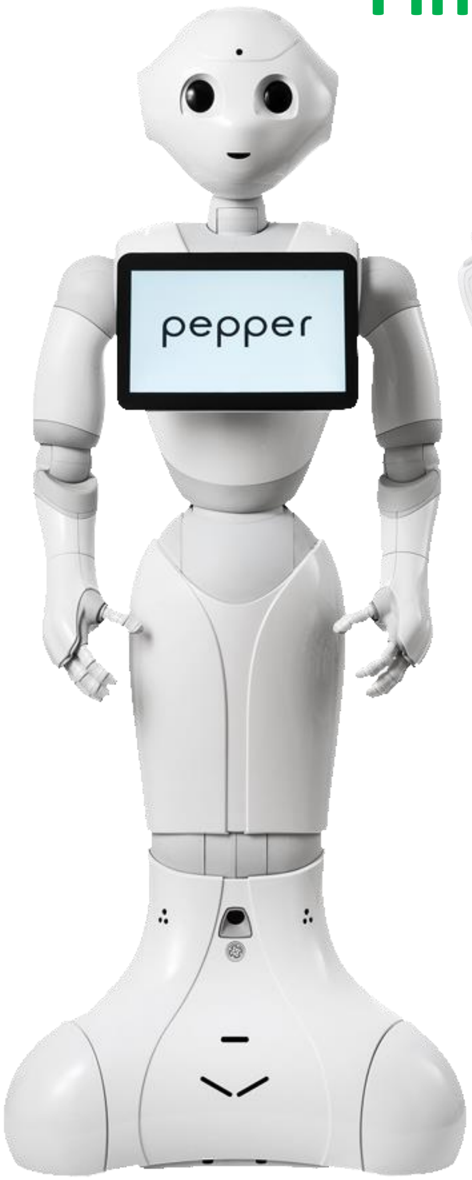
Fundamentals of Robotics

Robot Manipulators 05

Finding Orientation of a robot



Description of position and orientation of an end-effector

Displacement vector

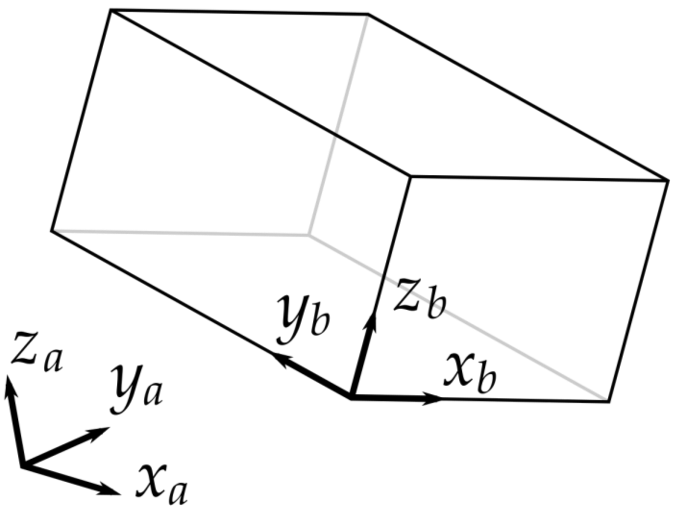
Displacement vector examples

Homogenous transformation matrix

Denavit-Hartenberg parameter table for Homogeneous transformation matrix

Description of Position and Orientation of an End-effector

- R_n^m, expresses the rotation of frame n relative to frame m.
- The rotation matrix is a 3x3 matrix that tells us the projection of x, y, z-axes in the frame n on x, y, z-axes of frame m.
- We use rotation matrix to figure out the rotation of the end-effector frame relative to the base frame in a robot manipulator.
- However, the rotation is not the only thing that changes when an end-effector moves.



Displacement Vector

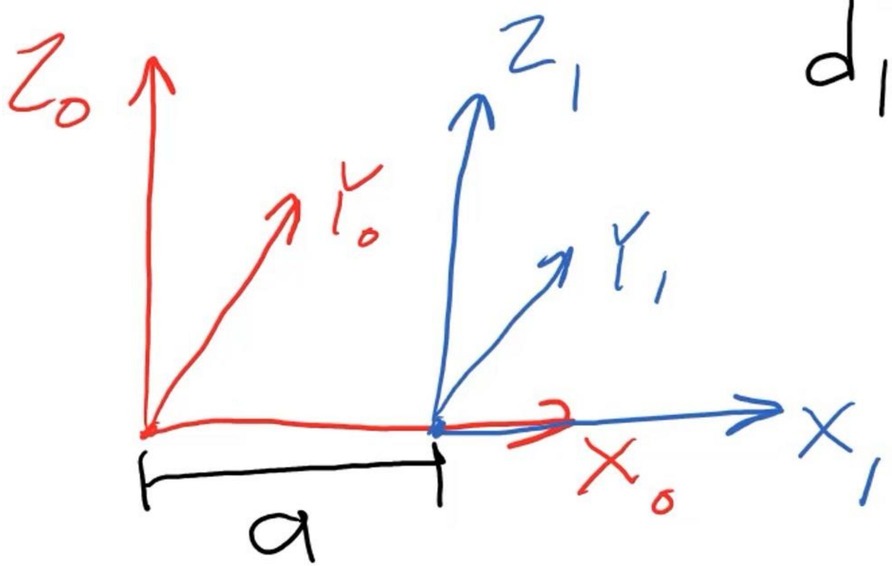
- The displacement vector expresses the change in the position of an end-effector.
- A displacement vector d_n^m is used to show the position change of frame n in frame m.

$$d_n^m = \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix} \begin{array}{l} \leftarrow x_n^m \\ \leftarrow y_n^m \\ \leftarrow z_n^m \end{array}$$

Displacement Vector

$$d_n^m = \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix} \begin{array}{l} \leftarrow X_n^m \\ \leftarrow Y_n^m \\ \leftarrow Z_n^m \end{array}$$

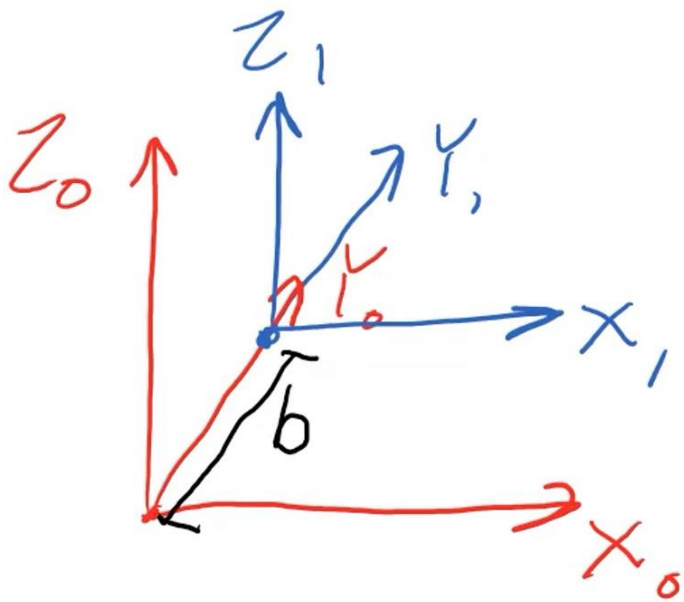
$$d_1^0 = \begin{bmatrix} a \\ 0 \\ 0 \end{bmatrix}$$



Displacement Vector

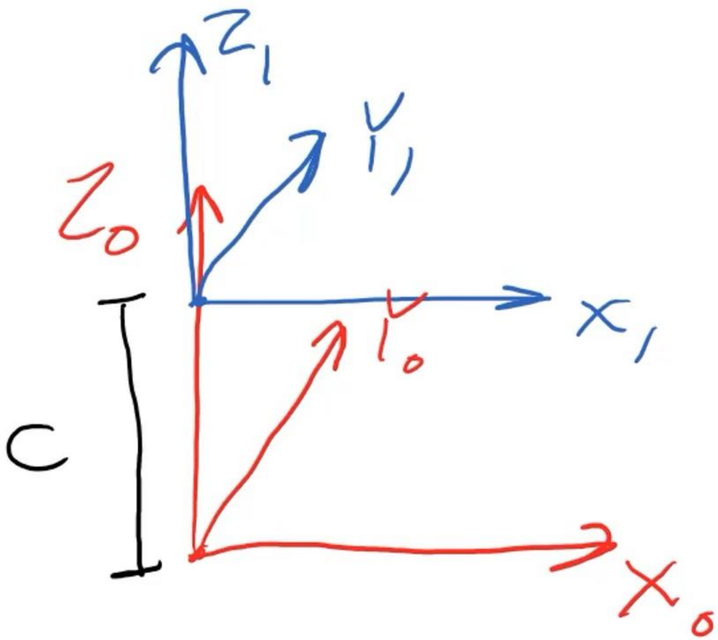
$$d_n^m = \begin{array}{c} \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix} \end{array} \begin{array}{l} \leftarrow X_n^m \\ \leftarrow Y_n^m \\ \leftarrow Z_n^m \end{array}$$

$$d_1^0 = \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix}$$



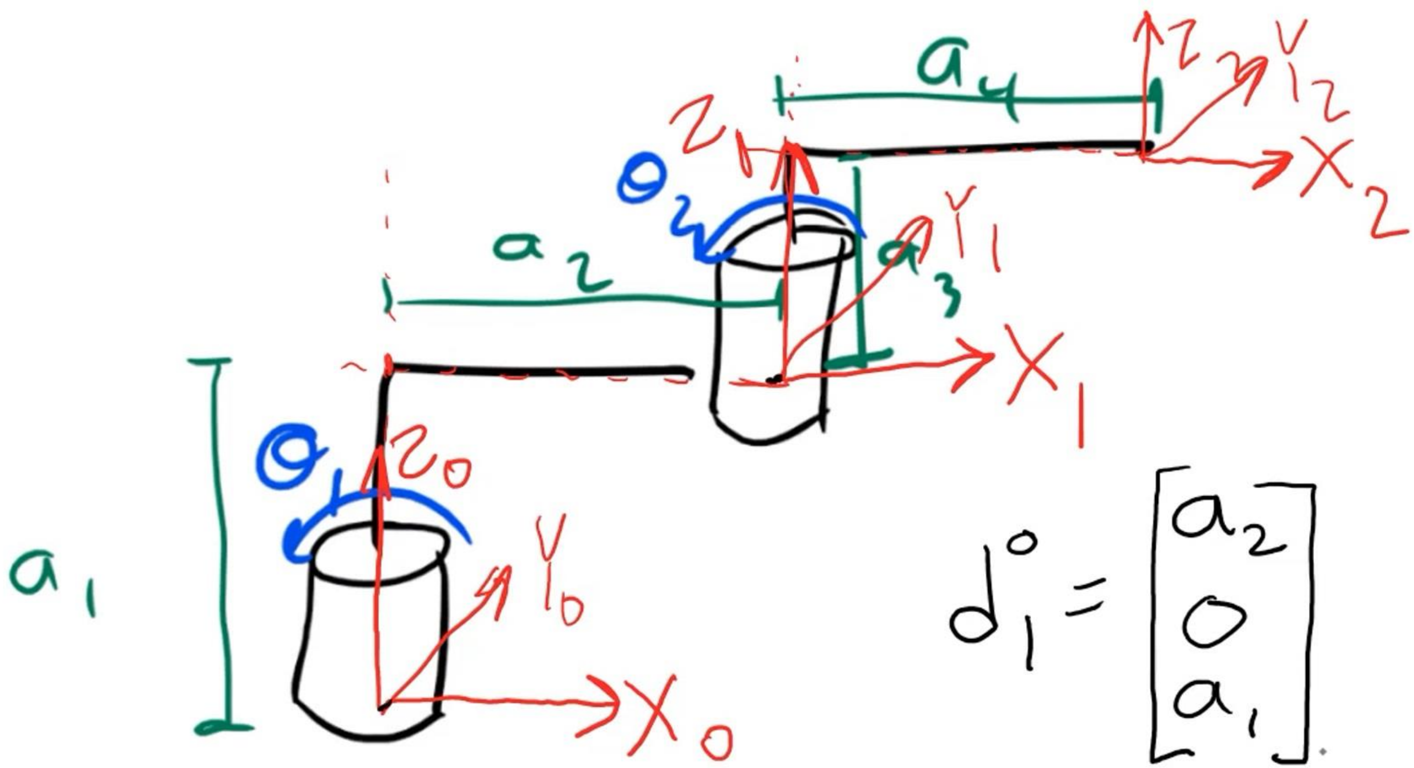
Displacement Vector

$$d_n^m = \left[\begin{array}{c} \square \\ \square \\ \square \end{array}\right] \begin{array}{l} \leftarrow X_n^m \\ \leftarrow Y_n^m \\ \leftarrow Z_n^m \end{array}$$



$$d_1^0 = \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix}$$

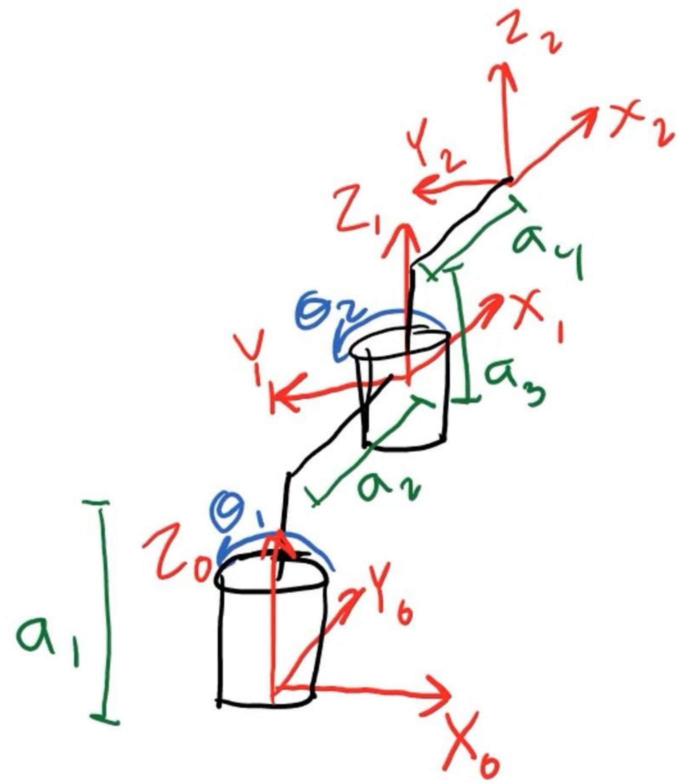
Displacement Vector for 2R manipulator



$$d_1^0 = \begin{bmatrix} a_2 \\ 0 \\ a_1 \end{bmatrix}$$

Displacement Vector for 2R manipulator

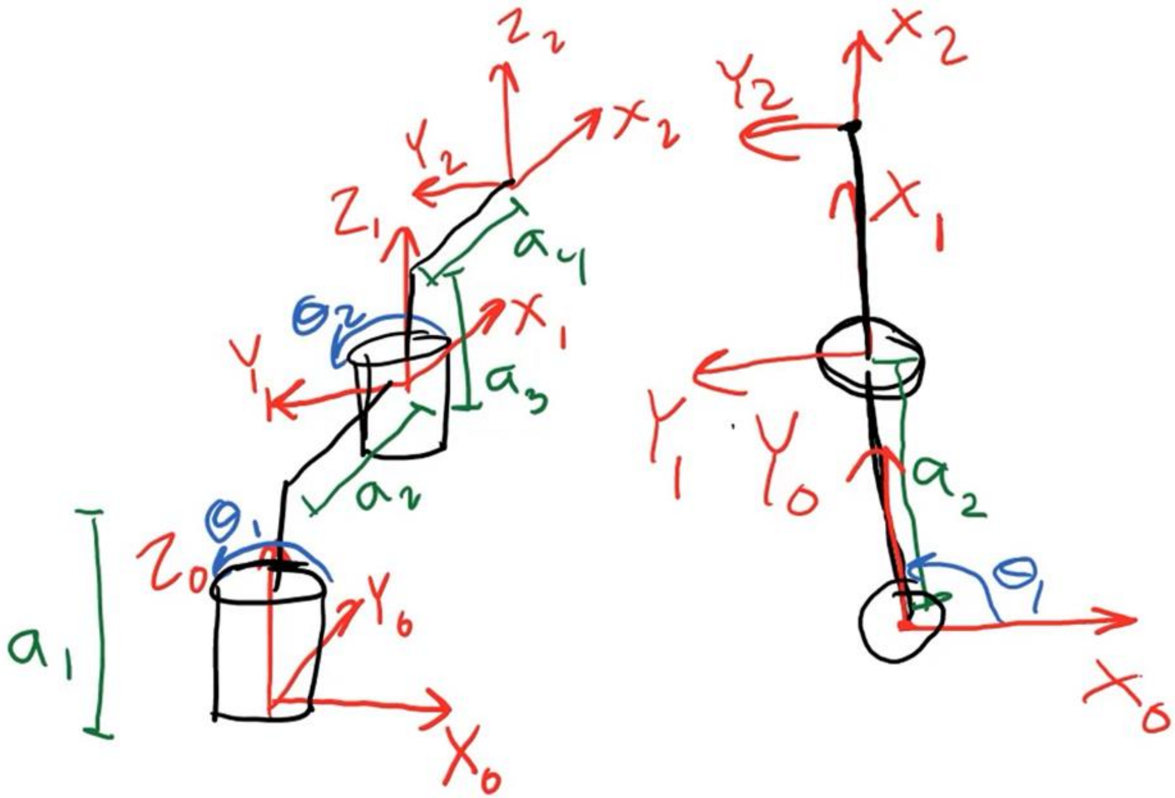
- With the change in the joint angle the displacement vector is also changed.
- There is need to create a displacement vector that is valid for all joint angles.



