-

ROBOTICS

MCTE 4352

SECTION 1

MINI PROJECT

MUHAMMAD ZULKARNAIN BIN ZAKARIA

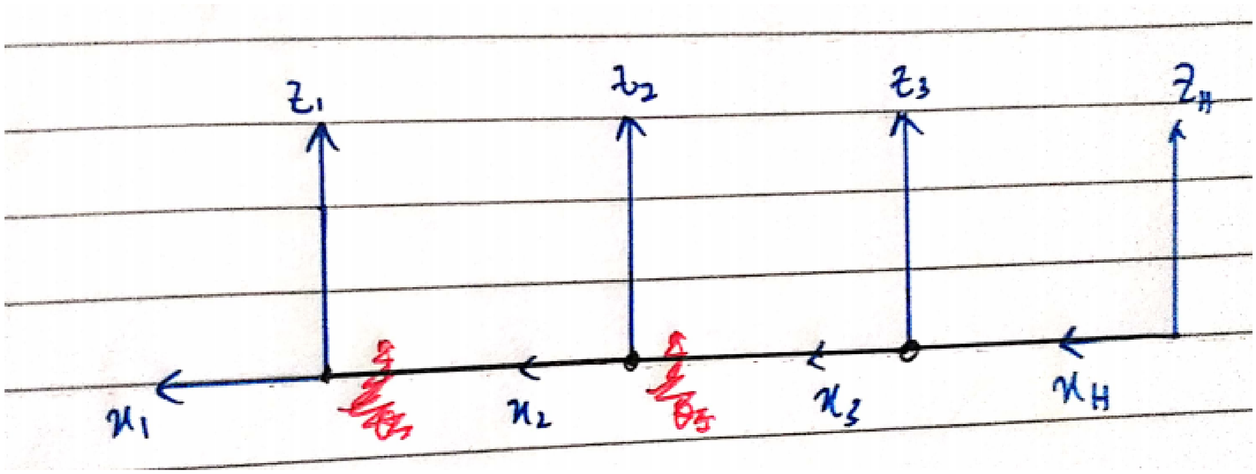
1723395

INTRODUCTION

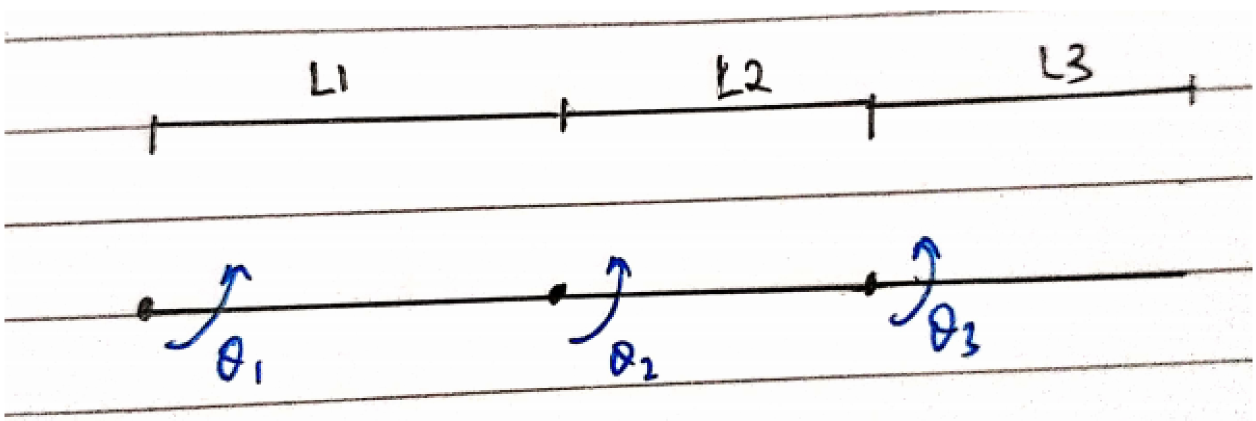
Robotics is an industry that has been growing with the modernization of this era. It has been developed time by time to ease human work. In this course, learning how to determine the frame of the simple robot and how to generate a robot with some calculation is a very precious knowledge. These projects allow us to learn using the modern software to simulate the robot. This is very advanced knowledge for students to get their hand in this kind of chance. For this project, MATLAB and RTB Toolbox is used to simulate the robot. Starting from determining the robot joint and translating all the points to matrix, find inverse kinematic using the software and lastly create a trajectory path for the robot. Exposure to modern solutions is a good approach to learning robotics. Robotics industries surely will be more challenging and could be the main industry in the near future.

