

**UNIVERSITY OF OTTAWA**  
**Faculty of Engineering**



uOttawa

**GNG1103 Design Project – Deliverable F**

Group C2

Aдриanna Chouliotis	300225514
Hunter Fleming	300213232
Jim Zhou	300197576
Minjung Gong	300080238
Toby Thai	300201869

**Submitted to:** Professor David Knox

**Submitted on:** March 6th, 2022

**Wrike Snapshot Link:**

<https://www.wrike.com/frontend/ganttchart/index.html?snapshotId=xH9ZzWTfshXFNcrx1t6L7nlmGXFxZG9W%7CIE2DSNZVHA2DELSTGIYA>

## TABLE OF CONTENTS

<b>TABLE OF CONTENTS</b> .....	<b>2</b>
<b>LIST OF TABLES</b> .....	<b>2</b>
<b>LIST OF FIGURES</b> .....	<b>2</b>
<b>CODE</b> .....	<b>3</b>
<b>TEST RUN DATA</b> .....	<b>8</b>
ASSEMBLY .....	8
<b>CALCULATIONS</b> .....	<b>9</b>
MAXIMUM REACH CALCULATIONS: .....	9
<b>UPDATED BILL OF MATERIALS</b> .....	<b>10</b>
<b>UPDATED DESIGN SPECIFICATIONS</b> .....	<b>11</b>
<b>REFERENCES</b> .....	<b>11</b>
OPEN-SOURCE CODE USED .....	11
INVERSE KINEMATICS .....	11
CODE DICTIONARY .....	11

## LIST OF TABLES

TABLE 1. INVERSE KINEMATICS CODE .....	3
TABLE 2. USER INTERFACE CODE .....	4
TABLE 3. UPDATED BILL OF MATERIALS .....	10
TABLE 4. UPDATED TARGET DESIGN SPECIFICATIONS .....	11

## LIST OF FIGURES

FIGURE 1. ARM AND END EFFECTOR CONNECTION .....	8
-------------------------------------------------	---

## CODE

All code was completed using Python.

<b>Table 1. Inverse Kinematics Code</b>	
1	<code>from numpy import *</code>
2	
3	<code>#a: lengths of arm pieces in cm</code>
4	<code>a1=7.4</code>
5	<code>a2=23</code>
6	<code>a3=25</code>
7	
8	<code>#insert desired positioning</code>
9	<code>px=</code>
10	<code>py= </code>
11	
12	<code>#phi is the angle of the end effector with respect to the x-axis</code>
13	<code>phi= #insert phi value</code>
14	<code>phi=deg2rad(phi)</code>
15	
16	
17	<code>wx=px-a3*cos(phi)</code>
18	<code>wy=py-a3*sin(phi)</code>
19	
20	<code>delta=wx**2+wy**2</code>
21	<code>c2=(delta-a1**2-a2**2)/(2*a1*a2)</code>
22	<code>s2=sqrt(1-c2**2)</code>
23	<code>theta_2=arctan2(s2, c2)</code>
24	
25	<code>s1=((a1+a2*c2)*wy-a2*s2*wx)/delta</code>
26	<code>c1=((a1+a2*c2)*wx+a2*s2*wy)/delta</code>
27	<code>theta_1=arctan2(s1, c1)</code>
28	<code>theta_3=phi-theta_1-theta_2</code>
29	
30	<code>#converting angles to degrees</code>
31	<code>b1=rad2deg(theta_1)</code>
32	<code>b2=rad2deg(theta_2)</code>
33	<code>b3=rad2deg(theta_3)</code>
34	
35	<code>print('theta 1:', b1)</code>
36	<code>print('theta 2:', b2)</code>
37	<code>print('theta 3:', b3)</code>
..	

One can determine the required angles that need to be connected to the robotic arm through the use of the inverse kinematic code found in Table 1. The only input needed is the desired positioning and the angle of the end effector with respect to the x-axis. With the determined angles, one inputs them into the Arduino code that connects to the motors and allows the motion to proceed. The Arduino coding will be completed during the next deliverable.

