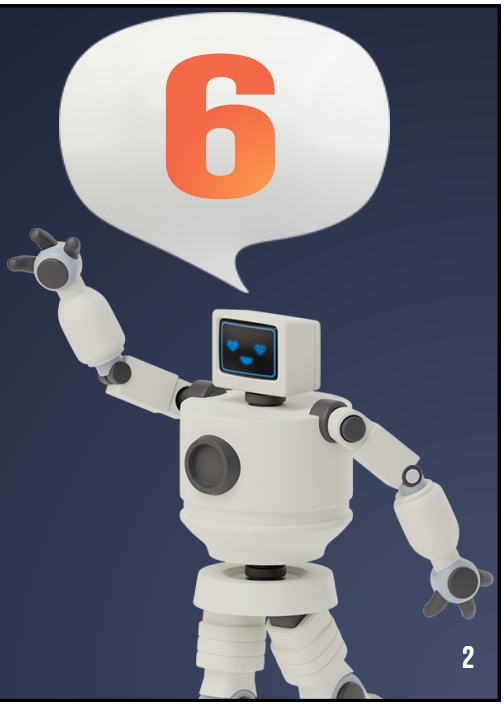
ROBOTIC ARM



Robotics Workshop

Umarah Qaseem

2

2

ROBOTIC ARM

Simulation

3

1 LINK ARM

Homogeneous transformation

```
>> a1 = 1

a1 =

     1

>> q1 = 0.2

q1 =

    0.2000

>> trchain2('R(q1) Tx(a1)', q1)

ans =

    0.9801   -0.1987    0.9801
    0.1987    0.9801    0.1987
         0         0    1.0000
```

Robotics – Umarah Qaseem

4

1 LINK ARM

Symbolic Representation

```
>> syms q1 a1
>> trchain2('R(q1) Tx(a1)', q1)

ans =

[ cos(q1), -sin(q1), a1*cos(q1)]
[ sin(q1),  cos(q1), a1*sin(q1)]
[       0,        0,          1]
```

