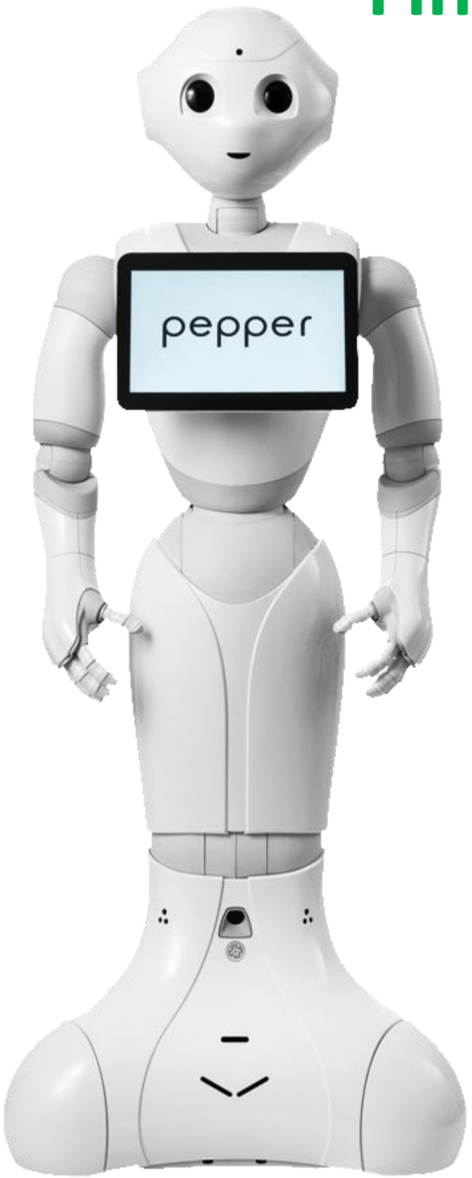
Fundamentals of Robotics

Robot Manipulators 04

Finding Orientation of a robot



Description of position and orientation of an end-effector

Projection of a vector

Rotation metrices

Rotation around z-axis

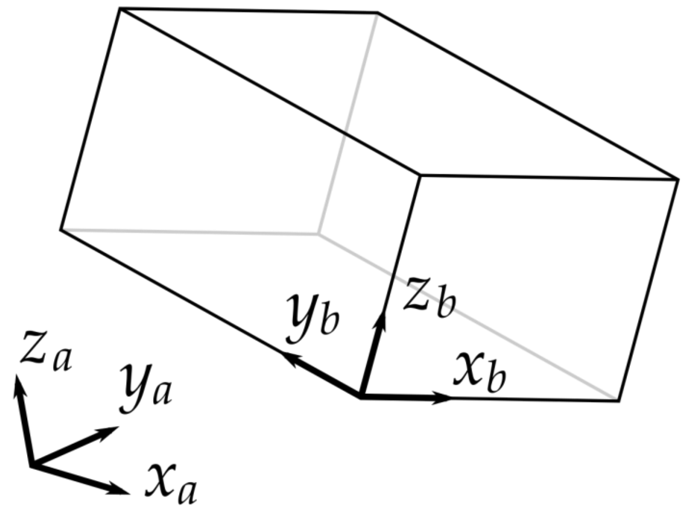
Rotation around x, y and z-axis

Random rotation around axes

Rotation matrix for 2 DoF manipulator

Description of Position and Orientation of an End-effector

- In general, a **rigid body** in **three-dimensional** space has **six DoF**: **three rotational** and **three translational**.
- The **position** and **orientation** of a **rigid body** can be described by **attaching a frame** to it.
- A **conventional** way to describe the position and orientation of a rigid body is to **attach a frame to it**.
- After defining a **reference coordinate system**, the position and orientation of the rigid body are fully described by the **position of the frame's origin** and the **orientation** of its axes, **relative** to the **reference frame**.

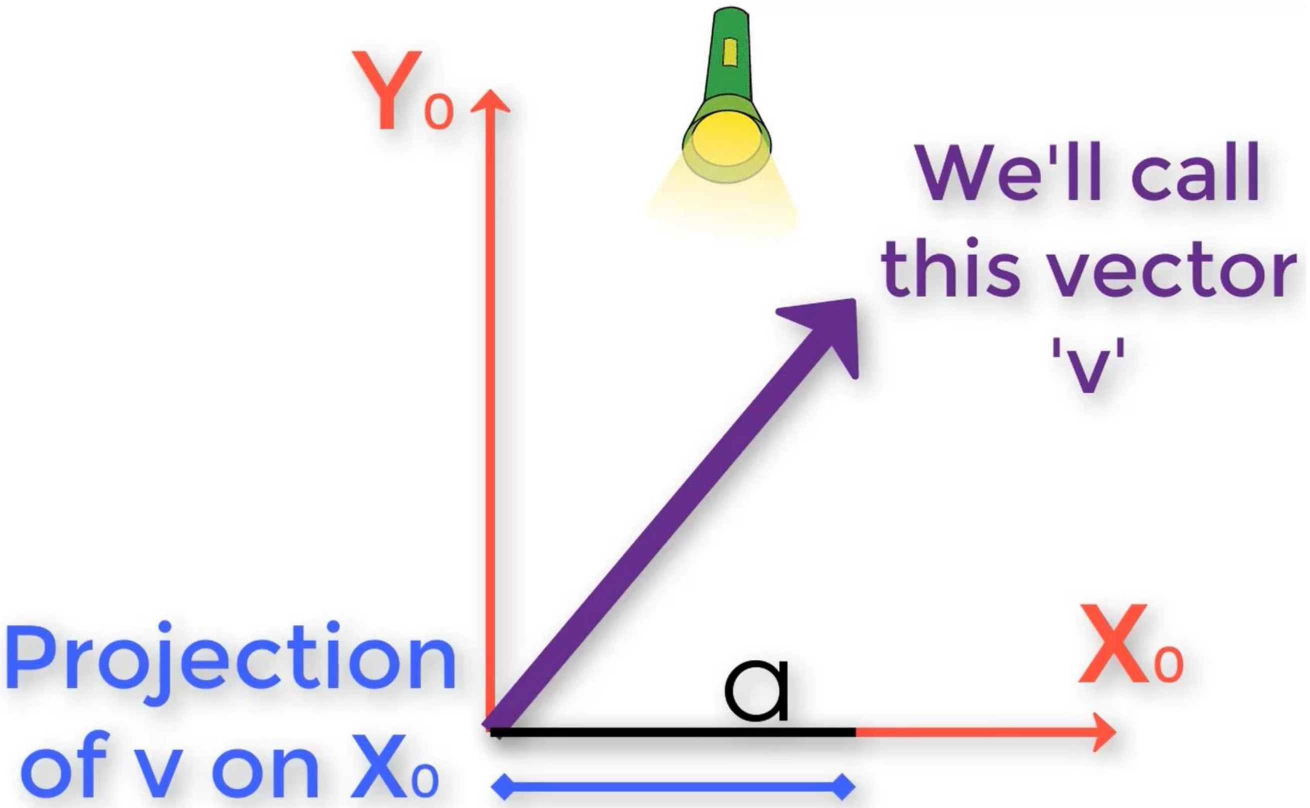


Description of Position and Orientation of an End-effector

- We describe the **position** of an end-effector using the **displacement vector**.
  - Position is e.g., location at x,y plane
- The **orientation** of an end-effector is described using the **rotation matrix**.
  - At the position of x, y to which direction the end-effector is facing.
- In Robotics, we care about both the position and the orientation (direction) of an end-effector.

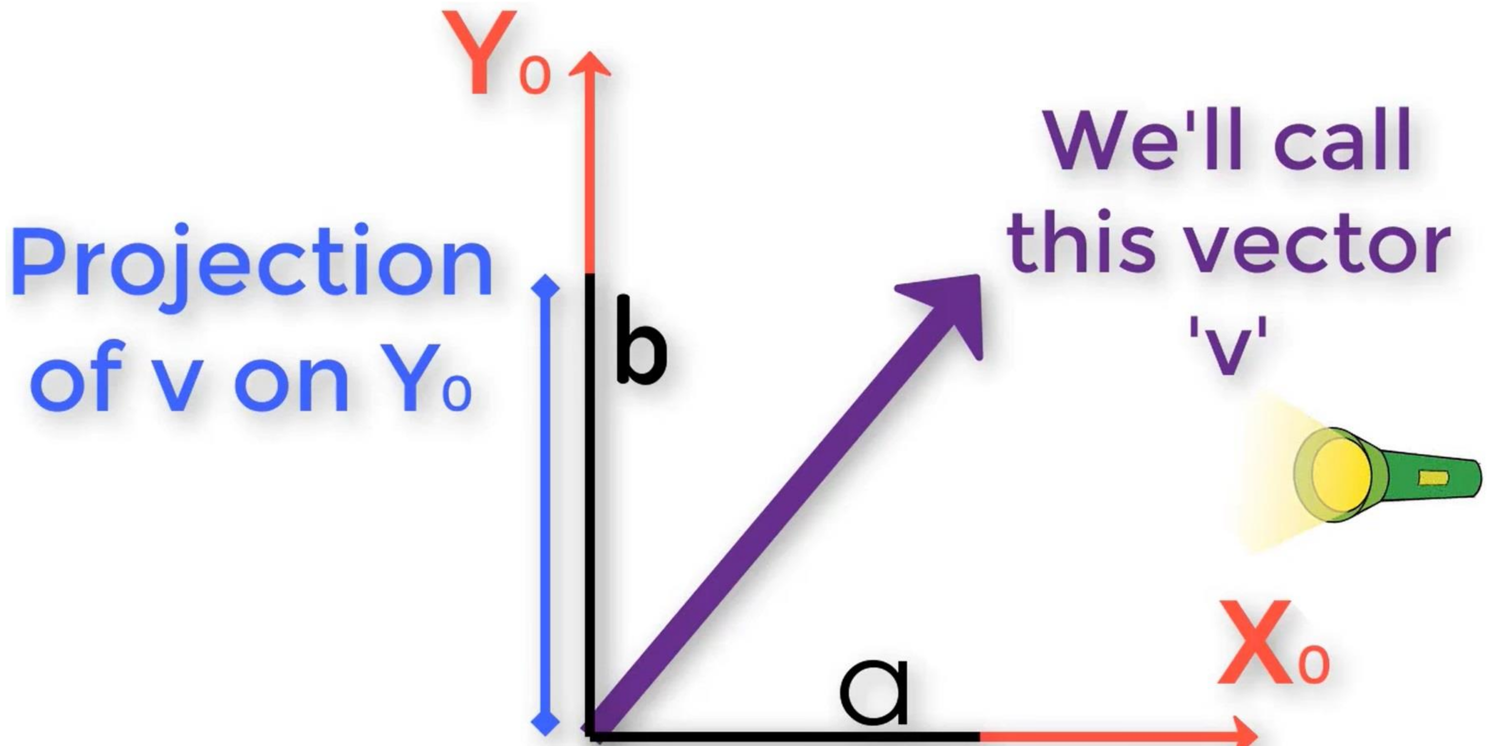
Projection of a vector

- Projection in 2-dimensional plane.



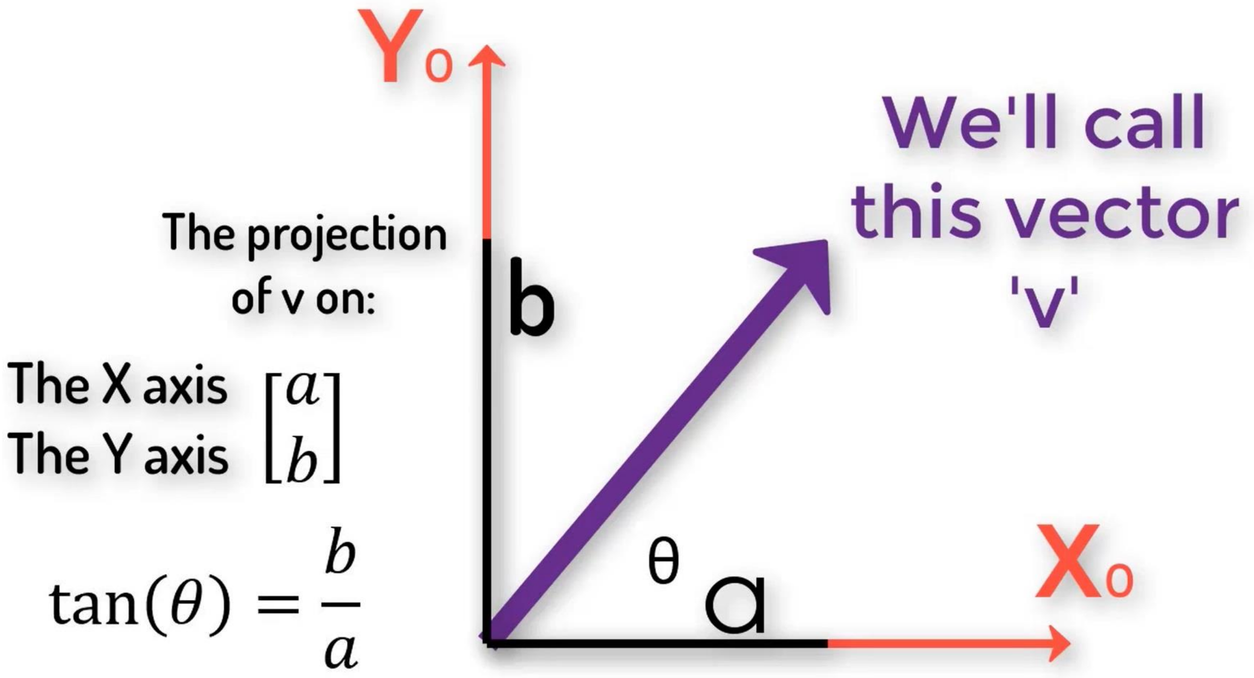
Projection of a vector

- Projection in 2-dimensional plane.



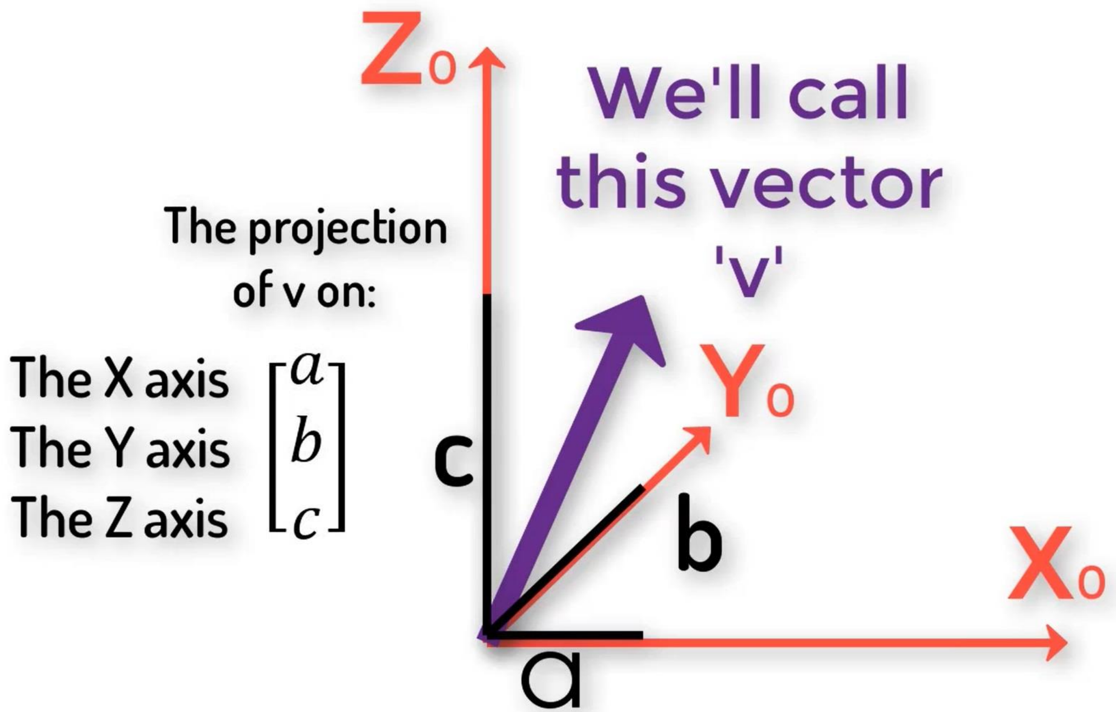
Projection of a vector

- Projection in a 2-dimensional plane.
- Suppose that you want to know the direction of vector **v** then **ϴ** would help you to find it.
- To find **ϴ** you need a **projection** of **v** in the **x** and **y** axes.



