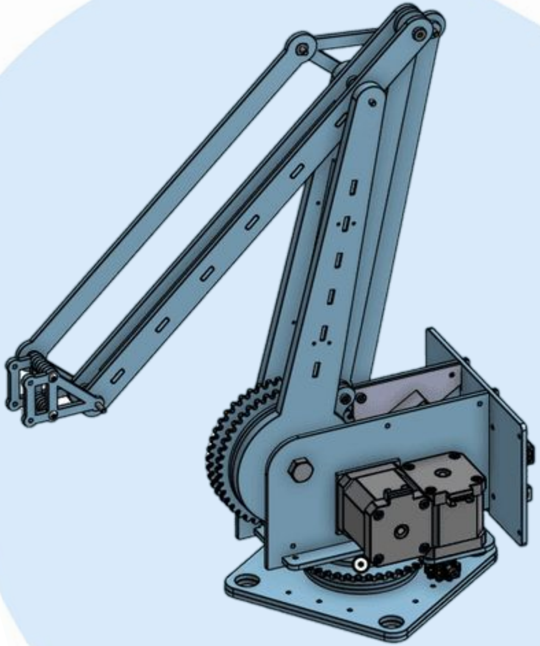
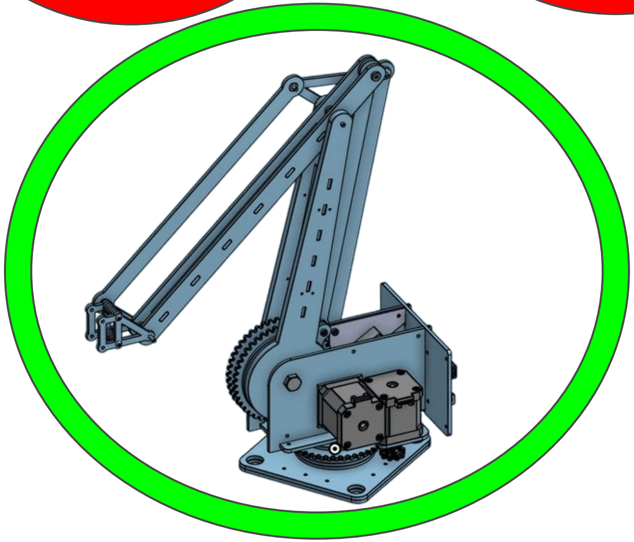


Inverse Kinematics Robot

Sarah Alkadri, Isabelle Barrette,
Jess Beardshaw, Rebeca Poulin, Benoit Tremblay

Sam





Problem Statement

John Faurbo, Theodore Eastmond and the Royal Canadian Navy need a robotic arm that has the ability to detect, remove and paint over corroded areas to ensure proper maintenance on the Halifax class warships. Cost, ease of use, proportions and safety of product are key to creating the best product for the client's needs.

Our Solution

01 Inverse Kinematics

3DOF, Open-Source, Scalable

02 Corrosion Detection Software

Coded in Python, Image filtering

03 User Interface

Coded in Python

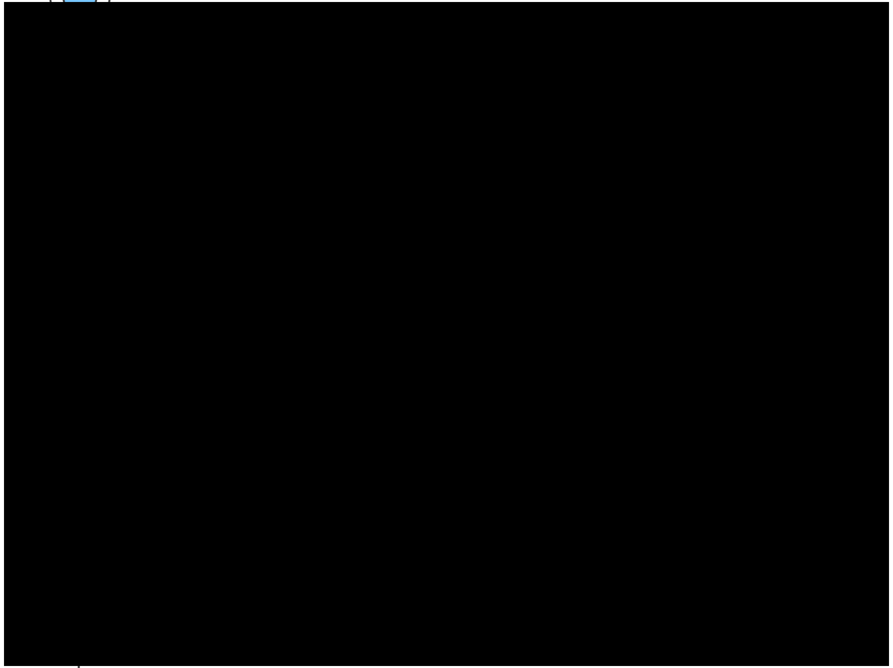
04 Safety

Emergency stop, PIR sensors

05 End Effectors

Camera scanning, Marker concept

What Is Inverse Kinematics?

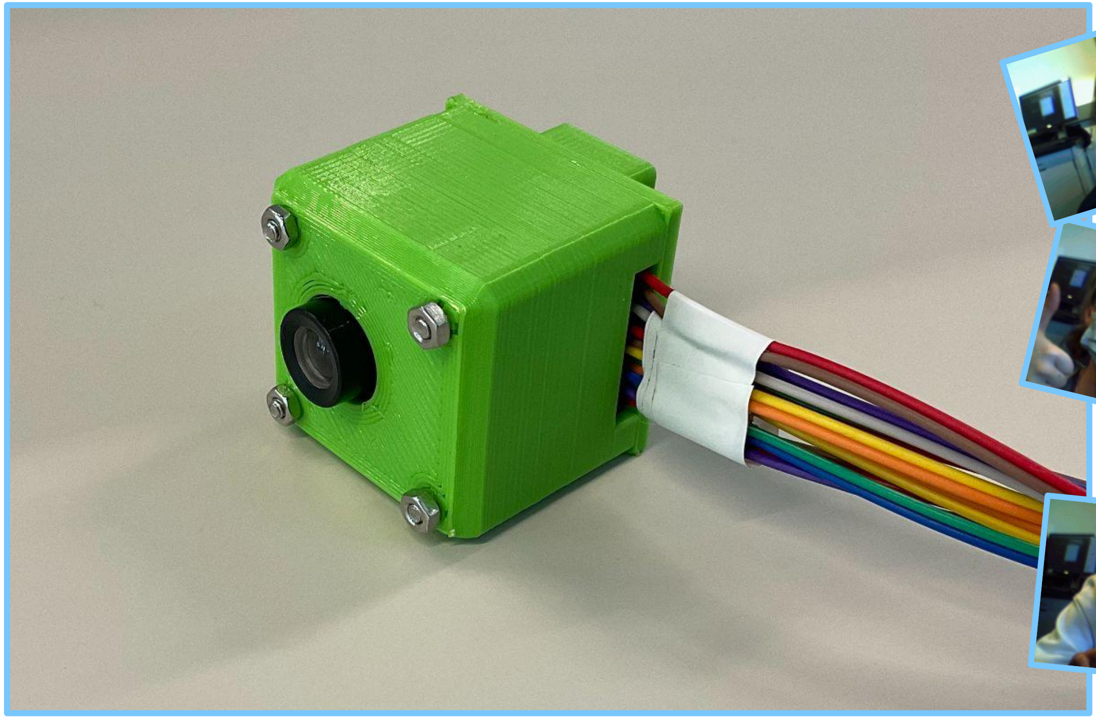


https://drive.google.com/file/d/1IbNPYFDLJrzp_Bf-3Jb8BZTqD_Bcbj2b/view

End Effectors

Setup thanks to Indrek Luuk!
https://circuitjournal.com/arduino-OV7670-to-pc

Our End Effectors



Taken with our OV7670 Camera!

1	2	3	4	5	6	7	8
16	15	14	13	12	11	10	9
17	18	19	20	21	22	23	24
32	31	30	29	28	27	26	25
33	34	35	36	37	38	39	40
48	47	46	45	44	43	42	41
49	50	51	52	53	54	55	56
64	63	62	61	60	59	58	57

