EE-381 Robotics-1

UG ELECTIVE COURSE

Lecture 2

Dr. Hafsa Iqbal
Department of Electrical Engineering,
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology,
Pakistan

Last Lecture

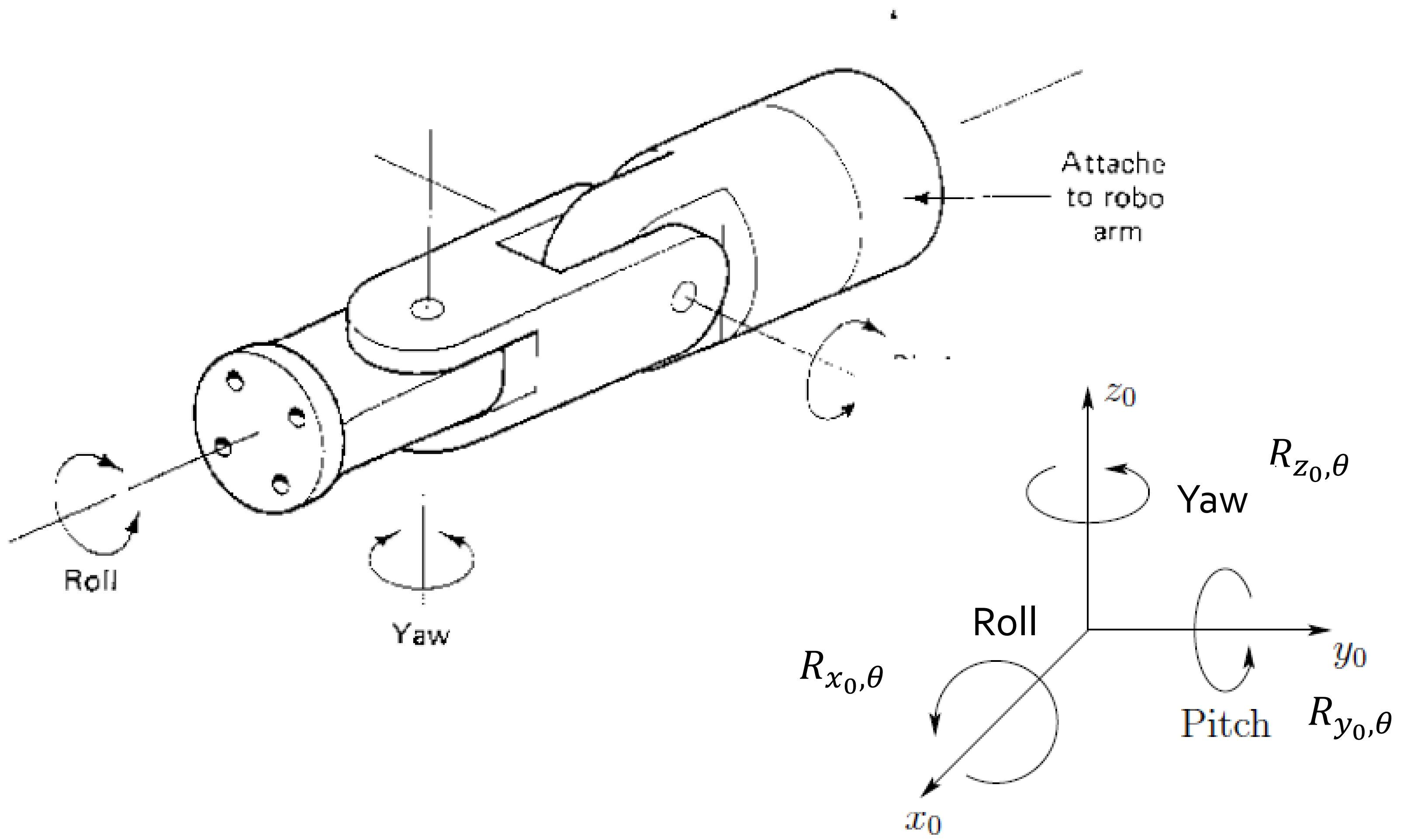
Enrollment Code: **983675410**

- Introduction to Robotics: Definition, history
- Robot accessories; joints
- Classification of Robots; power source, application, control systems, geometry, method of control