Robotics – Umarah Qaseem

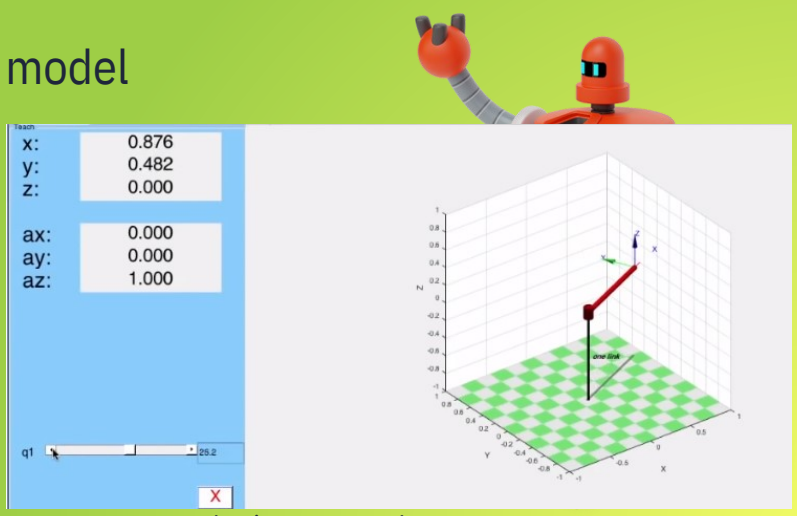
5

1 LINK ARM

Importing planer model

Try moving slider

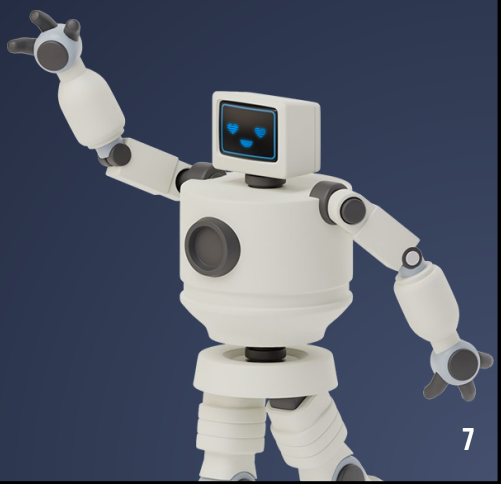
```
>> mdl_planar1
>> p1.teach
```



Robotics – Umarah Qaseem

6

Do same for 2 Link robotic arm yourself



7

7

2 LINK ARM

Homogeneous tranformation

```
>> syms q1 q2 a1 a2
>> trchain2('R(q1) Tx(a1) R(q2) Tx(a2)', [q1 q2])
```

```
>> mdl_planar2
>> p2.teach
>>
```

```
>> a1 = 1

a1 =

     1

>> a2 = 1

a2 =

     1

>> q1 = 0.2

q1 =

    0.2000

>> q2 = 0.3

q2 =

    0.3000

>> trchain2('R(q1) Tx(a1) R(q2) Tx(a2)', [q1 q2])

ans =

    0.8776   -0.4794    1.8576
    0.4794    0.8776    0.6781
```

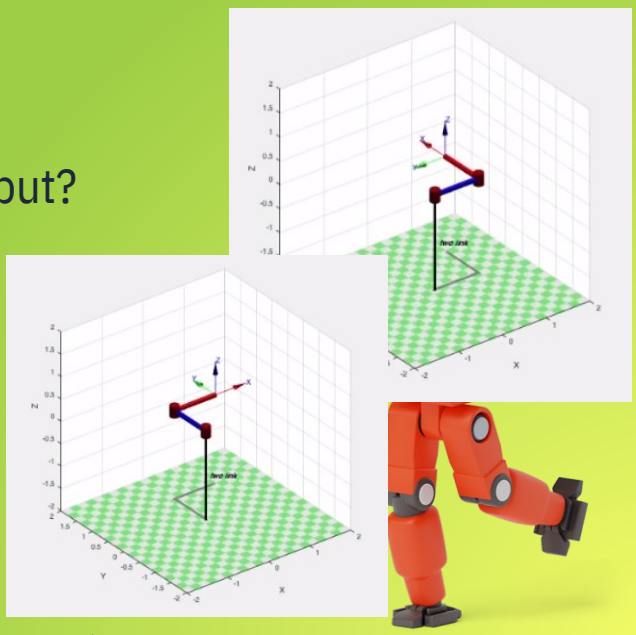
Robotics – Umarah Qaseem

8

2 LINK ARM

Different Configurations, but?

```
>> mdl_planar2
>> p2.teach
>> p2.plot([0 pi/2])
>> p2.plot([pi/2 -pi/2])
>>
```

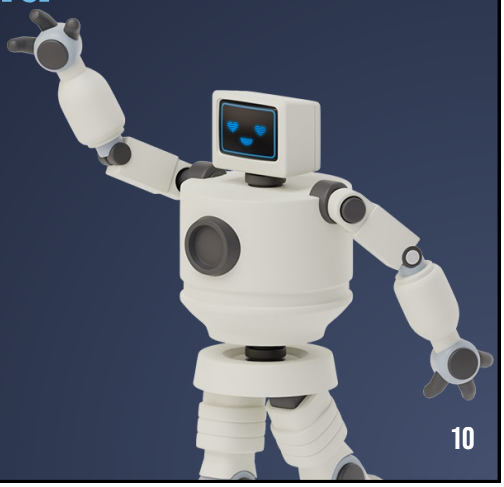


Robotics – Umarah Qaseem

9

3 link Robot Arm Symbolic expression and model only

Write!
Task.



10

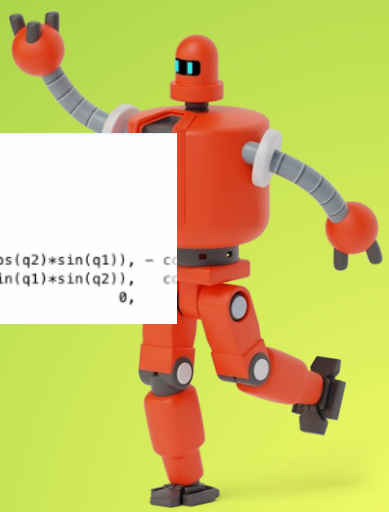
10

3 LINK ARM

Symbolic expression and model

```
>> syms a1 a2 a3 q1 q2 q3
>>
>> trchain2('R(q1) Tx(a1) R(q2) Tx(a2) R(q3) Tx(a3)', [q1 q2 q3])

ans =

[ cos(q3)*(cos(q1)*cos(q2) - sin(q1)*sin(q2)) - sin(q3)*(cos(q1)*sin(q2) + cos(q2)*sin(q1)), - c
[ cos(q3)*(cos(q1)*sin(q2) + cos(q2)*sin(q1)) + sin(q3)*(cos(q1)*cos(q2) - sin(q1)*sin(q2)),   c
[                                                                                          0,
```