Corrosion Detection Software

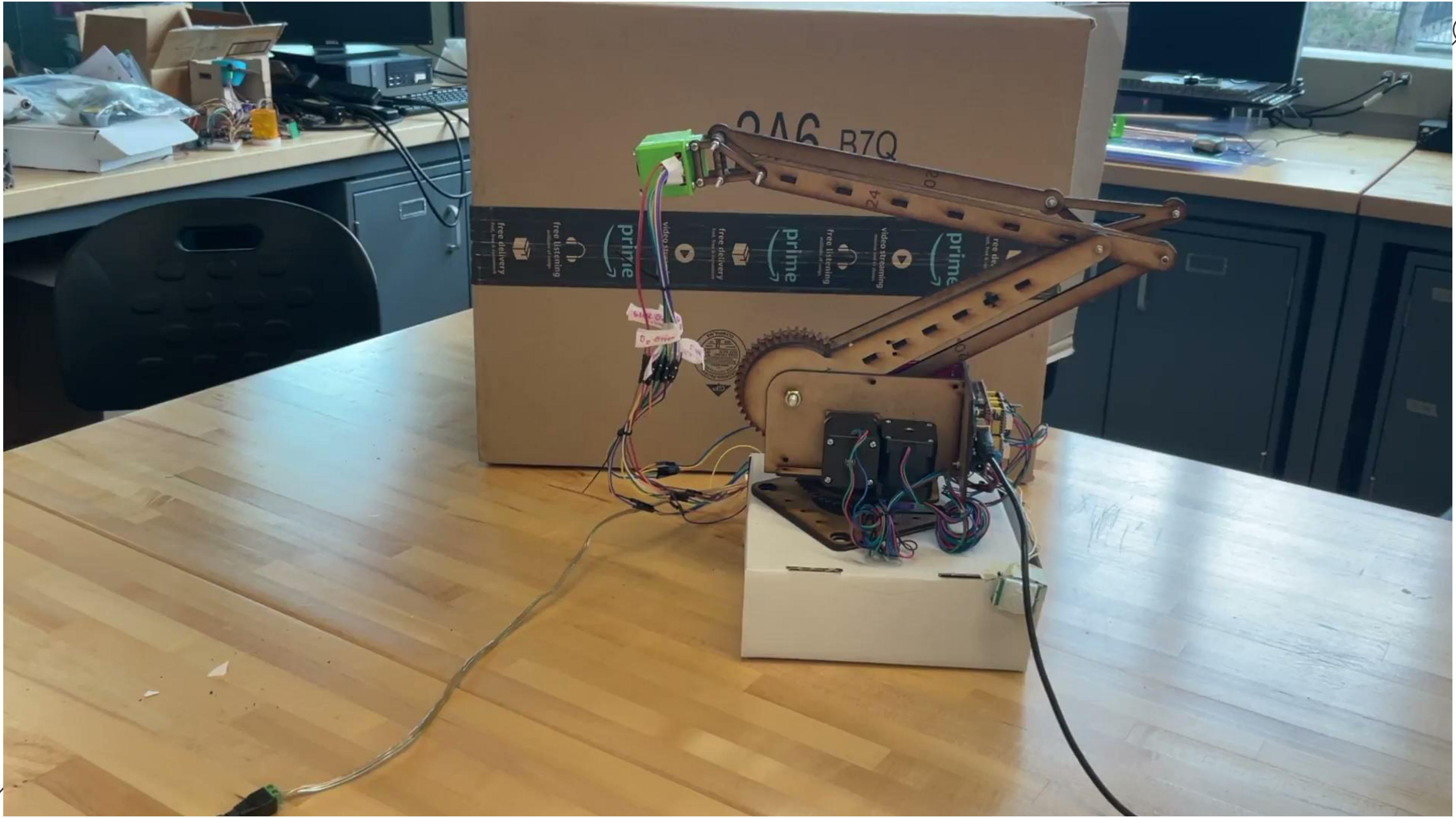
1 = Corrosion

0 = No Corrosion

Credit: Ben Zager and Ryan Dufour

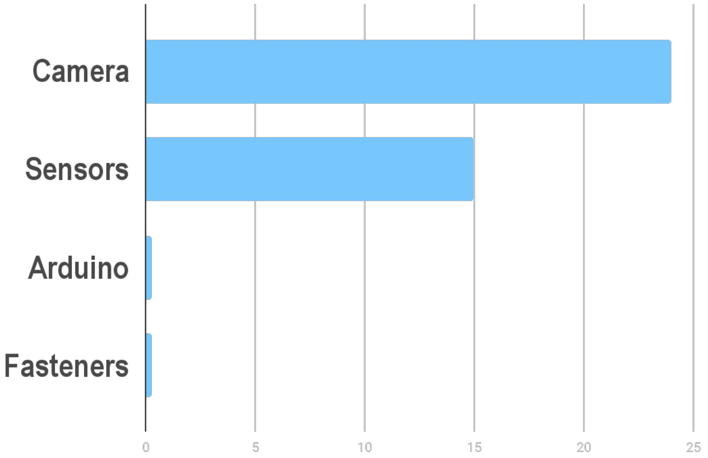

0%
Saturation
100%

Sam's Video Demo


free delivery
free listening
prime
video streaming

Budget

Cost Breakdown



$40

Amount Spent

$50

Allocated Budget

Automated

Functional Minimum Viable Product

Actualized

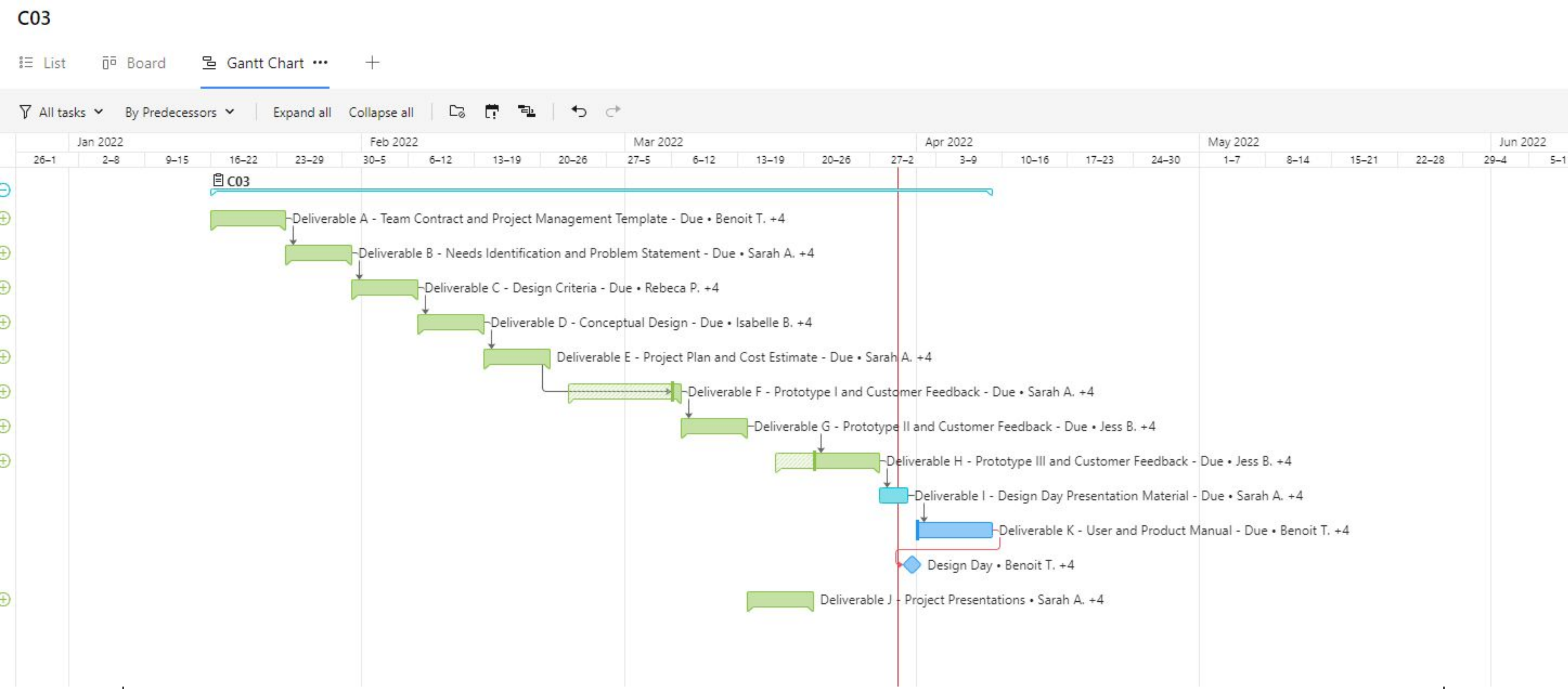
Corrosion Detection

Indicates whether images contain corrosion or not

Thank You!

Question?

Organization



Target Specifications

Robot Mechanics	
Degrees of Freedom	3
Weight supported by end effector (kg)	1.00
Weight of robot (kg)	9.07
Camera end-effector	
Back and Main piece diameter (mm)	60.00
Back depth (mm)	2.90
Main piece depth (mm)	22.00
Clips length	10.00
Weight (kg)	0.035
Camera	
Weight (kg)	0.016
Camera Resolution	VGA 640 x 480 at 30 fps
Working Power	60mW/15fps
Pixel coverage	3.6um x 3.6um
Painter/corrosion remover end-effector	
Main piece dimensions (mm)	(49.00 x 37.40 x 35) + 2(9.0 x 4.0 x 4.0)
Adjustable piece dimensions (mm)	(44.198 m x 22.0 x 34.5) + 2(32.5 x 4.8 x 4.8)
Weight (kg)	N/A

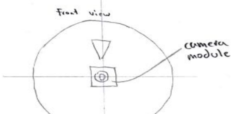
List of Prioritized Criteria

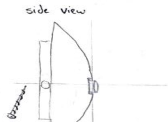
	Functional Requirements	Relation	Value	Units	Verification Method
1	The end effectors can hold and operate camera, paint and anti-corrosion sprayer.	=	Yes	N/A	Test
2	The arms supported weight.(i.e.the end effectors weight)	>=	750	g	Final test, weighing, analysis
3	The end effectors are easily interchangeable.	=	Yes	N/A	Test
9	The arm and end effectors remain stable and withstand pressure from hose and paint.	>=	180	psi	Test
4	The robot is easy to learn to operate.	<	45	minutes	Test
5	The end effector and parts are 3D printable	=	Yes	N/A	Analysis
6	The arm is powered by 120-volt outlets	=	Yes	N/A	Test, final check
8	The arm and end effector can be assembled quickly.	<	25	minutes	Test
10	The code uses a common programming language such as C or C++ or Python.	=	Yes	N/A	Test
11	The code uses inverse kinematics.	=	Yes	N/A	Test
12	The design of end effectors and code are open source.	=	Yes	N/A	Test
13	The arm is controlled using an Arduino Uno.	=	Yes	N/A	Test

	Constraints	Relation	Value	Units	Verification Method
1	The weight of the end effectors.	<=	750	g	Analysis
2	The dimensions of the end effectors.	<	60	mm	Analysis
3	The cost of the project.	<	50	$	Estimate, final check
4	The weight of the arm.	=	20	lbs	Analysis
5	The dimensions of the arm.	<	1	m^2	Analysis
6	The lighting of the arms surroundings.	>	Yes	N/A	Analysis
7	The area of sight/vision/range to spray and observe with camera.	=	1	m^2	Analysis

	Non-Functional Requirements	Relation	Value	Units	Verification Method
1	The robot is safe to operate and be around while working.	=	Yes	N/A	Test
2	The robot is operated by someone with limited technical experience. (High School Education)	=	Yes	N/A	Ask non-expert
3	The parts, robot and code are easily repairable.	=	Yes	N/A	Can be 3D printed
4	The robot is compact and transportable.	=	Yes	N/A	Analysis of dimensions
5	The robot's lifespan before minor repairs needed.	>=	2-3	months	Estimate, analysis
6	The robot's lifespan before major repairs needed.	>=	6	months	Estimate, analysis
7	The robot is aesthetically pleasing.	=	Yes	N/A	Client Meeting