$$d_1^0 = \begin{bmatrix} 0 \\ a_2 \\ a_1 \end{bmatrix}$$

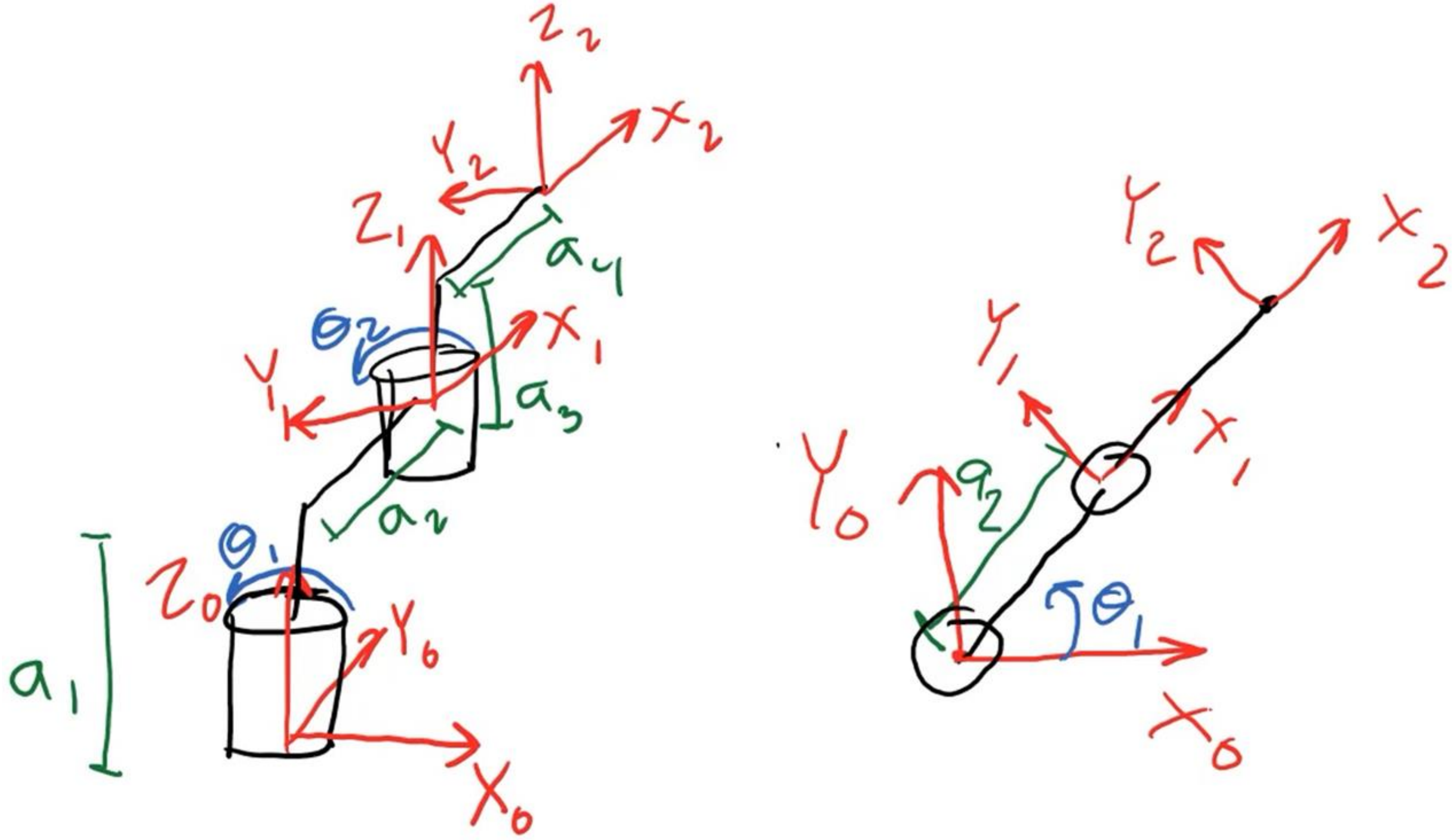
Displacement Vector for 2R manipulator

- Let's, create a top-view diagram from the kinematic diagram.
- Below is top-view diagram when joint angle is 90 degree.



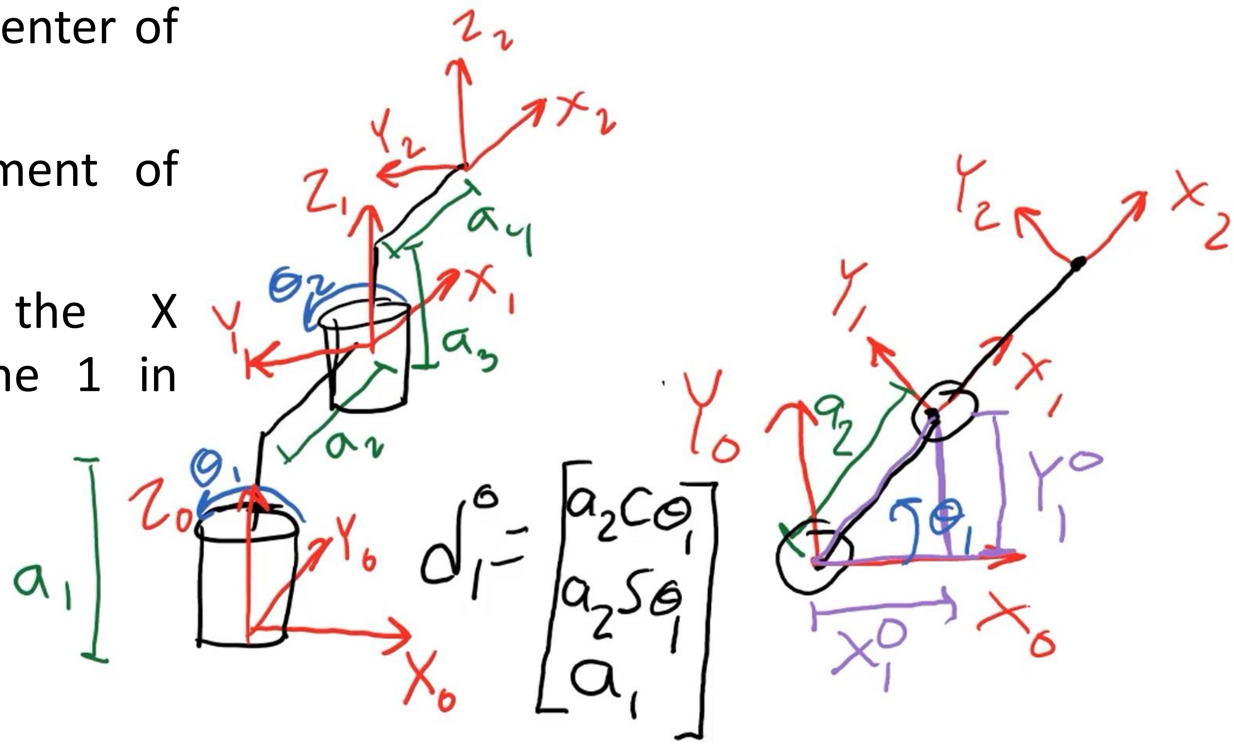
Displacement Vector for 2R manipulator

- Here, the joint angle is approximately 45 degree.



Displacement Vector for 2R manipulator

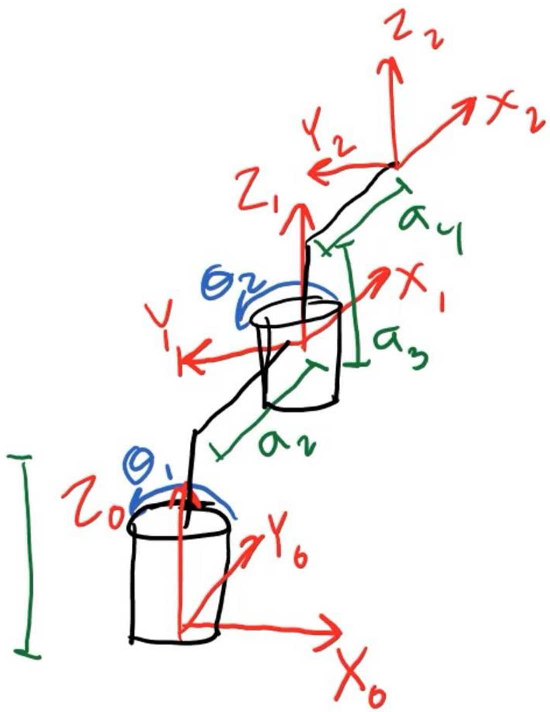
- We want to find out the displacement vector from the center of frame 0 to center of frame 1.
- Y_1^0 is the Y displacement of frame 1 in frame 0.
- Similarly, X_1^0 is the X displacement of frame 1 in frame 0.

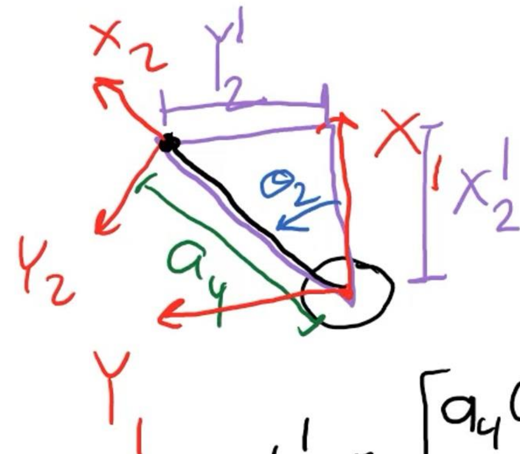


$$d_1^0 = \begin{bmatrix} a_2 C\theta_1 \\ a_2 S\theta_1 \\ a_1 \end{bmatrix}$$

Displacement Vector for 2R manipulator

- Finding the displacement vector d_2^1. We create 45^o angle for Θ_2.
- We draw top view diagram between frame 1 and 2 and ignore rest of the frames.

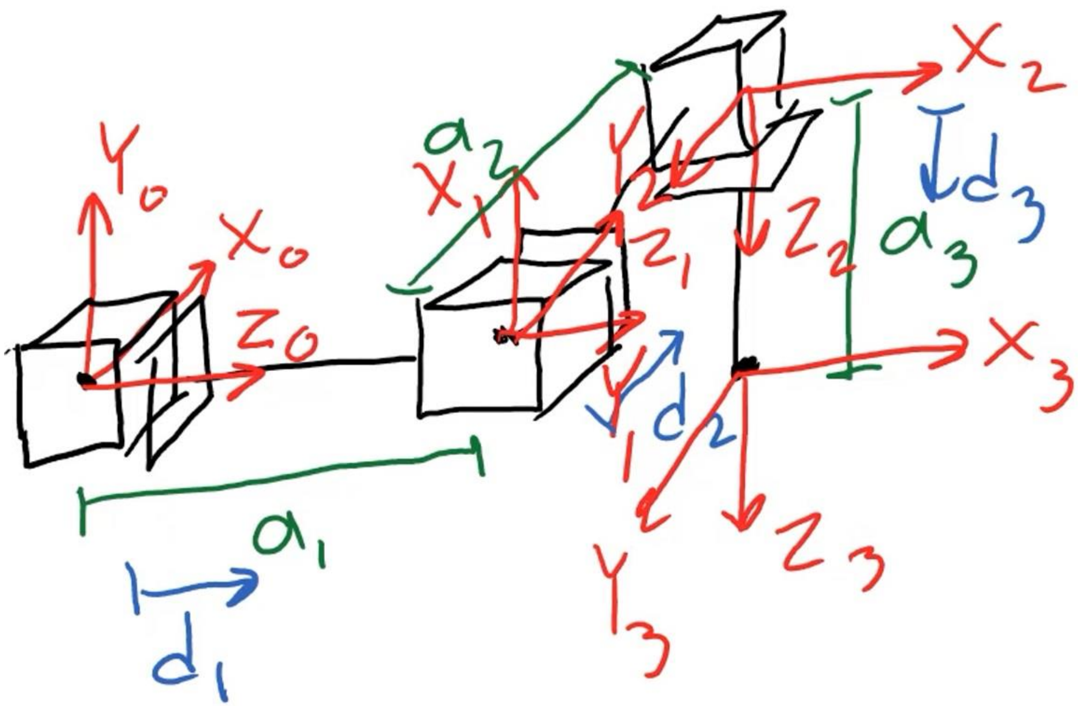




$$d_2^1 = \begin{bmatrix} a_4 C\theta_2 \\ a_4 S\theta_2 \\ a_3 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Cartesian)

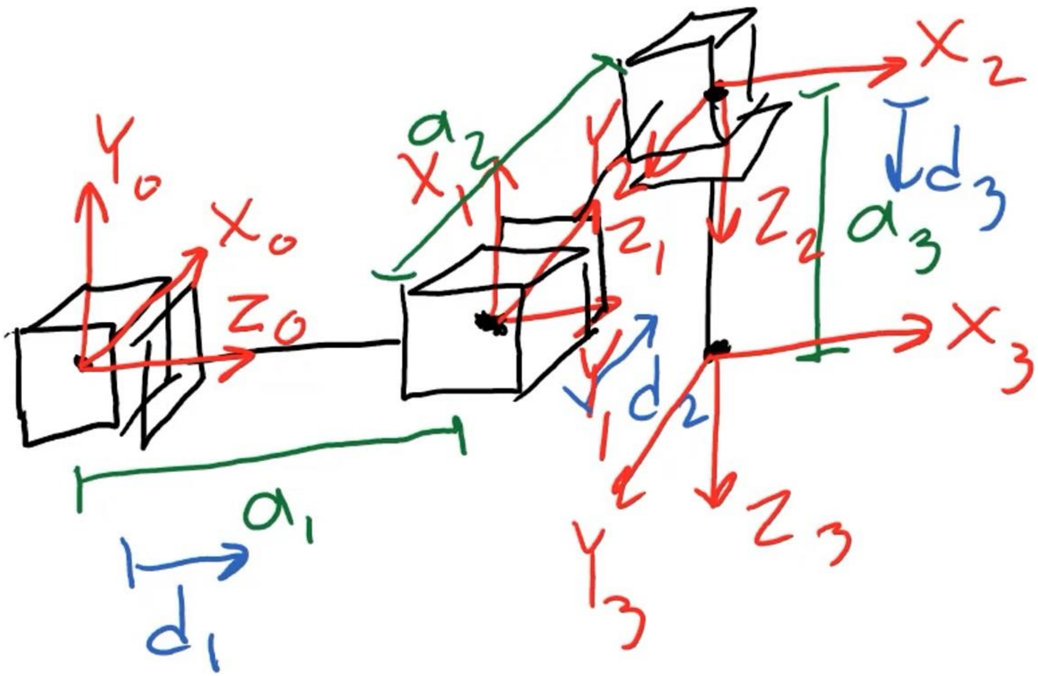
- To find d_2^0 we cannot use $d_2^0 \neq d_1^0 \mathrm{x} d_2^1$ or $d_2^0 \neq d_1^0 + d_2^1$
- Frame 01 is translating in z-axis direction plus addition joint displacement.
- There are not rotations in such manipulator type.



$$d_1^0 = \begin{bmatrix} 0 \\ 0 \\ a_1 + d_1 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators

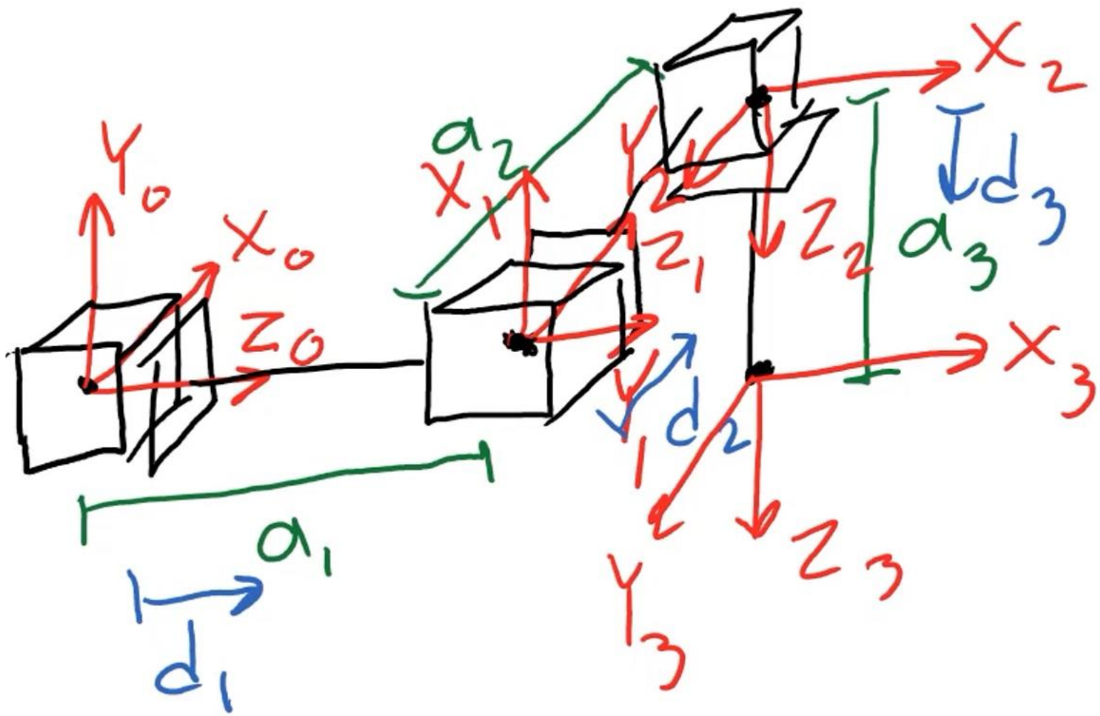
- Frame 02 is translating in z-axis direction plus the displacement.



$$d^1_2 = \begin{bmatrix} 0 \\ 0 \\ a_2 + d_2 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators

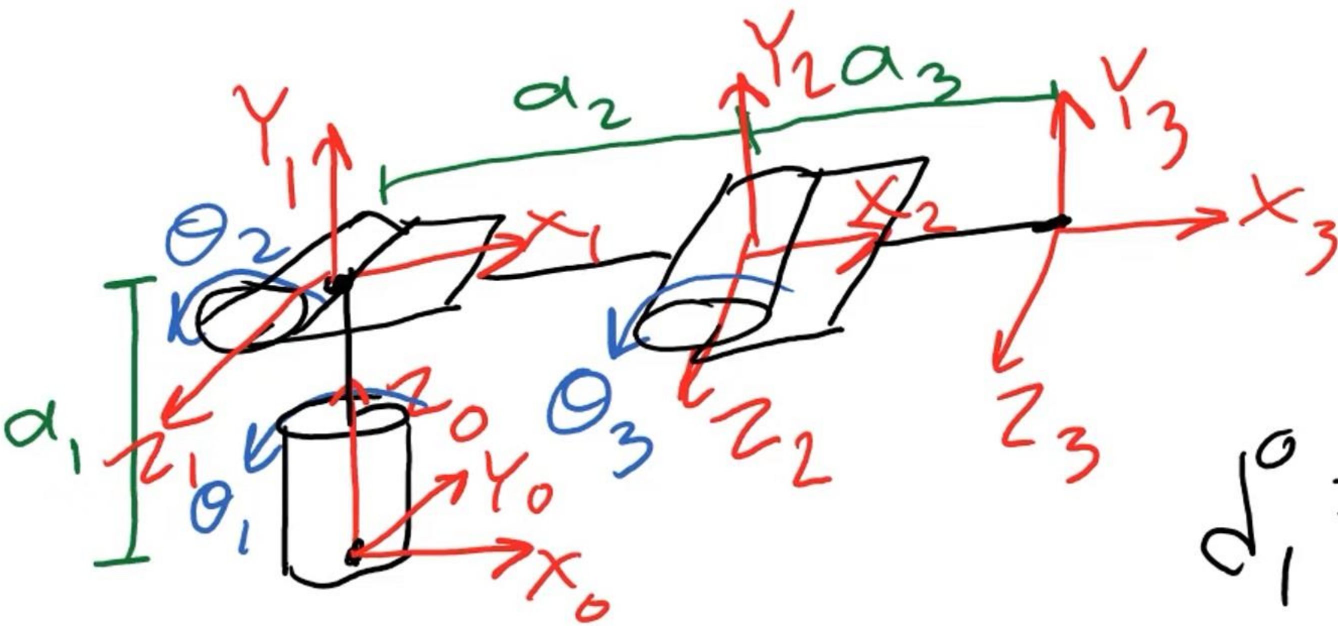
- The end-effector is only translating in z-axis.



$$d^2_3 = \begin{bmatrix} 0 \\ 0 \\ a_3 + d_3 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Articulated)

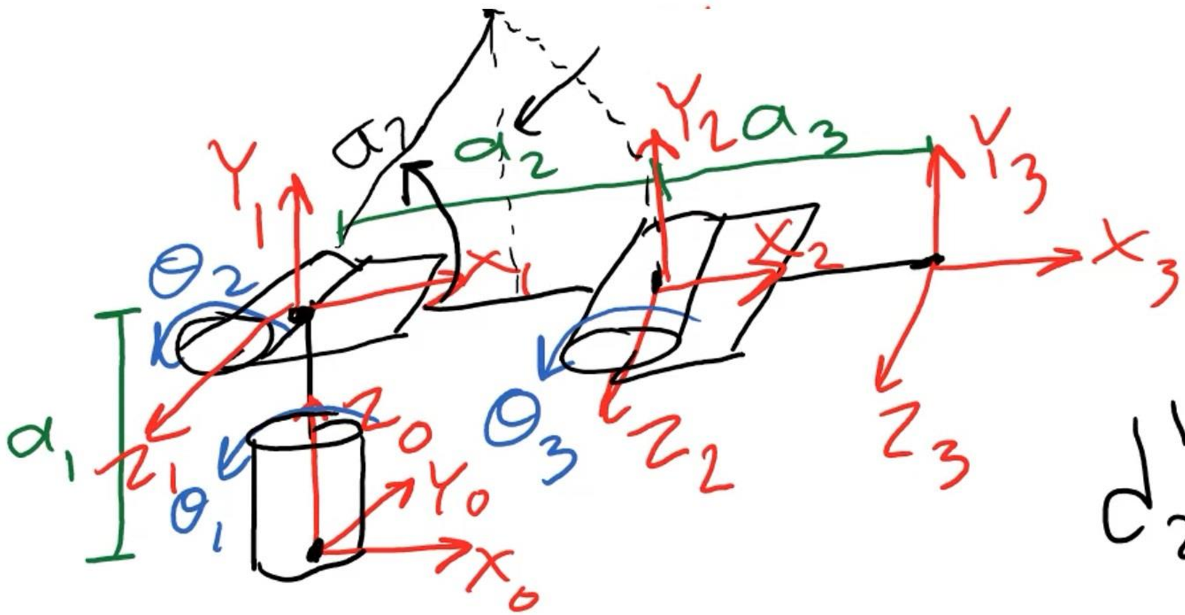
- Displacement vector for articulated manipulator.
- Displacement along z-axis and this vector is valid with any joint angle.
- The position of frame 1 is fixed.



$$d_1^0 = \begin{bmatrix} 0 \\ 0 \\ a_1 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Articulated)

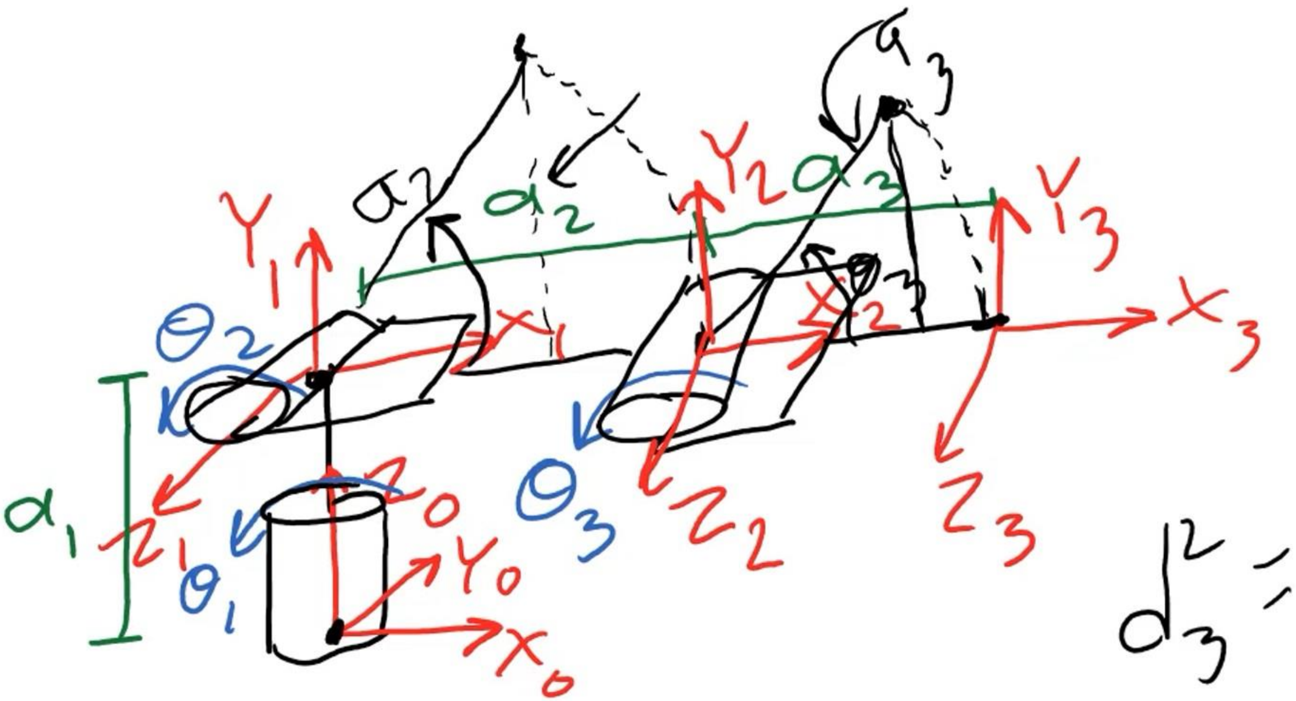
- Frame 2 has displacement along x and y-axis.
- We again 45^o angle for Θ_2.



$$d^1_2 = \begin{bmatrix} a_2 C\theta_2 \\ a_2 S\theta_2 \\ 0 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Articulated)

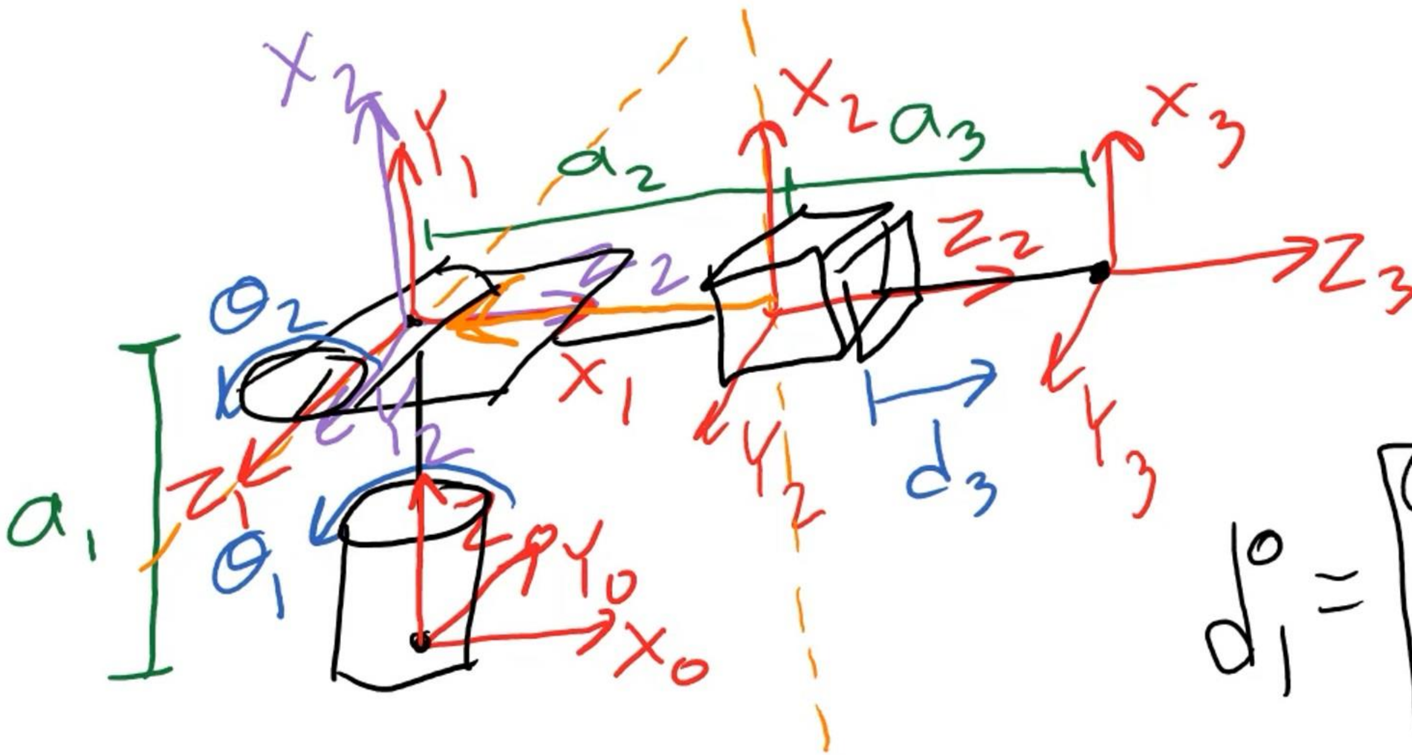
- Displacement along two axes. We again 45° angle for Θ_3.



$$d_3^2 = \begin{bmatrix} a_3 C\theta_3 \\ a_3 S\theta_3 \\ 0 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Spherical)

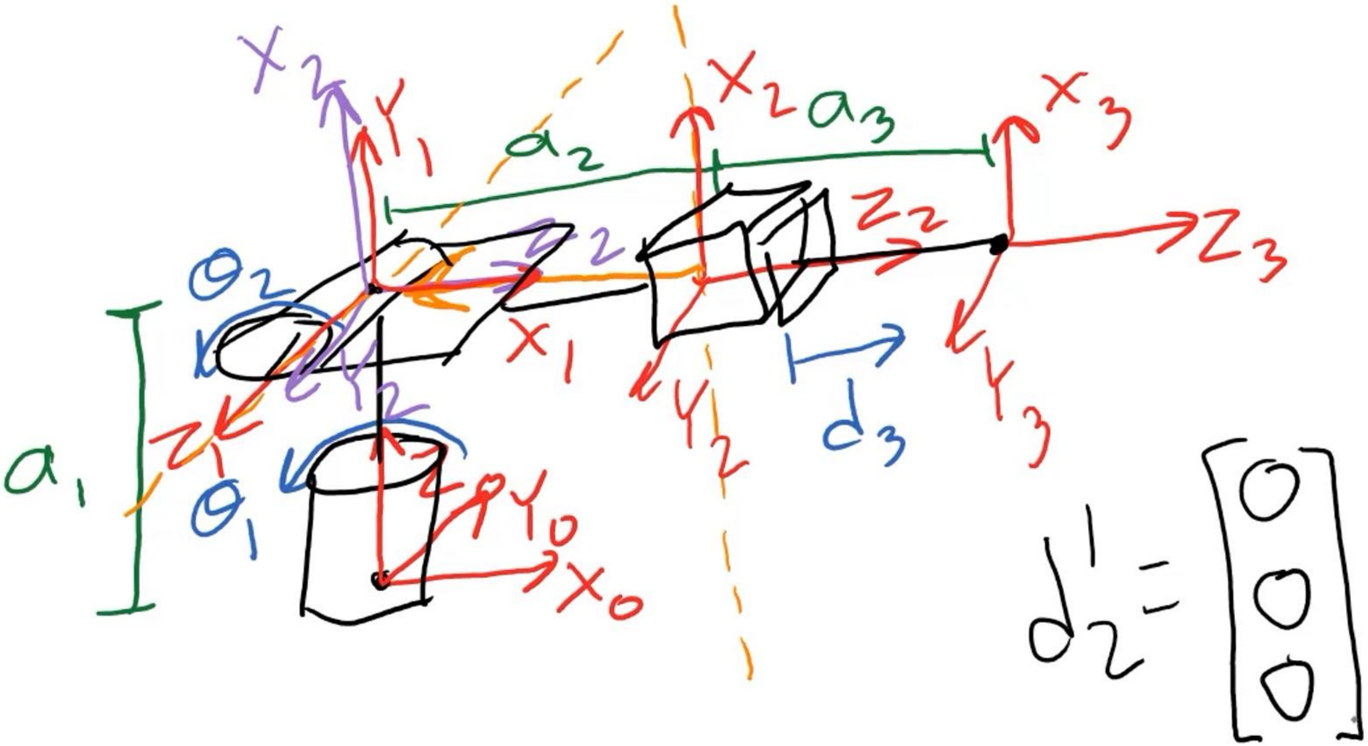
- Corrected kinematic will be for creating displacement vector.



$$d_1^0 = \begin{bmatrix} 0 \\ 0 \\ a_1 \end{bmatrix}$$

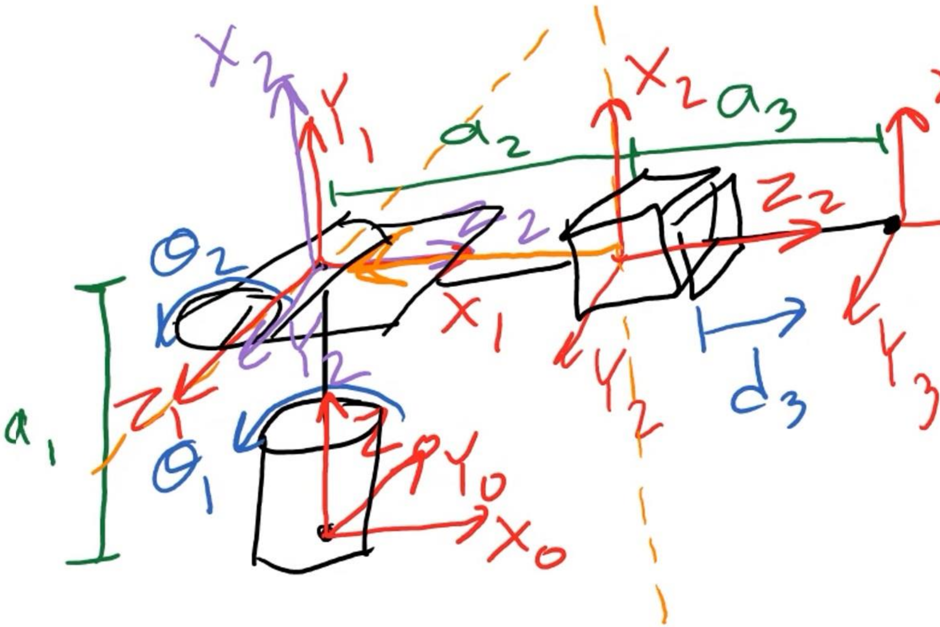
Displacement Vector in 3 DoF manipulators (Spherical)

- No displacement along any axis as both frame are on same point.