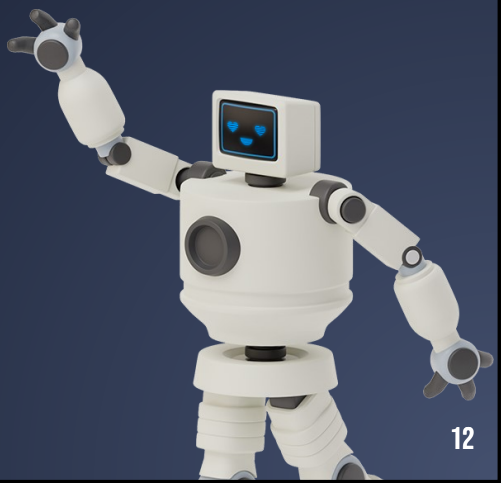
```
>> mdl_planar3
>> p3.teach
>>
```



Robotics – Umarah Qaseem

11

3 Dimensions!



12

12

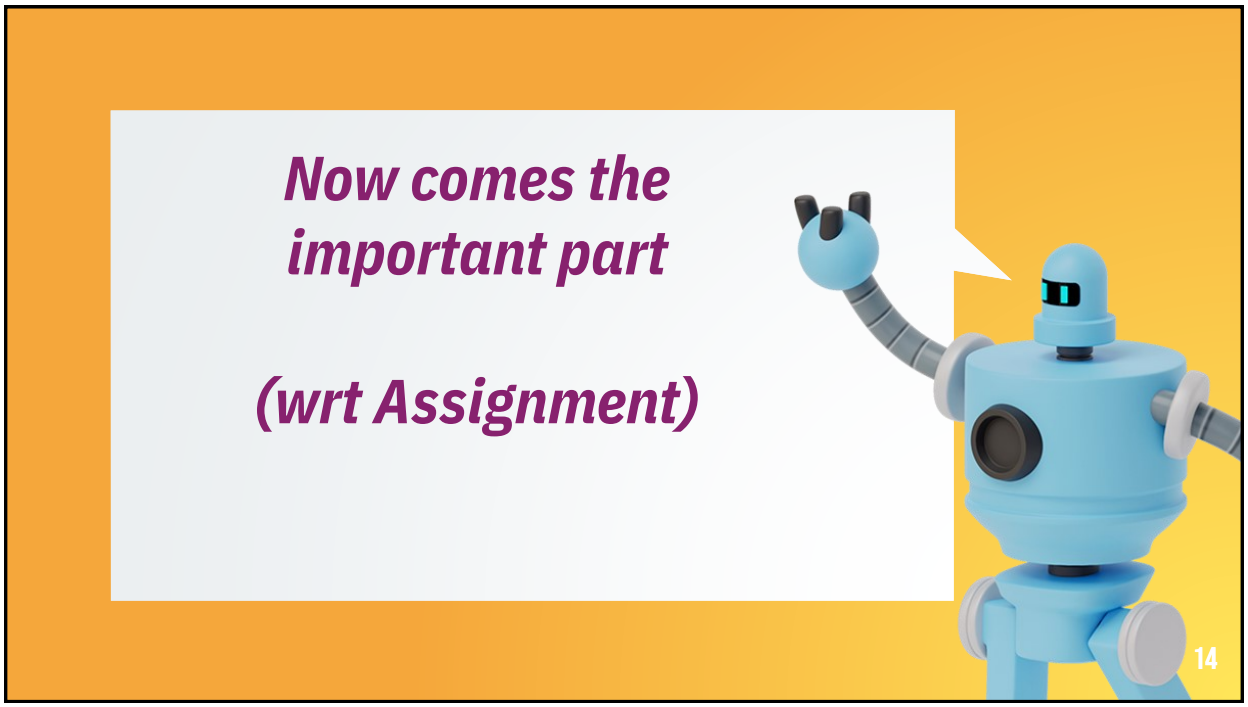
4 JOINT ARM IN 3D

Symbolic expression and model

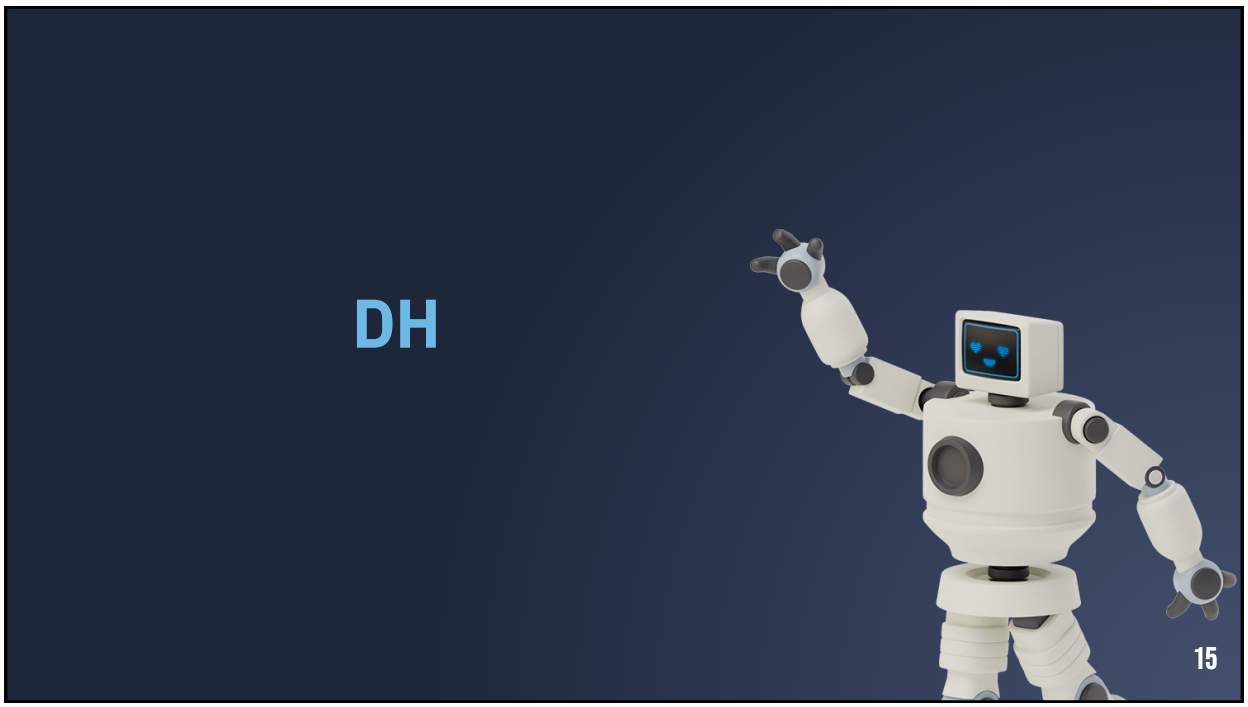
```
>> syms a1 a2 a3 a4 q1 q2 q3 q4
>> trchain('Rz(q1)Tz(a1)Ry(q2)Tz(a2)Ry(q3)Tz(a3)Ry(q4)Tz(a4)', [q1 q2 q3 q4])

ans =

[   cos(q4)*(cos(q1)*cos(q2)*cos(q3) - cos(q1)*sin(q2)*sin(q3)) - sin(q4)*(cos(q1)*cos(q2
[ - cos(q4)*(sin(q1)*sin(q2)*sin(q3) - cos(q2)*cos(q3)*sin(q1)) - sin(q4)*(cos(q2)*sin(q1
[                                 - cos(q4)*(cos(q2)*sin(q3) + cos(q3)*sin(q2)) - sin(q4)
[

>>
```

Robotics – Umarah Qaseem

13



14



15

```
>> dh = [
0 0 1 0
0 0 1 0
]

dh =

     0     0     1     0
     0     0     1     0

>> r = SerialLink(dh)

r =

noname (2 axis, RR, stdDH, fastRNE)

+---+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |
+---+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          1|          0|
|  2|         q2|          0|          1|          0|
+---+-----------+-----------+-----------+-----------+

grav =    0  base = 1  0  0  0   tool =  1  0  0  0
          0          0  1  0  0           0  1  0  0
       9.81          0  0  1  0           0  0  1  0
                     0  0  0  1           0  0  0  1

>>
```