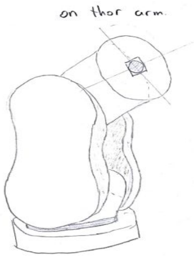
Camera end effector





* Since thor arm can rotate at endpoint arrow assures proper camera angle

* Hole in endpiece for crews/pins to hold endpiece in place

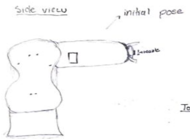


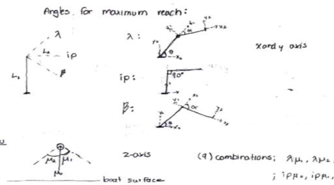
Pros: - Curved surface allows for no accidental obstruction of camera lens.
- Small in size so it will be relatively light.
- Circle can be made more bulbous for size

Cons: - being able to fit wiring & camera module inside semi circle might be difficult.
- shipping for arduino camera modules are somewhat long

Camera/sensor poses:

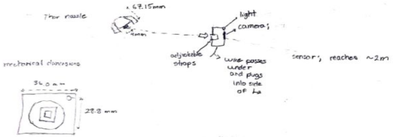
$L_1 = 160.00$ mm $L_2 = 195.00$ mm



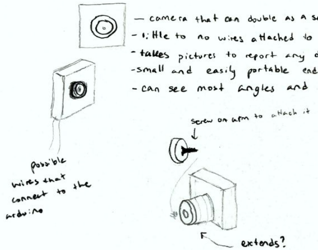


(9) combinations; $\lambda\mu_1, \lambda\mu_2, \lambda\mu_0$; $ip\mu_0, ip\mu_1, ip\lambda_2$; $\beta\mu_0, \beta\mu_1, \beta\mu_2$

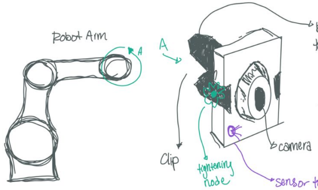
End-effector:



- camera that can double as a sensor
- little to no wires attached to facilitate the changing of end effectors
- takes pictures to report any damages it finds or "senses"
- small and easily portable end effector
- can see most angles and can ?extend? into tight places?

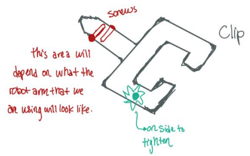


- adapt inverse kinematics equation to extended length?

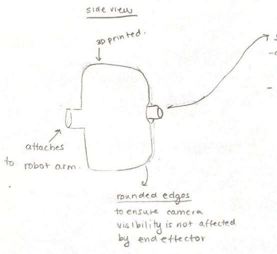


- back of clip contains screw holes to attach to the robot
- Material is 3D printable to follow wer need to cheap to acessible on the ship
- The clip is universal with all end effectors, making it easy to switch end effectors
- sensor to stop if someone walks infront of process
- Sends signal to stop analyzing, maybe 10 second stop?

Cons: not sure how strong the 3D printed tightening node is



END EFFECTOR: CAMERA ATTACHMENT

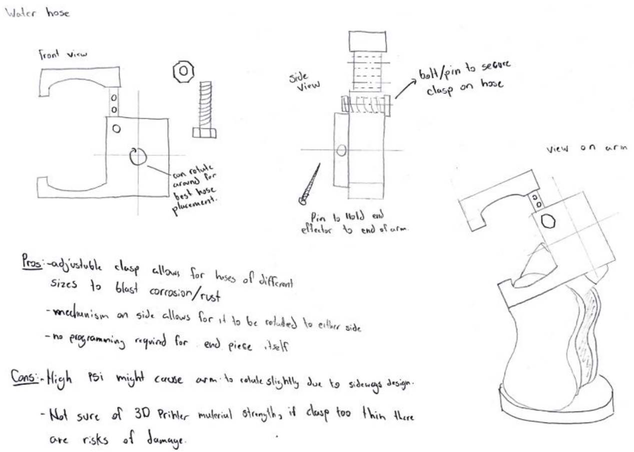


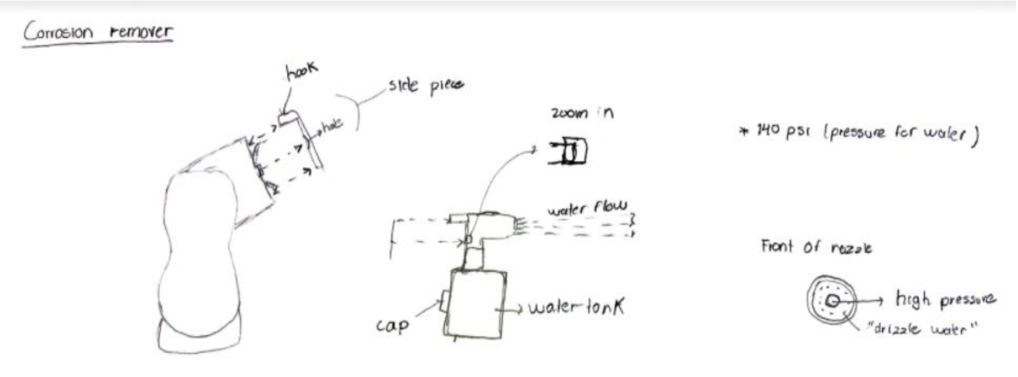
camera
- allows to take image of area through user interface for analysis
- camera used similar to borescope/endoscope (to illuminate area as well)

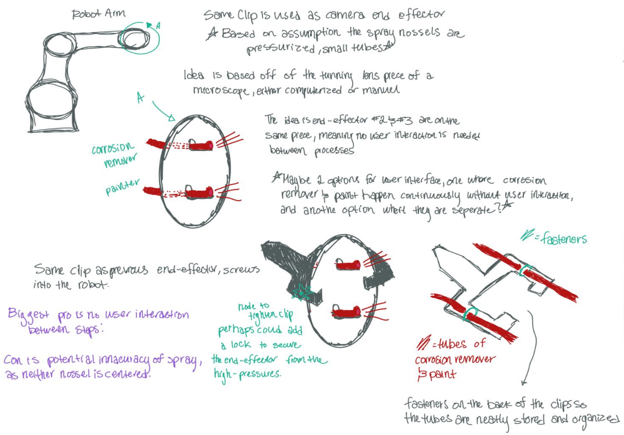
Questions
- should a camera be attached to each end effector?
- maybe attach camera to arm?

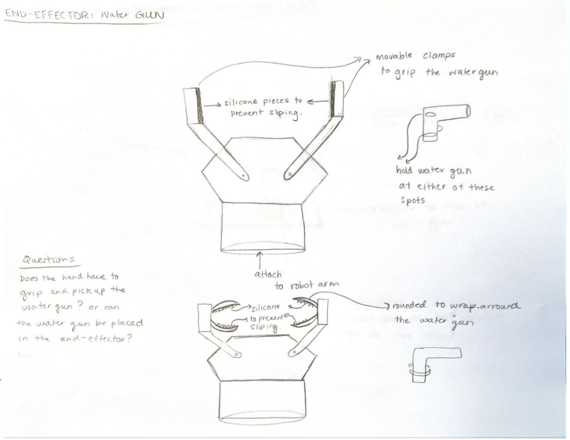
Pros
- small in size to allow access to tight areas
- rounded to prevent for an obstructive view

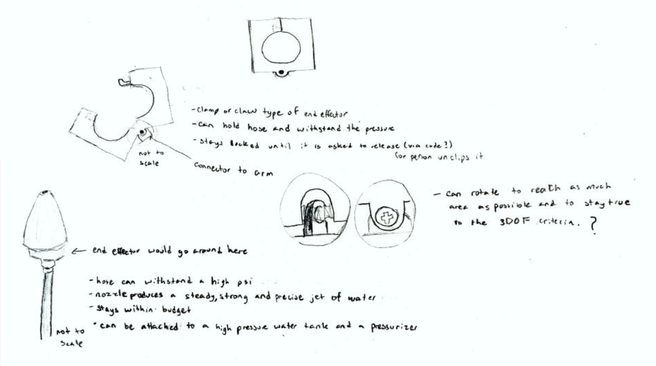
Cons
- camera wires might pose difficulty when detaching and attaching

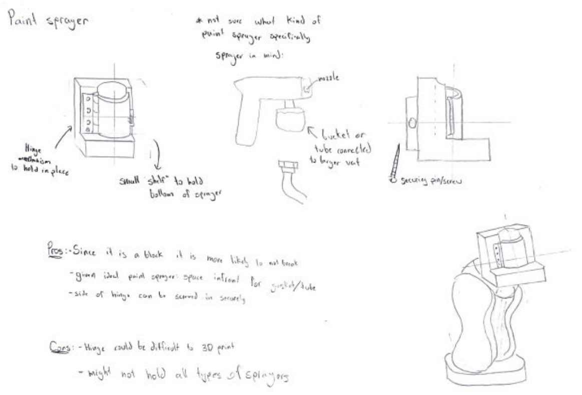

Water hose
Front view
can rotate around for best hose placement.
Side view
bolt/pin to secure clasp on hose
Pin to hold end effector to end of arm.
View on arm
Pros: - adjustable clasp allows for hoses of different sizes to blast corrosion/rust
- mechanism on side allows for it to be rotated to either side
- no programming required for end piece itself
Cons: - High PSI might cause arm to rotate slightly due to sideways design.
- Not sure of 3D Printer material strength, if clasp too thin there are risks of damage.