3DoF Planar Robot

RRR robots were chosen because we need all links to move to create. Three revolute joints.



Side View of the RRR robot.



Top View of the RRR robot.

D-H Table of the Robot

D-H	θ	a	d	alpha
1-2	θ_1	L1	0	0
2-3	θ_2	L2	0	0
3-H	θ_3	L3	0	0

$x = \theta_1$; $y = \theta_2$; $z = \theta_3$; a=L1 ; b=L2 ; c=L3

$\mathbf{A_1^2} =$

$$\begin{pmatrix} \cos(x) & -\sin(x) & 0 & a\cdot\cos(x) \\ \sin(x) & \cos(x) & 0 & a\cdot\sin(x) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$\mathbf{A_2^3} =$

$$\begin{pmatrix} \cos(y) & -\sin(y) & 0 & b\cdot\cos(y) \\ \sin(y) & \cos(y) & 0 & b\cdot\sin(y) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$\mathbf{A_3^H} =$

$$\begin{pmatrix} \cos(z) & -\sin(z) & 0 & c\cdot\cos(z) \\ \sin(z) & \cos(z) & 0 & c\cdot\sin(z) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$\mathbf{A_1^H} =$

$$\begin{pmatrix} \cos(x+y+z) & -\sin(x+y+z) & 0 & a\cdot\cos(x)+b\cdot\cos(x+y)+c\cdot\cos(x+y+z) \\ \sin(x+y+z) & \cos(x+y+z) & 0 & a\cdot\sin(x)+b\cdot\sin(x+y)+c\cdot\sin(x+y+z) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Inverse Kinematic

Next step is to find the inverse kinematic to determine the formula for θ_1, θ_2 and θ_3. The calculations will be done in Microsoft Excel using the formula below. Points x and y are points that we get from the image processing method. This is the example of calculating the theta by using analytical solution or inverse kinematic.

$$x^2 = a \cdot \cos(x) + b \cdot \cos(x+y) + c \cdot \cos(x+y+z)$$

$$y^2 = a \cdot \sin(x) + b \cdot \sin(x+y) + c \cdot \sin(x+y+z)$$

$$x^2+y^2 = (L1)^2 + (L2)^2 + (L3)^2 + 2(L1)(L2)\cos\theta_2 + 2(L1)(L3)\cos(-\theta_2 - \theta_3) + 2(L2)(L3)\cos\theta_3$$

Here are the calculations from the excel for each letter. Using this formula above, this is the result of angles at each point. This is the example of calculating the theta by using analytical solution or inverse kinematic.