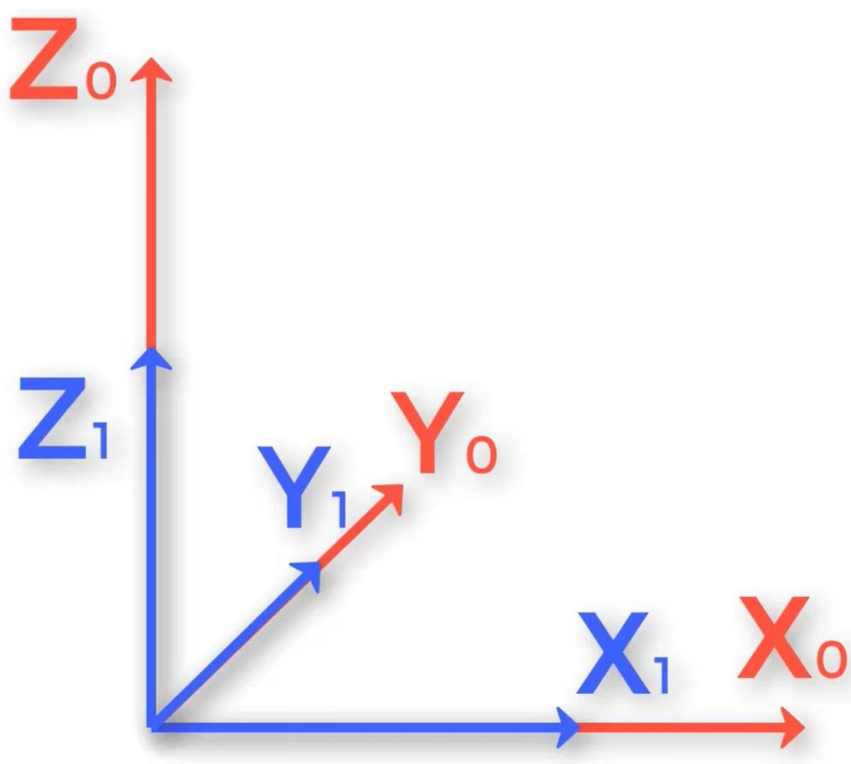
Projection of a vector

- In Robotics, we are usually working in 3D space.
- The idea of the projection also works in 3D space.
- Instead of using only projection in x and y axes will add projection on the z-axis.



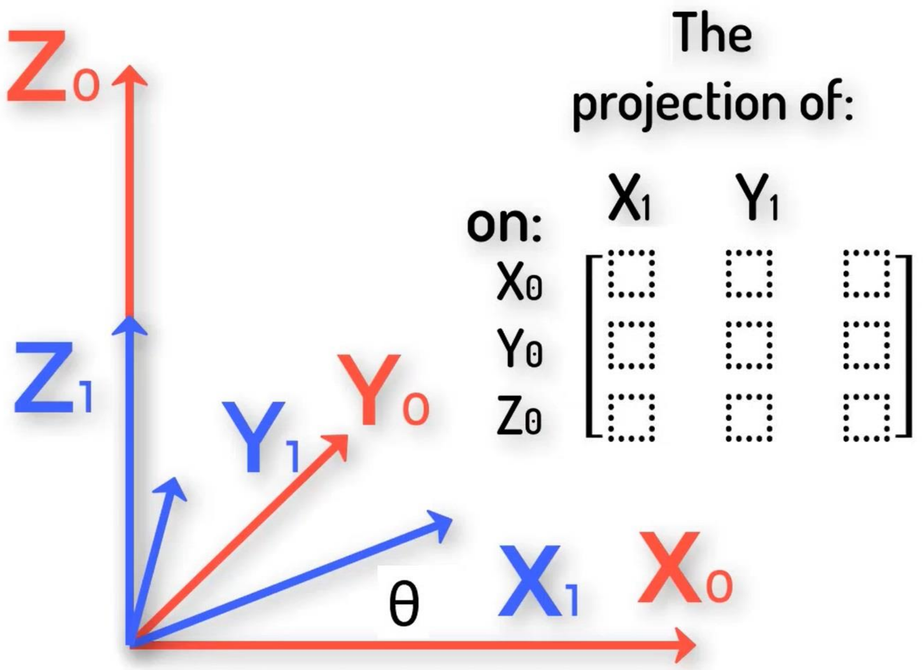
Projection of a frame

- In Robotics, we are not looking at the **projection of one vector** in space, instead, we look at the project on **a complete frame**.
- In other words, we have **three vectors** that are rotating together.



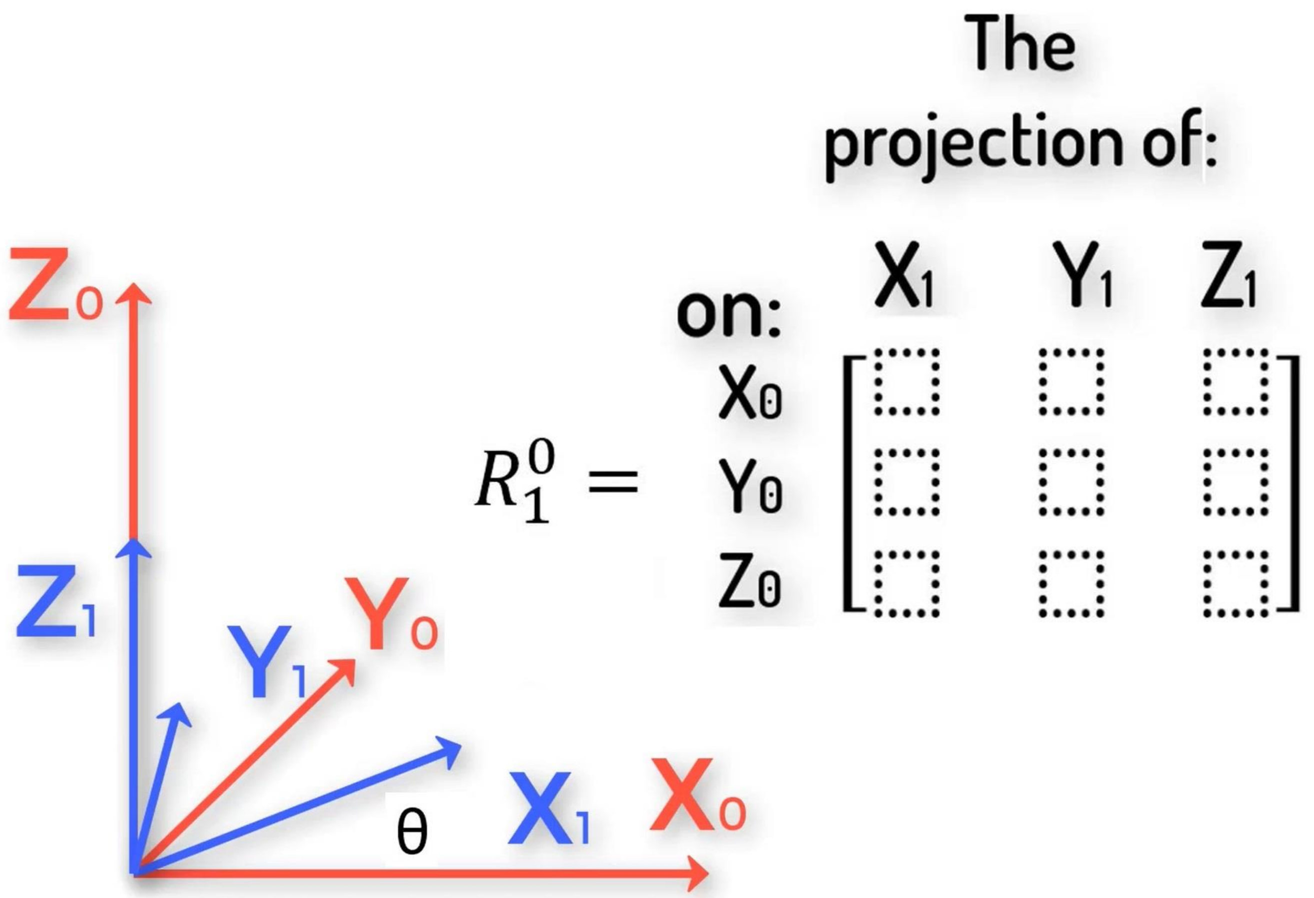
Projection of a frame

- We need a projection of $\mathbf{X_1}$, $\mathbf{Y_1}$, and $\mathbf{Z_1}$ on $\mathbf{X_0}$, $\mathbf{Y_0}$, and $\mathbf{Z_0}$.
- The blanks in the matrix should have the values of projections.



Rotation matrix

- $R^0{}_1$ is a rotation matrix between frames 0 and 1.



Rotation around z-axis

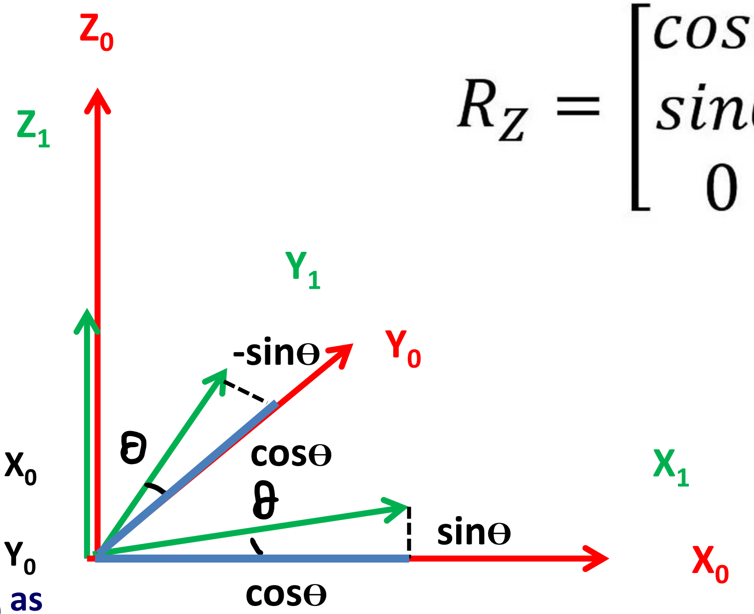
- In order to get values in the rotation matrix, we need to assume in the coordinate frame each of these individual axes is to be length 1.
- Suppose frame 0 rotated angle **Θ** around the z-axis and we know this angle.

$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

sinΘ = opp / hyp

cosΘ = adj / hyp

tanΘ = opp / adj



For projection of X_1 on X_0 is
cosΘ = adj / 1
cosΘ = adj = Projection of X_1 on X_0
sinΘ = opp / 1
cosΘ = adj = Projection of X_1 on Y_0
There is no projection of X_1 on Z_0 as X_1 doesn't overlap Z_0

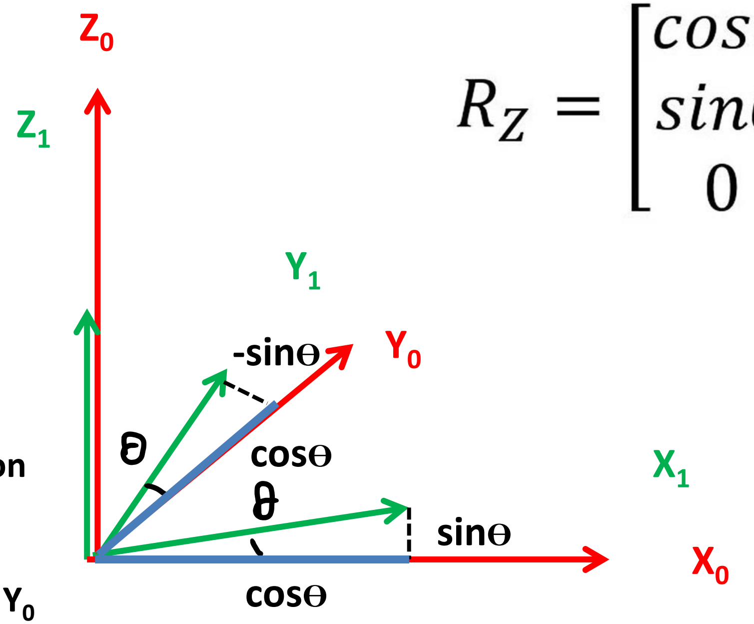
Rotation around z-axis

- The projection of Y_1 on X_0 is in negative direction. This projection will be on the opposite side of the triangle.

sinϴ = opp / hyp

cosϴ = adj / hyp

tanϴ = opp / adj



$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For projection of Y_1 on X_0 is
sinsϴ = -opp / 1
-sinsϴ = opp = Projection of Y_1 on X_0
cosϴ = adj / 1
cosϴ = adj = Projection of Y_1 on Y_0
There is no projection of Y_1 on Z_0 as Y_1 doesn't overlap Z_0

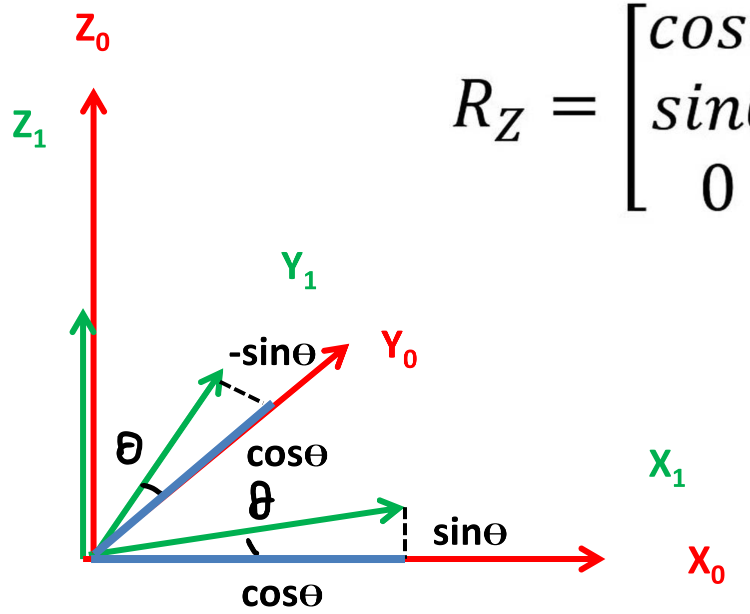
Rotation around z-axis

- There is no projection of Z_1 on X_0, Y_0.

sinΘ = opp / hyp

cosΘ = adj / hyp

tanΘ = opp / adj



$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation around z-axis

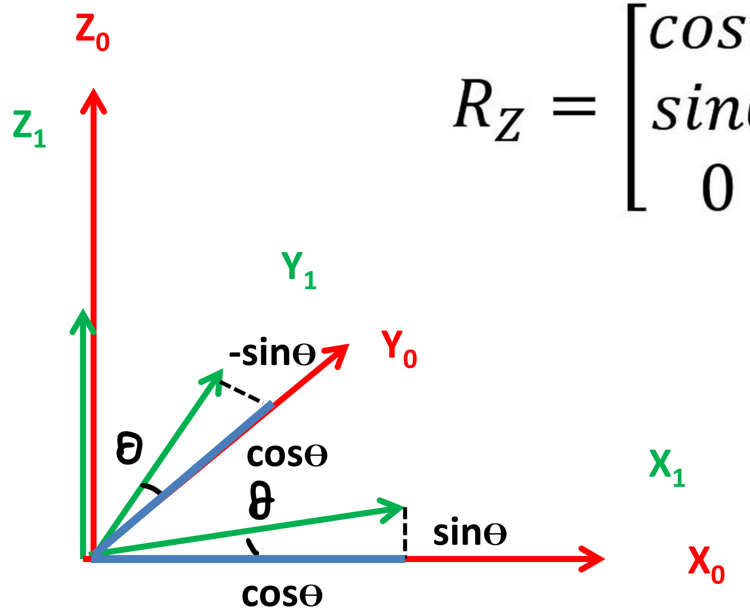
- R_z is a rotation matrix around the z-axis.
- There is no project of x and y on the z-axis as they are perpendicular to each other.
- Z_0 and Z_1 are lined up.