Displacement Vector in 3 DoF manipulators (Spherical)

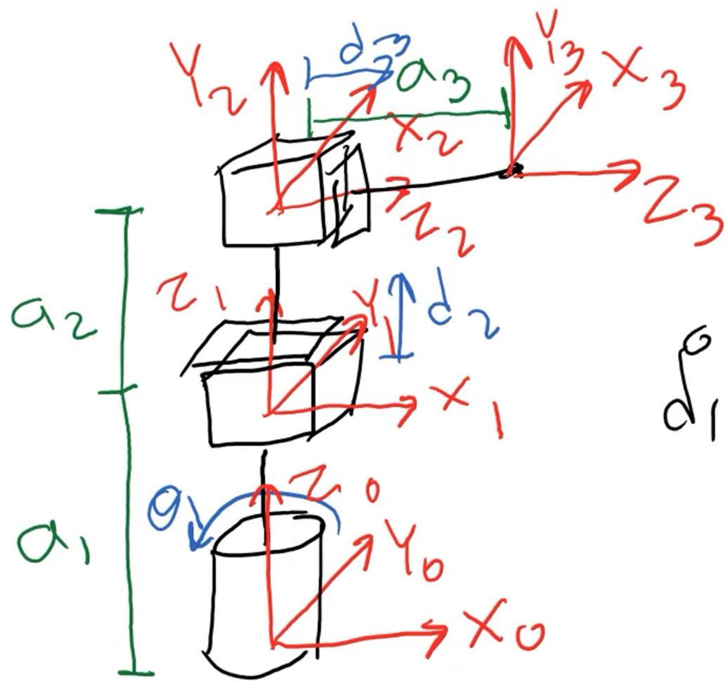
- Displacement along z-axis that included a_2 and a_3 links.



$$d_3^2 = \begin{bmatrix} 0 \\ 0 \\ a_2 + a_3 + d_3 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Cylinderical)

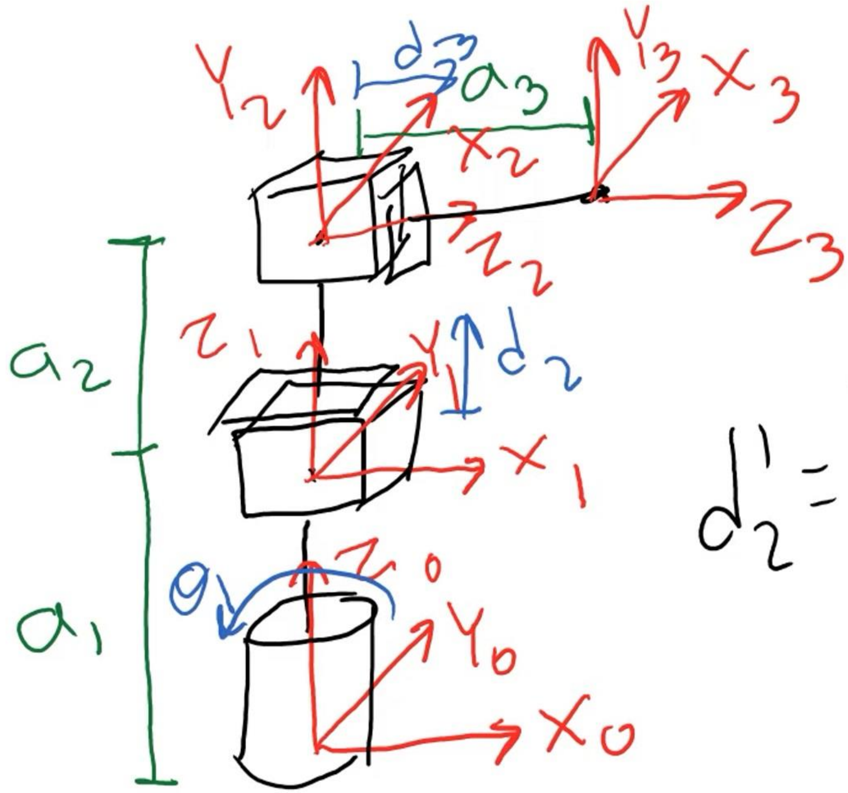
- Cylindrical manipulator
- Displacement along z-axis



$$d_1^0 = \begin{bmatrix} 0 \\ 0 \\ a_1 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Cylinderical)

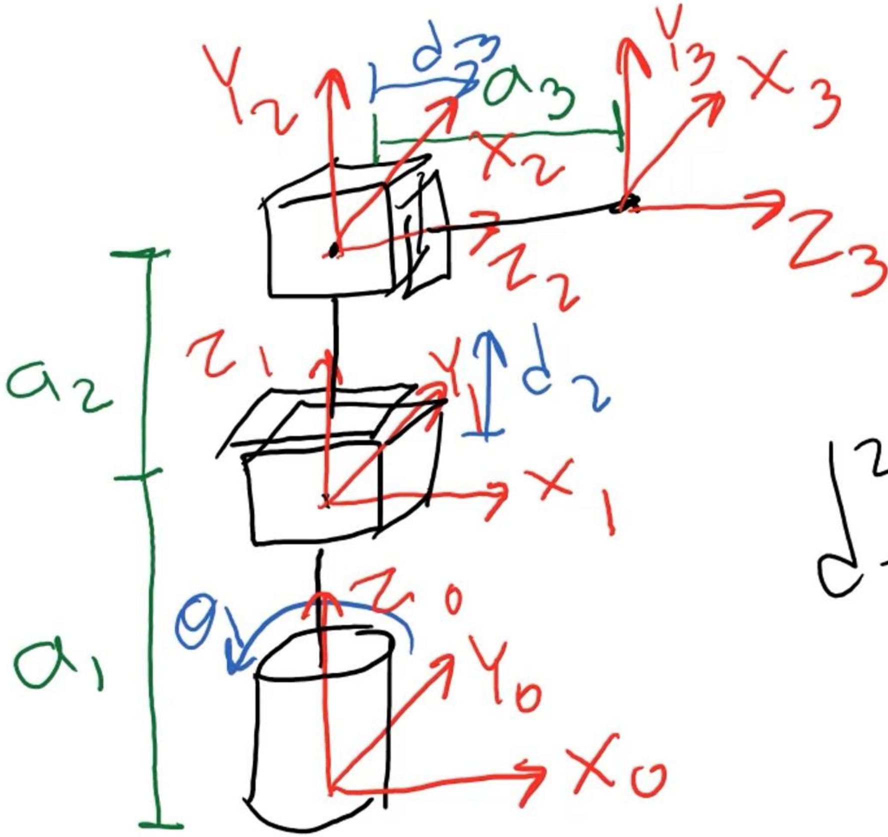
- Z-axis displacement contains a_2.



$$d_2^1 = \begin{bmatrix} 0 \\ 0 \\ a_2 + d_2 \end{bmatrix}$$

Displacement Vector in 3 DoF manipulators (Cylinderical)

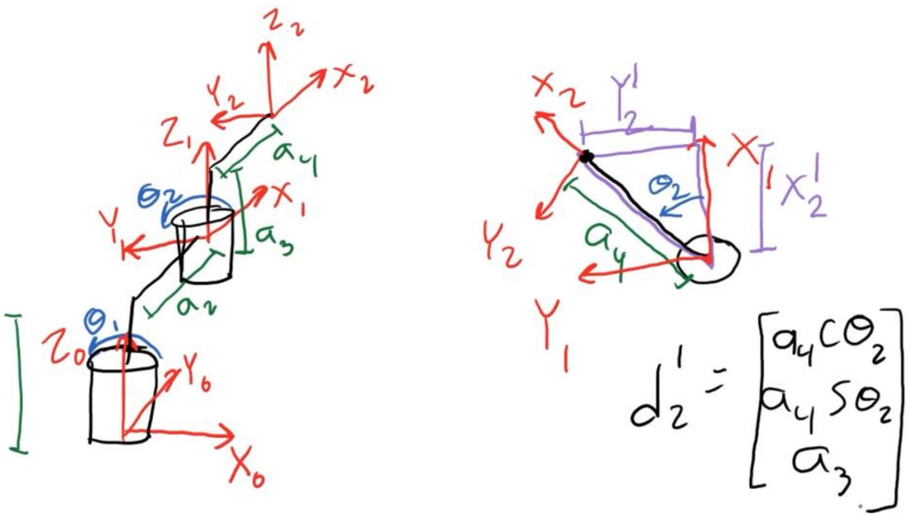
- Frame 3 displacement vector



$$d_3^2 = \begin{bmatrix} 0 \\ 0 \\ a_3 + d_3 \end{bmatrix}$$

Homogeneous transformation matrix

- To find d_2^0 we cannot use $d_2^0 \neq d_1^0 \mathrm{x} d_2^1$ or $d_2^0 \neq d_1^0 + d_2^1$.
- The displacement vector is combined with rotation matrix to form homogeneous transformation matrix.
- Using homogeneous transformation matrix, we can find d_2^0.
- These matrices can be combined by multiplication the same way rotation matrices can, allowing us to find the position of the end-effector in the base frame.



Homogeneous transformation matrix

- Homogeneous transformation matrix of frame m in frame n.
- We pad 0 0 0 1 as fourth row to make two homogenous matrices compatible for multiplication.

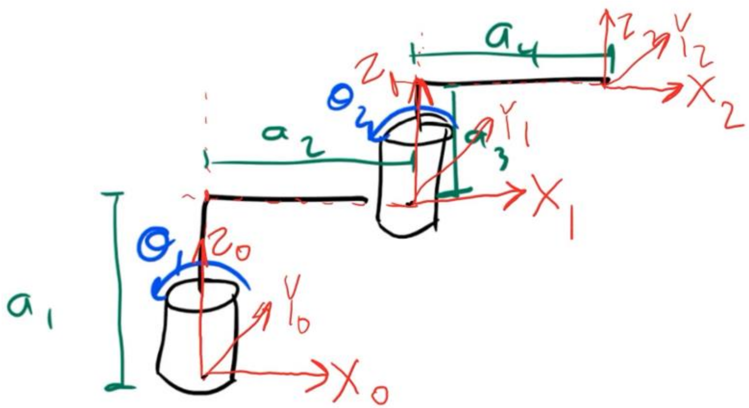
$$H_n^m = \begin{bmatrix} R_n^m & d_n^m \\ 0\ 0\ 0 & 1 \end{bmatrix}$$

Homogeneous transformation matrix for 2R manipulator

- Calculating H_1^0

$$R_1^0 = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad d_1^0 = \begin{bmatrix} a_2 cos\theta_1 \\ a_2 sin\theta_1 \\ a_1 \end{bmatrix}$$

$$H_1^0 = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 & a_2 cos\theta_1 \\ sin\theta_1 & cos\theta_1 & 0 & a_2 sin\theta_1 \\ 0 & 0 & 1 & a_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

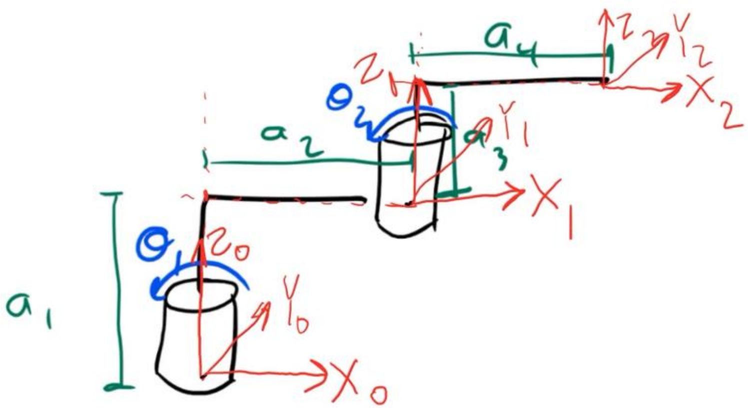


Homogeneous transformation matrix for 2R manipulator

- Calculating H_2^1

$$R_2^1 = \begin{bmatrix} cos\theta_2 & -sin\theta_2 & 0 \\ sin\theta_2 & cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad d_2^1 = \begin{bmatrix} a_4 cos\theta_2 \\ a_4 sin\theta_2 \\ a_3 \end{bmatrix}$$

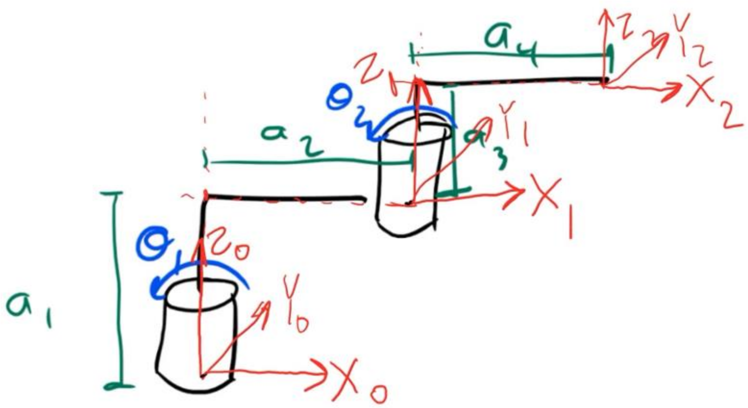
$$H_2^1 = \begin{bmatrix} cos\theta_2 & -sin\theta_2 & 0 & a_4 cos\theta_2 \\ sin\theta_2 & cos\theta_2 & 0 & a_4 cos\theta_2 \\ 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Homogeneous transformation matrix for 2R manipulator

- Calculating H_2^0

$$H_2^0 = H_1^0 H_2^1 \qquad H_2^0 = \begin{bmatrix} R_2^0 & d_2^0 \\ 0\ 0\ 0 & 1 \end{bmatrix}$$