Letter Z

	x	y	z	tan_Theta1	Theta1_rad	Theta1_deg	cos_Theta 3	Theta3_rad	Theta3_deg	L1 =
Z	22	90	0	-0.0871835	-0.0869636	-4.98264756	-0.7564286	2.4286318	139.1503522	L3 =
	29	95	0	-0.0996014	-0.099274	-5.68798225	-0.7182738	2.37211445	135.9121462	
	40	100	0	-0.1310531	-0.1303105	-7.46623919	-0.6666667	2.30052398	131.8103149	
	47	105	0	-0.1262777	-0.1256128	-7.19708551	-0.6180357	2.23703795	128.1728334	
	54	107	0	-0.1440943	-0.1431093	-8.19955647	-0.584375	2.19490598	125.7588489	
	64	106	0	-0.1977877	-0.1952675	-11.1880012	-0.5555952	2.15987502	123.7517231	
	70	103	0	-0.2497695	-0.2447617	-14.0238131	-0.5503274	2.15355261	123.3894756	
	69	100	0	-0.2766601	-0.2699089	-15.4646431	-0.5725893	2.18045698	124.9309826	
	69	98	0	-0.2975313	-0.2891904	-16.5693887	-0.584375	2.19490598	125.7588489	
	66	95	0	-0.3181943	-0.3080641	-17.6507742	-0.6136607	2.23148495	127.8546697	
	63	91	0	-0.3502755	-0.3369202	-19.3041077	-0.6473214	2.27486131	130.3399522	
	59	89	0	-0.3548289	-0.3409702	-19.536154	-0.6725595	2.30845832	132.2649188	
	53	85	0	-0.3708027	-0.3550858	-20.3449161	-0.7132738	2.36495447	135.5019096	
	51	84	0	-0.3713129	-0.3555342	-20.3706089	-0.724494	2.38109641	136.4267748	
	49	82	0	-0.3839262	-0.3665734	-21.0031063	-0.7403274	2.40435355	137.7593109	
	41	78	0	-0.3791059	-0.3623655	-20.7620154	-0.7808036	2.46674729	141.3342086	
	35	71	0	-0.424984	-0.4018571	-23.0247163	-0.8254167	2.54173621	145.6307577	
	44	74	0	-0.457623	-0.4291751	-24.5899219	-0.7913095	2.48374415	142.3080575	
	52	75	0	-0.4990552	-0.4628914	-26.5217263	-0.7640179	2.44031404	139.8196952	
	58	72	0	-0.5814355	-0.5266573	-30.1752402	-0.7575	2.43027144	139.2442963	
	60	69	0	-0.6426635	-0.5712005	-32.7273755	-0.7630655	2.43883921	139.7351939	
	57	60	0	-0.8031267	-0.6766446	-38.7688785	-0.8080655	2.5116571	143.9073515	
	56	66	0	-0.6739907	-0.5930559	-33.979602	-0.7889286	2.47985975	142.0854973	
	54	49	0	-1.0873435	-0.8272181	-47.3961066	-0.8536607	2.59377027	148.6120897	
	45	42	0	-1.3011396	-0.9155241	-52.4556667	-0.8991369	2.6885898	154.0448481	
	40	39	0	-1.4095022	-0.9537427	-54.6454296	-0.9190179	2.73637815	156.7829191	
	30	33	0	-1.7134152	-1.0425008	-59.7308946	-0.9527083	2.83282352	162.3088315	
	24	31	0	-1.7789221	-1.0586818	-60.6579977	-0.9661607	2.88070235	165.0520864	
	25	36	0	-1.3241972	-0.9239917	-52.9408266	-0.9547321	2.83955434	162.6944795	
	29	46	0	-0.9043667	-0.7352224	-42.1251427	-0.9238988	2.74894395	157.5028864	
	38	53	0	-0.8129274	-0.682574	-39.1086072	-0.8853274	2.65799396	152.2918359	
	41	54	0	-0.8194123	-0.6864661	-39.3316108	-0.8750893	2.6364166	151.0555443	
	50	62	0	-0.7120703	-0.618781	-35.4535384	-0.8230952	2.5376363	145.3958501	
	66	70	0	-0.6543367	-0.5794178	-33.1981967	-0.7364286	2.39857225	137.4280668	