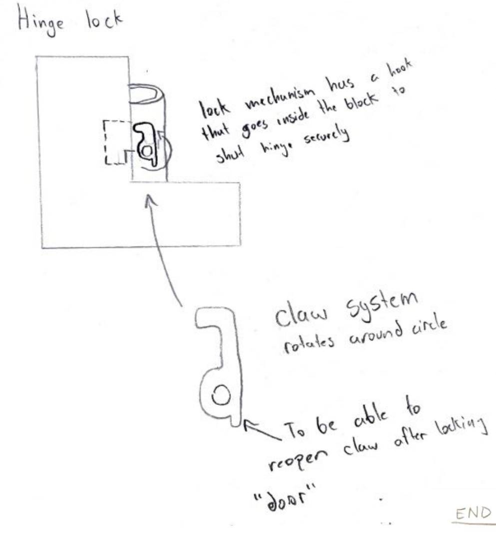

Corrosion remover
hook
side piece
hole
zoom in
water flow
cap
water tank
* 140 psi (pressure for water)
Front of nozzle
high pressure
"drizzle water"

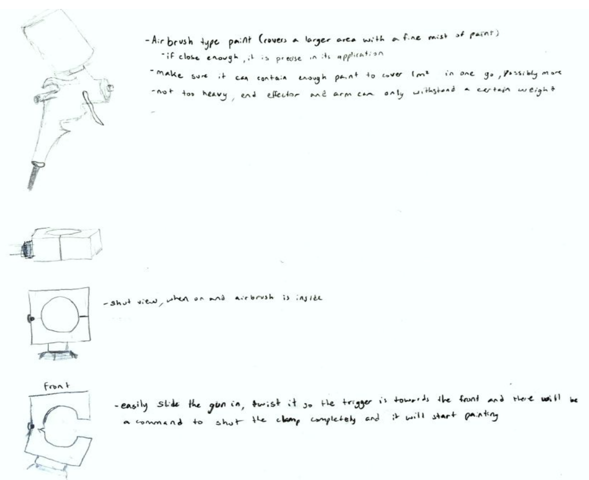

Robot Arm
Same Clip is used as camera end effector
Based on assumption the spray nossels are pressurized, small tubes
Idea is based off of the turning lens piece of a microscope, either computerized or manual
The idea is end-effector #2 & #3 are on the same piece, meaning no user interaction is needed between processes
Maybe 2 options for user interface, one where corrosion remover & paint happen continuously without user interaction, and another option where they are seperate?
corrosion remover
painter
Same clip as previous end-effector, screws into the robot.
Biggest pro is no user interaction between steps.
Con is potential innacuracy of spray, as neither nossel is centered.
node to tighten clip
perhaps could add a lock to secure the end-effector from the high-pressures.
= fasteners
= tubes of corrosion remover & paint
fasteners on the back of the clips so the tubes are neatly stored and organized

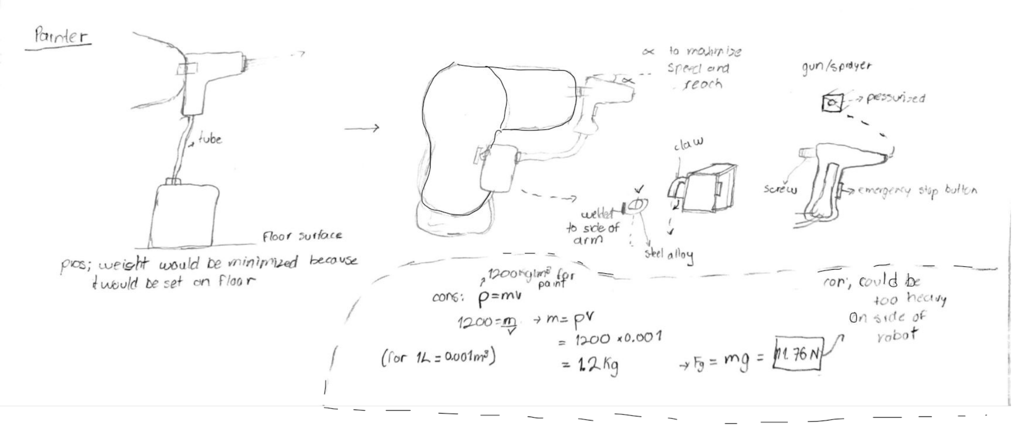

END-EFFECTOR: Water GUN
movable clamps to grip the watergun
silicone pieces to prevent sliping.
hold water gun at either of these spots
attach to robot arm
silicone to prevent sliping
rounded to wrap around the water gun
Questions
Does the hand have to grip and pick up the water gun? or can the water gun be placed in the end-effector?

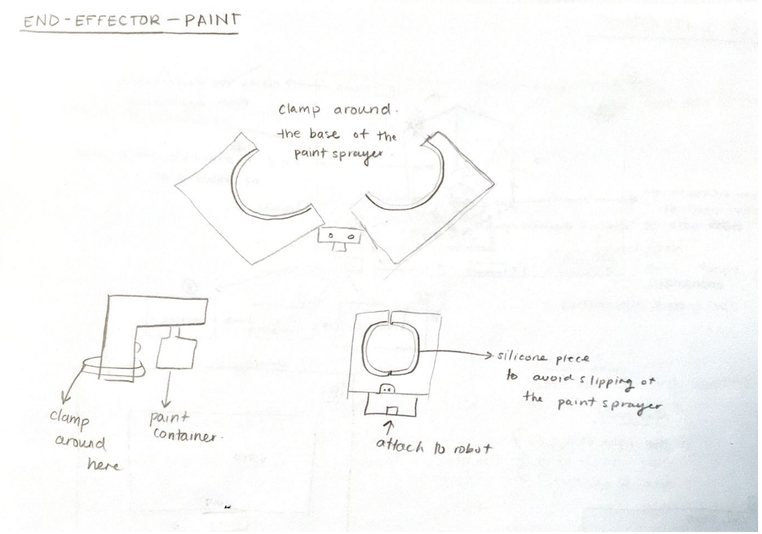

- clamp or claw type of end effector
- can hold hose and withstand the pressure
- stays locked until it is asked to release (via code?) (or person unclips it
not to scale
connector to arm
- can rotate to reach as much area as possible and to stay true to the 3DOF criteria. ?
← end effector would go around here
- hose can withstand a high psi
- nozzle produces a steady, strong and precise jet of water
- stays within budget
- can be attached to a high pressure water tank and a pressurizer
not to scale

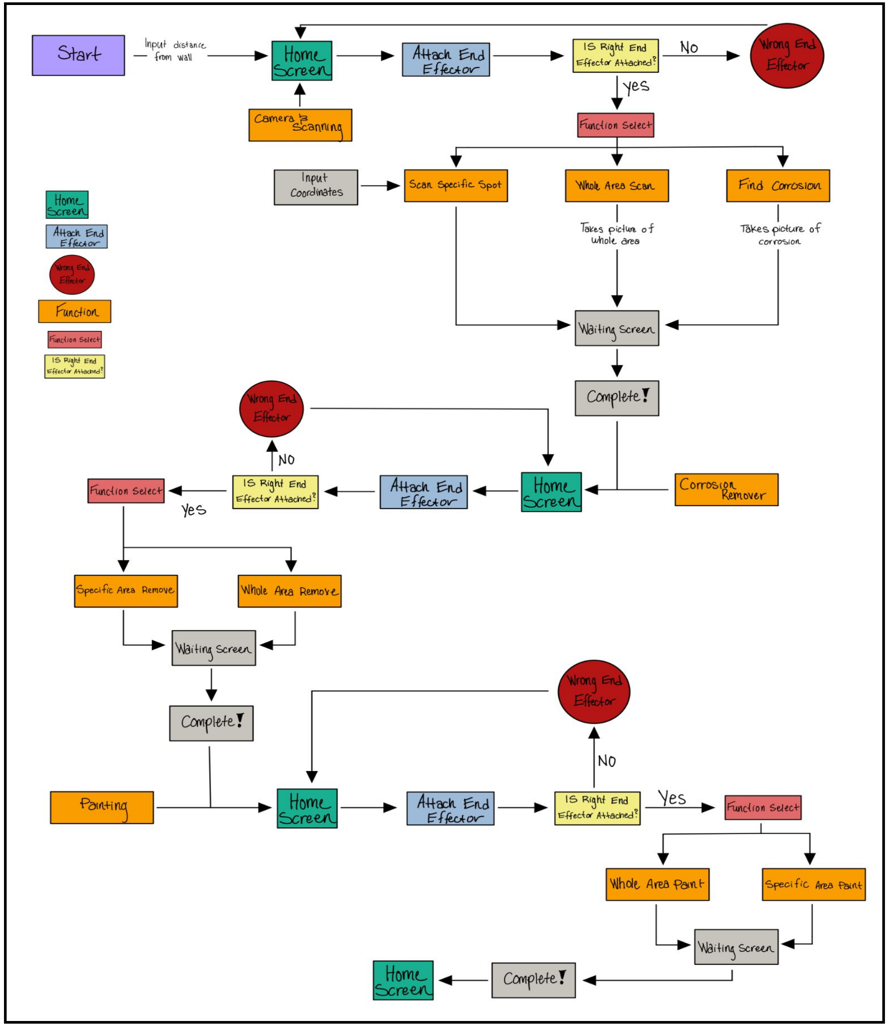

Paint sprayer
* not sure what kind of paint sprayer specifically
sprayer in mind:
nozzle
bucket or tube connected to larger vat
Hinge mechanism to hold in place
small "shelf" to hold bottom of sprayer
securing pin/screw
Pros: - Since it is a block it is more likely to not break
- given ideal paint sprayer: space infront for gasket/tube
- side of hinge can be screwed in securely
Cons: - Hinge could be difficult to 3D print
- might not hold all types of sprayers