**Table 2.** User Interface Code

```

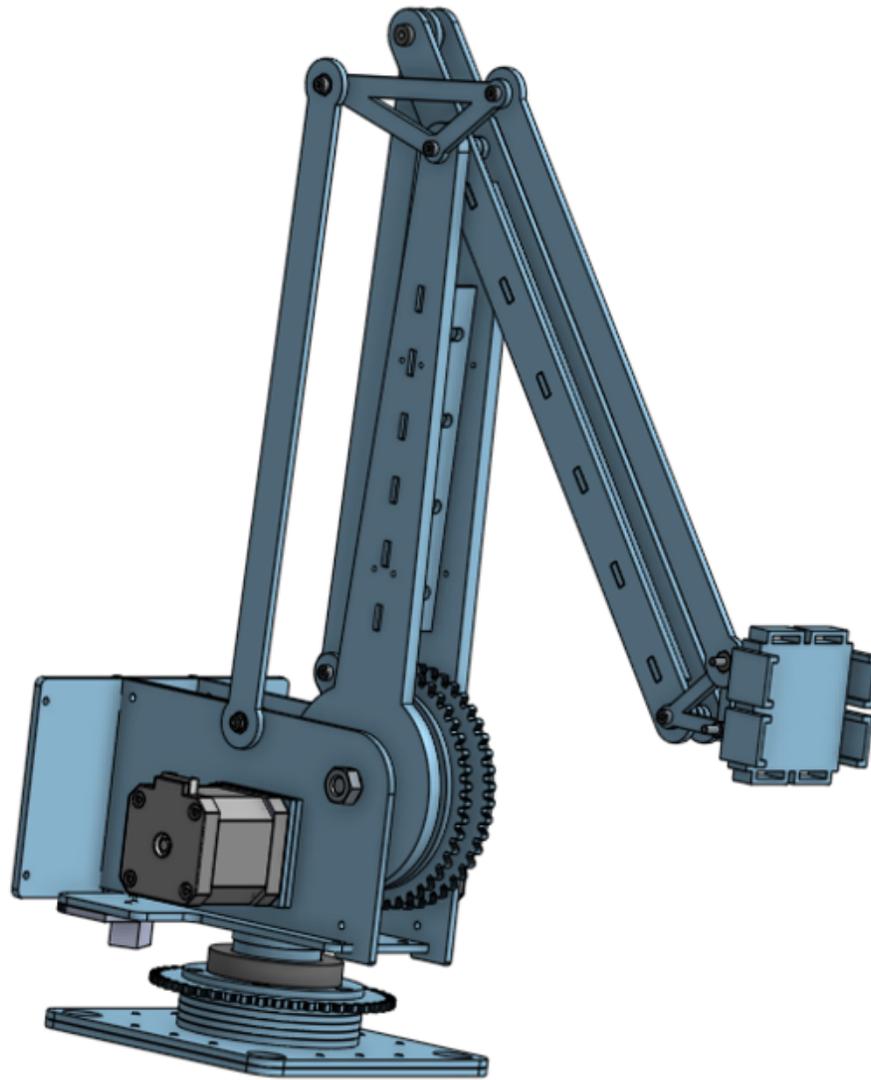
1 from tkinter import *
2 from numpy import *
3
4 root=Tk()
5 frame=Frame(root)
6 frame.pack()
7
8 from numpy import *
9
10 def startik():
11     #a: lengths of arm pieces in cm
12     a1=7.4
13     a2=23
14     a3=25
15
16     #insert desired positioning
17     px=
18     py=
19
20     #phi is the angle of the end effector with respect to the x-axis
21     phi=
22     phi=deg2rad(phi)
23
24
25     wx=px-a3*cos(phi)
26     wy=py-a3*sin(phi)
27
28     delta=wx**2+wy**2
29     c2=(delta-a1**2-a2**2)/(2*a1*a2)
30     s2=sqrt(1-c2**2)
31     theta_2=arctan2(s2,c2)
32
33     s1=((a1+a2*c2)*wy-a2*s2*wx)/delta
34     c1=((a1+a2*c2)*wx+a2*s2*wy)/delta
35     theta_1=arctan2(s1,c1)
36     theta_3=phi-theta_1-theta_2
37
38     #converting angles to degrees
39     b1=rad2deg(theta_1)
40     b2=rad2deg(theta_2)
41     b3=rad2deg(theta_3)
42
43     print('theta 1:',b1)
44     print('theta 2:',b2)
45     print('theta 3:',b3)
46
47 #Legend:
48 #Verify:confirm position input. If confirm; it goes to the choose end effector menu.
49     #If cancel; it goes back to choose orientation
50 #Orientpost:input desired postioning by inserting coordinate values
51 #Endeffect: choose end effector (scanner, painter or blaster)
52 #Scannerend: confirm that the scanner was the end effector that was meant to be chosen,
53     #if not cancel brings back to choice menu
54 #Painterend: painting end effectorm, confirm or cancel to bring back to choice menu
55 #Blasterend: choose between water, sand or return to menu
56     #sblaster: sand blaster, confirm or return to menu
57     #wblaster: water blaster, confirm or return to menu
58 #StartIK:calculates required angles