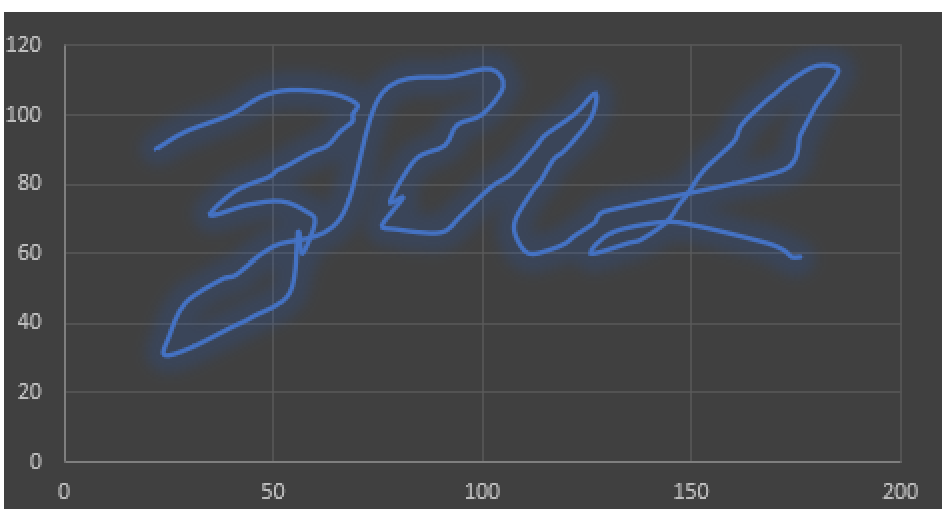
Letter U

U	76	106	0	-0.2395381	-0.2351082	-13.4707066	-0.5055952	2.10086807	120.3708736	
	93	111	0	-0.2303665	-0.2264164	-12.9727042	-0.3877976	1.96903735	112.8175301	
	102	113	0	-0.2235726	-0.2199554	-12.6025176	-0.3222321	1.89888278	108.7979693	
	105	108	0	-0.2728307	-0.2663483	-15.2606334	-0.3366369	1.91413938	109.6721081	
	100	100	0	-0.3472181	-0.3341944	-19.1479263	-0.4166667	2.00057176	114.6243184	
	94	97	0	-0.3713339	-0.3555527	-20.37167	-0.4688988	2.05883995	117.9628396	
	91	91	0	-0.4330715	-0.4086874	-23.4160623	-0.5189881	2.11646303	121.2643994	
	84	87	0	-0.4672792	-0.43713	-25.0457066	-0.5766369	2.18540262	125.2143464	
	78	75	0	-0.6154005	-0.5516665	-31.6081632	-0.6634226	2.29618006	131.5614265	
	81	76	0	-0.6078843	-0.5461966	-31.2947601	-0.6447321	2.27146915	130.1455954	
	76	68	0	-0.7242338	-0.6268058	-35.9133266	-0.702381	2.3495333	134.6183417	
	79	67	0	-0.7493556	-0.6430885	-36.8462597	-0.6925595	2.33582755	133.8330604	
	90	66	0	-0.7820951	-0.6637276	-38.0287888	-0.6411905	2.26684494	129.8806477	
	95	71	0	-0.7022444	-0.6122307	-35.0782339	-0.5932738	2.20591594	126.3896735	
	102	79	0	-0.5902047	-0.533186	-30.5493058	-0.5165179	2.11357566	121.0989647	
	107	83	0	-0.5396961	-0.4948979	-28.3555631	-0.466131	2.05570883	117.7834399	
	113	91	0	-0.4468621	-0.4202414	-24.0780583	-0.3854167	1.96645566	112.6696099	
	115	94	0	-0.4143903	-0.3928499	-22.5086417	-0.3553274	1.93406061	110.8135101	
	122	100	0	-0.3508252	-0.3374098	-19.3321552	-0.2713095	1.84554965	105.742206	
	127	106	0	-0.2907645	-0.2829625	-16.2125562	-0.1974702	1.769573	101.3890643	
	126	99	0	-0.3574866	-0.3433288	-19.6712904	-0.2477083	1.82111048	104.3419445	
	120	90	0	-0.4537971	-0.4260071	-24.4084098	-0.3422619	1.92011947	110.0147416	
	117	87	0	-0.4888062	-0.4546525	-26.0496719	-0.3792262	1.9597562	112.2857594	
	114	81	0	-0.5607067	-0.5110262	-29.2796432	-0.4298512	2.01512428	115.4581167	
	112	78	0	-0.5995472	-0.5400865	-30.9446756	-0.4575	2.045978	117.2259044	
	108	70	0	-0.7123514	-0.6189675	-35.464226	-0.5189286	2.1163934	121.2604096	
	108	65	0	-0.7888264	-0.6678905	-38.267309	-0.5390179	2.14006697	122.6168051	
	111	60	0	-0.8654037	-0.713369	-40.8730326	-0.5380655	2.13893668	122.5520443	
	116	61	0	-0.835549	-0.6960445	-39.8804108	-0.5006845	2.0951857	120.0452981	
	120	63	0	-0.7931169	-0.6705298	-38.4185276	-0.4652083	2.05466627	117.7237059	
	122	65	0	-0.7580816	-0.6486533	-37.1650983	-0.4431845	2.02994434	116.3072435	
	127	69	0	-0.6896489	-0.6037451	-34.5920442	-0.3901786	1.97162186	112.9656111	
	129	72	0	-0.6461854	-0.573689	-32.8699563	-0.3623512	1.94158561	111.2446611	