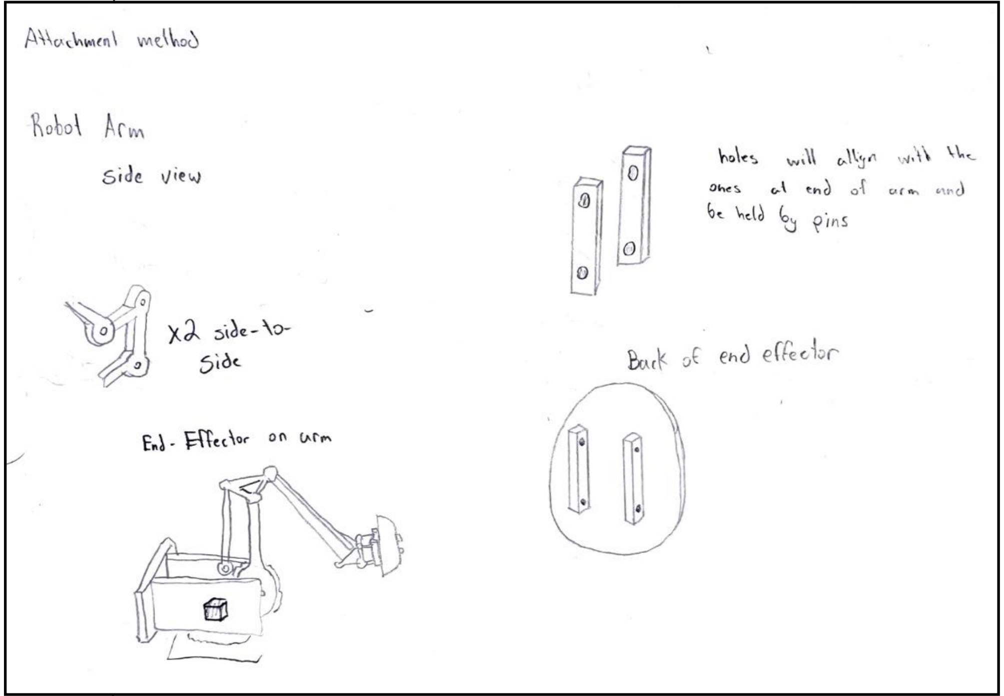

Hinge lock
lock mechanism has a hook that goes inside the block to shut hinge securely
claw system rotates around circle
To be able to reopen claw after locking "door"

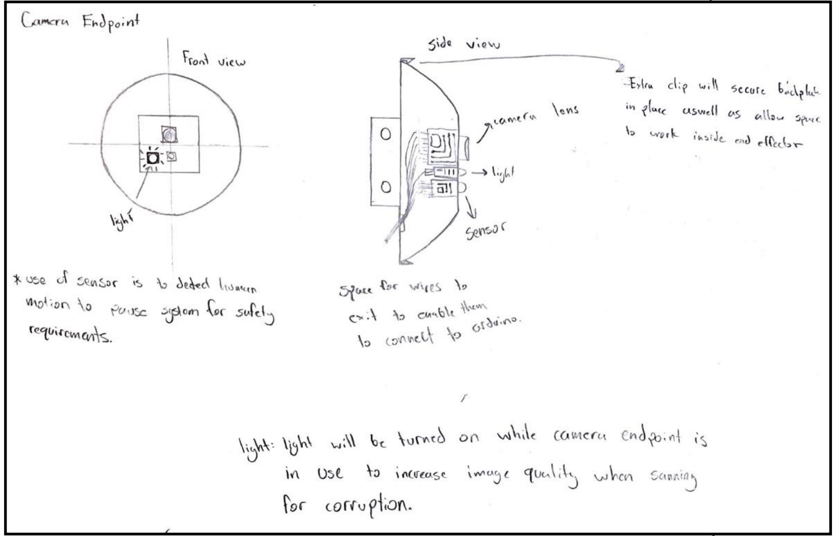

- Airbrush type paint (covers a larger area with a fine mist of paint)
- if close enough, it is precise in its application
- make sure it can contain enough paint to cover 1m² in one go, possibly more
- not too heavy, end effector and arm can only withstand a certain weight
- shut view, when on and airbrush is inside
Front
- easily slide the gun in, twist it so the trigger is towards the front and there will be a command to shut the clamp completely and it will start painting

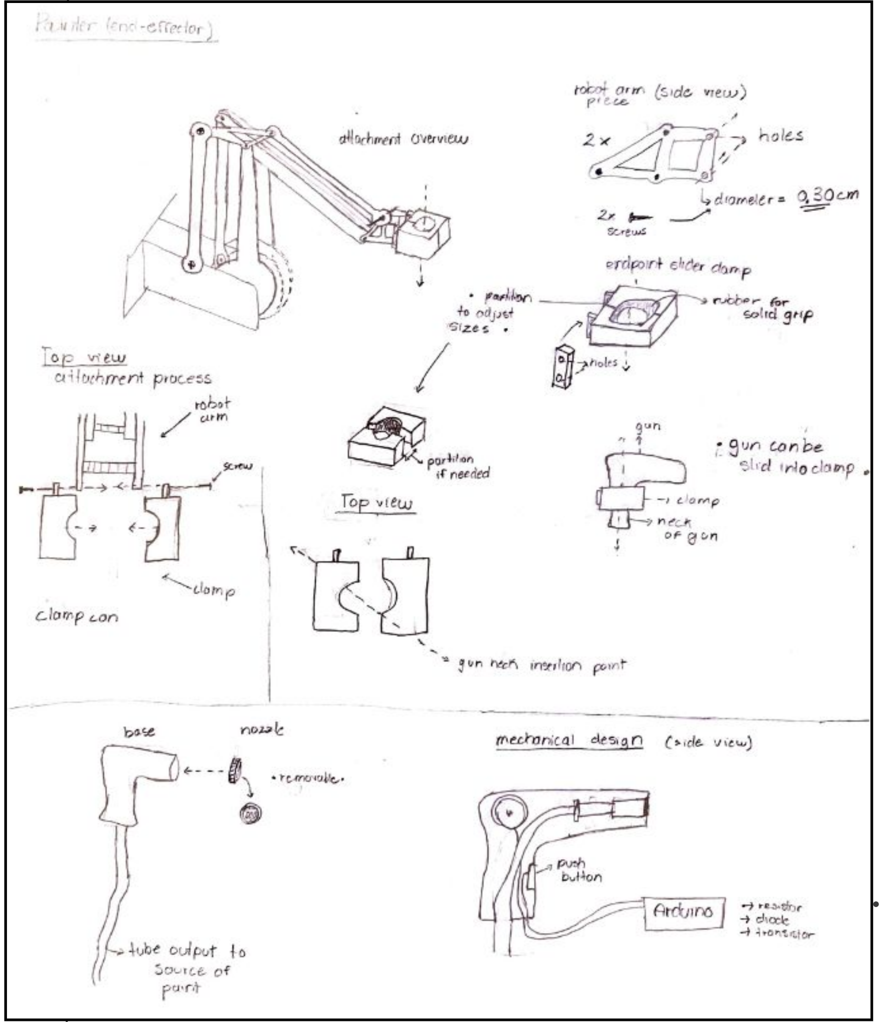

Painter
tube
Floor surface
pros; weight would be minimized because it would be set on floor
∝ to maximize speed and reach
gun/sprayer
pessurized
claw
screw
emergency stop button
welded to side of arm
steel alloy
cons: ρ=m/v 1200kg/m³ for paint
1200=m/v → m=ρv
= 1200×0.001
= 1.2Kg
(for 1L=0.001m³)
→ Fg = mg = 11.76N
cor; could be too heavy on side of robot

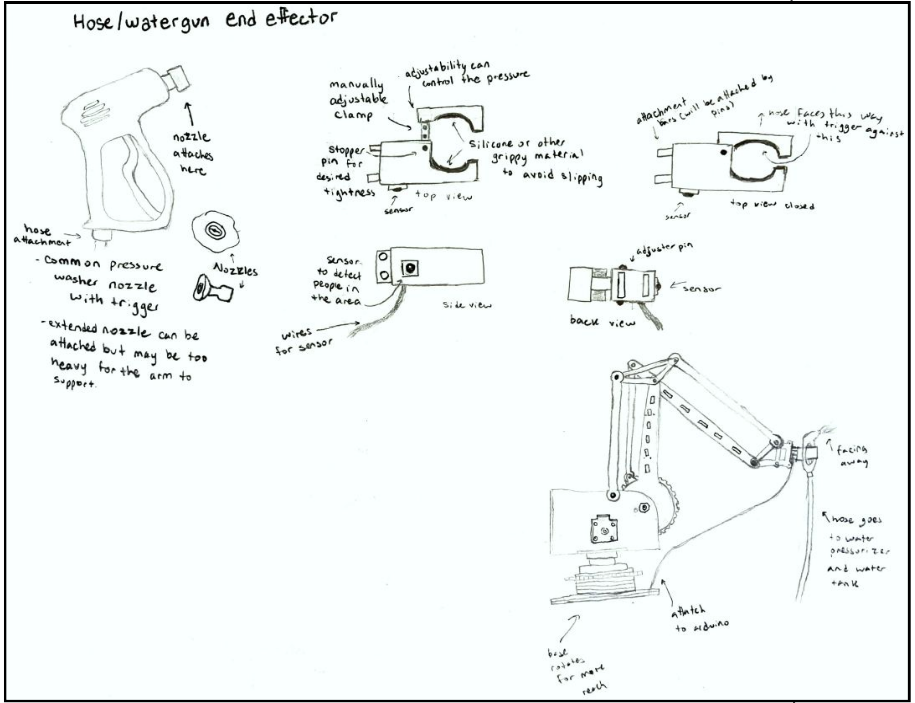

END-EFFECTOR – PAINT
clamp around the base of the paint sprayer
clamp around here
paint container
silicone piece to avoid slipping of the paint sprayer
attach to robot