```

```
59
60
61 def verify():
62     frame2=Frame(root,bg="blue")
63     frame.pack()
64     leftframe=Frame(root,bg="green",bd=3)
65     leftframe.pack(side=LEFT)
66     rightframe=Frame(root,bg="red",bd=3)
67     rightframe.pack(side=RIGHT)
68     label=Label(frame,text="Verify Data Input")
69     label.pack()
70     confirm=Button(leftframe,text="CONFIRM",command=endeffect)
71     confirm.pack(padx=3,pady=3)
72     cancel=Button(rightframe,text="CANCEL",command=orientpost)
73     cancel.pack(padx=3,pady=3)
74
75 def orientpost():
76     label=Label(frame,text="Insert Desired Postion")
77     label.pack()
78     entry1=Entry(frame,width=20)
79     entry1.insert(0,"x-coordinate")
80     entry1.pack(padx=5,pady=5)
81     entry2=Entry(frame,width=20)
82     entry2.insert(0,"y-coordinate")
83     entry2.pack(padx=5,pady=5)
84     entry3=Entry(frame,width=20)
85     entry3.insert(0,"z-coordinate")
86     entry3.pack(padx=5,pady=5)
87     xcoord=entry1.get()
88     ycoord=entry2.get()
89     zcoord=entry3.get()
90     button1=Button(text="Enter",command=verify)
91     button1.pack()
92
93
94 label=Label(frame,text="Insert Desired Postion")
95 label.pack()
96
97 entry1=Entry(frame,width=20)
98 entry1.insert(0,"x-coordinate")
99 entry1.pack(padx=5,pady=5)
100
101 entry2=Entry(frame,width=20)
102 entry2.insert(0,"y-coordinate")
103 entry2.pack(padx=5,pady=5)
104
105 entry3=Entry(frame,width=20)
106 entry3.insert(0,"z-coordinate")
107 entry3.pack(padx=5,pady=5)
108
109 xcoord=entry1.get()
110 ycoord=entry2.get()
111 zcoord=entry3.get()
112
113
114 button1=Button(text="Enter",command=verify)
115 button1.pack()
116
```

```
117
118 def endeffect():
119     frame=Frame(root)
120     frame.pack()
121     label=Label(frame,text="Choose End Effector Type:")
122     label.pack()
123
124     def scannerend():
125         frame=Frame(root,bg="blue")
126         frame.pack()
127         label=Label(frame,text="Verify End Effector Choice: Scanner?")
128         label.pack()
129         leftframe=Frame(root,bg="green",bd=3)
130         leftframe.pack(side=LEFT)
131         rightframe=Frame(root,bg="red",bd=3)
132         rightframe.pack(side=RIGHT)
133         confirm=Button(leftframe,text="CONFIRM",command=startik)
134         confirm.pack()
135         cancel=Button(rightframe,text="CANCEL",command=endeffect2)
136         cancel.pack()
137
138     def blasterend():
139         frame=Frame(root)
140         frame.pack()
141         label=Label(frame,text="Choose Blaster Type:")
142         label.pack()
143         RBtttn4=Radiobutton(frame,text="Water",command=wblaster)
144         RBtttn4.pack(padx=5,pady=5)
145         RBtttn5=Radiobutton(frame,text="Sand",command=sblaster)