16

DH

Give angle values to your arm

```
>> r.plot([0.2 0.3])
>> r.teach
```

17

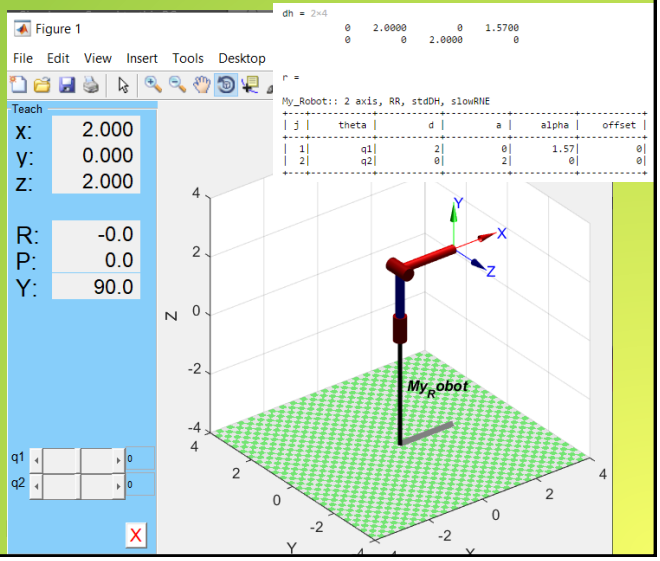
DH – FORWARD KINEMATICS

```
>> r.fkine([0.2 0.3])

ans =

    0.8776   -0.4794         0    1.8576
    0.4794    0.8776         0    0.6781
         0         0    1.0000         0
         0         0         0    1.0000
```

18

FORWARD KINEMATICS – METHOD 1

```
dh = [
0 2 0 1.57
0 0 2 0
]
r = SerialLink(dh,
'name','My_Robot')
r.plot([0.2,0.3])
r.teach
r.fkine([0.2, 0.3])
```



19

IMPORTANT

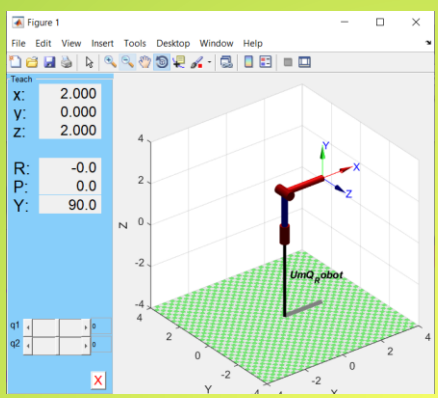
Clear your workspace by writing clear in the command window

20

FORWARD KINEMATICS – METHOD 2

```
L(1) = Link([ 0 2 0 pi/2], 'standard');
L(2) = Link([ 0 0 2 0], 'standard');
r=SerialLink(L, 'name', 'UmQ_Robot');
r.plot([0.2,0.3])
r.Teach
```

Sequence: Theta, d, a, alpha

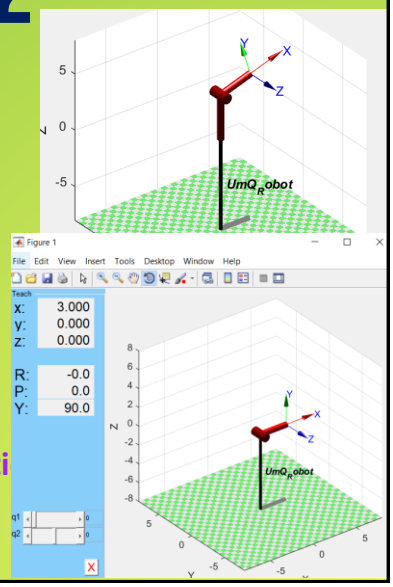


21

FK METHOD 2– EXAMPLE 2

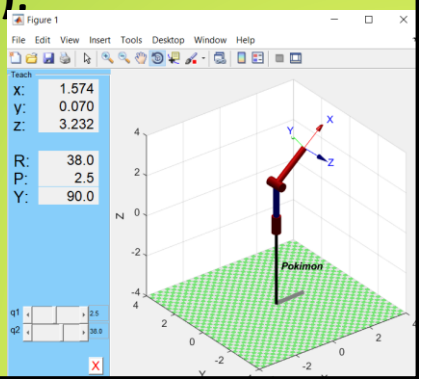
```
L(1) = Link([ 0 2 0 pi/2 1], 'standard');
L(2) = Link([ 0 0 3 0 0], 'standard');
L(1).qlim = [0,5]
r=SerialLink(L, 'name', 'UmQ_Robot');
r.plot([4,0.3])
r.teach
```