Letter L

L	172	84	0	-0.3795449	-0.3627493	-20.7840046	0.07857143	1.49214383	85.49354387	
	176	94	0	-0.279373	-0.2724272	-15.6089269	0.17297619	1.39694571	80.03909364	
	180	103	0	-0.1905599	-0.1883023	-10.7889263	0.268125	1.29935009	74.44727616	
	183	108	0	-0.1383632	-0.1374903	-7.87761321	0.33193452	1.23244269	70.61376466	
	185	113	0	-0.0898793	-0.0896385	-5.13590619	0.38672619	1.17371741	67.24905384	
	180	114	0	-0.1021502	-0.1017971	-5.83254263	0.33916667	1.22476541	70.17388896	
	174	110	0	-0.1564966	-0.1552375	-8.89445336	0.24928571	1.31885371	75.56475153	
	170	106	0	-0.2026019	-0.1998961	-11.4532048	0.18261905	1.38714669	79.47765077	
	167	103	0	-0.2370302	-0.232735	-13.3347342	0.13386905	1.43652418	82.30677251	
	162	97	0	-0.3031844	-0.2943757	-16.8664862	0.04919643	1.52158003	87.18011401	
	160	92	0	-0.3528796	-0.3392379	-19.4368983	0.00190476	1.56889156	89.89086512	
	153	84	0	-0.448124	-0.4212928	-24.1382981	-0.1052083	1.67619972	96.03916966	
	149	77	0	-0.5304841	-0.4877365	-27.9452417	-0.1747024	1.74639982	100.061339	
	147	74	0	-0.5688186	-0.5171764	-29.6320225	-0.2058036	1.77808111	101.8765434	
	144	69	0	-0.6350372	-0.5657845	-32.417066	-0.2530655	1.82664389	104.6589857	
	138	64	0	-0.7182398	-0.6228628	-35.6874113	-0.3232143	1.89992045	108.8574231	
	135	63	0	-0.7427823	-0.6388657	-36.6043096	-0.351369	1.92982932	110.5710751	
	126	60	0	-0.8187244	-0.6860545	-39.3080246	-0.4322619	2.01779596	115.6111922	
	131	66	0	-0.7171943	-0.6221728	-35.6478748	-0.3715179	1.95143969	111.809258	
	143	69	0	-0.6386983	-0.5683892	-32.5663	-0.2616071	1.83548329	105.1654457	
	151	68	0	-0.6189847	-0.554262	-31.7568746	-0.1956845	1.76775175	101.2847143	
	168	63	0	-0.5920187	-0.5345302	-30.6263245	-0.0537798	1.62460205	93.08284066	
	172	61	0	-0.5903571	-0.5332989	-30.5557789	-0.0206845	1.59148233	91.18522044	
	174	59	0	-0.5986512	-0.5394272	-30.9069005	-0.0072321	1.57802853	90.41437487	
	176	59	0	-0.5871023	-0.5308819	-30.4172937	0.01360119	1.55719472	89.22068516	

The shape of the plotted points manually using Microsoft Excel.