$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

sinϴ = **opp / hyp**

cosϴ = **adj / hyp**

tanϴ = **opp / adj**



Rotation around x, y and z-axis

- Like R_z is there are rotation matrices for x and y axes and are called R_x *and* R_y *matrices*.
- They are driven like R_z.

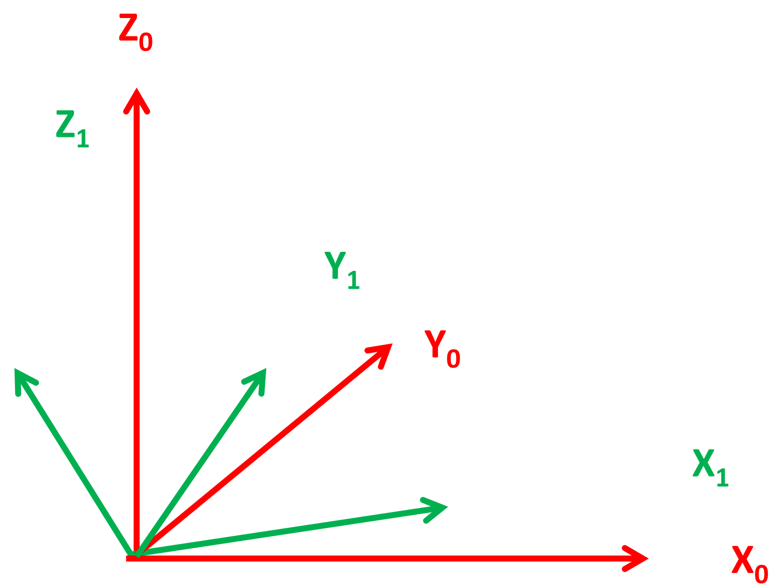
$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix}$$

$$R_Y = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix}$$

$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

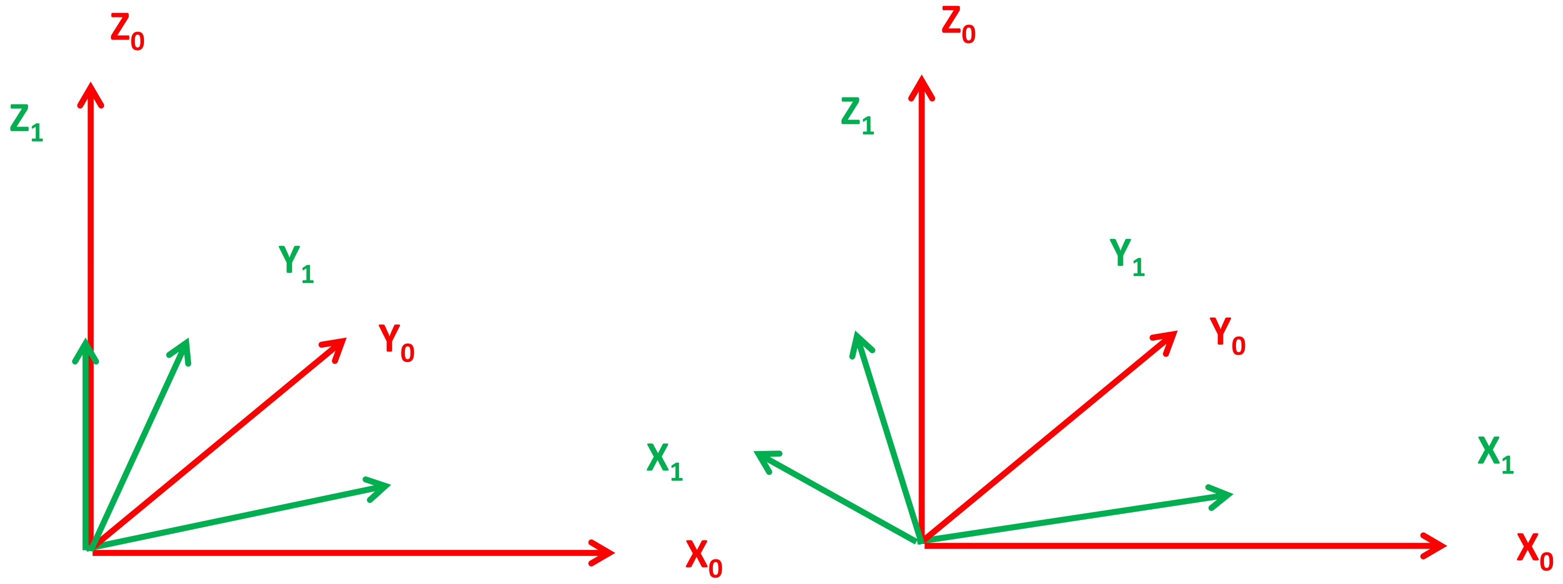
Random rotation around axes

- This shows a random rotation around the axes.
- Any rotation is rotation around the x, y, and z-axis.



Random rotation around axes

- First, we fixed the z-axis and rotate around it and then we move the z and y-axis by fixing the x-axis.
- The first rotation can be covered by the R_z rotation matrix and then we multiply it with R_x.
- The value of the newly formed matrix will be projections of $\mathbf{X_1}$, $\mathbf{Y_1}$, and $\mathbf{Z_1}$ on $\mathbf{X_0}$, $\mathbf{Y_0}$, and $\mathbf{Z_0}$.



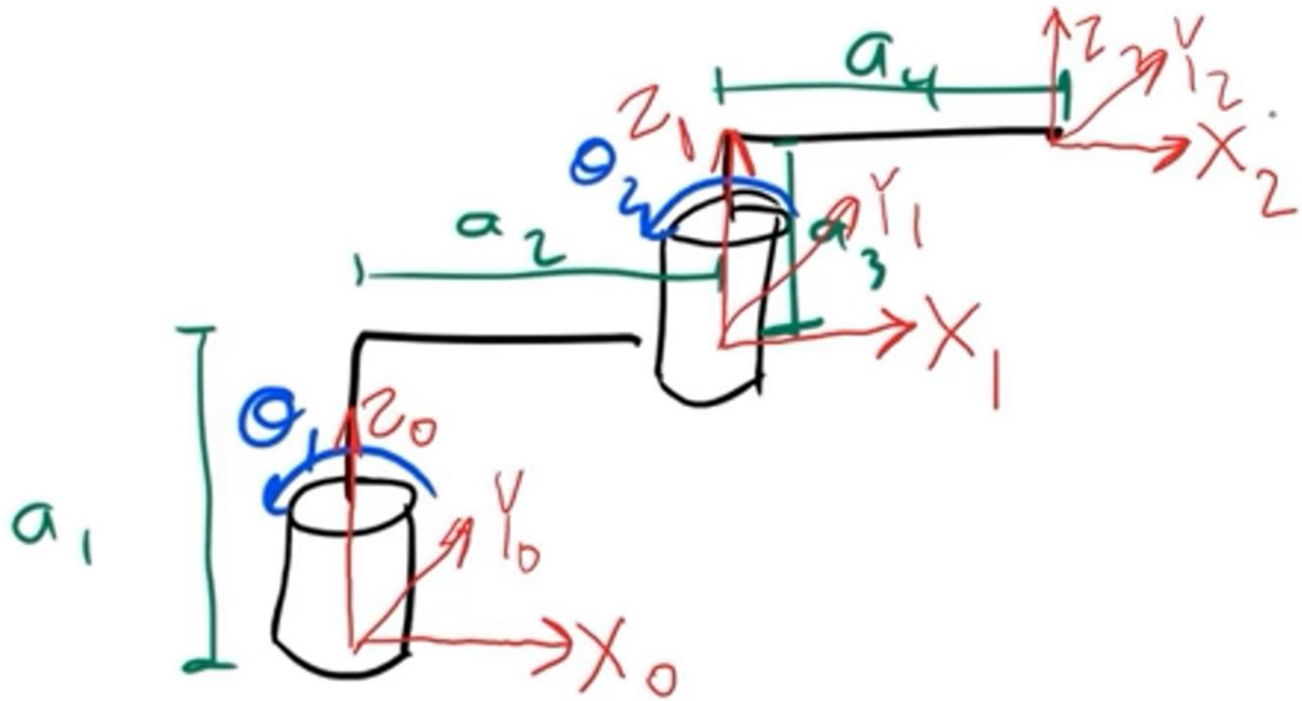
Random rotation around axes

- The first rotation can be covered by the R_z rotation matrix and then we multiply it with R_x.
- The value of the newly formed matrix will be projections of $\mathbf{X_1}$, $\mathbf{Y_1}$, and $\mathbf{Z_1}$ on $\mathbf{X_0}$, $\mathbf{Y_0}$, and $\mathbf{Z_0}$.
- In this way we can form these matrices without going through the process of projection.
- We can string together any number of rotations by using rotations around X, Y and Z.

$$\mathbf{R} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix}$$

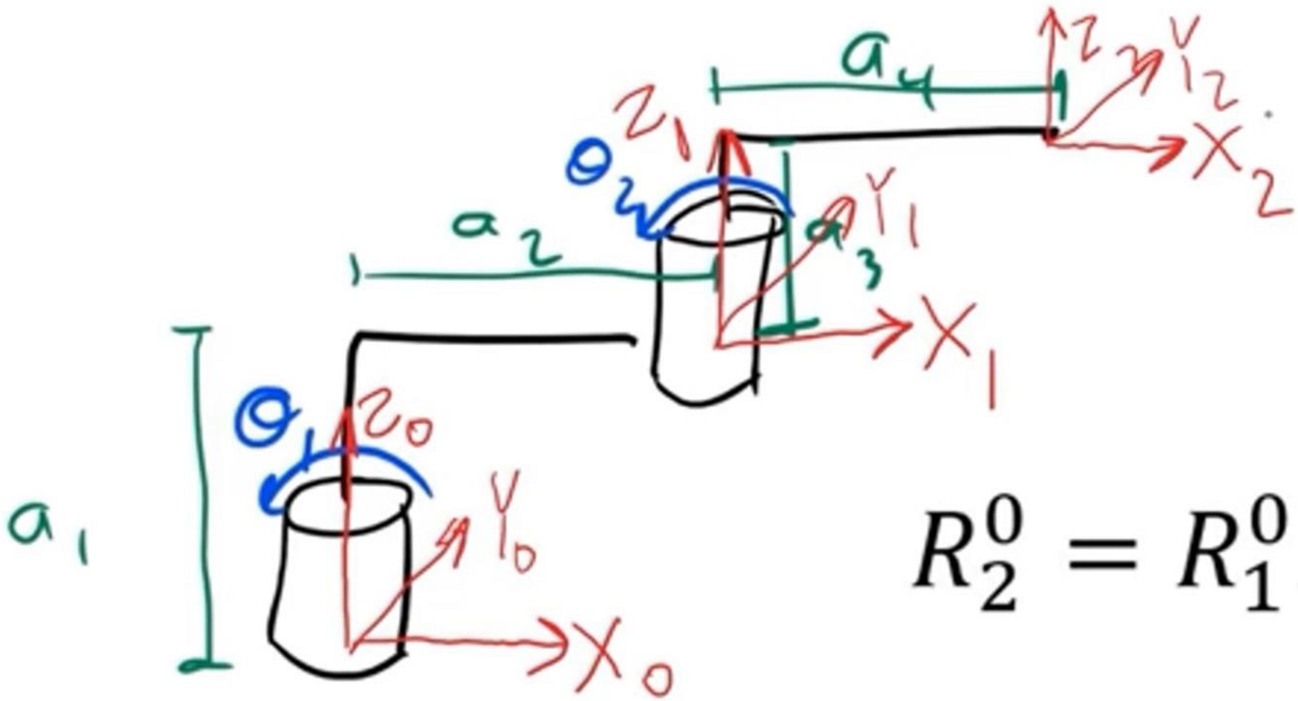
Rotation matrix for 2 DoF

- Let's consider the following kinematic diagram
- At the moment, there is not rotation at all as all axis are in the same direction.
- We need to find the rotation matrix which tell us how much an end-effector is rotation relative to the base frame (frame 0)