146         RBtttn5.pack(padx=5,pady=5)
147         RBtttn6=Radiobutton(frame,text="Return to End Effector Menu",command=endeffect2)
148         RBtttn6.pack(padx=5,pady=5)
149
150     def sblaster():
151         frame=Frame(root,bg="blue")
152         frame.pack()
153         frame=Frame(root,bg="purple")
154         frame.pack()
155         leftframe=Frame(root,bg="green",bd=3)
156         leftframe.pack(side=LEFT)
157         rightframe=Frame(root,bg="red",bd=3)
158         rightframe.pack(side=RIGHT)
159         label=Label(frame,text="Confirm Blaster Choice: Sand?")
160         label.pack()
161         confirm=Button(leftframe,text="CONFIRM",command=startik)
162         confirm.pack()
163         cancel=Button(rightframe,text="CANCEL",command=blasterend)
164         cancel.pack()
165
166     def wblaster():
167         frame=Frame(root,bg="purple")
168         frame.pack()
169         leftframe=Frame(root,bg="green",bd=3)
170         leftframe.pack(side=LEFT)
171         rightframe=Frame(root,bg="red",bd=3)
172         rightframe.pack(side=RIGHT)
173         label=Label(frame,text="Confirm Blaster Choice: Water?")
174         label.pack()
```

```
175     confirm=Button(leftframe,text="CONFIRM",command=startik)
176     confirm.pack()
177     cancel=Button(rightframe,text="CANCEL",command=blasterend)
178     cancel.pack()
179
180     def paintend():
181         frame=Frame(root,bg="pink")
182         frame.pack()
183         leftframe=Frame(root,bg="green",bd=3)
184         leftframe.pack(side=LEFT)
185         rightframe=Frame(root,bg="red",bd=3)
186         rightframe.pack(side=RIGHT)
187         label=Label(frame,text="Verify End Effector Choice: Painter?")
188         label.pack()
189         confirm=Button(leftframe,text="CONFIRM",command=startik)
190         confirm.pack()
191         cancel=Button(rightframe,text="CANCEL",command=endeffect2)
192         cancel.pack()
193
194     def endeffect2():
195         label=Label(frame,text="Choose End Effector Type:")
196         label.pack()
197         RBttn1=Radiobutton(frame,text="Scanner",command=scannerend)
198         RBttn1.pack(padx=5,pady=5)
199         RBttn2=Radiobutton(frame,text="Blaster",command=blasterend)
200         RBttn2.pack(padx=5,pady=5)
201         RBttn3=Radiobutton(frame,text="Painter",command=paintend)
202         RBttn3.pack(padx=5,pady=5)
203
204     RBttn1=Radiobutton(frame,text="Scanner",command=scannerend)
205     RBttn1.pack(padx=5,pady=5)
206     RBttn2=Radiobutton(frame,text="Blaster",command=blasterend)
207     RBttn2.pack(padx=5,pady=5)
208     RBttn3=Radiobutton(frame,text="Painter",command=paintend)
209     RBttn3.pack(padx=5,pady=5)
210
211     root.mainloop()
212
```