MATLAB CODING EXPLANATION

For the first step, declaring the joint of the robot and put in the values from DH Table into the code. In this code, the length of the joint is declared and the type of the robot is set up. From the code below, the robot will be set up as an RRR robot which has three revolute joints.

```
L1 = 65; % Length for joint1
L2 = 80;  %Length for joint2
L3 = 70; % length for joint3

%Set up DH parameter for the robot
L(1) = Link([0 0 L1 0 0]); % L = Link([Theta d a alpha 0/1]) >> 0 for revolute and 1 for prismatic
L(2) = Link([0 0 L2 0 0]);
L(3) = Link([0 0 L3 0 0]);
Rob = SerialLink([L(1) L(2) L(3)], 'name','RRR'); %name the robot RRR
```

To check on the DH table from this code whether it is the same as what has been calculated, put the SerialLink in the command window..

```
RRR (3 axis, RRR, stdDH, fastRNE)

+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|          0|         65|          0|          0|
|  2|         q2|          0|         80|          0|          0|
|  3|         q3|          0|         70|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
```

Next step is to translate all the points to matrix form. 91 points is translated to the matrix since the name have 91 coordinates Below is the code.

```
T0=transl(22,90,30);
T1=transl(29,95,0);
T2=transl(40,100,0);
T3=transl(47,105,0);
T4=transl(54,107,0);
T5=transl(64,106,0);
T6=transl(70,103,0);
T7=transl(69,100,0);
T8=transl(69,98,0);
T9=transl(66,95,0);
T10=transl(63,91,0);
T11=transl(59,89,0);
T12=transl(53,85,0);
T13=transl(51,84,0);
T14=transl(49,82,0);
T15=transl(41,78,0);
T16=transl(35,71,0);
T17=transl(44,74,0);
T18=transl(52,75,0);
T19=transl(58,72,0);
T20=transl(60,69,0);
T21=transl(57,60,0);
T22=transl(56,66,0);
T23=transl(54,49,0);
T24=transl(45,42,0);
T25=transl(40,39,0);
T26=transl(30,33,0);
T27=transl(24,31,0);
```

Next step is to get the inverse kinematic from the matrix. The inverse kinematic will be used to animate the trajectory of the robot. The list goes for all 91 points. P0 is the initial angle for all three joints. T represents the matrix, P represents the angle and [1,1,1,0,0,0] is a function to mask the matrix.

```
%Inverse Kinematic
P0=[0 0 0]; % Initial Coordinate of Robot
P1 = Rob.ikine(T0,P0,[1,1,1,0,0,0]);
P2 = Rob.ikine(T1,P1,[1,1,1,0,0,0]);
P3 = Rob.ikine(T2,P2,[1,1,1,0,0,0]);
P4 = Rob.ikine(T3,P3,[1,1,1,0,0,0]);
P5 = Rob.ikine(T4,P4,[1,1,1,0,0,0]);
P6 = Rob.ikine(T5,P5,[1,1,1,0,0,0]);
P7 = Rob.ikine(T6,P6,[1,1,1,0,0,0]);
P8 = Rob.ikine(T7,P7,[1,1,1,0,0,0]);
P9 = Rob.ikine(T8,P8,[1,1,1,0,0,0]);
P10 = Rob.ikine(T9,P9,[1,1,1,0,0,0]);
P11 = Rob.ikine(T10,P10,[1,1,1,0,0,0]);
P12 = Rob.ikine(T11,P11,[1,1,1,0,0,0]);
P13 = Rob.ikine(T12,P12,[1,1,1,0,0,0]);
P14 = Rob.ikine(T13,P13,[1,1,1,0,0,0]);
P15 = Rob.ikine(T14,P14,[1,1,1,0,0,0]);
P16 = Rob.ikine(T15,P15,[1,1,1,0,0,0]);
P17 = Rob.ikine(T16,P16,[1,1,1,0,0,0]);
P18 = Rob.ikine(T17,P17,[1,1,1,0,0,0]);
P19 = Rob.ikine(T18,P18,[1,1,1,0,0,0]);
P20 = Rob.ikine(T19,P19,[1,1,1,0,0,0]);
P21 = Rob.ikine(T20,P20,[1,1,1,0,0,0]);
P22 = Rob.ikine(T21,P21,[1,1,1,0,0,0]);
P23 = Rob.ikine(T22,P22,[1,1,1,0,0,0]);
P24 = Rob.ikine(T23,P23,[1,1,1,0,0,0]);
```