Homogeneous transformation matrix Python code 2R manipulator

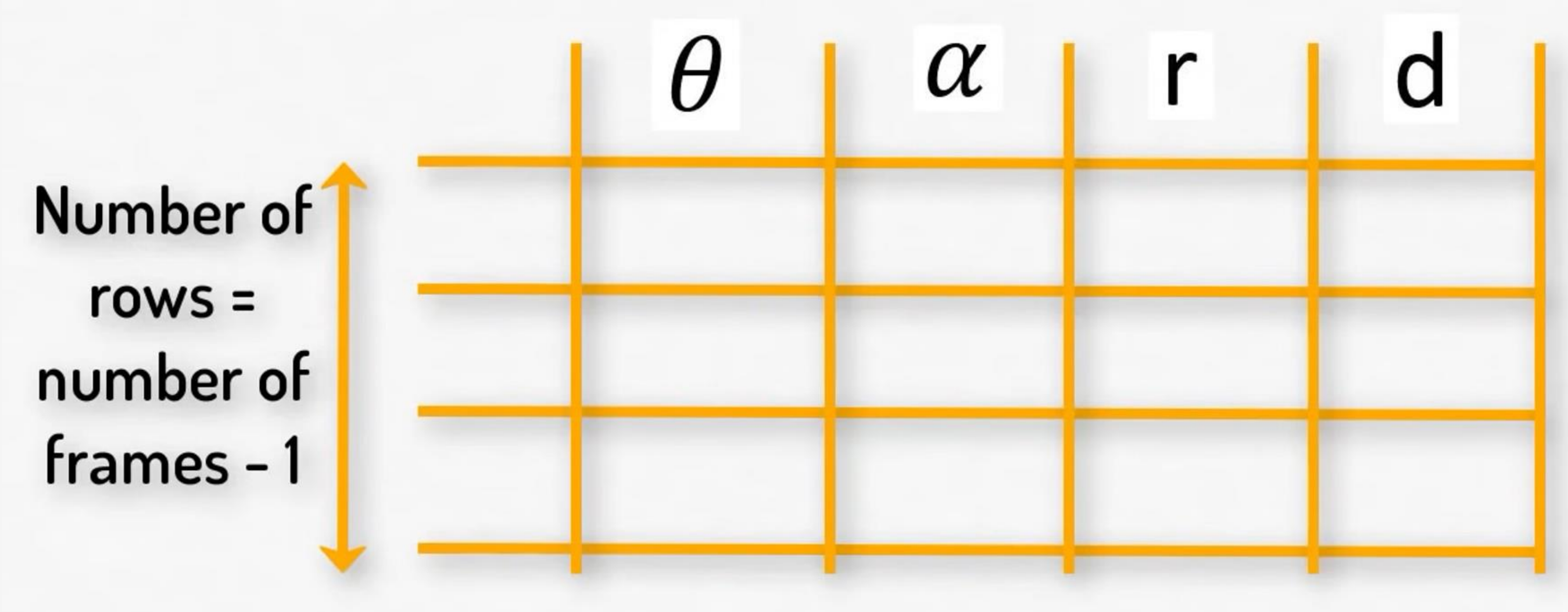
```
import numpy as np
a1 = 1
a2 = 1
a3 = 1
a4 = 1
T1 = 0
T2 = 90

R0_1 = [[np.cos(T1), -np.sin(T1), 0], [np.sin(T1), np.cos(T1), 0], [0, 0, 1]]
R1_2 = [[np.cos(T2), -np.sin(T2), 0], [np.sin(T2), np.cos(T2), 0], [0, 0, 1]]
R0_2 = np.dot(R0_1, R1_2)

d0_1 = [[a2*np.cos(T1)], [a2*np.sin(T1)], [a1]]
d1_2 = [[a4*np.cos(T2)], [a4*np.sin(T2)], [a3]]

H0_1 = np.concatenate((R0_1, d0_1), 1)
H0_1 = np.concatenate((H0_1, [[0, 0, 0, 1]]), 0)

H1_2 = np.concatenate((R1_2, d1_2), 1)
H1_2 = np.concatenate((H1_2, [[0, 0, 0, 1]]), 0)


H0_2 = np.dot(H0_1, H1_2)
print ("H0_2")
print(np.matrix(H0_2))
```

Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- Till now we have found homogenous transformation matrix by finding rotation matrix and displacement vector and assembling them in homogeneous transformation matrix.
- Denavit-Hartenberg method is a short method to find homogenous transformation matrix without finding rotation matrix and displacement vector.
  - First step is to assign frames according to 4 rule of Denavit-Hartenberg.
  - Second step is fill out the Denavit-Hartenberg parameter table.
    - The number of rows for this table are equal to number of frame minus 1.
    - The of columns for this table are fixed with ϴ, α, r and d
    - ϴ and α are rotations whereas, r and d are linear distance or displacement

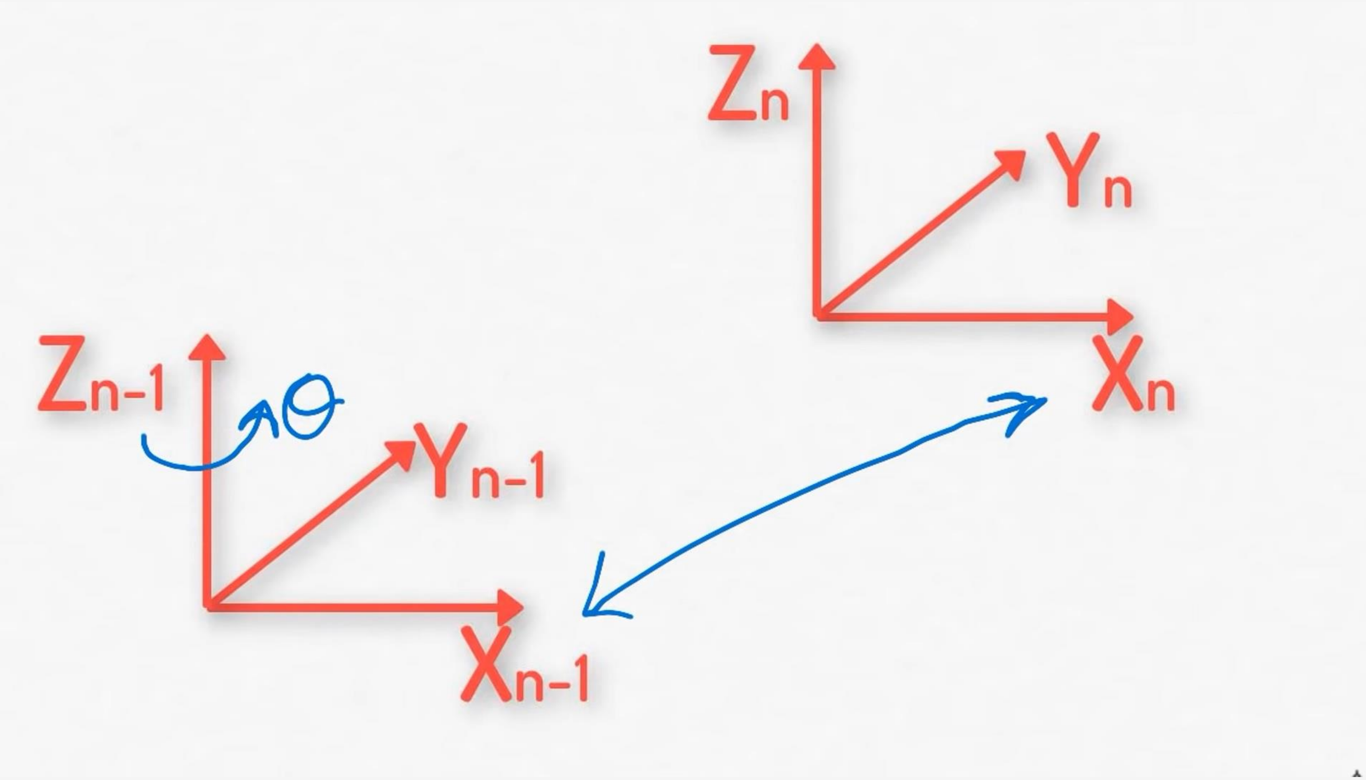
Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- Second step is fill out the Denavit-Hartenberg parameter table.
- The table is arranged in a way mentioned below:



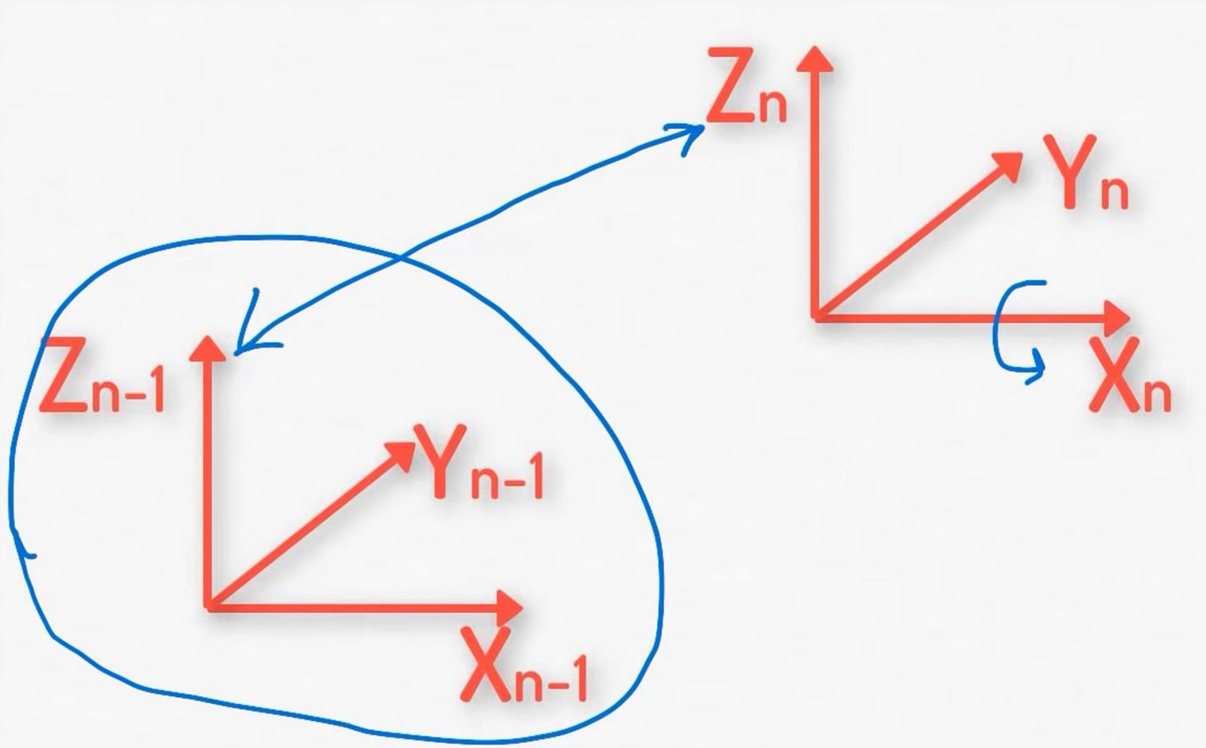
Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- ϴ is the rotation around Z_{n-1} to match X_{n-1} with X_n.



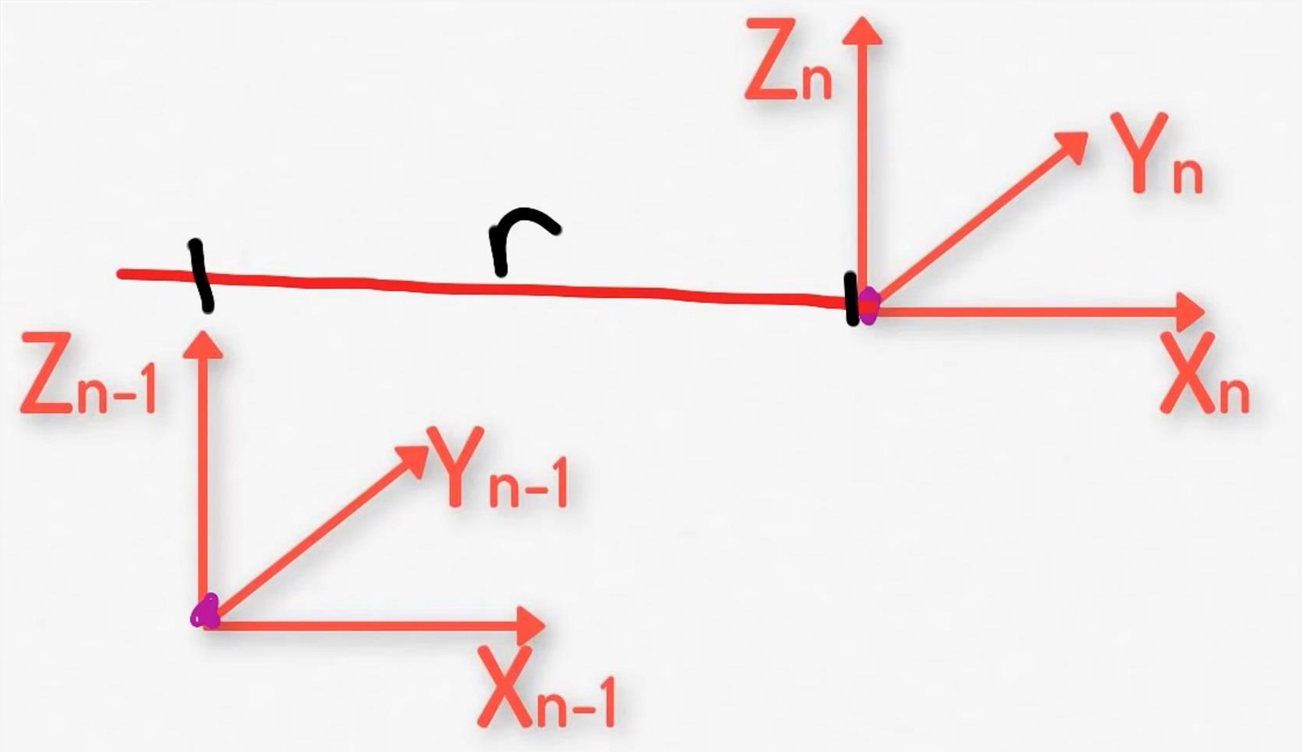
Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- α is the rotation around X_n required to match Z_{n-1} with Z_n.
- Here, the rotation will be around X_n, however, the frame n-1 will be rotated.



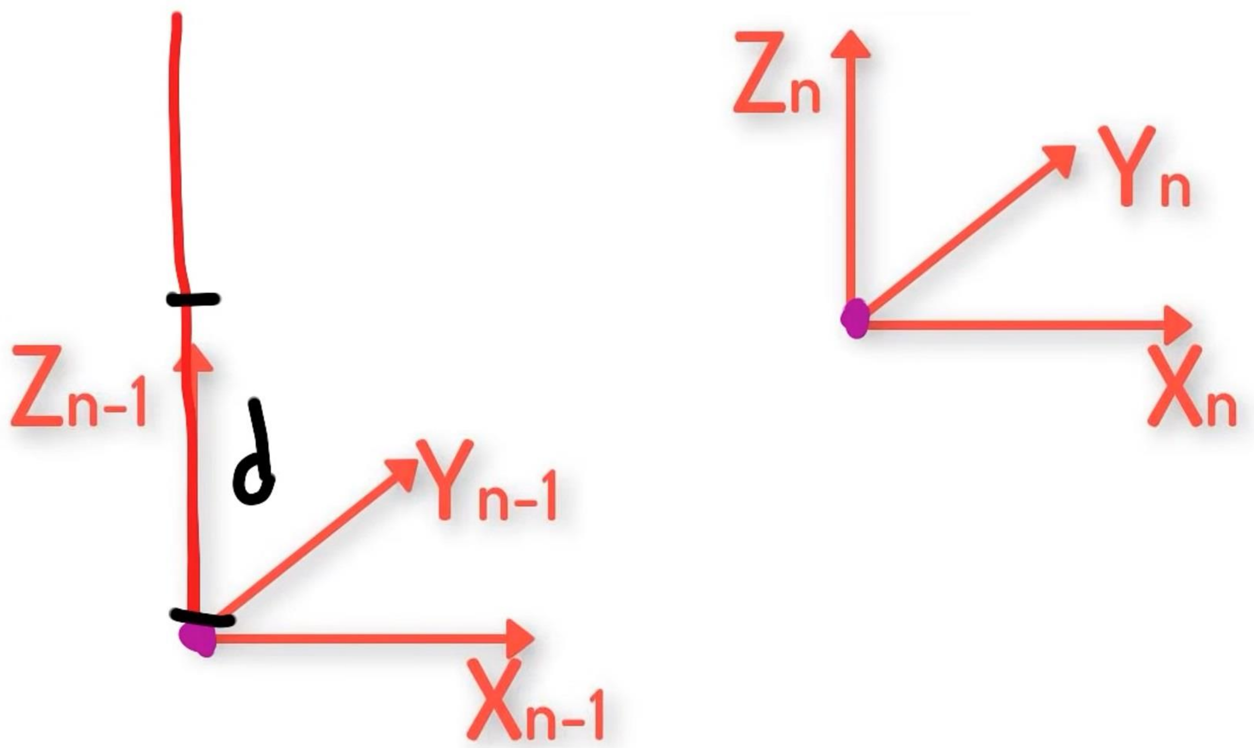
Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- *r* is distance between frame n and n-1 in the direction of X_n.