Rotation matrix for 2 DoF

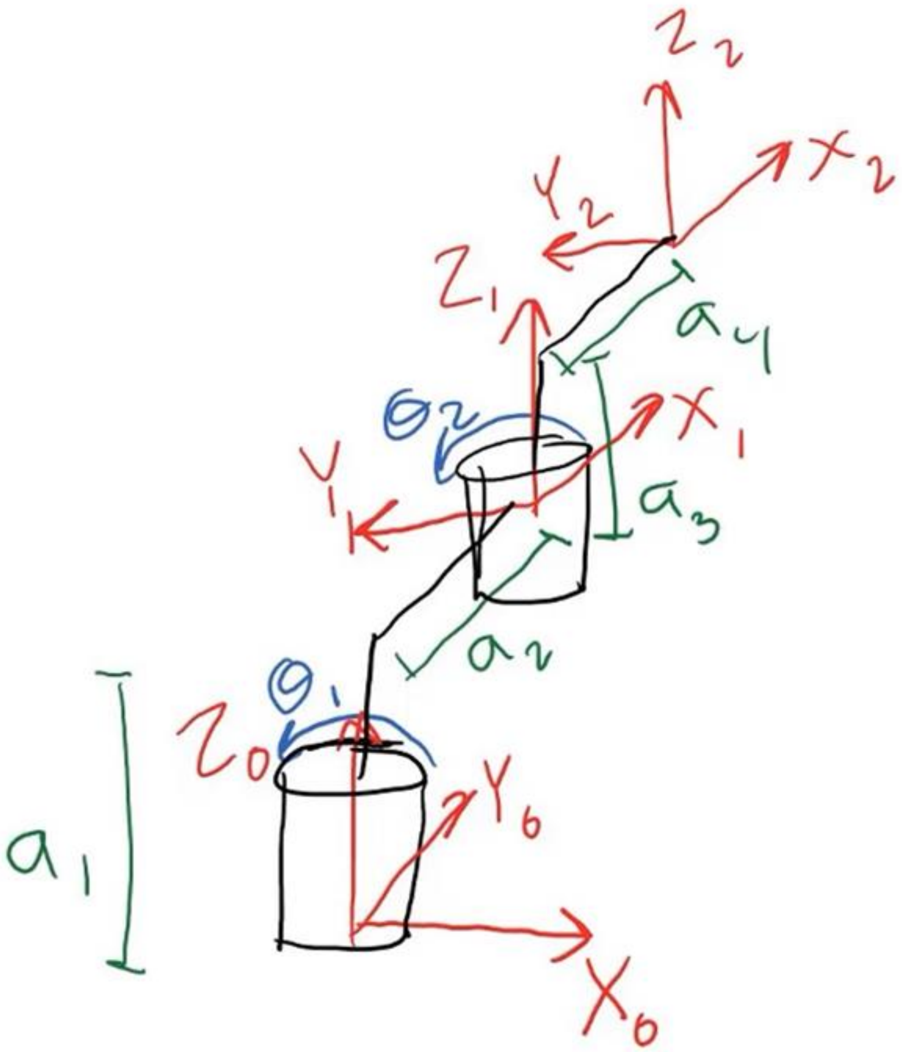
- As we can string together rotations.
- Therefore, first we would find rotation between based (frame 0) frame and first frame (frame 1) and then finding rotation matrix for first frame and second frame (frame 2).



$$R_2^0 = R_1^0 R_2^1$$

Rotation matrix for 2 DoF

- As the first joint moves the first frame would look like this:
- As you can see the position of frame 1 is also change. However, at the moment we are only concerned with change in the orientation of the end-effector.
- The position change will be covered in next topic called displacement vector.

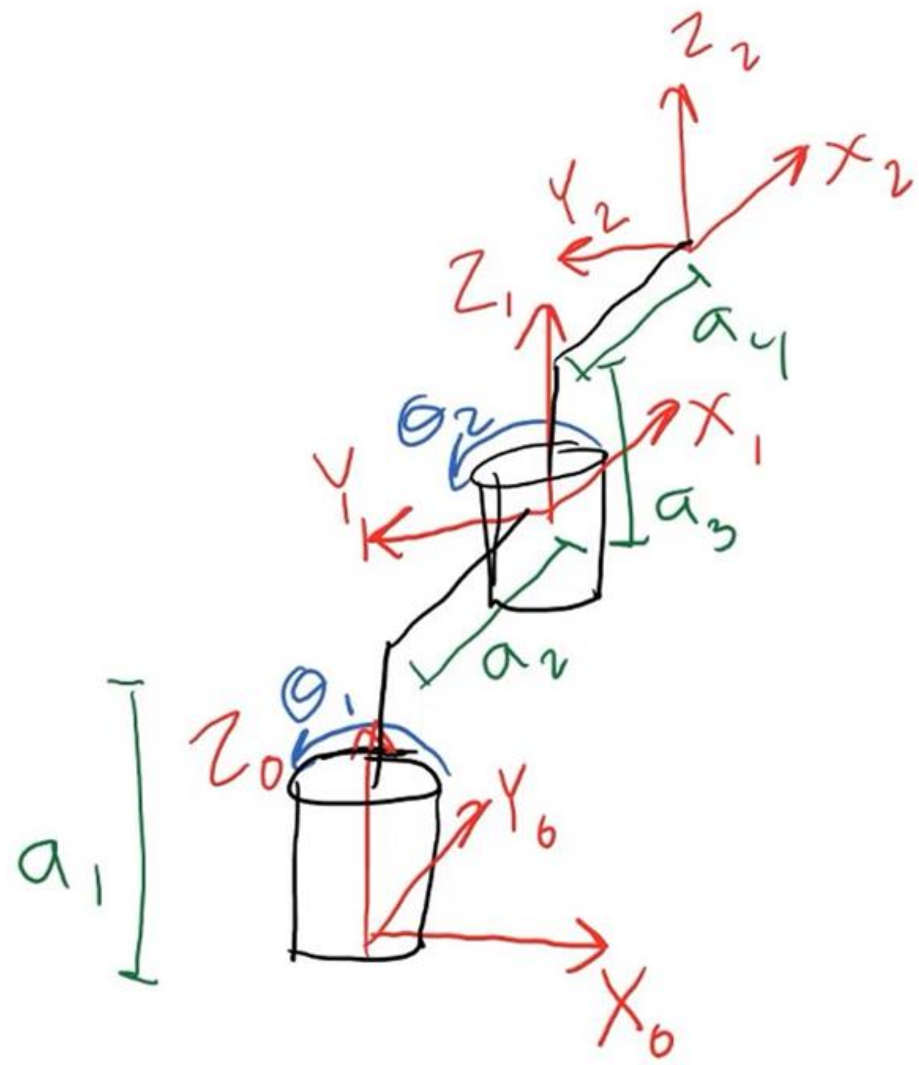


Rotation matrix for 2 DoF

- As the first joint moves the based frame rotates around z-axis.
- Here, we will use R_z matrix.
- Theta will be replaced by theta1 for R^0_1

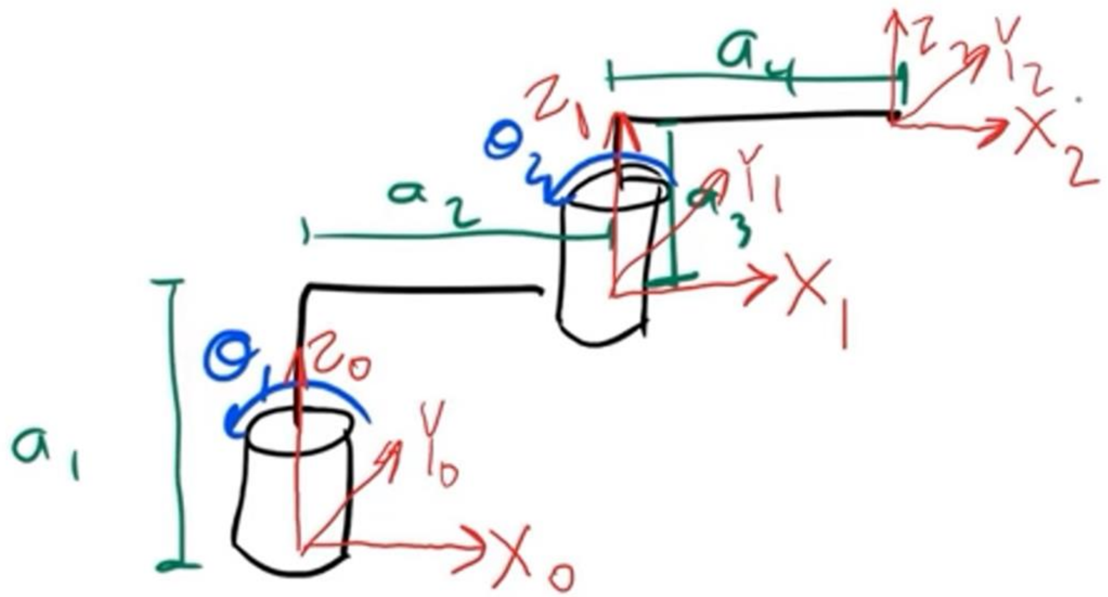
$$R_Z = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^0_1 = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotation matrix for 2 DoF

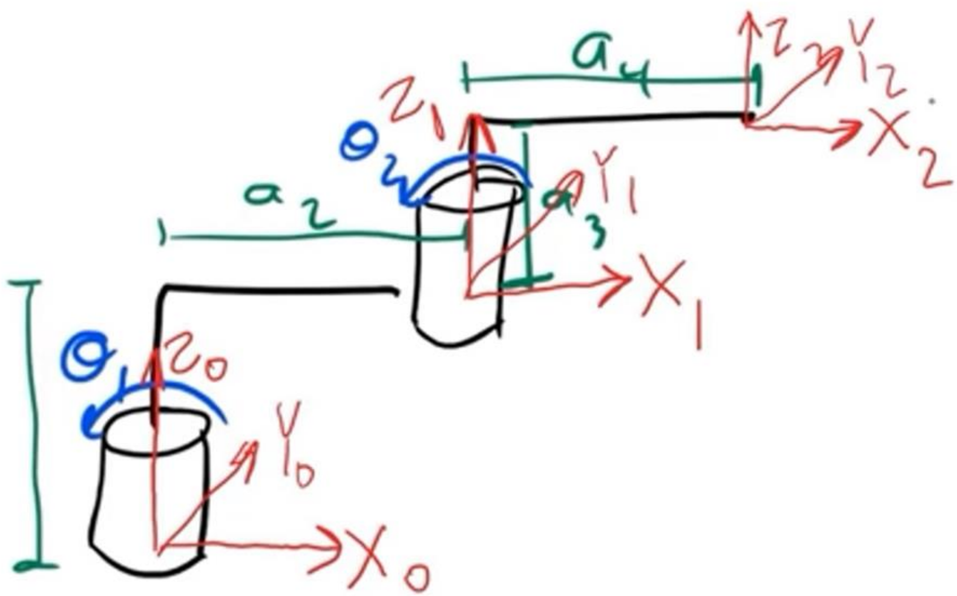
- The rotation matrix for R^0_1 is still incomplete.
- We need to find how much frame 0 is rotated relative to frame 1 when the joint angle is 0.
- Between frame 0 and 1 there is no rotation at all as you can see in the kinematic diagram X_0, Y_0 and Z_0 are facing he same direction as X_1, Y_1 and Z_1.



Rotation matrix for 2 DoF

- To show no rotation between frame 0 and frame 1 can be expressed using special matrix called the identity matrix.
- You can see in identity the projection the axis on itself is one and other is zero (that shows not rotations between frames).
- To get complete $R^0{}_1$ we need to left multiply identity with current $R^0{}_1$ to get it complete.

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotation matrix for 2 DoF

- To get complete R^0_1 we need to left multiply identity with current R^0_1 to get it complete.

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

Rotation matrix for 2 DoF

- Frame 2 is going to rotate as shown in figure.
- This rotation is still around z-axis.
- Therefore, R^1_2 is drive in the same way as R^0_1.

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

Rotation matrix for 2 DoF

- To find the rotation is the end-effector relative to based frame is R^0_2.

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

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

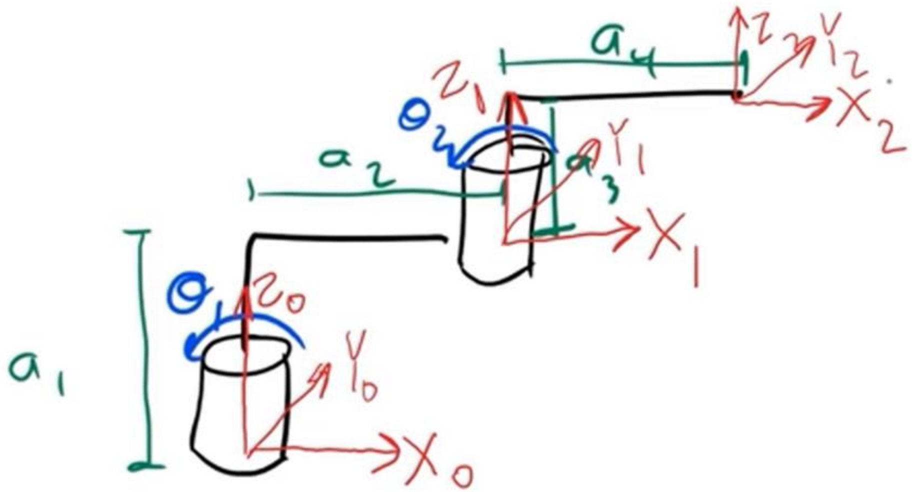
$$R_2^0 = R_1^0 R_2^1$$

Rotation matrix for 2 DoF with Python example

- At Theta1 0 show there is no rotation of frame 2 relative to frame 0.

```
import numpy as np
Theta1 = 0 # try 0 and then 90 degree joint angle
Theta2 = 0
Theta1 = (Theta1/180) * np.pi
Theta2 = (Theta2/180) * np.pi

R0_1 = [[np.cos(Theta1), -np.sin(Theta1), 0], [np.sin(Theta1), np.cos(Theta1), 0], [0, 0, 1]]
R1_2 = [[np.cos(Theta2), -np.sin(Theta2), 0], [np.sin(Theta2), np.cos(Theta2), 0], [0, 0, 1]]

R0_2 = np.dot(R0_1, R1_2)
print(R0_2)
```

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

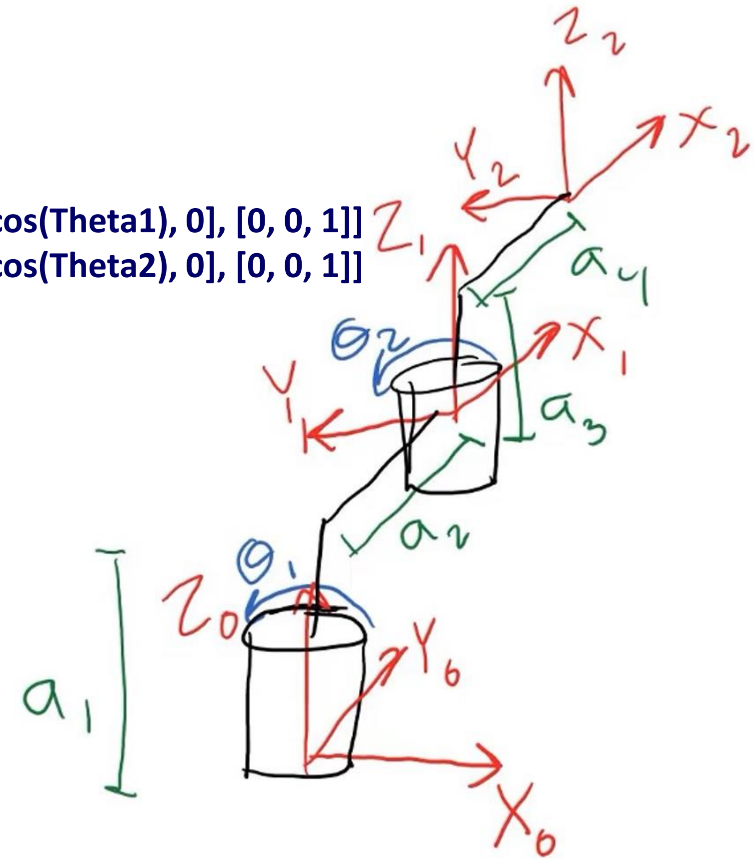


Rotation matrix for 2 DoF with Python example

- At Theta1 0 show there is no rotation of frame 2 relative to frame 0.

```
import numpy as np
Theta1 = 90 # try 0 and then 90 degree joint angle
Theta2 = 0

Theta1 = (Theta1/180) * np.pi
Theta2 = (Theta2/180) * np.pi

R0_1 = [[np.cos(Theta1), -np.sin(Theta1), 0], [np.sin(Theta1), np.cos(Theta1), 0], [0, 0, 1]]
R1_2 = [[np.cos(Theta2), -np.sin(Theta2), 0], [np.sin(Theta2), np.cos(Theta2), 0], [0, 0, 1]]

R0_2 = np.dot(R0_1, R1_2)print(R0_2)
```