Then, create the trajectory array using all coordinates of our points. This step is to initialize the size of the trajectory.

```
%Create trajectory path for the robot
trajectory = [
22  90  30;
22  90  30;
29  95  0;
40  100 0;
47  105 0;
54  107 0;
64  106 0;
70  103 0;
69  100 0;
69  98  0;
66  95  0;
63  91  0;
59  89  0;
53  85  0;
51  84  0;
49  82  0;
41  78  0;
35  71  0;
44  74  0;
52  75  0;
58  72  0;
60  69  0;
57  60  0;
56  66  0;
54  49  0;
45  42  0;
```

Below is the code to set up the size of the trajectory path that will be use in plotting the coordinates.

```
[nx,ny] = size(trajectory);
```

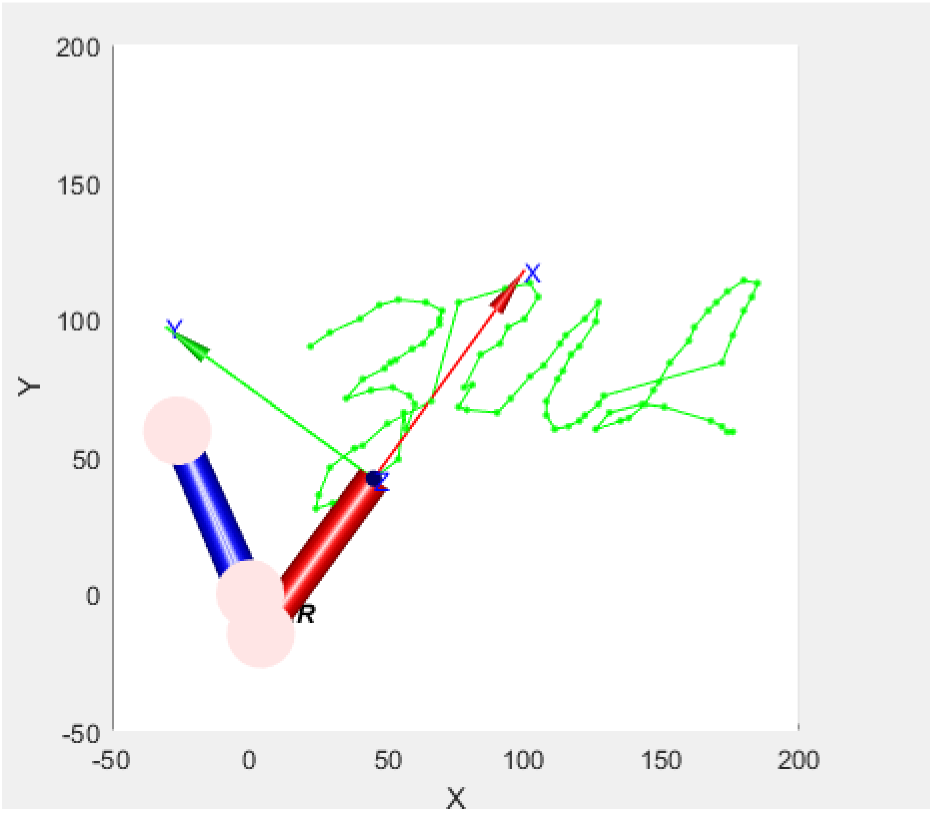
Then , all the points in the array of trajectory will be used in the code below to plot all the points. The for loop is used to make sure the trajectory path goes through all the points and neither exceeds nor less. The axis part is to declare the workspace or the Cartesian range that the plot will take place. The axis should be larger than any coordinates of the points. Then, labels the axis and set view for the animation.

```
figure
hold on

for i = 1:nx-1
    v=[trajectory(i,:);trajectory(i+1,:)];
    plot3(v(:,1),v(:,2),v(:,3),'g');
    plot3(v(:,1),v(:,2),v(:,3),'g.')
end

axis([-50 200 -50 200 -50 200]);
xlabel('X-Axis');
ylabel('Y-Axis');
zlabel('Z-Axis');

view(0,90);
```