Sequence: Theta, d, a, alpha, 0 for revolute & 1 for prismati



FORWARD KINEMATICS – METHOD 4

```
L(1) = Revolute('d', 2, 'a', 0, 'alpha', pi/2);
L(2) = Revolute('d', 0, 'a', 2, 'alpha', 0);
twolink = SerialLink(L, 'name', 'Pokimon');
twolink.teach
```



WITH PUMA 560

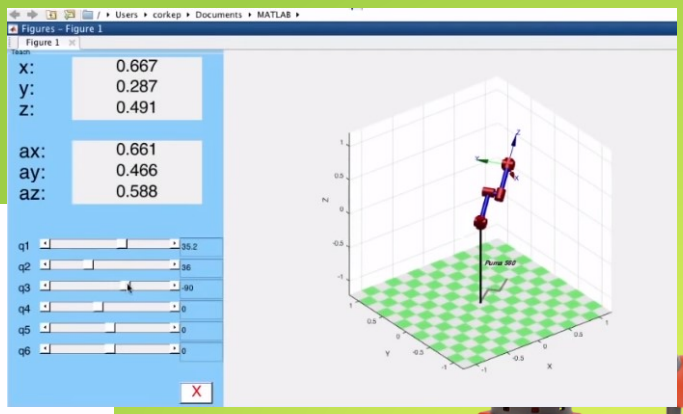
```
>> mdl_puma560
>> p560

p560 =

Puma 560 (6 axis, RRRRRR, stdDH, fastRNE)
 viscous friction; params of 8/95;
+---+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |
+---+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|      1.571|
|  2|         q2|          0|     0.4318|          0|
|  3|         q3|       0.15|     0.0203|     -1.571|
|  4|         q4|     0.4318|          0|      1.571|
|  5|         q5|          0|          0|     -1.571|
|  6|         q6|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+

grav =    0  base = 1  0  0  0   tool =  1  0  0  0
          0         0  1  0  0           0  1  0  0
       9.81         0  0  1  0           0  0  1  0
                    0  0  0  1           0  0  0  1

>> p560.plot(qz)
```

Do not use this for assignment. Why?

PUMA 560

```
>> p560.plot(qz)
>> p560.plot(qr)
>> p560.teach
>> p560.fkine([0.1 0.2 0.3 0 0 0])

ans =

    0.8732   -0.0998   -0.4770    0.2478
    0.0876    0.9950   -0.0479   -0.1259
    0.4794   -0.0000    0.8776    0.4745
         0         0         0    1.0000

>>
```

PUMA 560




```
>> p560.plot(qz)
>> p560.plot(qr)
>> p560.teach
>> p560.fkine([0.1 0.2 0.3 0 0 0])

ans =

    0.8732   -0.0998   -0.4770    0.2478
    0.0876    0.9950   -0.0479   -0.1259
    0.4794   -0.0000    0.8776    0.4745
         0         0         0    1.0000

>>
```

26

BASE TRANSFORM

p560.base

```
>> p560.base = transl(10, 15, 2)
p560 =

Puma 560 (6 axis, RRRRRR, stdDH, fastRNE)
 viscous friction; params of 8/95;
+---+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |
+---+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|      1.571|
|  2|         q2|          0|     0.4318|          0|
|  3|         q3|       0.15|     0.0203|     -1.571|
|  4|         q4|     0.4318|          0|      1.571|
|  5|         q5|          0|          0|     -1.571|
|  6|         q6|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+

grav =    0  base = 1   0   0  10   tool =  1  0  0  0
          0          0   1   0  15           0  1  0  0
       9.81          0   0   1   2           0  0  1  0
                     0   0   0   1           0  0  0  1

>> p560.fkine([0.1 0.2 0.3 0 0 0])
```

```
ans =

    0.8732   -0.0998   -0.4770   10.2478
   -0.0876   -0.9950    0.0479   15.1259
   -0.4794    0.0000   -0.8776    1.5255
         0         0         0    1.0000
```