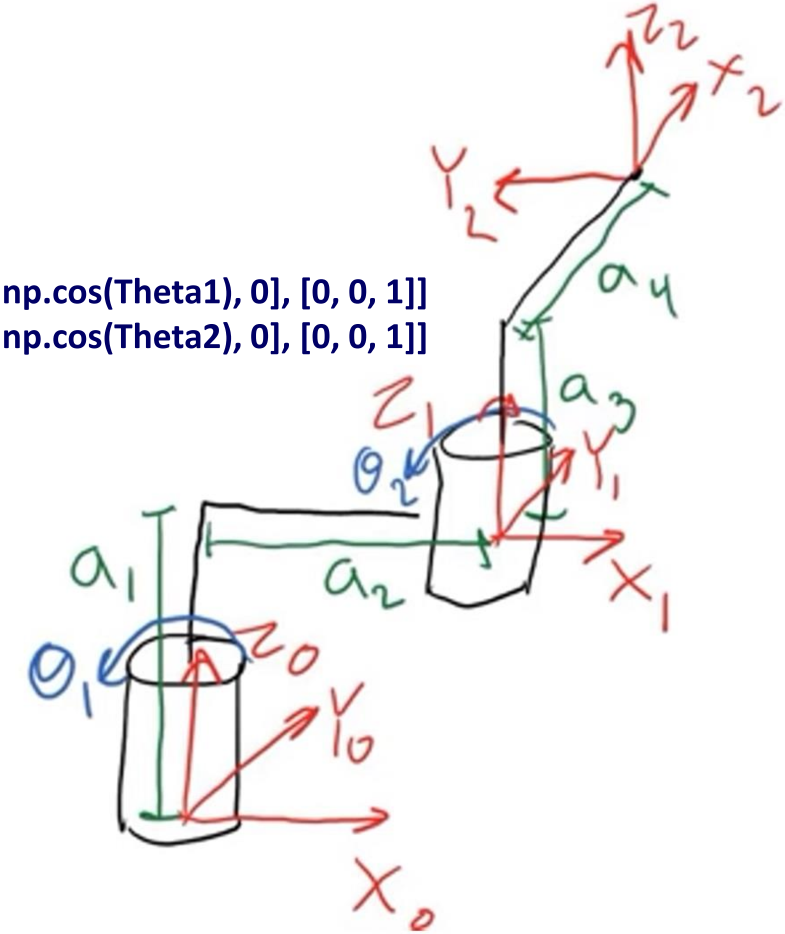
```
[ [ 6.123234e-17 -1.000000e+00  0.000000e+00]

  [ 1.000000e+00  6.123234e-17  0.000000e+00]

  [ 0.000000e+00  0.000000e+00  1.000000e+00] ]
```



Rotation matrix for 2 DoF with Python example

- At Theta1 0 show there is no rotation of frame 2 relative to frame 0.

```
import numpy as np
Theta1 = 0
Theta2 = 90

Theta1 = (Theta1/180) * np.pi
Theta2 = (Theta2/180) * np.pi

R0_1 = [[np.cos(Theta1), -np.sin(Theta1), 0], [np.sin(Theta1), np.cos(Theta1), 0], [0, 0, 1]]
R1_2 = [[np.cos(Theta2), -np.sin(Theta2), 0], [np.sin(Theta2), np.cos(Theta2), 0], [0, 0, 1]]

R0_2 = np.dot(R0_1, R1_2)
print(R0_2)
```



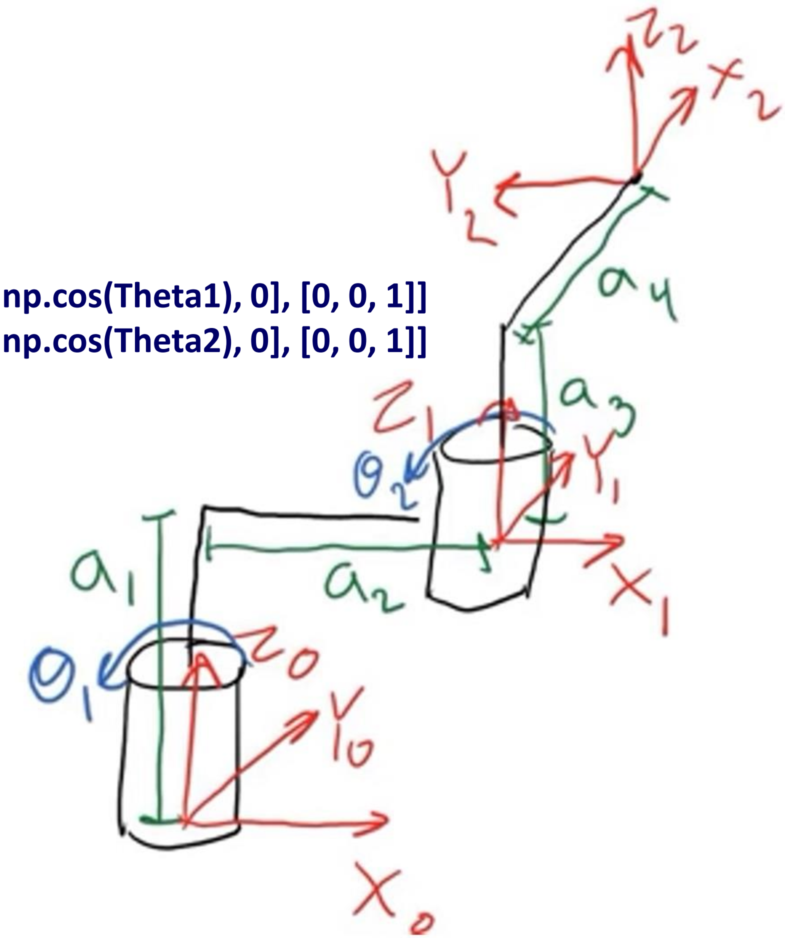

```
[ [ 6.123234e-17  -1.000000e+00  0.000000e+00]

  [ 1.000000e+00   6.123234e-17  0.000000e+00]

  [ 0.000000e+00   0.000000e+00  1.000000e+00] ]
```

Rotation matrix for 2 DoF with Python example

- At Theta1 0 show there is no rotation of frame 2 relative to frame 0.

```
import numpy as np
Theta1 = 0
Theta2 = 90

Theta1 = (Theta1/180) * np.pi
Theta2 = (Theta2/180) * np.pi

R0_1 = [[np.cos(Theta1), -np.sin(Theta1), 0], [np.sin(Theta1), np.cos(Theta1), 0], [0, 0, 1]]
R1_2 = [[np.cos(Theta2), -np.sin(Theta2), 0], [np.sin(Theta2), np.cos(Theta2), 0], [0, 0, 1]]

R0_2 = np.dot(R0_1, R1_2)
print(R0_2)
```

```
[ [ 6.123234e-17  -1.000000e+00  0.000000e+00]

  [ 1.000000e+00   6.123234e-17  0.000000e+00]

  [ 0.000000e+00   0.000000e+00  1.000000e+00] ]
```

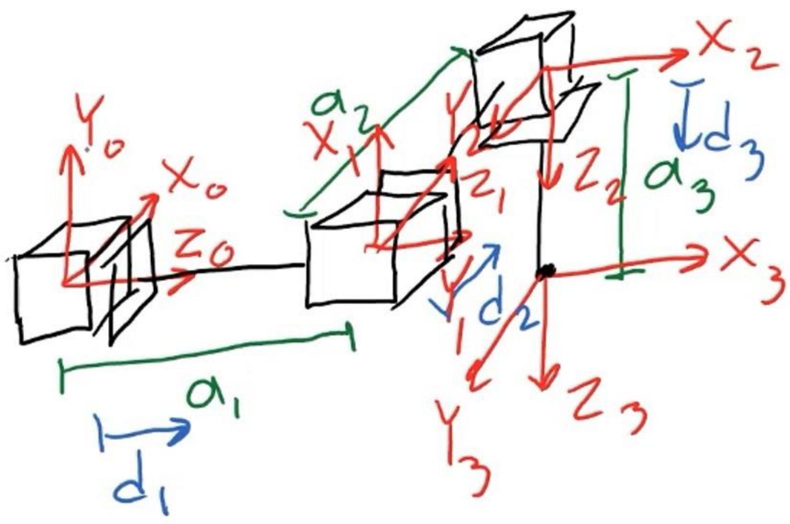


Rotation matrix for Cartesian manipulator

- Now we look for rotation matrices for standard 3 DoF manipulators
- To find the rotation matrix for the end-effector of below manipulator, we use following equation:

$$R_3^o = R_1^o \ R_2^1 \ R_3^2$$

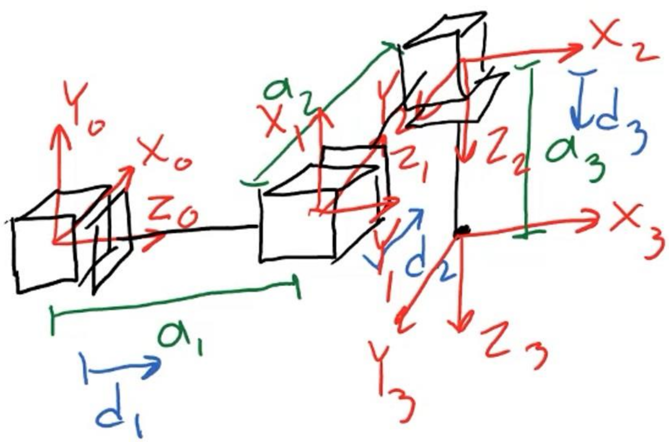
- To find the R_3^o there are two way:
  - To find the combination of rotations and apply standard rotation matrix.
    - e.g,. we rotate frame 0 around its z-axis 90^o and then rotate 90^oaround x-axis and then apply standard R_X and R_Z.
  - The second way is a shortcut using meaning of rotation which is projection of rotated frame on the original frame.



Rotation matrix for Cartesian manipulator

$$R_3^o = R_1^o \; R_2^1 \; R_3^2$$

- To find the R_1^o there are two ways:
  - To find the combination of rotations and apply standard rotation matrix.
    - e.g,. We rotate frame 0 around its z-axis 90^o and then rotate 90^o around x-axis and then apply standard R_X and R_Z.
  - The second way is a shortcut using meaning of rotation which is projection of rotated frame on the original frame.



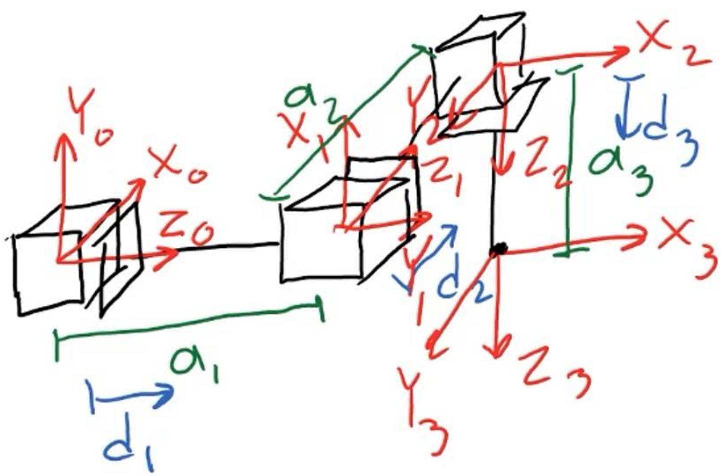
$$R_1^0 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_1^0 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Rotation matrix for Cartesian manipulator

$$R_3^o = R_1^o\ R_2^1\ R_3^2$$

- To find the R_2^1 there are two ways:
  - To find the combination of rotations and apply standard rotation matrix.
  - Here, we are using prismatic joint. Therefore, no rotation is performed.
  - The second way is a shortcut using meaning of rotation which is projection of rotated frame on the original frame.



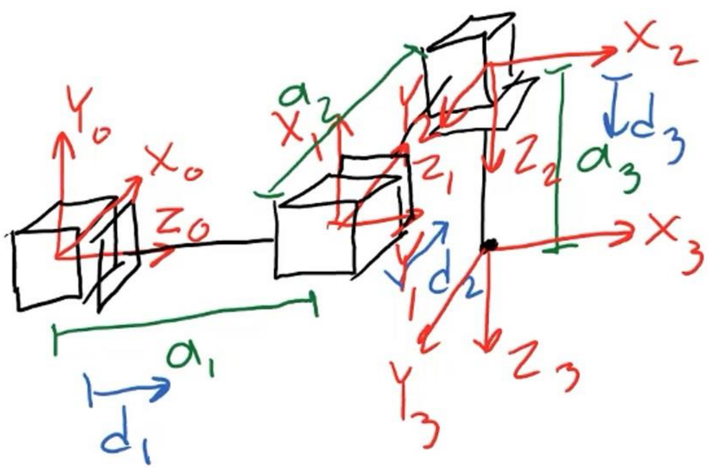
$$R_2^1 = \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_2^1 = \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

Rotation matrix for Cartesian manipulator

$$R_3^o = R_1^o \; R_2^1 \; R_3^2$$

- To find the R_3^2 there are two ways:
  - To find the combination of rotations and apply standard rotation matrix.
  - Here, we are using prismatic joint. Therefore, no rotation is performed.
  - The second way is a shortcut using meaning of rotation which is projection of rotated frame on the original frame.



$$R_3^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

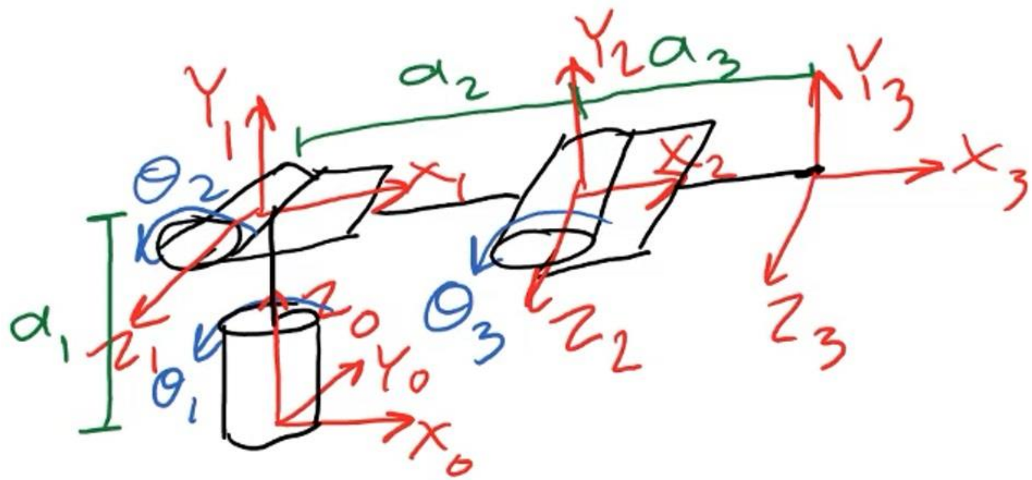
$$R_3^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation matrix for Articulated manipulator

- To find the rotation matrix for 3 DoF articulated manipulator's end-effector, we used following equation:

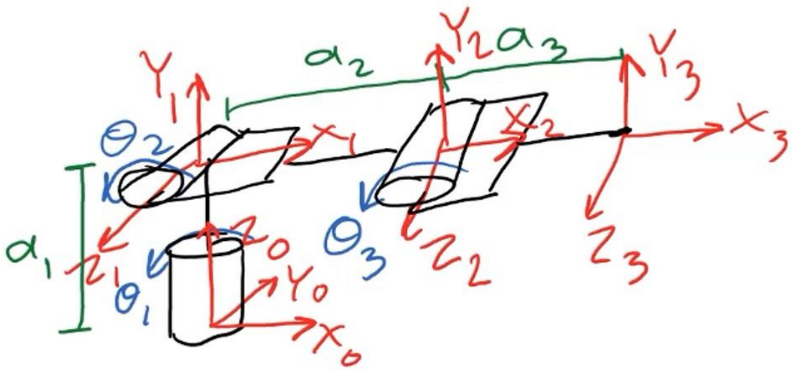
$$R_3^o = R_1^o \; R_2^1 \; R_3^2$$

- To find the R_3^o there are two way:



Rotation matrix for Articulated manipulator

- To find R_1^o
- Firstly, we need to find the matrix that represents the rotation of frame 1 relative to frame 0 before the joint variables has moved (use shortcut method).
- Secondly, we need to find the matrix that represents the rotation due to joint variable.
  - Here, we must be careful in choosing the axis of rotation in such cases like this one. Θ_1 is rotation around Z_0-axis and Y_1-axis. The questions arises here is which standard rotation matrix we use R_y or R_z.
  - The answer is, we can use any of R_y and R_z. However, we need to place R_y on the right side of first matrix and R_z on the left side.