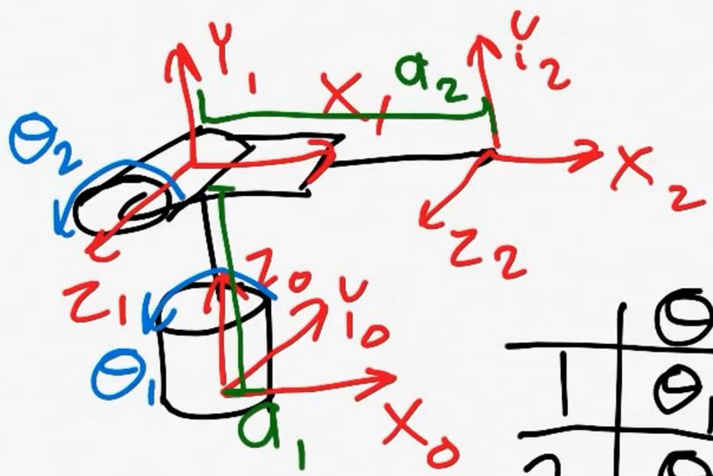
Denavit-Hartenberg Approach to Homogenous Transformation Matrix

- *d* is distance between frame n and n-1 in the direction of Z_{n-1}.



Denavit-Hartenberg Approach to Homogenous Transformation Matrix

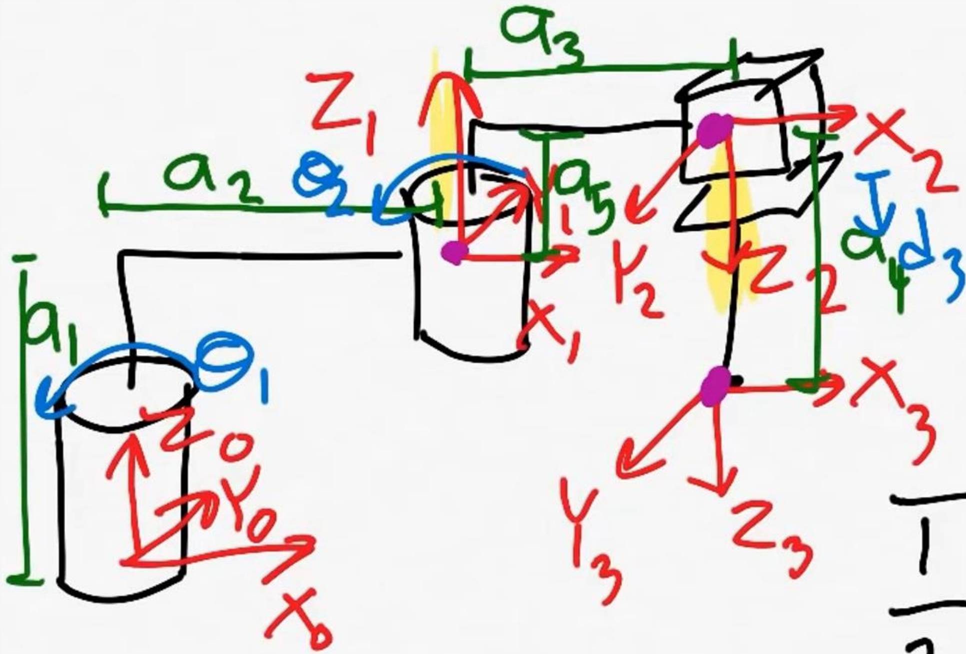
- Parameter table for 2 R manipulator
- This manipulator is like articulated manipulator, however, not complete as articulated manipulator has 3 DoF.
  - ϴ is the rotation around Z_{n-1} to match X_{n-1} with X_n.
  - α is the rotation around X_n required to match $Z\_(n-1)$ with Z_n.
    - Here, the rotation will be around X_n, however, the frame n-1 will be rotated
  - r is distance between frame n and n-1 in the direction of X_n.
  - d is distance between frame n and n-1 in the direction of Z_{n-1}.



	θ	α	r	d
1	θ_1	90	0	a_1
2	θ_2	0	a_2	0

Denavit-Hartenberg Approach to Homogenous Transformation Matrix

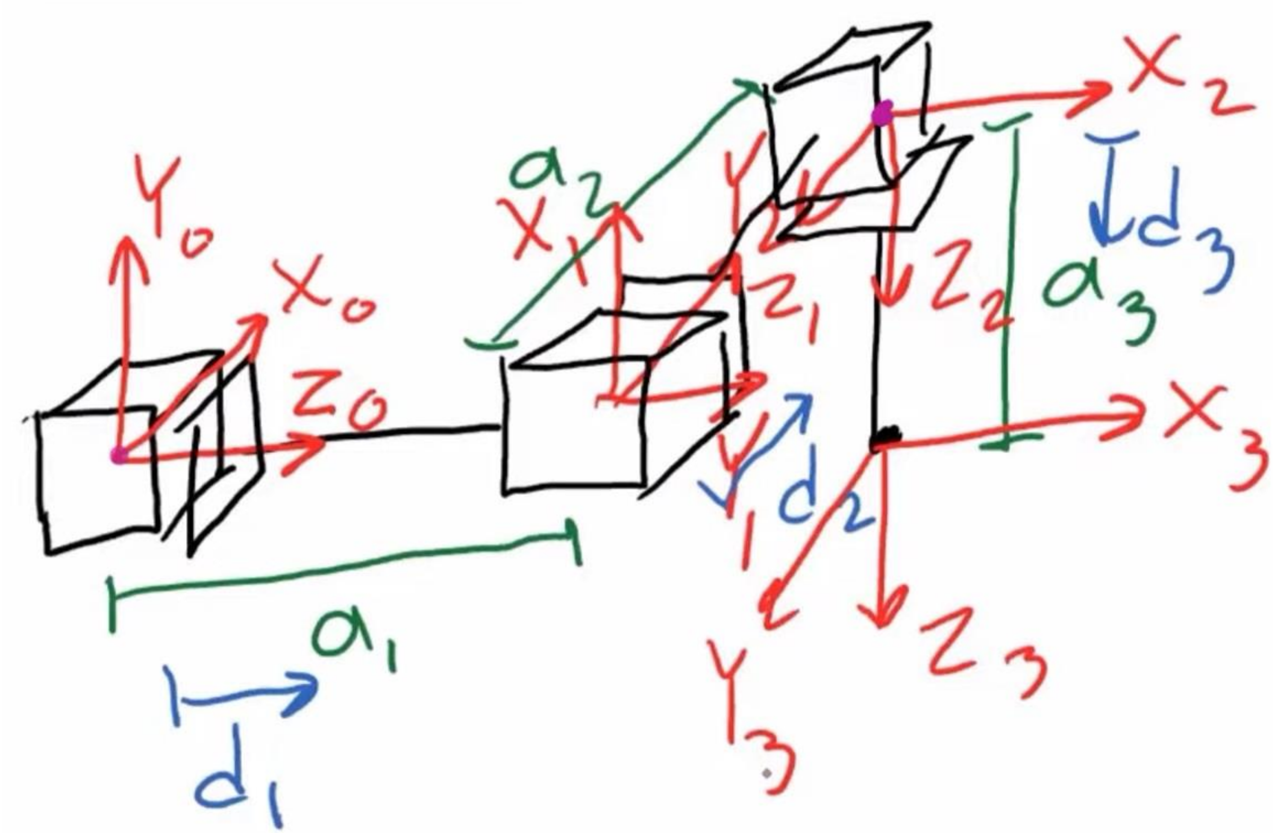
- 3 DoF manipulator parameter table.



	θ	α	r	d
1	θ_1	0	a_2	a_1
2	θ_2	180	a_3	a_5
3	0	0	0	a_4+d_3

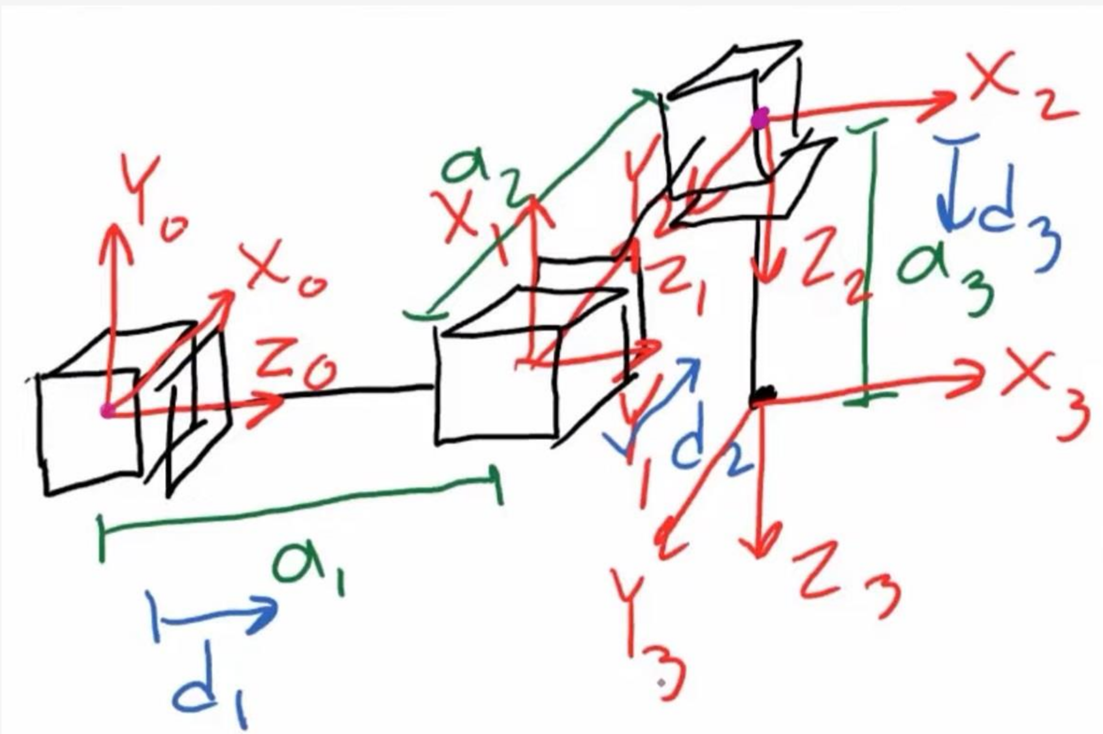
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

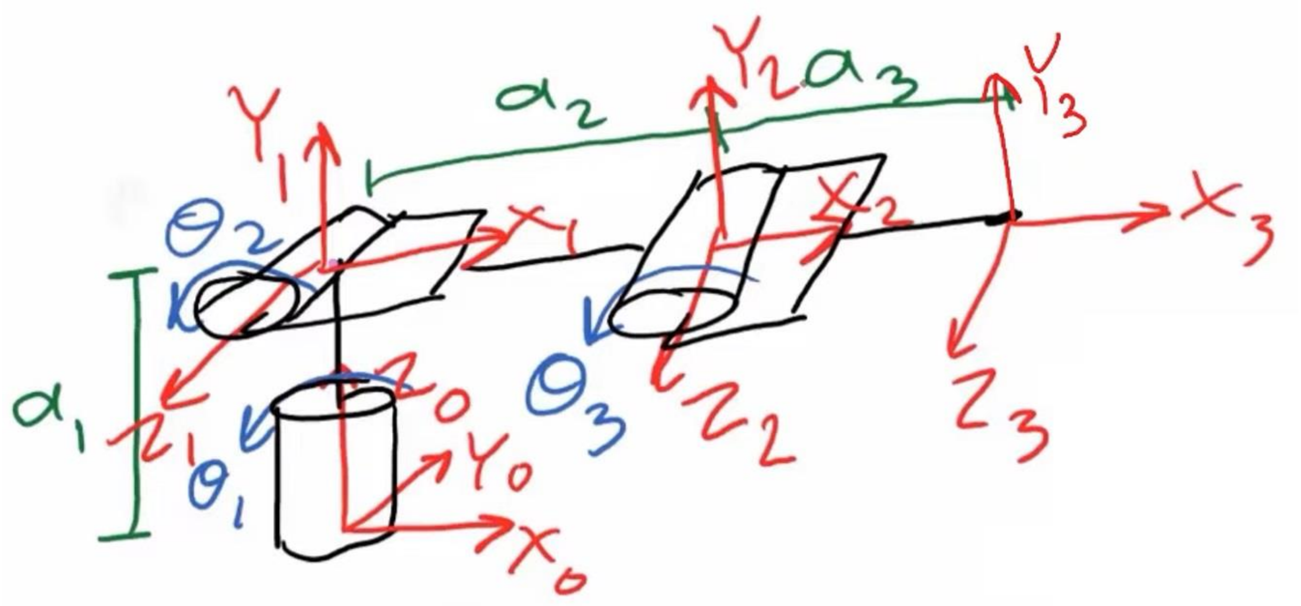
- Cartesian manipulator



	θ	α	r	d
1	90	90	0	a_1+d_1
2	90	-90	0	a_2+d_2
3	0	0	0	a_3+d_3

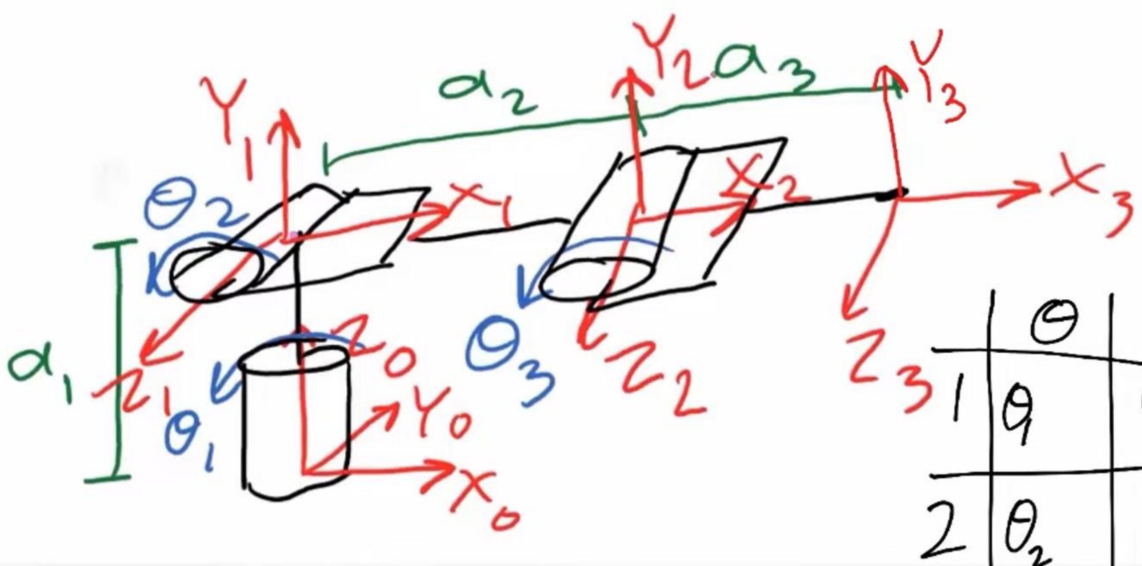
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