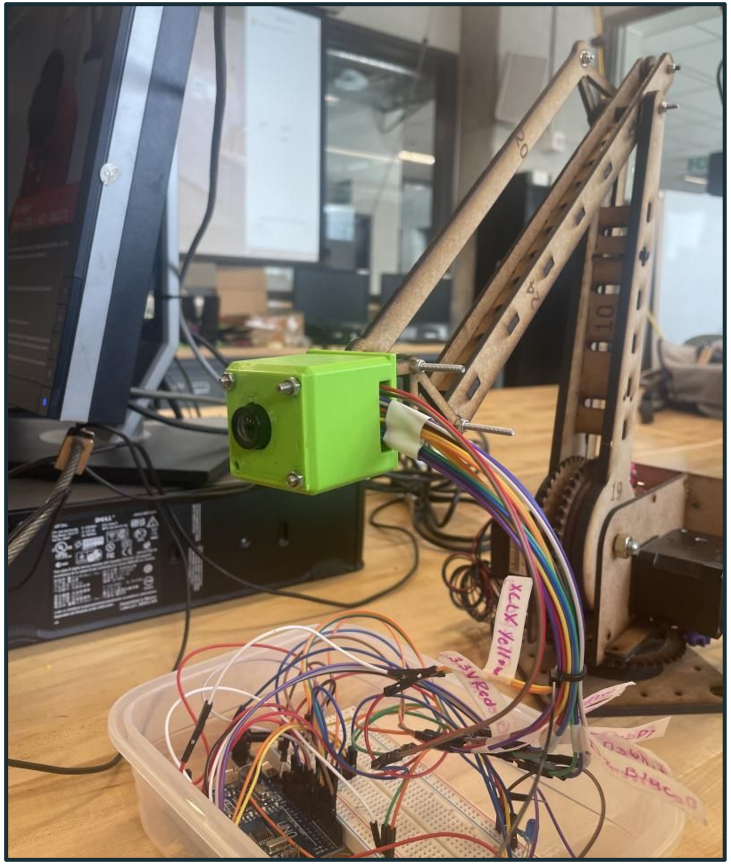

Start
Input distance from wall
Home Screen
Attach End Effector
IS Right End Effector Attached?
No
Wrong End Effector
Yes
Camera is Scanning
Function Select
Input Coordinates
Scan Specific Spot
Whole Area Scan
Find Corrosion
Takes picture of whole area
Takes picture of corrosion
Waiting Screen
Complete!
Home Screen
Attach End Effector
Wrong End Effector
Function
Function Select
IS Right End Effector Attached?
Corrosion Remover
Specific Area Remove
Whole Area Remove
Painting
Whole Area Paint
Specific Area Paint

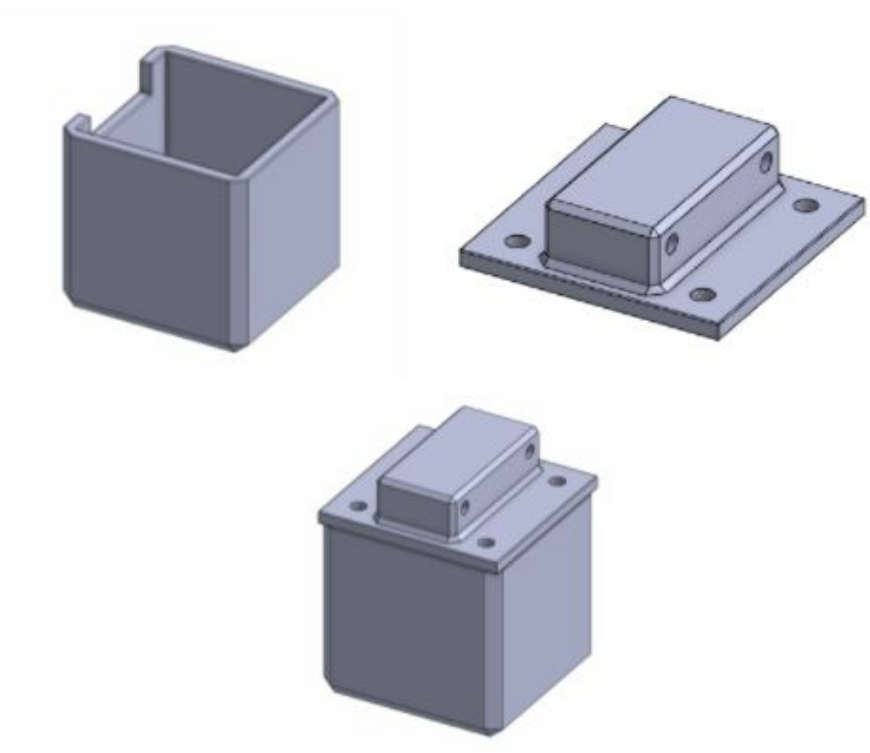

Attachment method
Robot Arm
side view
x2 side-to-side
End-Effector on arm
holes will allyn with the ones at end of arm and be held by pins
Back of end effector


Camera Endpoint
Front view
light
side view
camera lens
light
sensor
-Extra clip will secure backplate in place aswell as allow space to work inside end effector
* use of sensor is to detect human motion to pause system for safety requirements.
Space for wires to exit to enable them to connect to arduino.
light: light will be turned on while camera endpoint is in use to increase image quality when scanning for corruption.

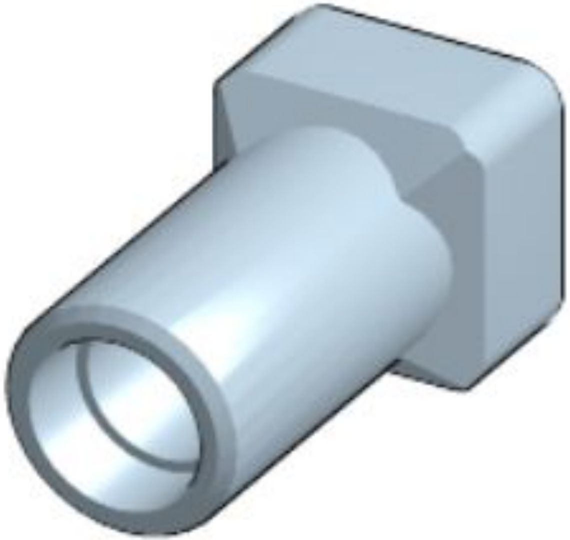

Painter (end-effector)
robot arm (side view) piece
attachment overview
2x
holes
diameter = 0,30cm
2x screws
endpoint slider clamp
partition to adjust sizes
rubber for solid grip
holes
Top view attachment process
robot arm
screw
clamp
clamp can
partition if needed
Top view
gun
gun can be slid into clamp
clamp
neck of gun
gun neck insertion point
base
nozzle
removable
tube output to source of paint
mechanical design (side view)
push button
Arduino
resistor
diode
transistor


Hose/watergun end effector
nozzle attaches here
hose attachment
- Common pressure washer nozzle with trigger
- extended nozzle can be attached but may be too heavy for the arm to support.
Nozzles
manually adjustable clamp
adjustability can control the pressure
Stopper pin for desired tightness
Silicone or other grippy material to avoid slipping
sensor
top view
attachment bars (will be attached by pins)
hose faces this way with trigger against this
top view closed
Sensor to detect people in the area
side view
wires for sensor
adjuster pin
sensor
back view
facing away
hose goes to water pressurizer and water tank
attach to arduino
base rotates for more reach






FINE

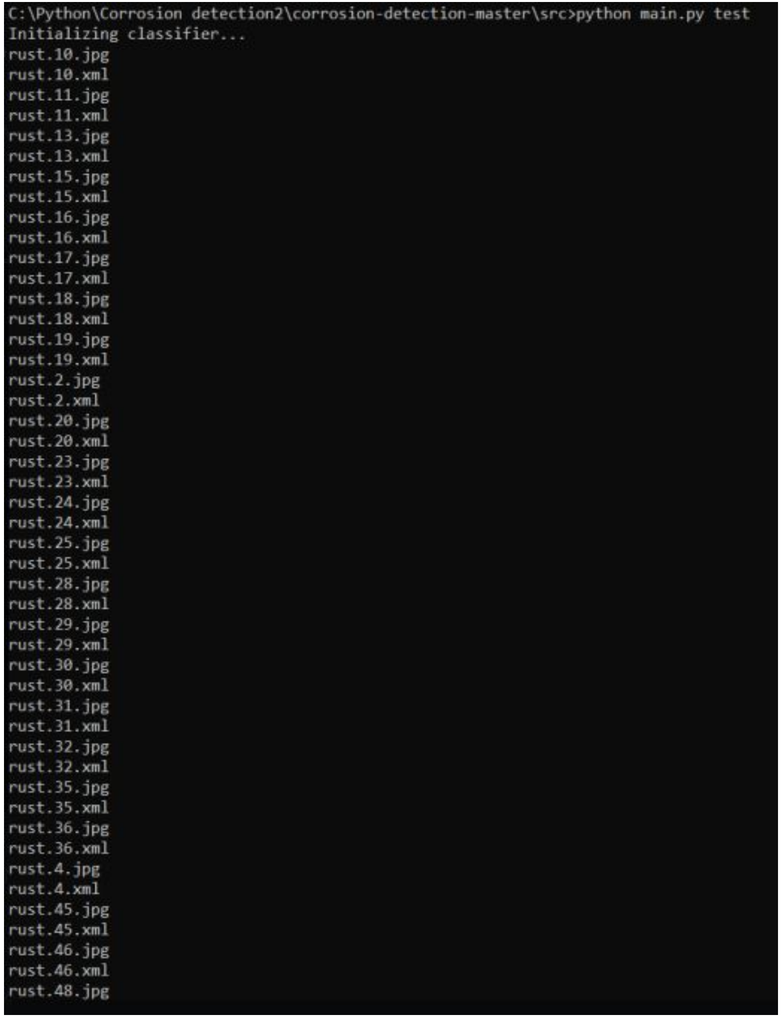


Item Name and Link	Quantity	Cost ($)	Justification
Camera OV7670 VGA CMOS Camera	1	23.68	The camera chosen needs to be compatible with Arduino software in order to access the data (live video feed) and send it to other devices.
PIR motion sensors PIR motion sensors	1	14.99	These sensors will be added to different end-effectors to ensure safety while operation. (soldered)
3D printing materials		0.00	Since most of our end effector components will be 3D printed, we will be using the machines and materials provided in the Maker Lab.
Arduino kit and wires	1	0.00 (Free at Maker Lab)	The Arduino will be useful for the spray guns in order to connect the sensors and triggers to a specific output in our software. This kit includes a breadboard and some resistors in order
Bolts and Nuts	14	0.00(Free at MakerLab)	Used to attach end effector parts together and attach the end effectors to the robot arm.
Sharpie	1	0.00	The Sharpie is used for the paint/corrosion remover end effector to demonstrate its functionality to the client on design day.
Total product cost (w/o taxes or shipping)		38.67	
Total product cost (including taxes and shipping)		41.16	