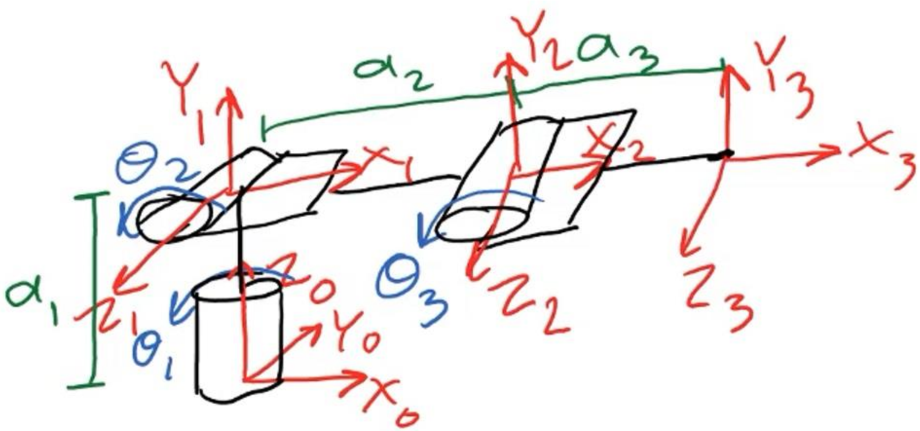


Z Y

$$R_1^0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

Rotation matrix for Articulated manipulator

- To find R_1^o
- Firstly, we need to find the matrix that represents the rotation of frame 1 relative to frame 0 before the joint variables has moved (use shortcut method).
- Secondly, we need to find the matrix that represents the rotation due to joint variable.
  - Here, we must be careful in choosing he axis of rotation in such cases like this one. Θ_1 is rotation around Z_0-axis and Y_1-axis. The questions arises here is which standard rotation matrix we use R_y or R_z.
  - We cane use any of R_y and R_z. However, we need to place R_y on the right side of first matrix and R_z on the left side.

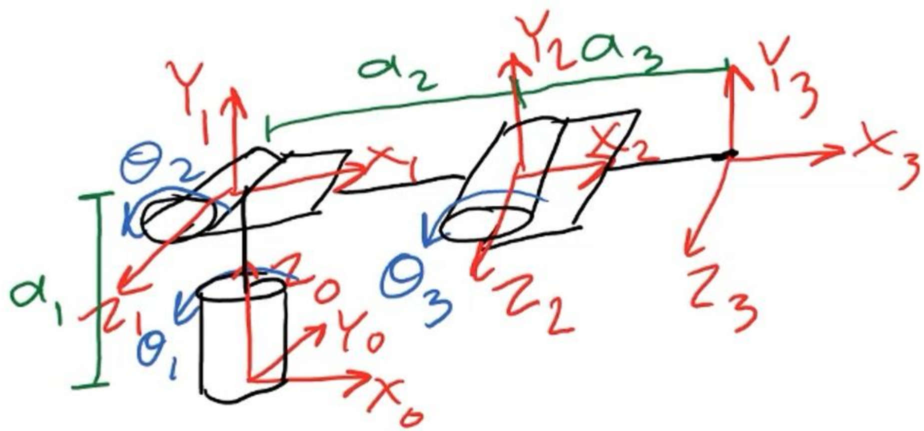


$$R_1^0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 \\ 0 & 1 & 0 \\ -\sin\theta_1 & 0 & \cos\theta_1 \end{bmatrix}$$

$$R_1^0 = \begin{bmatrix} C\theta_1 & 0 & S\theta_1 \\ S\theta_1 & 0 & -C\theta_1 \\ 0 & 1 & 0 \end{bmatrix}$$

Rotation matrix for Articulated manipulator

- To find R_2^1
- Firstly, we need to find the matrix that represents the rotation of frame 1 relative to frame 0 before the joint variables has moved (use shortcut method).
- Secondly, we need to find the matrix that represents the rotation due to joint variable.
  - Here, we must be careful in choosing he axis of rotation in such cases like this one. Θ_1 is rotation around Z_0-axis and Y_1-axis. The questions arises here is which standard rotation matrix we use R_y or R_z.
  - We cane use any of R_y and R_z. However, we need to place R_y on the right side of first matrix and R_z on the left side.

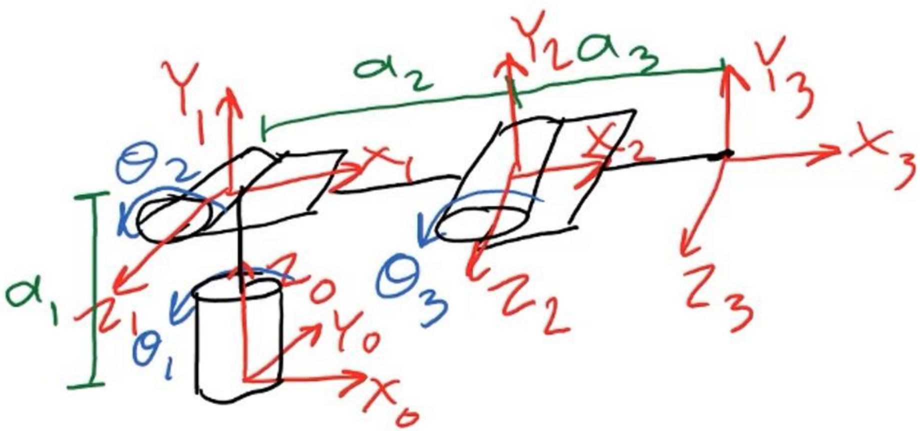


$$R_2^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta_2 & -S\theta_2 & 0 \\ S\theta_2 & C\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_2^1 = \begin{bmatrix} C\theta_2 & -S\theta_2 & 0 \\ S\theta_2 & C\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation matrix for Articulated manipulator

- To find R_3^2
- Firstly, we need to find the matrix that represents the rotation of frame 1 relative to frame 0 before the joint variables has moved (use shortcut method).
- Secondly, we need to find the matrix that represents the rotation due to joint variable.
  - Here, we have to be careful in choosing he axis of rotation in such cases like this one. Θ_1 is rotation around Z_0-axis and Y_1-axis. The questions arises here is which standard rotation matrix we use R_y or R_z.
  - We cane use any of R_y and R_z. However, we need to place R_y on the right side of first matrix and R_z on the left side.

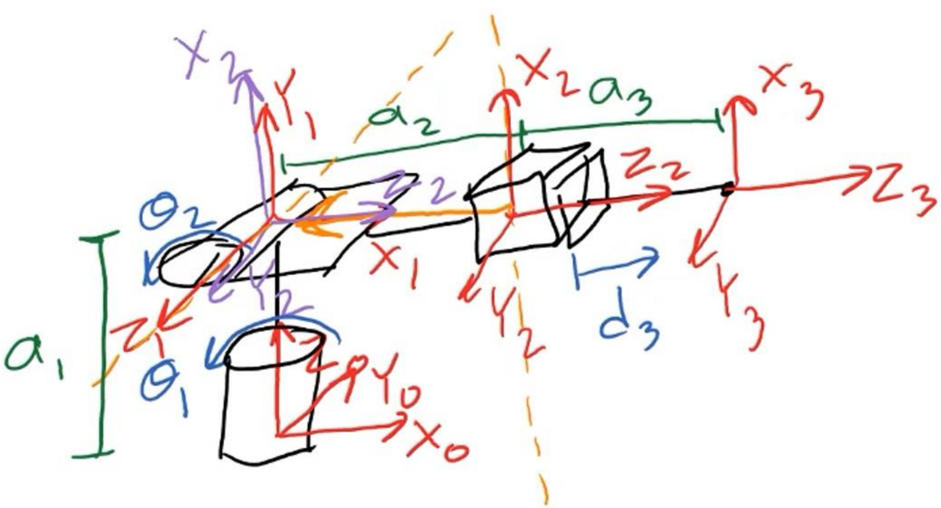


$$R_3^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta_3 & -S\theta_3 & 0 \\ S\theta_3 & C\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_3^2 = \begin{bmatrix} C\theta_3 & -S\theta_3 & 0 \\ S\theta_3 & C\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation matrix for Spherical manipulator

$$R_3^o = R_1^o\ R_2^1\ R_3^2$$



$Z \quad Y$

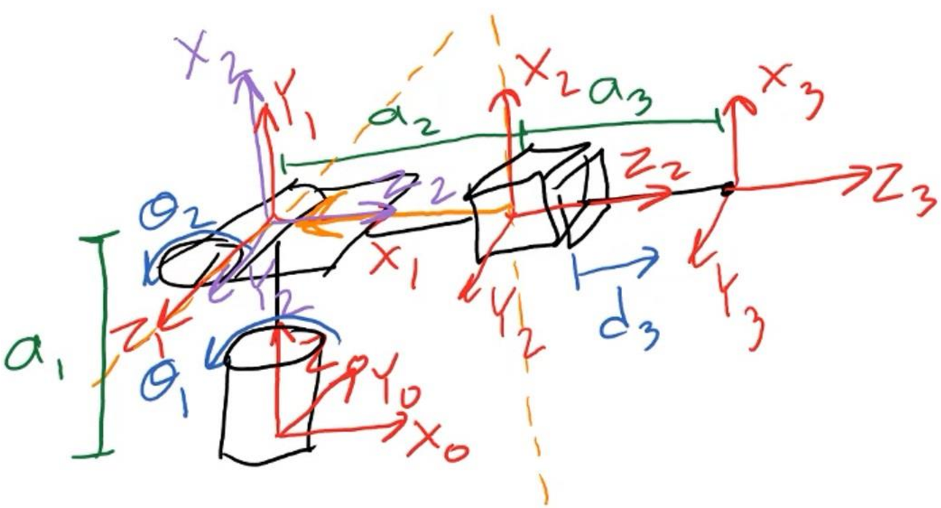
$$R_1^0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} c\theta_1 & -s\theta_1 & 0 \\ s\theta_1 & c\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$R_1^0 = \begin{bmatrix} c\theta_1 & 0 & s\theta_1 \\ s\theta_1 & 0 & -c\theta_1 \\ 0 & 1 & 0 \end{bmatrix}$$

Rotation matrix for spherical manipulator

- To find R_2^1 . We would use the purple colour frame 2 instead of red one.

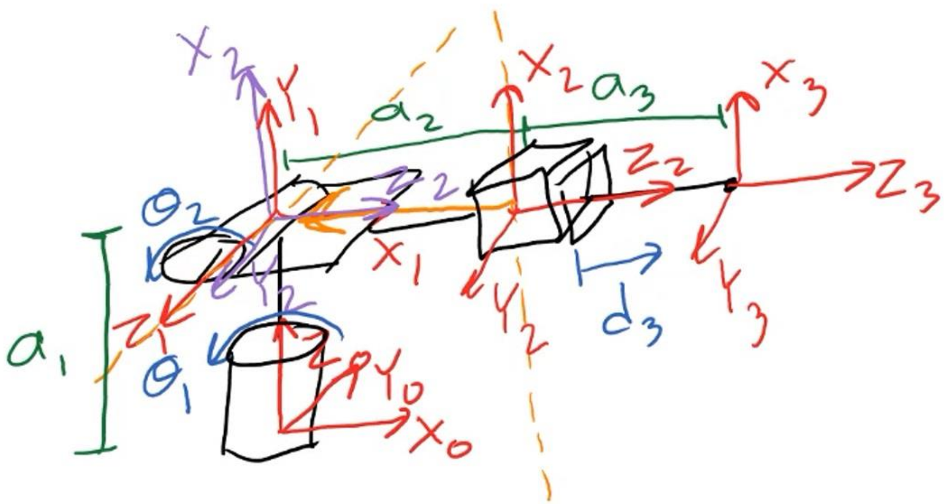


$$Z \qquad Y$$

$$R_2^1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} c\theta_2 & 0 & s\theta_2 \\ 0 & 1 & 0 \\ -s\theta_2 & 0 & c\theta_2 \end{bmatrix}$$

$$R_2^1 = \begin{bmatrix} -s\theta_2 & 0 & c\theta_2 \\ c\theta_2 & 0 & s\theta_2 \\ 0 & 1 & 0 \end{bmatrix}$$