27

TOOL TRANSFORM

```
>> p560.tool = transl(0, 0, 0.2)
p560 =

Puma 560 (6 axis, RRRRRR, stdDH, fastRNE)
 viscous friction; params of 8/95;
+---+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |
+---+-----------+-----------+-----------+-----------+
|  1|        q1|          0|          0|      1.571|
|  2|        q2|          0|     0.4318|          0|
|  3|        q3|       0.15|     0.0203|     -1.571|
|  4|        q4|     0.4318|          0|      1.571|
|  5|        q5|          0|          0|     -1.571|
|  6|        q6|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+

grav =    0  base = 1   0   0  10   tool =  1          0          0          0
          0          0  -1   0  15           0          1          0          0
       9.81          0   0  -1   2           0          0          1        0.2
                     0   0   0   1           0          0          0          1

>> p560.fkine([0.1 0.2 0.3 0 0 0])
```

P560.tool
P560.T6w7

Robotics – Umarah Qaseem

28

TASKS

29

TASKS

ALREADY COMPLETED!

1- Simulate 1, 2 and 3 joint planar arm
2- Simulate 4 joint 3D robotic Arm
3- Create a 2 link serial manipulator using denavit hartenberg Notation and apply forward kinematics on it.

30

30

TASK 1

Do it now

Create a 6-link manipulator with all revolute joints

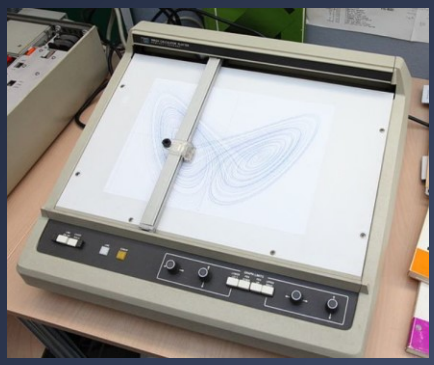
31

31

TASK 2

Do it now

Simulate this robot! Create a cartesian robot with joints which are all prismatic



32

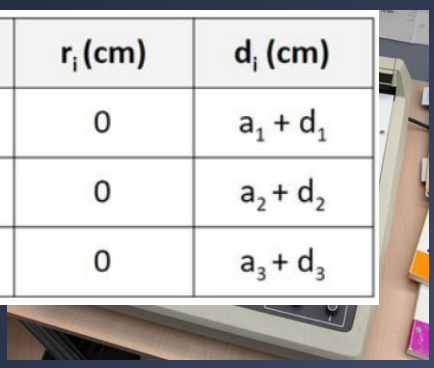
32

TASK 2

Do it now

Simulate this robot! Cr ro wi prismatic

Joint i	θ_i (deg)	α_i (deg)	r_i (cm)	d_i (cm)
1	90	90	0	$a_1 + d_1$
2	90	-90	0	$a_2 + d_2$
3	0	0	0	$a_3 + d_3$



33

33

TASK 3

Do it now

Simulate SCARA robot!
2 revolute and 1 prismatic

Joint i	θ_i (deg)	α_i (deg)	r_i (cm)	d_i (cm)
1	θ_1	0	a_2	a_1
2	θ_2	180	a_4	a_3
3	0	0	0	$a_5 + d_3$

34

34

HOME TASK

Complete Task 1, 2 and 3. Submit a hard copy of code (either hand-written or printed) before next class, i.e on Monday 30th September 2024 at 2:30 PM

35

35

ASSIGNMENT – MORE DISCUSSION

Schematics
DH Table
Forward Kinematics
Plot workspace
Inverse Kinematics

36

ASSIGNMENT 1 – MORE DISCUSSION

- **Goal:** Design a robotic manipulator.
- **Work-space**: Half doughnut
- It is not hollow.
- Robot should not have redundant joints
- Deliverables: Code file, Report (which should also contain detailed output plots)
- Code should be well commented, modular (use functions), generic (do not hard code everything). More details in the assignment document.
- Deadline: 7th October, Monday

37

37