: February 27
9	Check whether the Arduino can instruct the motors to rotate precisely to a specified angle.	The Arduino will make each stepper motor rotate by 90º. A straight object, such as a stick, will be attached to the motor to help measure the accuracy of the rotation.	The angle of rotation will be measured using a protractor. The start and end points of the stick will be marked as a reference for the measurement.	Test duration: 30 minutes Test date: February 27
10	Determine whether the end-effector can pick up an object with a mass higher than what is expected	This prototype is analytical so that success of the end-effector can be predicted prior to printing a physical prototype.	The analytical test will produce a number that when compared with a design constraint, will indicate whether the end-effector will be successful.	Test date: February 27
11	Test the conceptual design of a spring-based clamp end-effector	Build a physical prototype of the clamp end effector using inexpensive materials such as wood	This test is qualitative, with the purpose of gaining insight to the type of design. Qualitative observations can be used to modify the actual 3D printed design before it is printed.	Test duration:15 minutes Test date: March 3
12	Test to see if a simple app can be built using the MIT App Inventor.	Design a simple application on the MIT App Inventor that prints “Hello” to the user interface.	If the app runs successfully, the program will print “Hello” to the user interface. This proof-of-concept test can then allow the team to build more complicated	Test duration: 5 minutes Test date: February 26

			apps.	
13	Test to see if a simple app that uses user input can be built using the MIT App Inventor.	Design a simple application on the MIT App Inventor that asks a user for a string and prints the string back to the user/	If the app prints the string that the user entered, then the test confirms that user input is working successfully.	Test duration: 5 minutes Test date: February 26

5.0 Materials for Each Prototype

This section contains a table that outlines materials required to build each prototype and conduct its corresponding test. The numbers in Table 1 correspond to the numbers in the "Test ID" column of Table 2.

Table 2. Materials for each prototype test

Test ID	Materials Needed
1	Python
2	Python
3	Python
4	Python
5	Python
6	Python
7	Python Aruino Pyfirmata library Motors
8	Motors Arduino Breadboard
9	Arduino Stepper motor Plastic Bracket
10	Onshape

11	Wood/other cheap material
12	MIT app inventor
13	MIT app inventor

6.0 Risks and Contingency Plans

Table 3. Risks and Contingency plans

Risk	Issue	Contingency plan
1. High Costs	- The budget for this project is 50$, certain electronics cost a fair amount of this budget. This could be an issue moving forward.	- A detailed bill of materials must be created and followed. Things must be ordered in advance to avoid extra shipping costs.
2. Parts not coming in	- If parts don't come in on time it could greatly affect the progress of our arm.	- To avoid this issue, we must have a detailed list of materials and order online the parts as soon as possible
3. Lack of clarity	- Changes to the project and the responsibility of the arm will pose a great risk to the overall project.	- If something is not clear specific questions must be directed to the client about what the arm must accomplish.
4. Scheduling issues (prototype)	- There is not much time to develop each prototype, this could lead to time management issues down the line.	- The schedule that has been created on Wrike must be adamantly followed to make sure the project stays on track. If a team member falls behind on their portion they must reach out and ask for help.
5. Technology risks	- Technology risks such	- The prototype plan

	as the 3D printing machines or laser cutting machines not working could pose a threat to the success of the prototypes.	must be followed. The plan will make sure that the construction of the end effector is completed early, this way if the machines are not working we will still have plenty of time to complete the task.
6. Scheduling issues (software)	- Software is difficult to schedule because we will not know how long it will take to do some of the parts of it.	- To avoid this we will take time to estimate how long each portion will take and divide it up equally. If someone realizes that their section is going to take longer than originally planned they will reach out to the group and another team member can help them out.
7. Time issues (3D printing)	- 3D printing takes a considerable amount of time and if we do not take the time to calculate how long the 3D printing will take, it will be a big problem.	- To make sure we have plenty of time to complete the sections of the end effector using 3D printing the prototype test plan must be followed. It must allow time for the 3D printing to be completed and it must not be attempted the night before.
8. Code Issues	- Bugs and errors could be a result of rushing near the end of the project to get the software done on time.	- Frequently checking the code for bugs and errors can prevent this. Codes should also not be left with error messages to make sure nothing is forgotten.

7.0 Conclusion

The Department of National Defence has a need for a robotic arm that uses inverse kinematics to paint surfaces. In this deliverable, the ideal conceptual design developed in Deliverable D was used to create a prototype testing plan and a bill of materials necessary for developing the final product.

Appendix: Prospective Test Plan Ideas

Although test plans up until March 6 have been listed in Section 4.0, it is important to have foresight for better time and project management. In this appendix, tentative ideas for future test plans are suggested.