Rotation matrix for spherical manipulator

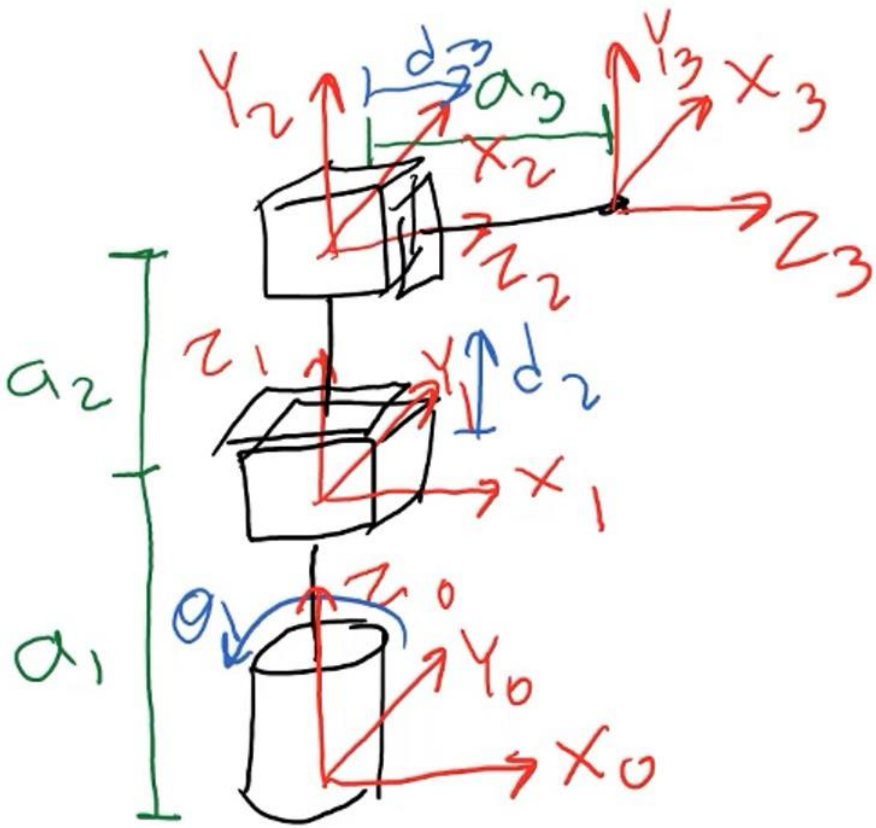


$$R^2_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^2_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation matrix for cylindrical manipulator

- Finding R_1^0

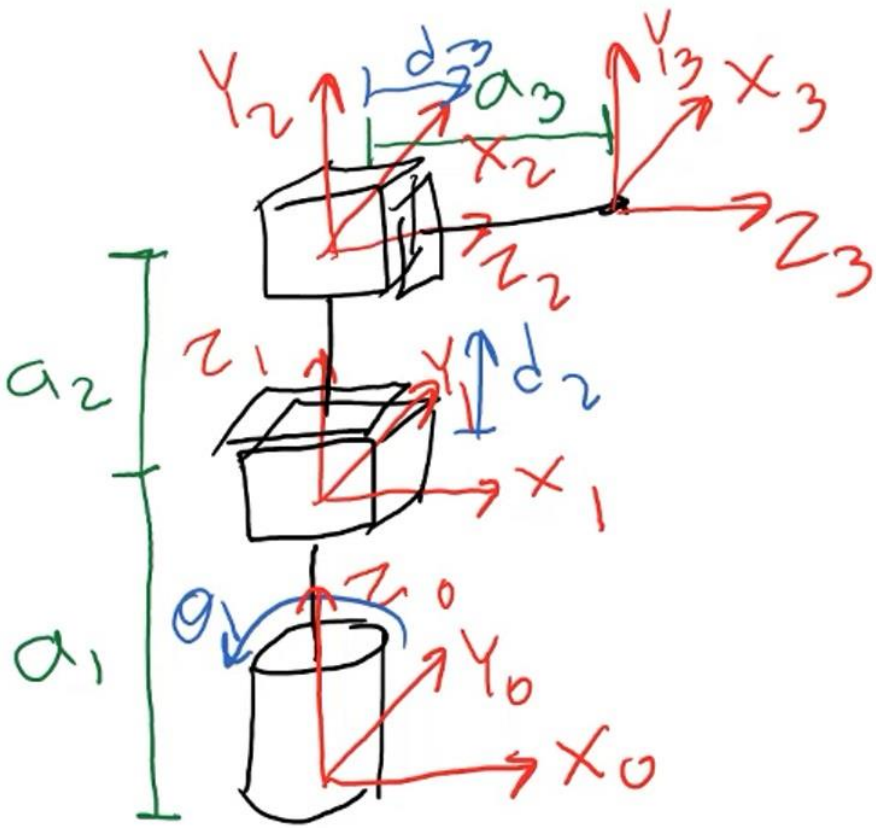


$$R_1^0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 \\ s\theta_1 & c\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_1^0 = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 \\ s\theta_1 & c\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation matrix for cylindrical manipulator

- Finding R_2^1 and R_3^2

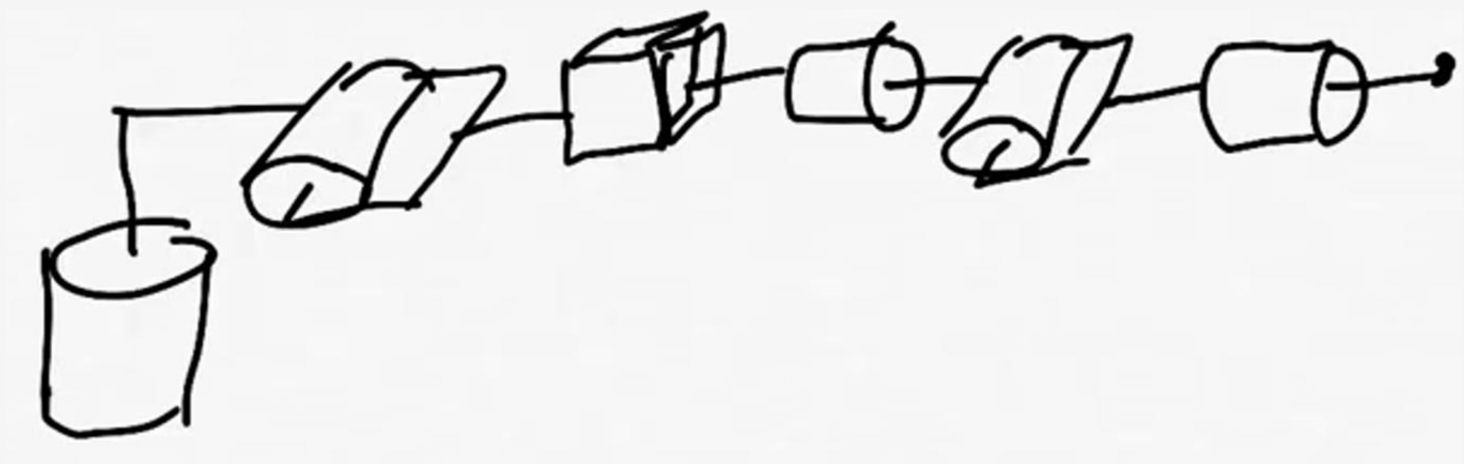


$$R_2^1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$R_3^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

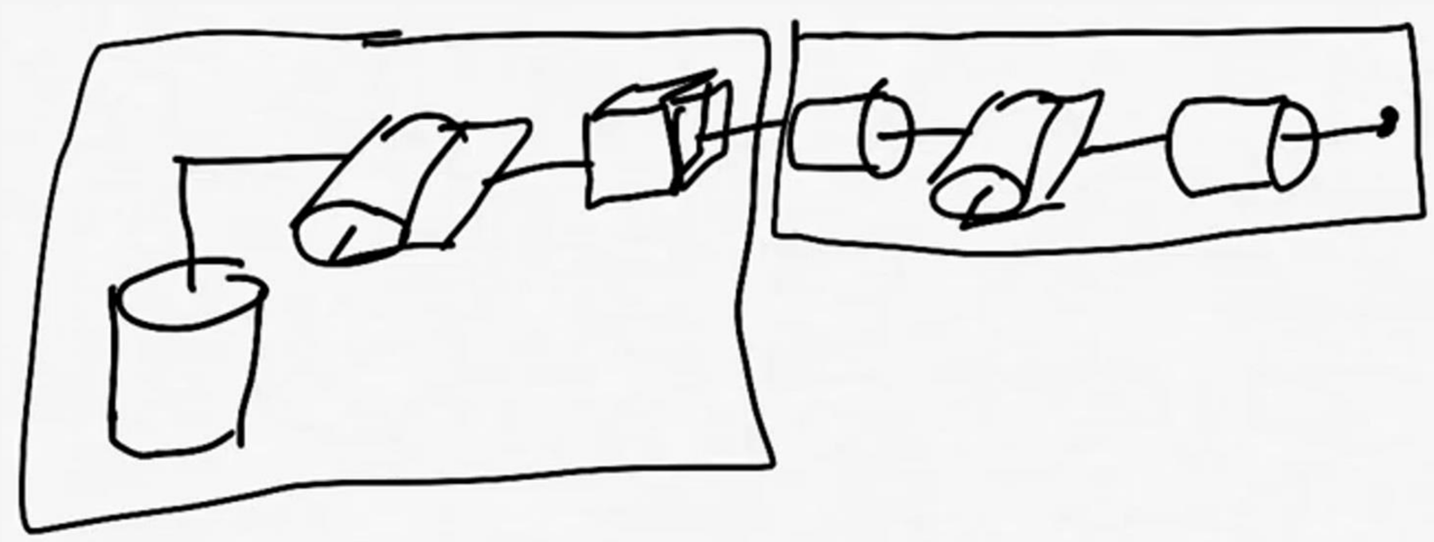
Rotation matrix for 6 DoFmanipulator

- Which standard manipulator it is?
  - Half of the manipulator is spherical
  - The later half is spherical wrist



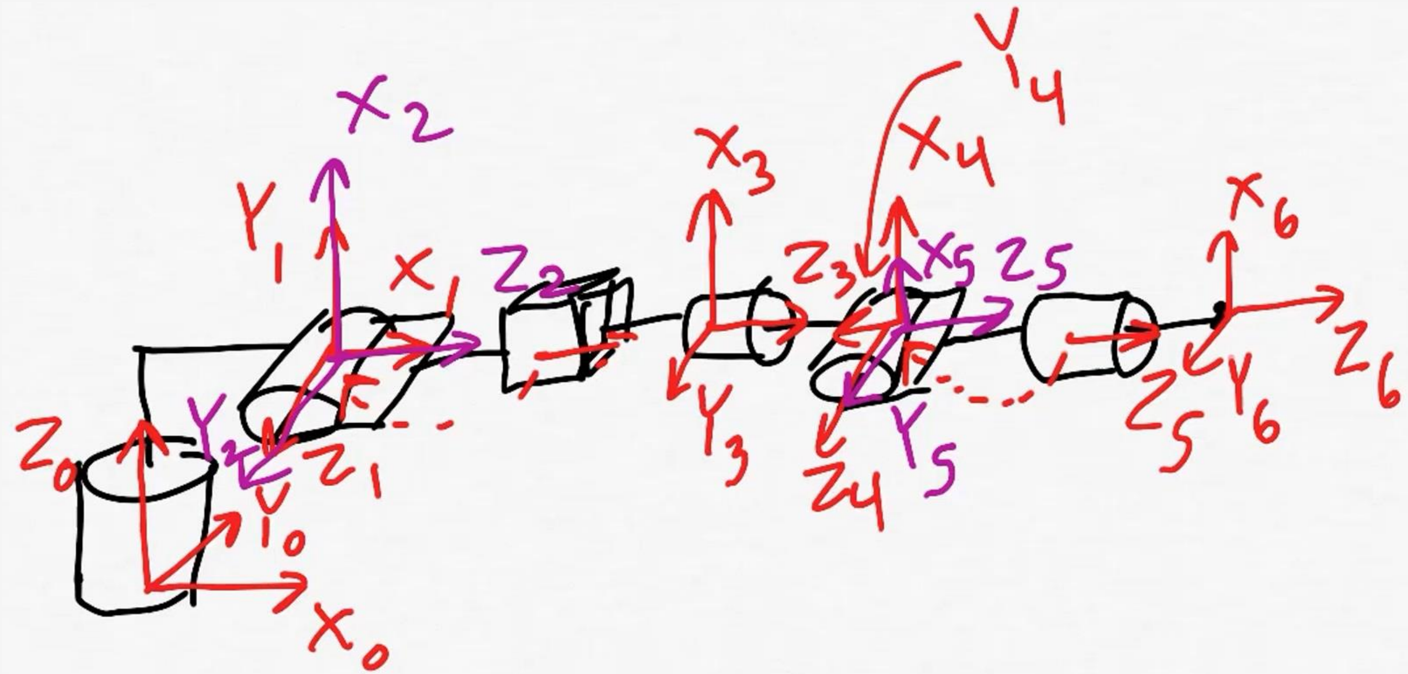
Rotation matrix for 6 DoFmanipulator

- Which standard manipulator it is?



Rotation matrix for 6 DoFmanipulator

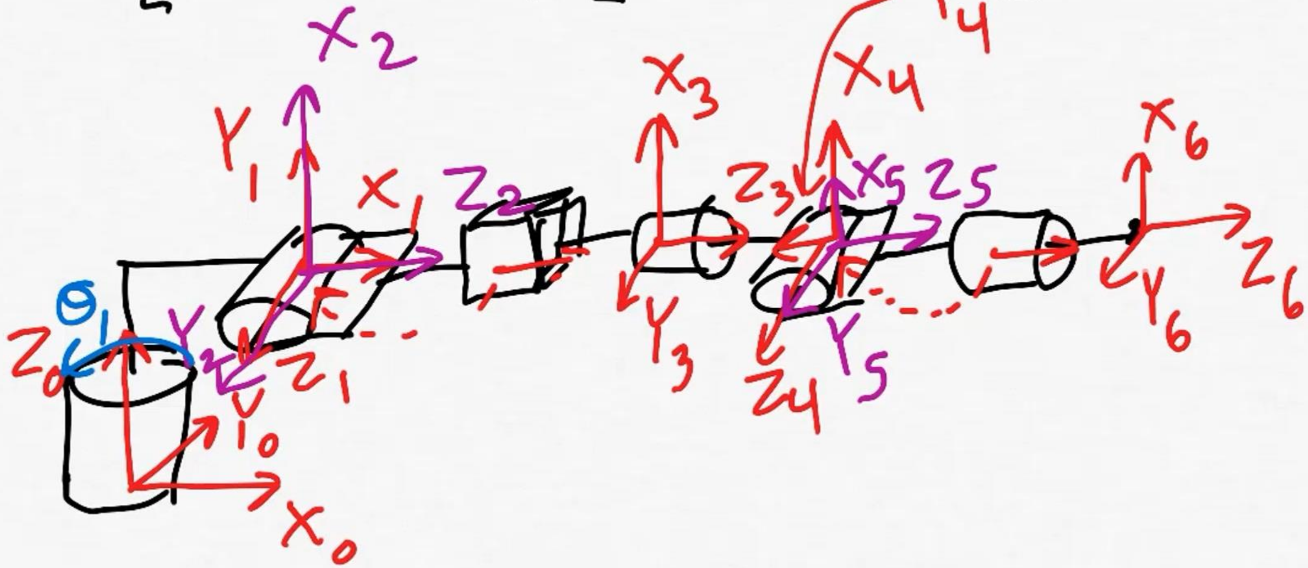
- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