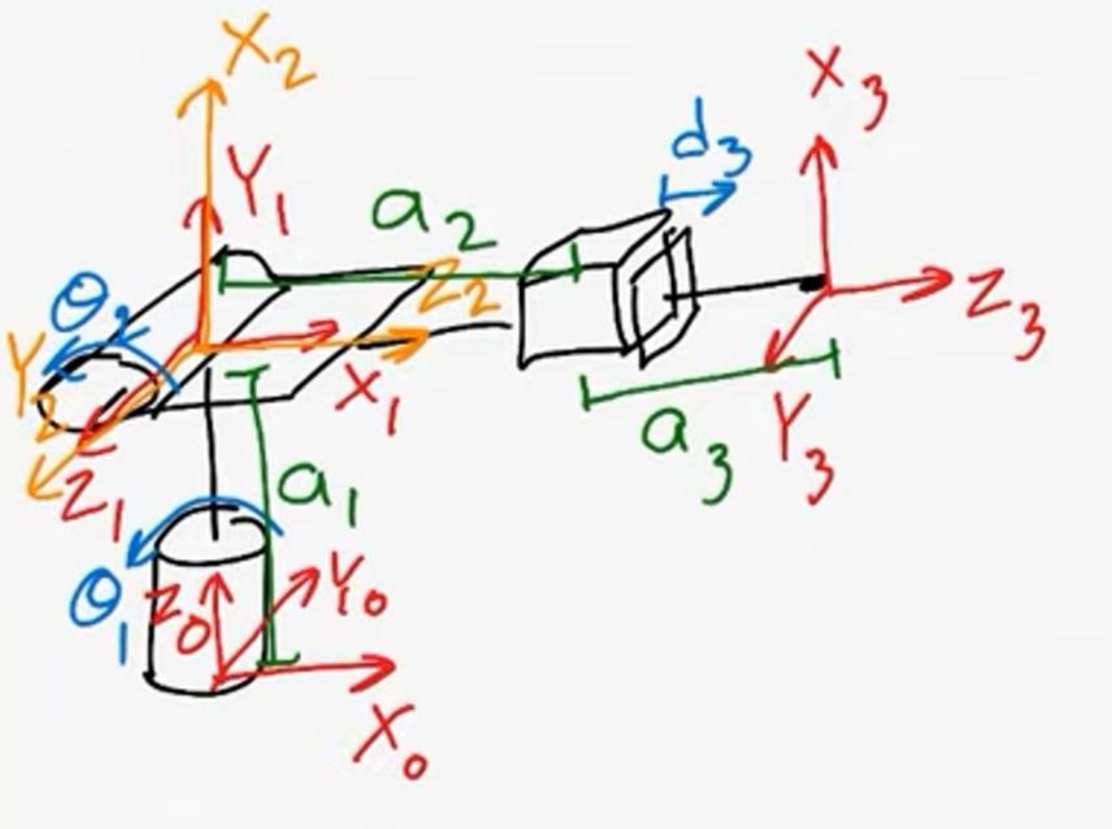
- Articulated manipulator



	θ	α	r	d
1	θ	90	0	a_1
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0

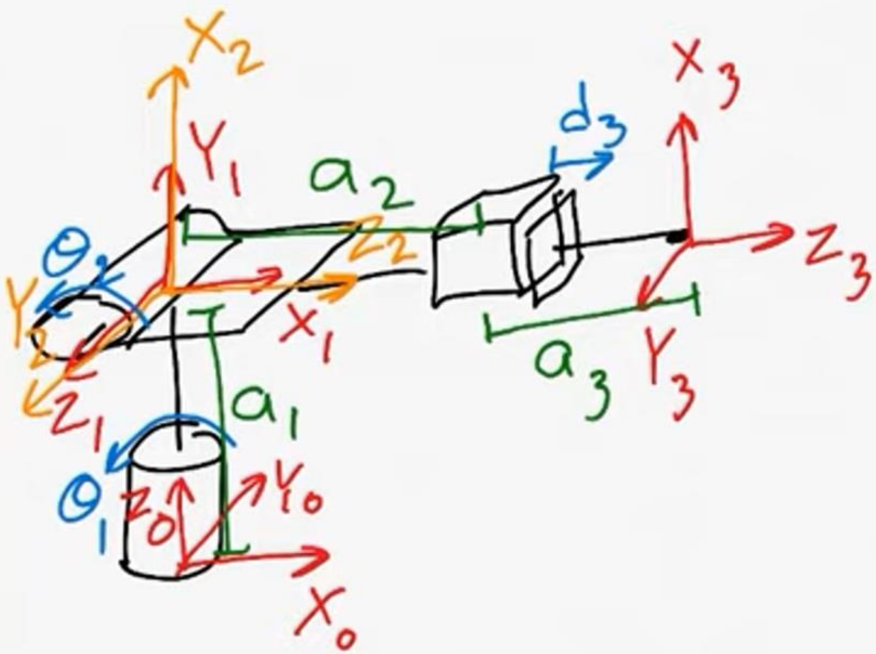
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

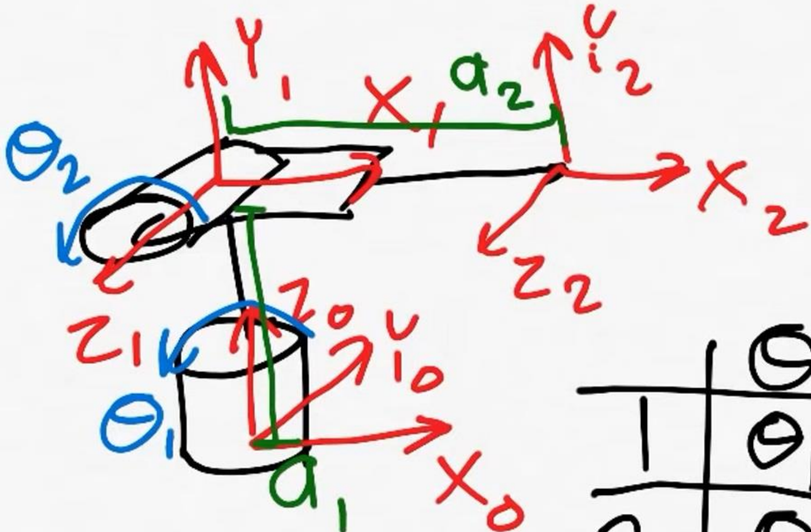
- Spherical manipulator



	θ	α	r	d
1	θ_1	90	0	a_1
2	θ_2+90	90	0	0
3	0	0	0	$a_2+a_3+d_3$

Denavit-Hartenberg Approach to Homogenous Transformation Matrix

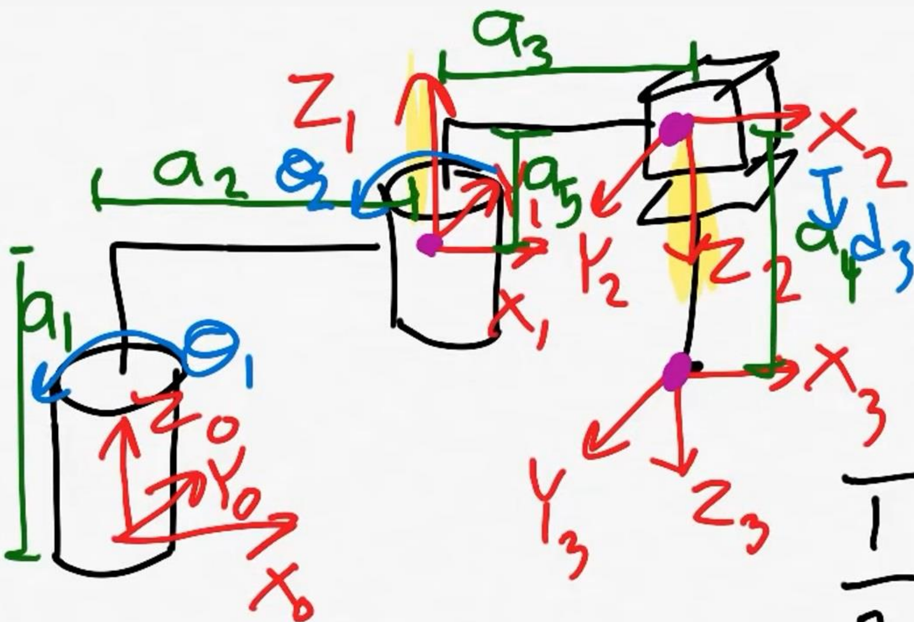
- Parameter table for 2 R manipulator
- This manipulator is like articulated manipulator, however, not complete as articulated manipulator has 3 DoF.



	θ	α	r	d
1	θ_1	90	0	a_1
2	θ_2	0	a_2	0

Denavit-Hartenberg Approach to Homogenous Transformation Matrix

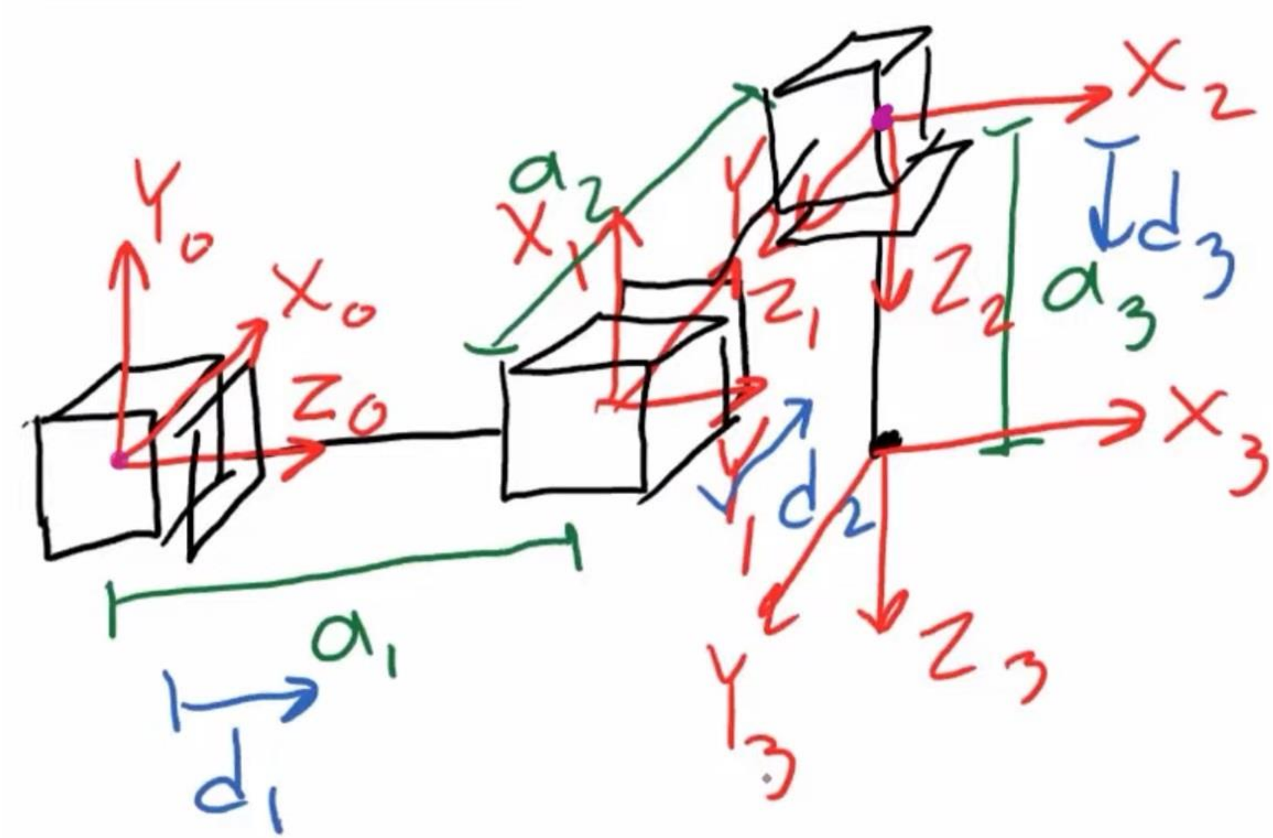
- 3 DoF manipulator parameter table.



	θ	α	r	d
1	θ_1	0	a_2	a_1
2	θ_2	180	a_3	a_5
3	0	0	0	a_4+d_3

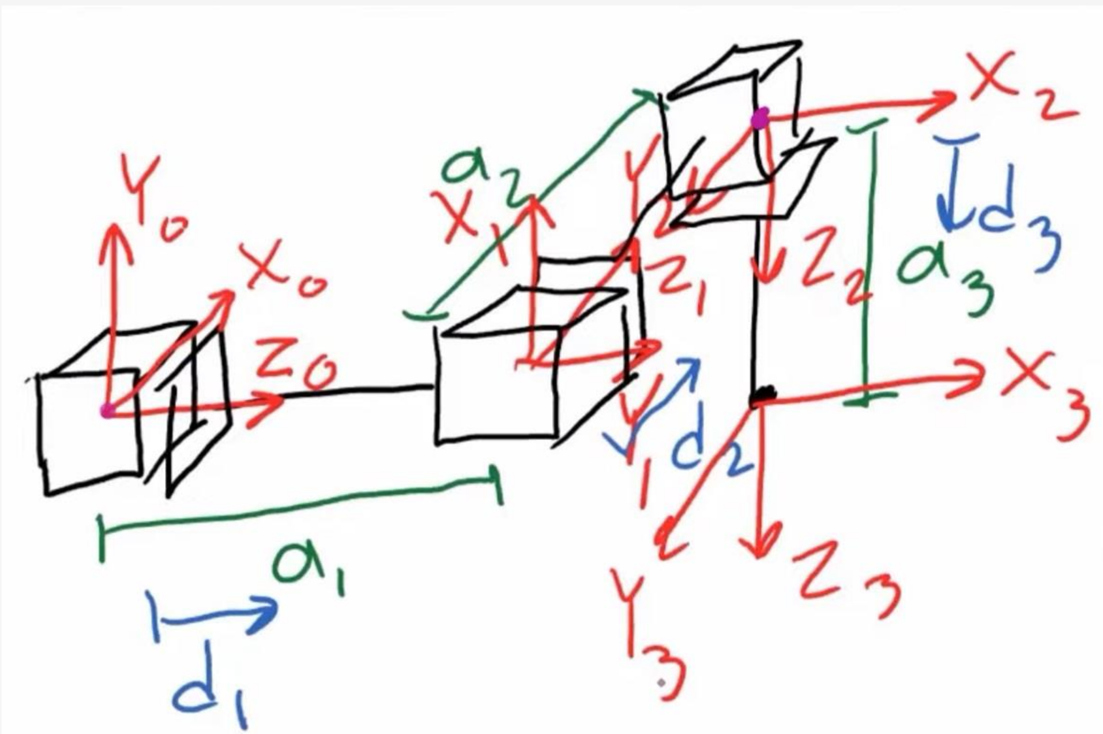
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

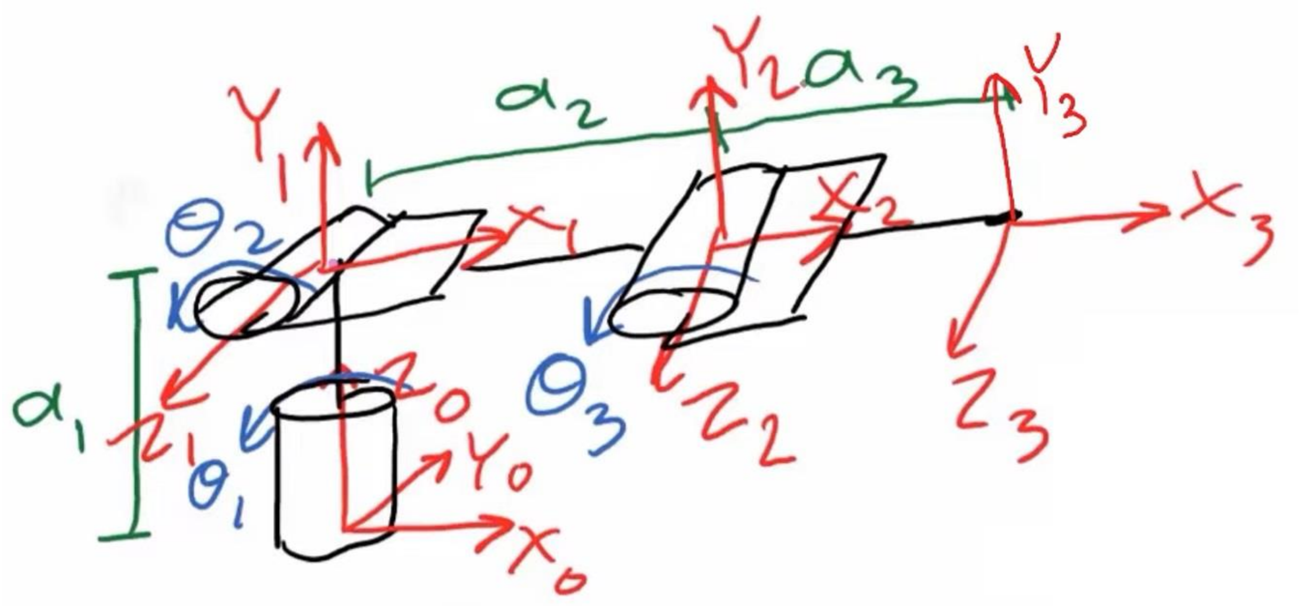
- Cartesian manipulator



	θ	α	r	d
1	90	90	0	a_1+d_1
2	90	-90	0	a_2+d_2
3	0	0	0	a_3+d_3

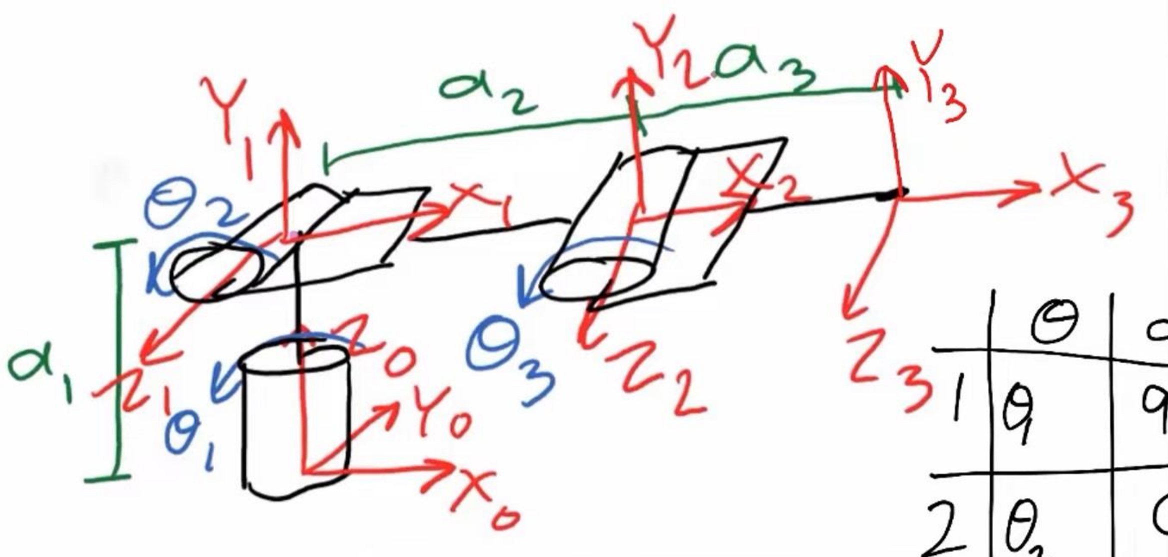
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