Table 4. Prospective Prototype Test Plan Ideas

Test ID	Test Objective	Description of Prototype used and of Basic Test Method (What)	Description of Results to be Recorded and how these results will be used (How)	Estimated Test duration and planned start date (When)
14	Test if user input can be used to control a hardware system.	Design a simple application on the MIT App Inventor (or UI) that takes in a user input and passes it on to the Python code. The user will indicate how many times a motor should rotate and change direction.	If the motor rotates and changes direction the same number of times that the user indicated through the user interface, then this result confirms that user input can be passed through Python and the Arduino to specify the action of hardware.	Test duration: 20 minutes Test date: March 6

15	Ensure that the end-effector has enough compressive force structural integrity to lift objects such as a paint brush, nozzle, and camera.	Make a 3D printed prototype of the end effector. A paint brush and a cell phone will be measured to estimate the mass required for the end-effector to support. A safety factor will be determined to multiply the mass of the heavier object between the paint brush and cell phone. An object of similar mass to the required mass will be picked up by the robot. If such a mass cannot be found, a bucket of water will be used to create a custom mass by calculating the required volume of water using the desired mass and the density of water.	If the end-effector can lift the mass for one minute without dropping it and show no signs of slippage and breakage anywhere on the robotic arm, then the result confirms that any object with a mass less than the mass tested can be lifted by the arm.	Test duration: 30 minutes Test date: March 6
16	Test whether the end-effector spring has enough compressive force to hold a paint brush and a cell phone.	Make a 3D printed prototype of the end effector and use it to grip and pick up an object without the use of the robot.	This is a pass-fail test. If the end-effector can hold the paint brush and the cell phone and they do not move due to	Test duration: 20 minutes Test date: March 6

		Someone's hand will open the end-effector and place a paint brush and a cell phone in the clamp and the spring force will hold the objects in place once the person lets go of the end-effector.	gravity, then the test is successful. If the end-effector drops an object, a spring with a higher spring constant will have to be used to provide more compressive force	
17	-Test whether the end-effector spring has enough compressive force to hold objects -Test if the robot arm can support the end-effector and the object and that the end-effector can connect to the robot arm without any problems	Use a 3D printed prototype of the end-effector to connect to the robot and pick up an object.	The test will be successful if the robot can pick up a paint brush and a cell phone, and if the robot arm shows no sign of breakage or deformation. If a failure occurs, the cause will be identified to redesign the end-effector.	Test duration: 20 minutes Test date: March 6
18	Determine whether an ultrasonic sensor can be used for safety to detect whether a person is too close to the robot while in use	In the code, a selected distance of 2 metres will be declared. An ultrasonic sensor will be placed at various distances from a wall: closer and farther than 2 metres.	If the sensor is within 2 metres from the wall, the code will output True. Otherwise, it will output False. The value in this test is that if a person is too close to the robot while in use, the robot	Test duration: 20 minutes Test date: March 10

			will stop moving as a safety measure to prevent any injuries.	
19	Find the maximum radius of the robot’s workspace.	To find the maximum radius, motors will be removed from the robot and the arm will be extended manually and measured using a measuring tape.	The value of the distance on the measuring tape will be recorded. This value will be used in the code to prevent possible errors that could occur if the user instructs the robot to place its end-effector outside its workspace.	Test duration: 20 minutes Test date: March 10
20	Test the kill switch	The Aruduino will be rotating a motor indefinitely. The kill switch button will be connected to the Arduino and if it is pressed, the Arduino should stop the motor from rotating.	If the motor stops rotating, the kill switch works. Otherwise, there is a hardware problem or a software problem that must be identified and corrected to test the switch again.	Test duration: 10 minutes Test date: March 10
21	Test if the code can instruct the end-effector to move to a precise position.	A position in 3D space will be passed into the code, which should make the motor rotate by the angles calculated using inverse kinematics to	To confirm that the end-effector moves to the correct location, the location in 3D space will be identified using a measuring tape. A physical marking will be	Test duration: 45 minutes Test date: March 13

		place the end-effector in the correct location.	placed in the location. If the end-effector moves to the location of the physical marking, the test is successful. If the robot fails the test, the error must be identified and fixed before repeating the test.	
22	Test if the robot can draw a straight line on a surface using a marker.	The test conductor will place a marker in the end effector clamp and measure the distance from the base of the robot to the surface to draw on. The conductor will then input the distance in the user interface and press start. The robot should then draw the programmed line on the surface. The specifications of the line will be coded so that the user does not have to pass a vector drawing into the user interface.	The robot should be able to successfully draw the line on the surface. The success of the test will be based on whether the line is of the distance specified in the code and if the line is drawn in the correct location. If this comprehensive test of a simple drawing is successful, more complicated comprehensive tests can be conducted, such as drawing more complicated shapes and receiving drawing instruction through the user	Test duration: 45 minutes Test date: March 13

			interface in the form of vector drawings.	
23	Test if the robot can draw a circle on a surface using a marker.	This test is identical to Test 22, but the robot will draw a circle.	The results of the test will be processed identically as in Test 22, but the radius of the circle and the position of its centre will be measured.	Test duration: 45 minutes Test date: March 13
24	Determine whether the robot can draw a drawing given a vector drawing passed into the user interface.	This test is identical to Test 22, but instead of a preprogrammed line, a vector drawing will be passed into the user interface and processed to instruct the robot to draw the given diagram.	If the software can process the vector drawing in a timely manner without any errors and the arm can correctly draw the line given by the vector drawing, then the test is successful. The same measuring techniques in Test 22 will be used to validate the line.	Test duration: 45 minutes Test date: March 17

Table 5. Materials for Each Prototype Test

Test ID	Materials Needed
14	MIT app inventor Python Motors
15	Printed end effector Cellular Phone Paint brush

16	Printed end effector Cellular Phone Paint brush
17	Printed end effector Paint brush Robotic arm
18	Arduino Ultrasonic sensor
19	Motors Arm Measuring tape
20	Arduino Motors Switch
21	Robotic arm Arduino Motors End effector
22	Marker End effector Robotic arm Arduino Motors
23	Marker End effector Robotic arm Arduino Motors
24	MIT app inventor Marker End effector Robotic arm Arduino Motors Inkscape