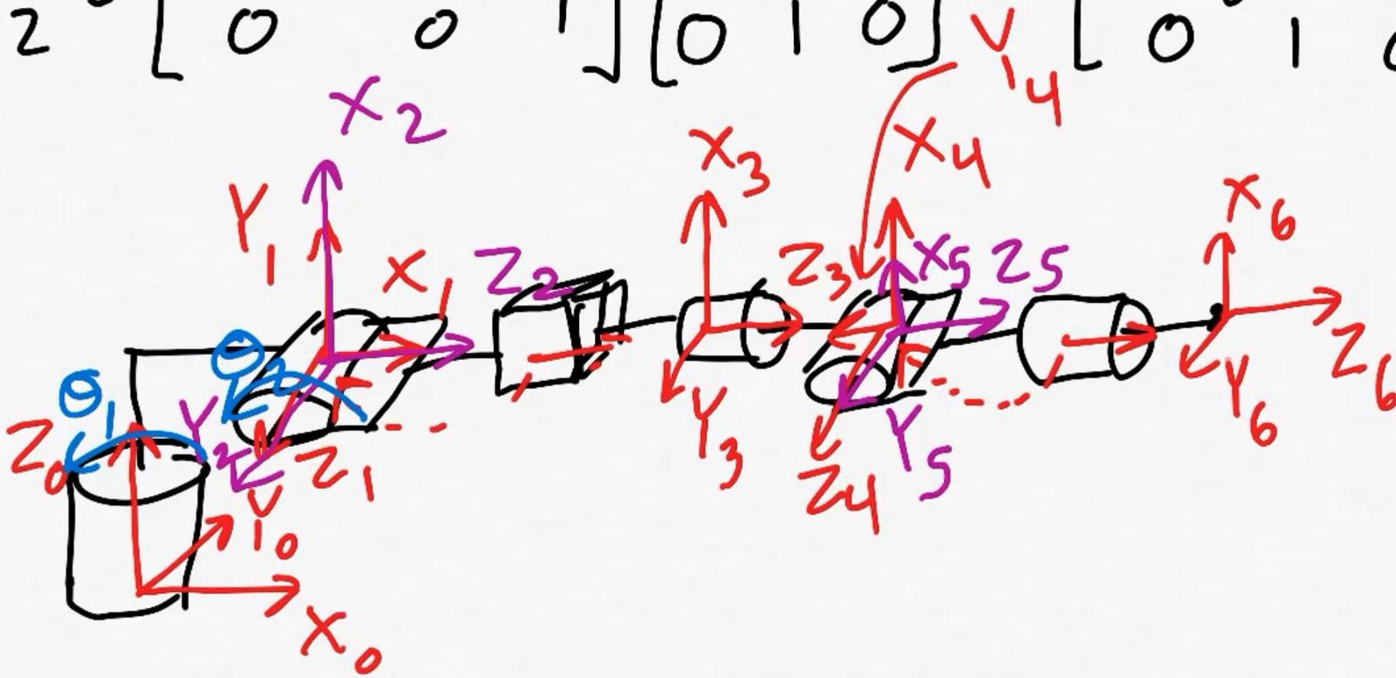
$$R_1^0 = \begin{bmatrix} C\theta_1 & -S\theta_1 & 0 \\ S\theta_1 & C\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} C\theta_1 & 0 & S\theta_1 \\ S\theta_1 & 0 & -C\theta_1 \\ 0 & 1 & 0 \end{bmatrix}$$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

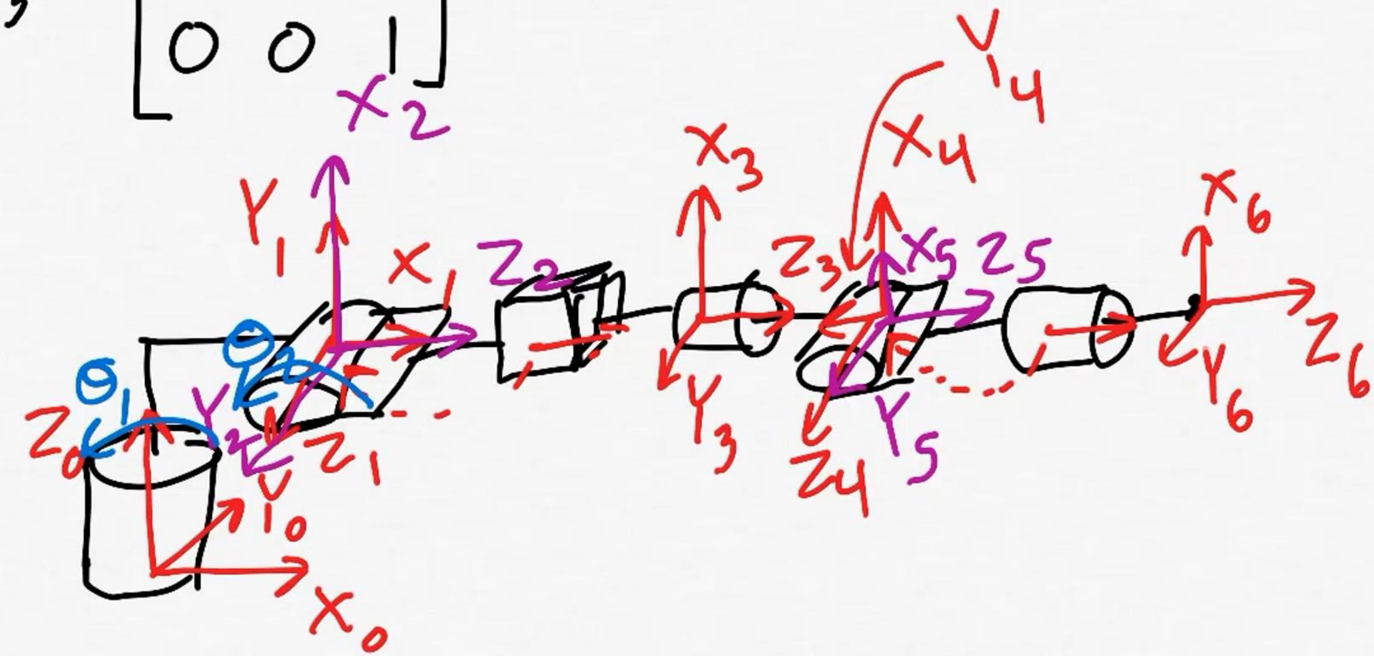
$$R_2^1 = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 \\ s\theta_2 & c\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -s\theta_2 & 0 & c\theta_2 \\ c\theta_2 & 0 & s\theta_2 \\ 0 & 1 & 0 \end{bmatrix}$$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

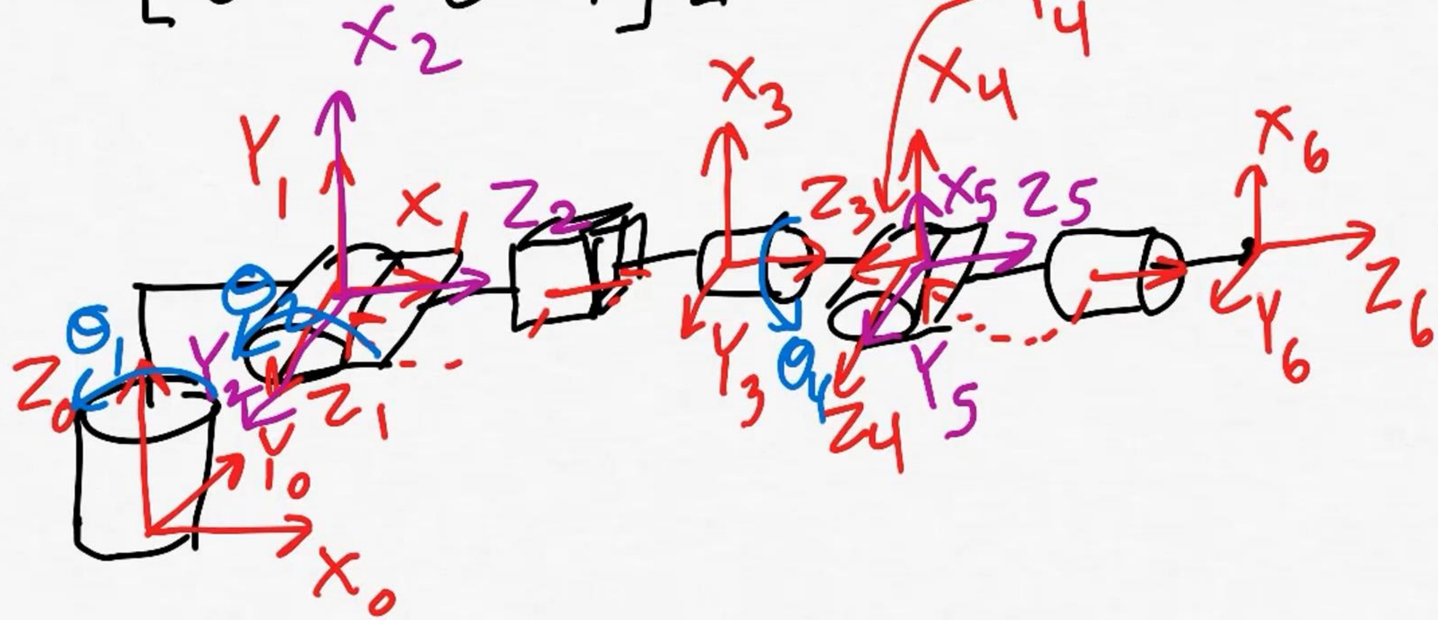
$$R_3^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

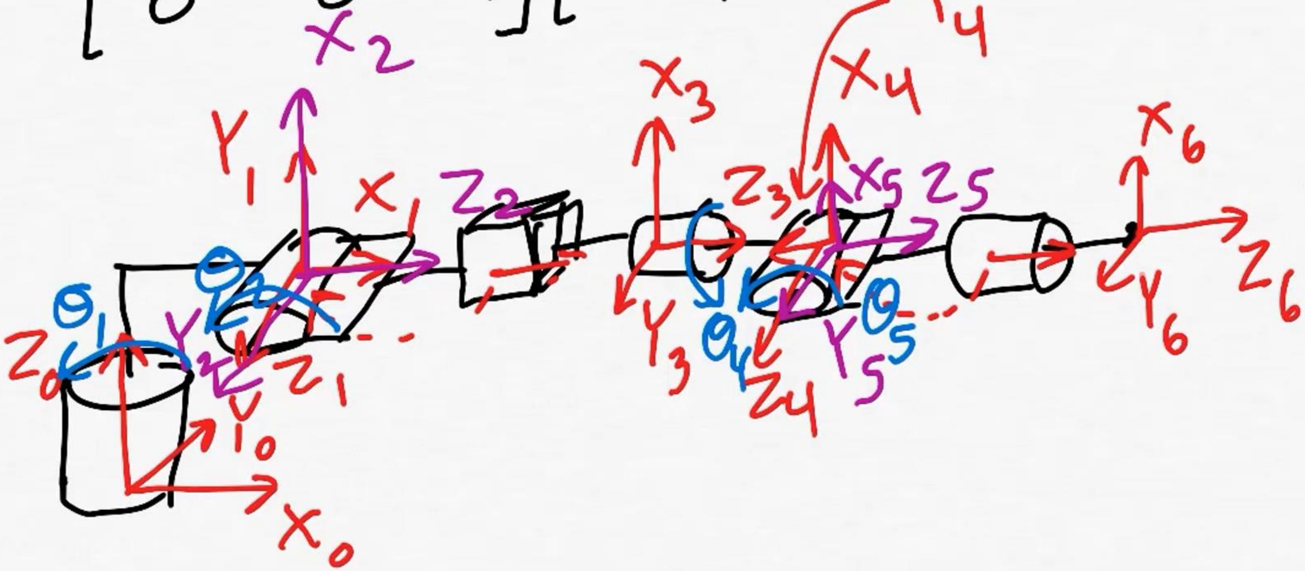
$$R_4^3 = \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 \\ s\theta_4 & c\theta_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} c\theta_4 & 0 & -s\theta_4 \\ s\theta_4 & 0 & c\theta_4 \\ 0 & -1 & 0 \end{bmatrix}$$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

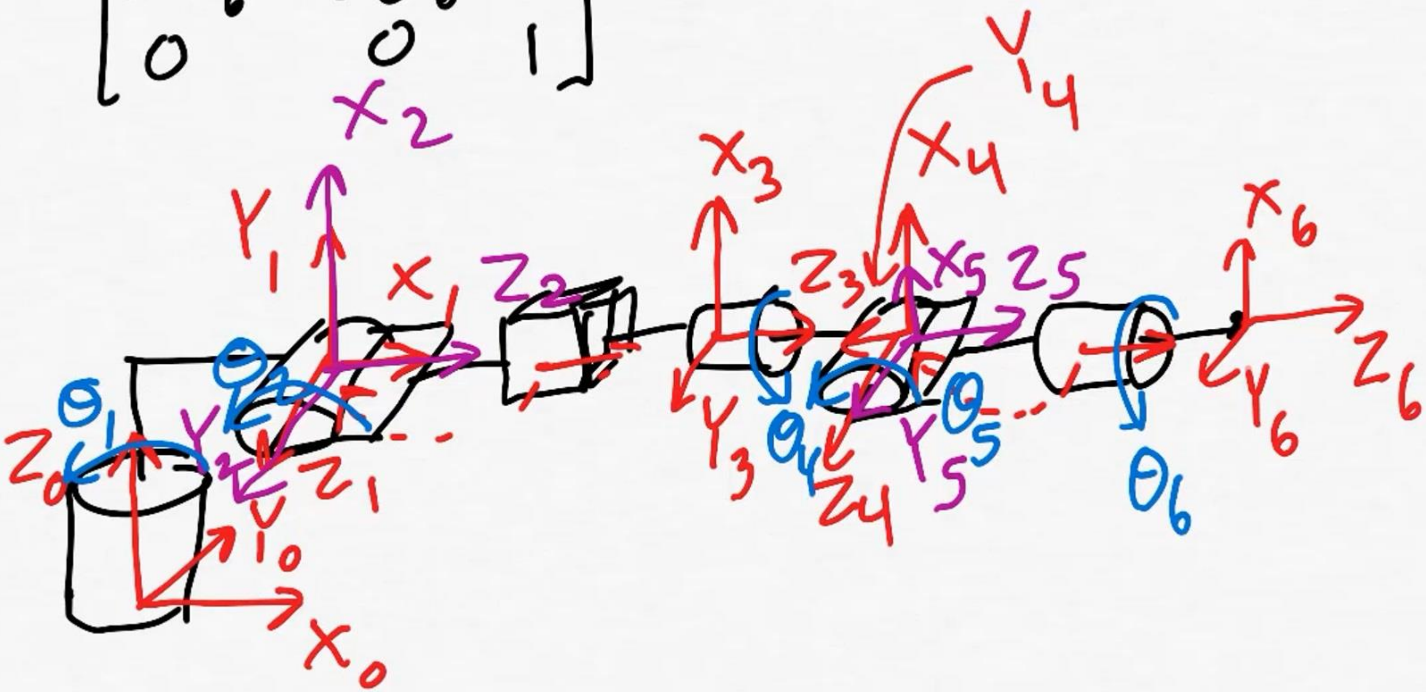
$$R_5^4 = \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 \\ s\theta_4 & c\theta_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} c\theta_4 & 0 & s\theta_4 \\ s\theta_4 & 0 & -c\theta_4 \\ 0 & 1 & 0 \end{bmatrix}$$



Rotation matrix for 6 DoFmanipulator

- $R_6^o = R_1^o\ R_2^1\ R_3^2\ R_4^3\ R_5^4\ R_6^5$

$$R_6^5 = \begin{bmatrix} C\theta_6 & -S\theta_6 & 0 \\ S\theta_6 & C\theta_6 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



End