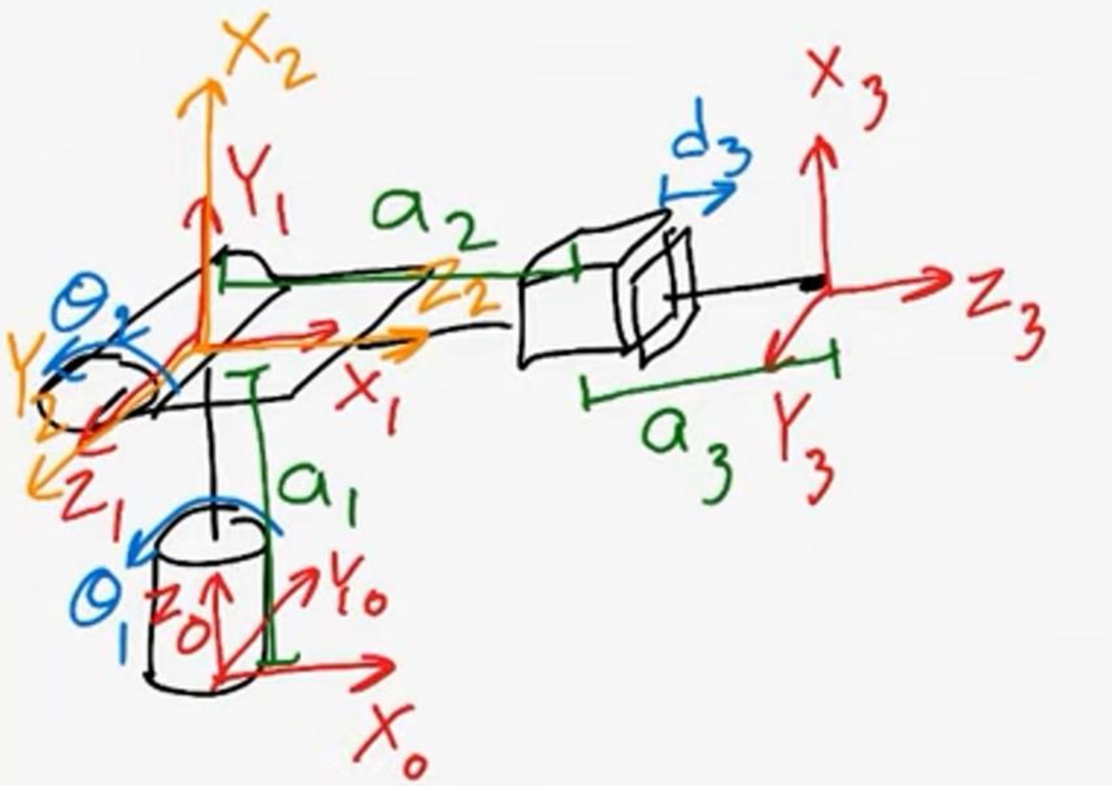
- Articulated manipulator



	θ	α	r	d
1	θ_1	90	0	a_1
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0

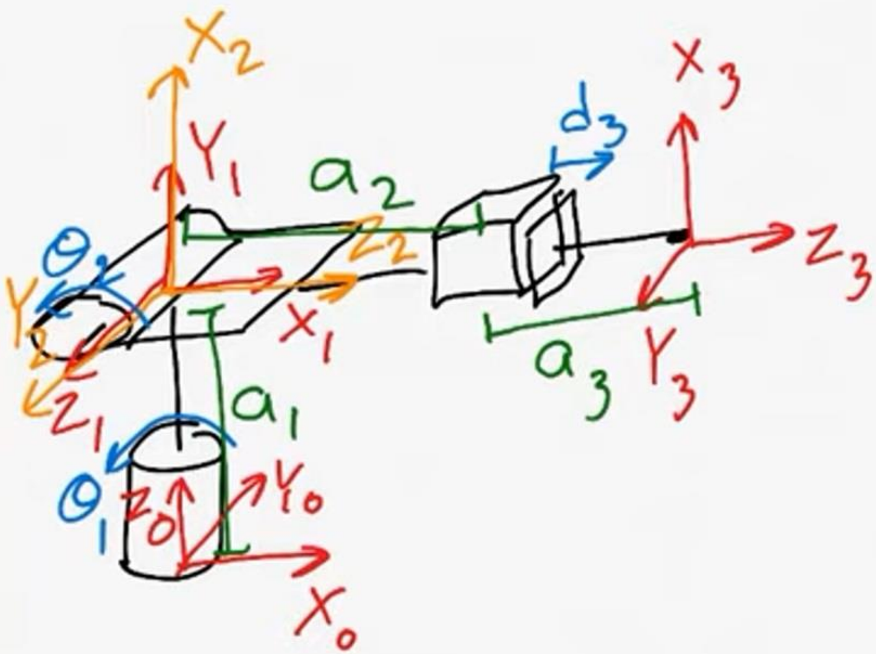
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Which standard manipulator?



Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

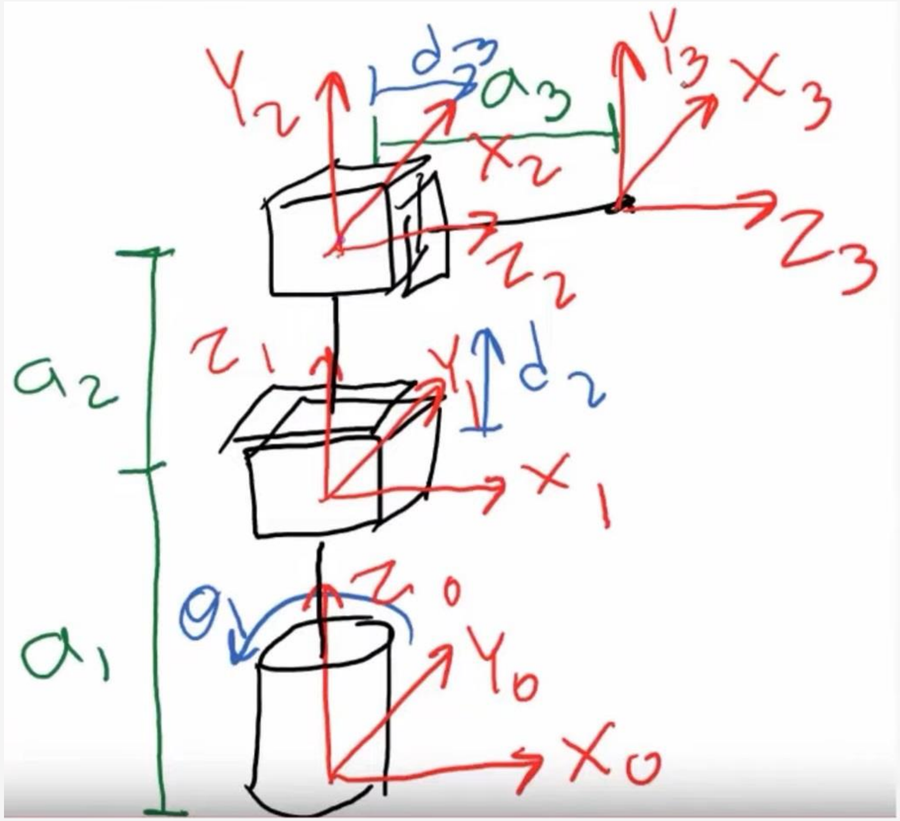
- Spherical manipulator



	θ	α	r	d
1	θ_1	90	0	a_1
2	θ_2+90	90	0	0
3	0	0	0	$a_2+a_3+d_3$

Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

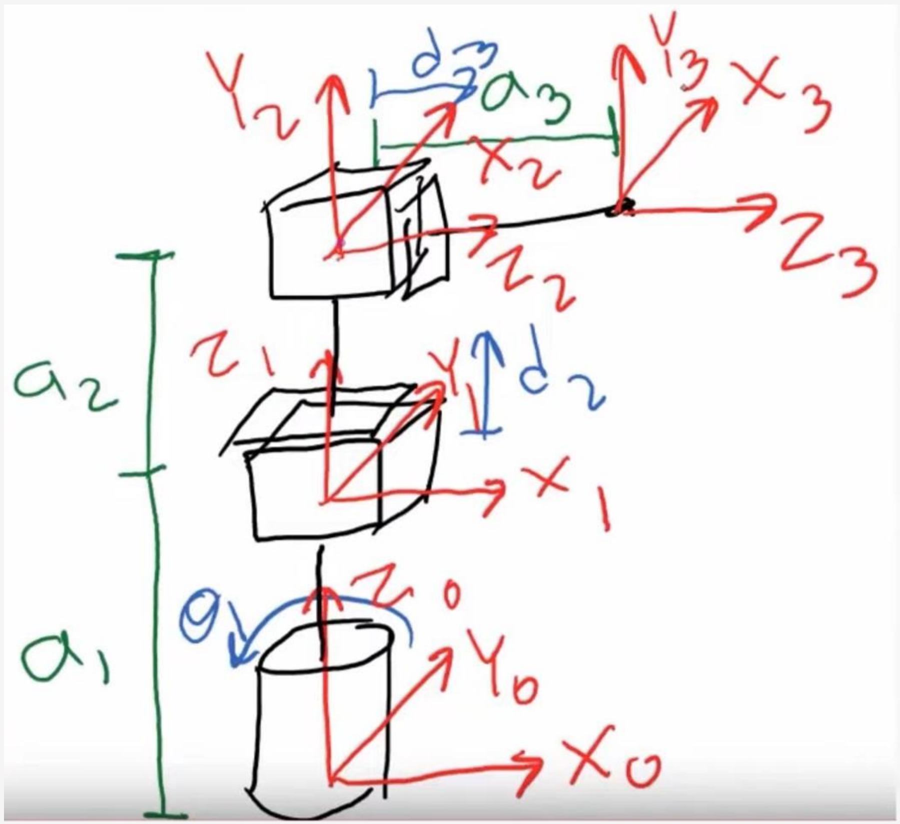
- Which manipulator is this?



	θ	α	r	d
1	θ_1	0	0	a_1
2	90	90	0	a_2+d_2
3	0	0	0	a_3+d_3

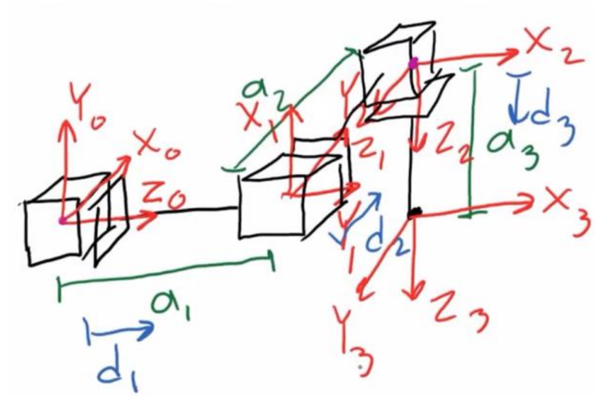
Denavit-Hartenberg Approach to Homogenous Transformation Matrix with 3 DoF

- Cylindrical manipulator



	θ	α	r	d
1	θ_1	0	0	a_1
2	90	90	0	a_2+d_2
3	0	0	0	a_3+d_3

Homogenous Transformation Matrix using Denavit-Hartenberg Approach (Python code)



	θ	α	r	d
1	90	90	0	a_1+d_1
2	90	-90	0	a_2+d_2
3	0	0	0	a_3+d_3

```
import numpy as np
T1=0
T2=0
T3=0
T4=0
T5=0
T6=0
d1=1
d2=1
d3=1
a1=1
a2=1
a3=1
a4=1
T1=(T1/180.0)*np.pi
T2=(T2/180.0)*np.pi
T3=(T3/180.0)*np.pi
T4=(T4/180.0)*np.pi
T5=(T5/180.0)*np.pi
T6=(T6/180.0)*np.pi
PT = [[(90.0/180.0)*np.pi, (90.0/180.0)*np.pi, 0, a1+d1], [(90.0/180.0)*np.pi, (-90.0/180.0)*np.pi, 0, a2+d2], [0, 0, 0, a3+d3]]
i=0
H0_1 = [[np.cos(PT[i][0]), -np.sin(PT[i][0])*np.cos(PT[i][1]), np.sin(PT[i][0])*np.sin(PT[i][1]), PT[i][2]*np.cos(PT[i][1])],
    [np.sin(PT[i][0]), np.cos(PT[i][0])*np.cos(PT[i][1]), -np.cos(PT[i][0])*np.sin(PT[i][1]), PT[i][2]],
                    [0, np.sin(PT[i][1]), np.cos(PT[i][1]), PT[i][3]],
                    [0, 0, 0, 1]]
i=1
H1_2 = [[np.cos(PT[i][0]), -np.sin(PT[i][0])*np.cos(PT[i][1]), np.sin(PT[i][0])*np.sin(PT[i][1]), PT[i][2]*np.cos(PT[i][1])],
    [np.sin(PT[i][0]), np.cos(PT[i][0])*np.cos(PT[i][1]), -np.cos(PT[i][0])*np.sin(PT[i][1]), PT[i][2]],
                    [0, np.sin(PT[i][1]), np.cos(PT[i][1]), PT[i][3]],
                    [0, 0, 0, 1]]
i=2
H2_3 = [[np.cos(PT[i][0]), -np.sin(PT[i][0])*np.cos(PT[i][1]), np.sin(PT[i][0])*np.sin(PT[i][1]), PT[i][2]*np.cos(PT[i][1])],
    [np.sin(PT[i][0]), np.cos(PT[i][0])*np.cos(PT[i][1]), -np.cos(PT[i][0])*np.sin(PT[i][1]), PT[i][2]],
                    [0, np.sin(PT[i][1]), np.cos(PT[i][1]), PT[i][3]],
                    [0, 0, 0, 1]]
print "H0_1"
print np.matrix(H0_1)
print "H1_2"
print np.matrix(H1_2)
print "H2_3"
print np.matrix(H2_3)
print "H0_3"
print np.matrix(H0_3)
```

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- Once Denavit-Hartenberg parameter table is filled then we can also verify if there is any error in the table.
- An error free parameter table must include all joint angles and link lengths.
- Now the next thing is to find out how to use parameter table to find the homogenous transformation matrix.
- Once the Denavit-Hartenberg parameter table is filled out then we can find each row of the homogenous transformation matrix between frame n-1 and frame n by plugging in the parameter into from each row of the table into homogenous transformation matrix row.

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- Here, n is the row of parameter table.
- The general homogeneous transformation matrix would look like:

$$H_n^{n-1} = \begin{bmatrix} C\theta_n & -S\theta_n C\alpha_n & S\theta_n S\alpha_n & r_n C\theta_n \\ S\theta_n & C\theta_n C\alpha_n & -C\theta_n S\alpha_n & r_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- Here, n = 1 mean first row of parameter table plugged into the first row of homogeneous transformation matrix.
- With H_1^0, n is 1 and n-1 is 0.

$$
\begin{array}{cc}
 & n=1 \\
H_1^0 = & \begin{bmatrix} C\theta_n & -S\theta_n C\alpha_n & S\theta_n S\alpha_n & r_n C\theta_n \\ S\theta_n & C\theta_n C\alpha_n & -C\theta_n S\alpha_n & r_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{array}
$$

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- Here, n = 2 mean second row of parameter table plugged into the second row of homogeneous transformation matrix.
- With H_2^1, n is 2 and n-1 is 1.

$$n=2$$

$$H_2^1 = \begin{bmatrix} C\theta_n & -S\theta_n C\alpha_n & S\theta_n S\alpha_n & r_n C\theta_n \\ S\theta_n & C\theta_n C\alpha_n & -C\theta_n S\alpha_n & r_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- Here, n = 2 means third row of parameter table plugged into the third row of homogeneous transformation matrix.
- With H_3^2, n is 3 and n-1 is 2.

$$n=3$$

$$H_3^2 = \begin{bmatrix} C\theta_n & -S\theta_n C\alpha_n & S\theta_n S\alpha_n & r_n C\theta_n \\ S\theta_n & C\theta_n C\alpha_n & -C\theta_n S\alpha_n & r_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

How to get Homogenous Transformation Matrix using Denavit-Hartenberg Approach to

- In the end, all homogeneous transformation matrices are multiplied to get H_3^0.
- The first three rows and first three columns represents rotation matrix.
- Last column with top three rows is displacement vector.

$n=3$

$$H_3^2 = \begin{bmatrix} C\theta_n & -S\theta_n C\alpha_n & S\theta_n S\alpha_n & r_n C\theta_n \\ S\theta_n & C\theta_n C\alpha_n & -C\theta_n S\alpha_n & r_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_3^0 = H_1^0 H_2^1 H_3^2$$

End