And for the last part is the code to declare the trajectory of the robot from one point to the next point. The command 'jtraj' is to connect all the angles of joints from one point to another point and move the robot. Then use the Rob.plot to show the robot and declare it workspace.

```
t=[0:3:3];
%Animate
A1 = jtraj(P0,P1,t);
Rob.plot(A1,'workspace',[-50 200 -50 200 -50 200]);
A2 = jtraj(P1,P2,t);
Rob.plot(A2,'workspace',[-50 200 -50 200 -50 200]);
A3 = jtraj(P2,P3,t);
Rob.plot(A3,'workspace',[-50 200 -50 200 -50 200]);
A4 = jtraj(P3,P4,t);
Rob.plot(A4,'workspace',[-50 200 -50 200 -50 200]);
A5 = jtraj(P4,P5,t);
Rob.plot(A5,'workspace',[-50 200 -50 200 -50 200]);
A6 = jtraj(P5,P6,t);
Rob.plot(A6,'workspace',[-50 200 -50 200 -50 200]);
A7 = jtraj(P6,P7,t);
Rob.plot(A7,'workspace',[-50 200 -50 200 -50 200]);
A8 = jtraj(P7,P8,t);
Rob.plot(A8,'workspace',[-50 200 -50 200 -50 200]);
A9 = jtraj(P8,P9,t);
Rob.plot(A9, 'workspace',[-50 200 -50 200 -50 200]);
A10 = jtraj(P9,P10,t);
Rob.plot(A10, 'workspace',[-50 200 -50 200 -50 200]);
```

The result of the plotted points by using the simulation tools in RTB Toolbox.



DISCUSSION AND CONCLUSION

Settling up the robot using two methods which is manually and using the RTB Toolbox is surely different. However, the first step is to identify the robot and in this project RRR robots were chosen. Next is to construct DH Table parameters from the robot. Then, find the transformation matrix from the DH Table. Next is to find the inverse kinematic for the robot. By using the software, we could easily find the inverse kinematic for software. But for the manual method, calculations need to be done. However, inverse kinematic for the manual method cannot be done as the formula could not be obtained as the value of the angles remain unknown and lack of equations. The software part runs smoothly and manages to plot all the points. For the manual method, a line graph was created to compare it with the software method. However, the plot is not very smooth. This is because smaller points are plotted. The greater the number of points, the smoother the plot will be. The plot that has been obtained from both methods is the same. The value of the inverse kinematic could not be compared as the manual method cannot be done.

In conclusion, both methods are good, but the software method is easier and faster to do compared to the manual method.