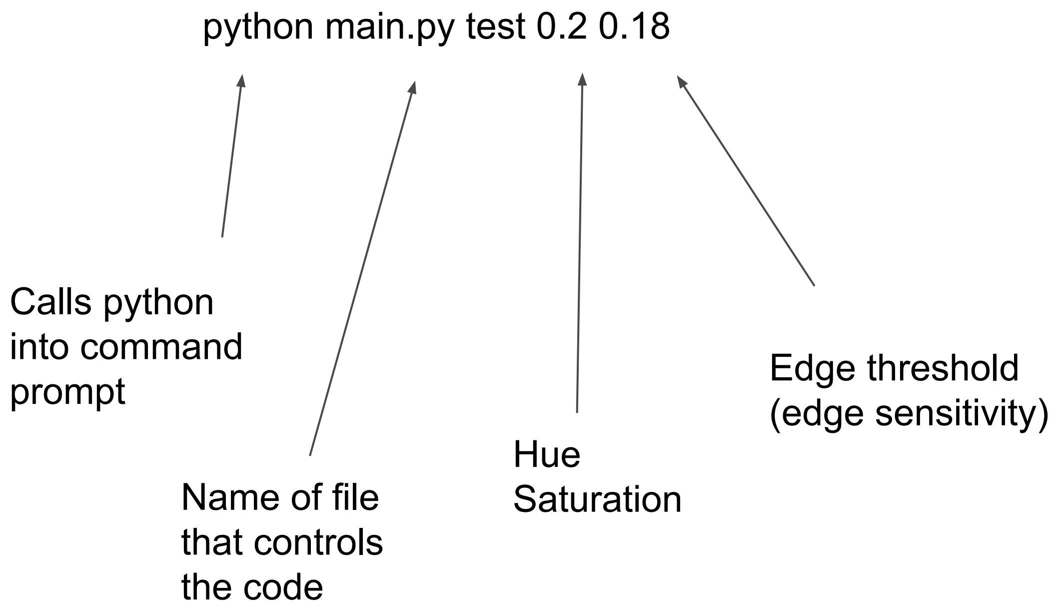
Item Name	Description	Type	Prototype #	Source
CAD software (Onshape)	This will be used to create a computer-aided design of different end effectors.	Analytical (Software)	1	https://www.onshape.com/en/
Arduino Studio (Tinkercad)	To test circuits.	Temporary software	2	https://www.tinkercad.com
3D printer	To 3D print all end effectors and attachment pieces.	Equipment.	3	MarkerSpace
Coding Software(CLion, CodeBlocks)	To implement code.	Software	4	Personal device

Test #	Objective	Description and Test Method	Expected Result/Stopping Criteria	Test Duration and Date
1	Mathematical code concept	To have a logical and functional mathematical approach of the functionality and movement of the arm.	Applicable to our code concept further on.	1 or 2 days Reading week
2	Analysis of materials	Lots of materials are used in this design, such as different 3D printable materials, cameras, sensors and Arduino components such as the wires and diodes. These will have to be tested for their effectiveness and researched extensively,	Be approved by the TA/PM and purchase materials ASAP.	2 or 3 days Reading week
3	Engineering drawing of end-effectors	Detailed engineering drawing on paper of our design and the orthographic projections to show all sides and important components	Functional drawing with all technical components provided.	2 days Reading week
4	Basic code for arm movement	Once the mathematical concept is achieved and the inverse kinematics equation is understood, the equations can be translated to code for future testing	Test on the robot model with school Arduino, capable of performing defined tasks	2 or 3 days While the drawings are being made
5	3D modelling on Onshape or Solidworks	3D drawing or model on a 3D modelling site to determine our "final" design with more precision and to better our understanding of our design and ensure our understanding of it.	Is complete and able to be 3D printed successfully	2 days As soon as engineering drawing is done
6	Camera and corrosion detection code	If all goes well, the corrosion code we have found may be accessible to us and may be able to be translated, and that translation to a language that we understand would be this step.	Granted permission of the detection code, successfully translated from Python to C. Functional with robot testing.	4 days While drawings and models are being made
7	Create user interface and test with what we have	Attempt different user inputs and see how these are processed and outputted compared to the expected outcome.	The code successfully directs the user to each desired input screen	1 day Before the first session with robot
8	Test materials with what we currently have	Materials have been analyzed, and the best ones are chosen and must be put to the test to see if they are good for our product. They will be tested in durability and compatibility with the arm and the code.	Test Arduino parts with arm and code. The camera can fit in our end effector piece.	2 days First session with robot arm
9	Test arm movement code on arm	The algorithm and code for the inverse kinematics movement of the arm should be completed, and it will be tested on the arm as soon as the opportunity presents itself so that any issues are discovered and it can be modified accordingly quickly	The robot successfully performs the inputted function	1 day First session with robot arm
10	Paper or cardboard quick prototype	Quickly make a 2D and/or 3D tangible model of end-effectors as a size comparison to the actual robot and objects that will be used with them to be sure of our dimensions	Production is successful	> 1 day First session with robot arm
11	Retouch engineering drawing and 3D modeling of end-effectors	Any miscalculations or wrong dimensions are discovered through the previous tests and now the drawings and models can be readjusted to accommodate our new discoveries	Successfully implement changes for second improved prototype designs.	1 day After first session with robot
12	Second Paper or cardboard prototype	Another comparison with a quick and easy prototype and the arm with the new calculations and retouched dimensions to see if it is correct, if not repeat steps 5 and 6 until the prototype works	Successfully implement changes for second improved prototype designs.	1 to 5 days For next session with robot arm
13	3D printed model of what we have designed so far	The 3D model is adjusted and can now have the pieces printed and assembled for testing on the robot. If the previous analysis and prototyping were effective, this should be done once or twice to minimize the number of materials used and the overall cost	Successful printing process according to the measurements of the designs.	1 or 2 days Second session with robot arm

14	Test code and user interface with newly 3D printed pieces and arm	Pieces are printed and the end-effectors are assembled, everything can be wired and plugged into the Arduino in its respective place, and the code can be tested on the arm and the user interface. If any errors occur, the code and user interface will have to be modified accordingly	Consistent with prototype testing. Five consecutive test trials with no errors.	1 or 2 days Once robot and 3D printer is accessible
15	Make sure attachments and necessary scenarios are compatible with end-effectors and code	The final test will entail putting all pieces together for one last test, running multiple scenarios with the user interface, arm and all the end effectors to simulate the users' experience and ensure that it is possible, simple and easy to understand for the high school students who will most likely be running the interface and interchanging the end effectors	Consistent prototype testing. Five consecutive test trials with no errors.	3 days Second last step, leave time to fix mistakes and get feedback
16	Adjust all necessary things and create the final versions of end-effectors, code and user interface	Once the group and client have settled on a final version and has been through the tests previously mentioned, it is time to bring it to life and create the final version of everything necessary, test it on the robot arm and if all goes well, there will no longer be any need for prototyping	Either run out of time or be satisfied with the final product before design day	1 to 3 days Last step, must be before design day



```
C:\Python\Corrosion detection2\corrosion-detection-master\src>python main.py test
Initializing classifier...
rust.10.jpg
rust.10.xml
rust.11.jpg
rust.11.xml
rust.13.jpg
rust.13.xml
rust.15.jpg
rust.15.xml
rust.16.jpg
rust.16.xml
rust.17.jpg
rust.17.xml
rust.18.jpg
rust.18.xml
rust.19.jpg
rust.19.xml
rust.2.jpg
rust.2.xml
rust.20.jpg
rust.20.xml
rust.23.jpg
rust.23.xml
rust.24.jpg
rust.24.xml
rust.25.jpg
rust.25.xml
rust.28.jpg
rust.28.xml
rust.29.jpg
rust.29.xml
rust.30.jpg
rust.30.xml
rust.31.jpg
rust.31.xml
rust.32.jpg
rust.32.xml
rust.35.jpg
rust.35.xml
rust.36.jpg
rust.36.xml
rust.4.jpg
rust.4.xml
rust.45.jpg
rust.45.xml
rust.46.jpg
rust.46.xml
rust.48.jpg
```



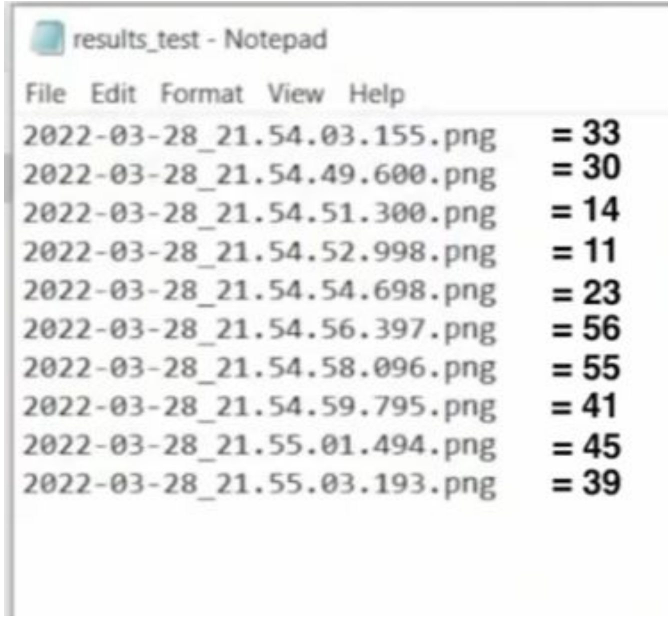
Implemented Coordinate System

```
C:\Users\jessb\Documents\Python\Rust>rust_detectiion.py
82
1.28125
2022-03-17_11.11.18.090.png = 1
2022-03-17_11.11.19.797.png = 2
2022-03-17_11.11.21.497.png = 3
2022-03-17_11.11.23.196.png = 4
2022-03-17_11.11.24.881.png = 5
2022-03-17_11.11.28.293.png = 6
2022-03-17_11.11.29.995.png = 7
2022-03-17_11.11.31.693.png = 8
2022-03-17_11.11.33.391.png = 9
2022-03-17_11.11.36.788.png = 10
2022-03-17_11.11.38.487.png = 11
2022-03-17_11.11.40.187.png = 12
2022-03-17_11.11.41.886.png = 13
2022-03-17_11.11.43.585.png = 14
2022-03-17_11.11.45.284.png = 15
2022-03-17_11.11.46.982.png = 16
2022-03-17_11.11.48.676.png = 17
2022-03-17_11.11.50.382.png = 18
2022-03-17_11.11.52.074.png = 19
2022-03-17_11.11.53.779.png = 20
2022-03-17_11.11.55.478.png = 21
2022-03-17_11.11.57.177.png = 22
2022-03-17_11.11.58.876.png = 23
2022-03-17_11.12.00.577.png = 24
2022-03-17_11.12.02.275.png = 25
2022-03-17_11.12.03.976.png = 26
2022-03-17_11.12.05.673.png = 27
2022-03-17_11.12.07.372.png = 28
2022-03-17_11.12.09.071.png = 29
2022-03-17_11.12.10.771.png = 30
2022-03-17_11.12.14.168.png = 31
2022-03-17_11.12.15.877.png = 32
2022-03-17_11.12.17.566.png = 33
2022-03-17_11.12.19.265.png = 34
2022-03-17_11.12.20.970.png = 35
2022-03-17_11.12.22.669.png = 36
2022-03-17_11.12.24.363.png = 37
2022-03-17_11.12.26.061.png = 38
2022-03-17_11.12.27.765.png = 39
2022-03-17_11.12.29.458.png = 40
2022-03-17_11.12.31.165.png = 41
2022-03-17_11.12.32.865.png = 42
2022-03-17_11.12.34.569.png = 43
2022-03-17_11.12.36.255.png = 44
2022-03-17_11.12.37.954.png = 45
```