**Figure 1. Arm and End Effector Connection**

## TEST RUN DATA

### Assembly

To test the assembly of the mechanical components and build a physical arm with an end effector. Through the CAD model, it was confirmed that the arm can move flexibly within a range of 1 meter. The end effector will also be able to hold the blaster or painter using a Velcro band.

## CALCULATIONS

Approximate Volume Calculation: 0.974 m<sup>2</sup> (Onshape)

Maximum Reach Calculations:

$$\theta_2 = \cos^{-1} \left( \frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2} \right) = \cos^{-1} \left( \frac{1^2 + 1^2 - (0.3\text{m})^2 - (0.3\text{m})^2}{2(0.3\text{m})^2} \right) = N/A$$

⇒ No need to calculate  $\theta_1$ , as a 1m reach is already impossible.

Estimation with 45-degree value for each arm

$$A1 = \begin{matrix} & 0.707 & -0.707 & 0.707 & 0.212 \\ & 0.707 & 0.707 & -0.707 & 0.212 \\ & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 1 \end{matrix}$$

$$A2 = \begin{matrix} & 0.707 & -0.707 & 0.707 & 0.212 \\ & 0.707 & 0.707 & -0.707 & 0.212 \\ & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 1 \end{matrix}$$

### UPDATED BILL OF MATERIALS

**Table 3. Updated Bill of Materials**

Name	Description	Dimensions	Quantity	Unit Cost	Total Cost
<b>Software</b>					
Python	End effector needs open-source Python to work. Python can be download for free from the Python site and is useful and easy for coding.	-	1	\$0	\$0
<b>Hardware</b>					
Laptop	To code, prototype, and test with Python, we need a laptop. But it is free because everyone on the team has their own laptop already.	-	1	\$0	\$0
Arduino Uno	The Arduino uno is an open-source microcontroller board. It is required to operate the robot arm using Python.	-	1	\$0.00	\$0.00
ABS Filament	ABS filament is 3d printer plastic. It is easy to process, has high impact resistance and good heat resistance. It is often used as a substitute for metal in industrial products such as automobile parts and electronic device parts because of its very high strength.	-	1	\$0	\$0
Screw	Screws are used to connect the arm to the end effector.	M5-0.8×45mm	25	\$0.00	\$0.00
Wire Set	Wires are used to connect the Arduino to the end effector, supply and operate power.	-	120pcs	\$0.00	\$0.00
Velcro Industrial Roll	Velcro band is used in this project to strap the items together by attaching them to the End effector. This is suitable for strapping items as it has a good adhesion if only a certain amount of contact area is secured.	Length: 5m Width: 3cm	1	\$20.99	\$20.99
Servo Motor	Servo motor is a motor that can move exactly as much as inputted by the control and measurement circuit when movement is specified. This is where the movement of the end effector helps.	13.69×9.8× 3.61 cm 50g	4	\$0.00	\$0.00
			<b>TOTAL</b>	<b>\$20.99</b>	
			<b>TAX (13%)</b>	<b>\$2.73</b>	
			<b>TOTAL COST</b>	<b>\$23.72</b>	

## UPDATED DESIGN SPECIFICATIONS

**Table 4. Updated Target Design Specifications**

	<u>Design Specifications</u>	<u>Relation</u>	<u>Value</u>	<u>Units</u>	<u>Verification</u>	<u>Prototype 1: Specs Met?</u>
<b>Functional Requirements</b>						
1	3D Printed	-	Yes	>	Design	N/A
2	Coded off Python	-	Yes	-	Design	Yes
3	Open Source	-	Yes	-	Uploading	Yes
4	Ground Mounted	>	Yes	-	Design	Yes
<b>Constraints</b>						
1	Cost per Arm	$\leq$	50	\$ CAD	Cost Calculations	Yes
2	Temperature to Withstand	Range	-30 - 60	°C	Testing/Analysis	N/A
3	Degrees of Freedom	=	3	-	Design	Yes
4	Weight	$\leq$	9	Kg	Weighing	Yes
5	Size (L x H x W)	$\leq$	1	m <sup>2</sup>	Calculation	Yes
6	Payload	$\approx$	1	Kg	Testing	N/A
7	Pressure to Withstand	$\leq$	8	Bar	Testing/Analysis	
8	Range of Motion	$\geq$	$\pm 0.5$	m	Testing/Calculations	
9	Time to Paint 4 ft <sup>2</sup>	$\approx$	4	Hours	Testing	N/A
<b>Non-Functional Requirements</b>						
1	Ease of Use	$\leq$	6	Hours	Training Testing	Yes
2	Pinch Points	$\geq$	4	-	Design	Yes
3	UV/Corrosion Resistance	-	Yes	-	Testing	N/A
4	Life Span of the Product	$\geq$	1	Year	Use	N/A

## REFERENCES

### Open-Source Code Used

<https://github.com/aakieu/3-dof-planar/blob/master/InverseKinematics.py>

### Inverse Kinematics

<https://os.mbed.com/users/ms523/notebook/3d-inverse-kinematics/>  
[http://motion.cs.illinois.edu/software/klampt/latest/pyklampt\\_docs/Manual-IK.html](http://motion.cs.illinois.edu/software/klampt/latest/pyklampt_docs/Manual-IK.html)

### Code Dictionary

<https://realpython.com/python-gui-tkinter/>  
<https://coderslegacy.com/python/python-gui/python-tkinter-radio-button/>