File Edit Format Run Options Window Help

```python
import os
from os.path import isfile, join
import os.path
path = r"C:\Users\jessb\Documents\Python\corrosion\corrosion-detection-master\data\test"

picturenumber = len(os.listdir(path))
print(picturenumber)

x = picturenumber/64
print(float(x))

gridnumber = 1
y = 1

for dirpath, dirnames, filenames in os.walk(path):
        while gridnumber <= 64:
                for filename in [f for f in filenames][:64]:
                        while y <= 2:
                                print ("%s = %d" %(filename,gridnumber))
                                y +=1
                gridnumber +=1
                y = 1
quit()
```

Improved Coordinate Concept



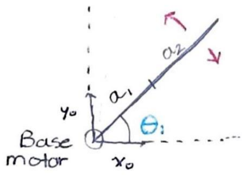
results_test - Notepad

File Edit Format View Help

2022-03-28_21.54.03.155.png = 33
2022-03-28_21.54.49.600.png = 30
2022-03-28_21.54.51.300.png = 14
2022-03-28_21.54.52.998.png = 11
2022-03-28_21.54.54.698.png = 23
2022-03-28_21.54.56.397.png = 56
2022-03-28_21.54.58.096.png = 55
2022-03-28_21.54.59.795.png = 41
2022-03-28_21.55.01.494.png = 45
2022-03-28_21.55.03.193.png = 39

- Values assigned to each picture
- Value is also assigned to a *motor position*
- IK goes back to position with code for rust remover step

Top

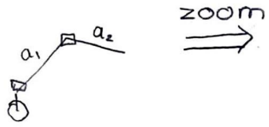


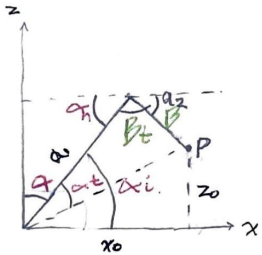
imput → x_0, y_0

Base rotation = θ_1 = $\boxed{\tan^{-1}(y_0/x_0)}$

if theta > 0, direction = HIGH   else, direction = LOW

Side





a_1 = shoulder length

a_2 = elbow length

B = Hypothenuse

① $Hypot = \sqrt{x_0^2 + z_0^2}$

$\alpha t = \arccos((B^2 + a_1^2 - a_2^2)/2) \cdot B \cdot a_1$

$\alpha i = \alpha t + \arctan(z_0/x_0)$

$\boxed{\alpha = 90 - \alpha i}$

$\beta t = \arccos((a_2^2 + a_1^2 - B^2)/2) \cdot a_1 \cdot a_2$

$\boxed{\beta = 180 - (\beta t + (90 - \alpha))}$

Language = C++

```cpp
#include <Stepper.h>

int i;
const int StepX = 2; //elbow joint
const int DirX = 5;
const int StepY = 3;//shoulder joint
const int DirY = 6;
const int StepZ = 4; //base joint
const int DirZ = 7;

float var = 1.8;
float direction;
float ShoulderLength = 9.555; //a1
float ElbowLength = 9.859; //a2
float Hypot;
float x0 = 5;//Target end point TEST
float y0 = 8;//Target end point TEST
float z0 = 4;//Target end point TEST
float A, B, C,
    Theta,
    Alpha,
  Alpha_temp,
  Alpha_i,
  Beta_temp,
    Beta;
```

```cpp
//HIGH for clockwise and LOW for anticlockwise
if(Beta<0) direction = LOW;
else direction = HIGH;
digitalWrite(DirX,direction);
Serial.println("X driection is :");
Serial.print("direction");

if(Alpha<0) direction = LOW;
else direction = HIGH;
digitalWrite(DirY,direction);
Serial.println("Y driection is :");
Serial.print("direction");

if(Theta>0) direction = LOW;
else direction = HIGH;
digitalWrite(DirZ,direction);
Serial.println("Z driection is :");
Serial.print("direction");

//other functions
serial();
movement();
```

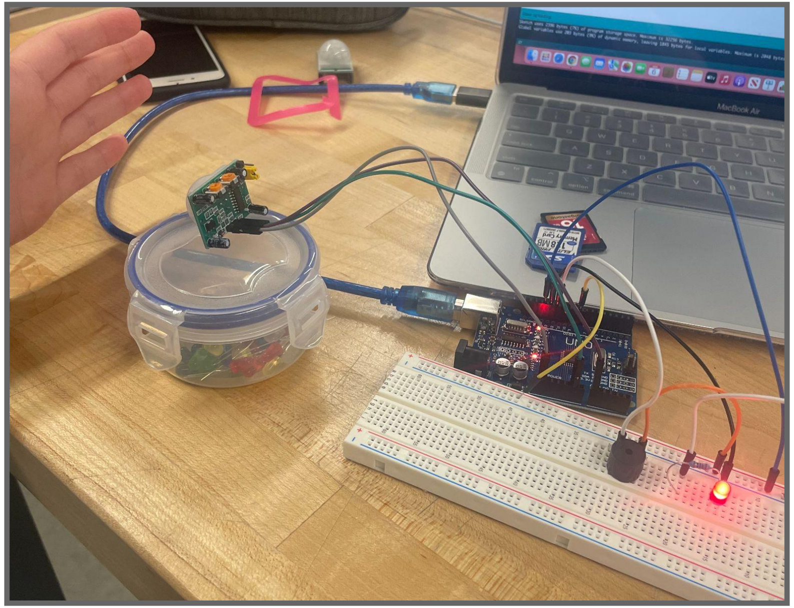
```cpp
void calculation() {

  Hypot = sqrt(sq(x0) + sq(z0));
  A = ElbowLength;
  B = Hypot;
  C = ShoulderLength;

  Theta = atan(y0/x0)* (180 / PI);

  Alpha_temp = acos((sq(B) + sq(C) - sq(A)) / (2 * B * C)) * (180 / PI);
  Alpha_i = 90 - Alpha_temp + atan(z0/x0)*(180 / PI);
  Alpha = 90 - Alpha_i;

  Beta_temp =  acos((sq(C) + sq(A) - sq(B)) / (2 * A * C)) * (180 / PI);
  Beta = 180 -(Alpha_i + Beta_temp);

}//end calc
```

```cpp
void movement(){

  //Move until target steps
  for(int x = 0; x<Theta/var ; x++){
  digitalWrite(StepZ,HIGH);
  delay(100);
  digitalWrite(StepZ,LOW);
  }

  for(int x = 0; x<Alpha/var ; x++){
  digitalWrite(StepY,HIGH);
  delay(100);
  digitalWrite(StepY,LOW);
  }

  for(int x = 0; x<Beta/var ; x++){
  digitalWrite(StepX,HIGH);
  delay(100);
  digitalWrite(StepX,LOW);
  }

}
```

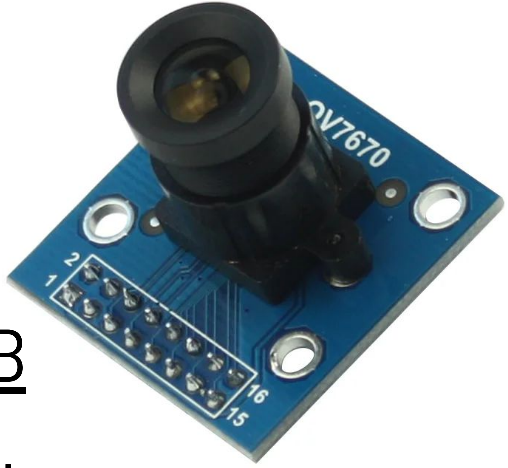
No analog pins left on arduino... Needed other solution!

Sensor_Code

```
int buzzer = 9;
int ledPin = 11;                // choose the pin for the LED
int inputPin = 10;               // choose the input pin (for PIR sensor)
int pirState = LOW;             // we start, assuming no motion detected
int val = 0;                    // variable for reading the pin status
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);      // declare LED as output
  pinMode(inputPin, INPUT);     // declare sensor as input
  pinMode(buzzer, OUTPUT);
}
 void loop(){
  val = digitalRead(inputPin);  // read input value
  if (val == HIGH) {            // check if the input is HIGH
    digitalWrite(ledPin, HIGH);  // turn LED ON
    if (pirState == LOW) {
      // we have just turned on

digitalWrite(buzzer,HIGH);
     delay(100);

      Serial.println("Motion detected!");
      // We only want to print on the output change, not state
      pirState = HIGH;
    }
  } else {
    digitalWrite(ledPin, LOW); // turn LED OFF
    if (pirState == HIGH){
      // we have just turned of

      digitalWrite(buzzer,LOW);
      delay(100);
      Serial.println("Motion ended!");
      // We only want to print on the output change, not state
      pirState = LOW;
    }
  }
}
```



- **OV7670** *<u>– 0.3 Megapixels</u>*
- <u>VGA sends picture output to USB</u>
- No external *Ram or Fifo* required
- *No Arduino shield* required!
- **Very weak!** 1Mhz capabilities with USB vs Arduino clock speed of 16Mhz



–