Todays Lecture Agenda

- Robot Configurations
- Robot Programming

Wrist


Attache
to robo
arm
Roll
Yaw
z_0
$R_{z_0,\theta}$
Yaw
Roll
$R_{x_0,\theta}$
y_0
Pitch
$R_{y_0,\theta}$
x_0

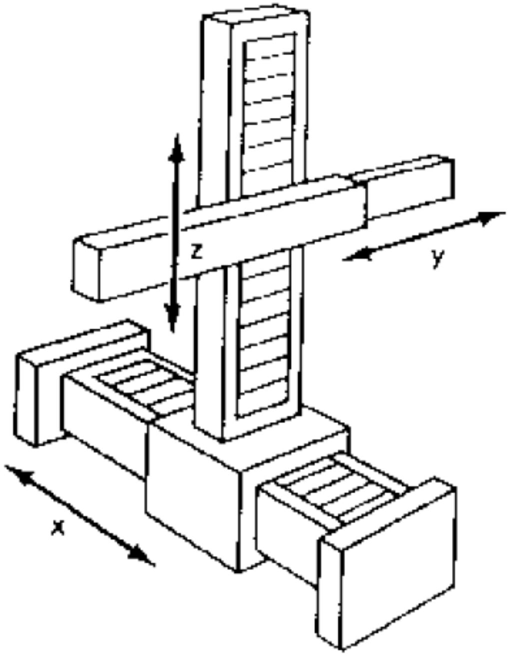
Robot Configurations

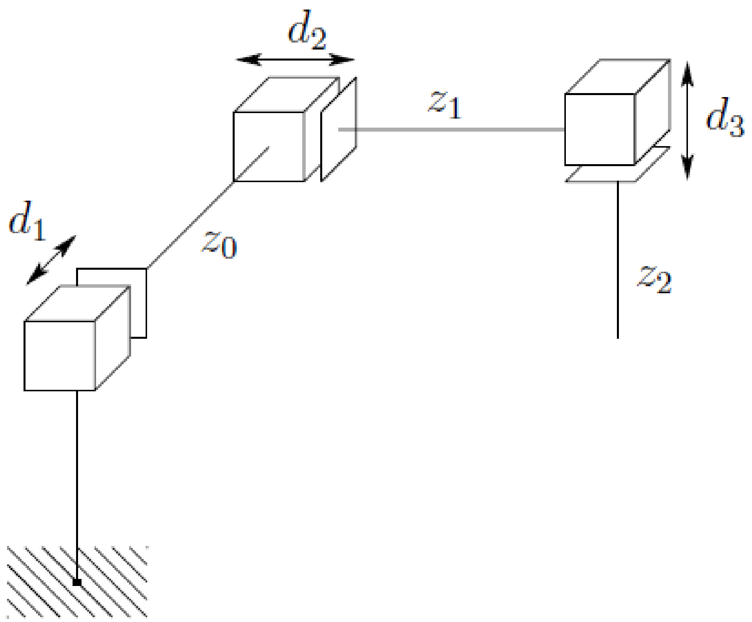
Based on coordinate system

1. Cartesian/Rectangular Robot (PPP)
2. Cylindrical Robot (RPP)
3. Spherical Robot (RRP)
4. Articulated Robot (RRR)
5. SCARA (special types of spherical) (RRP)

1-Cartesian Robot (PPP)

- 3 Prismatic Joints that orient the end effector, which are usually followed by additional revolute joints





Configuration of Cartesian Robot

https://www.youtube.com/watch?v=ci_mpRERMog

1-Cartesian Robot

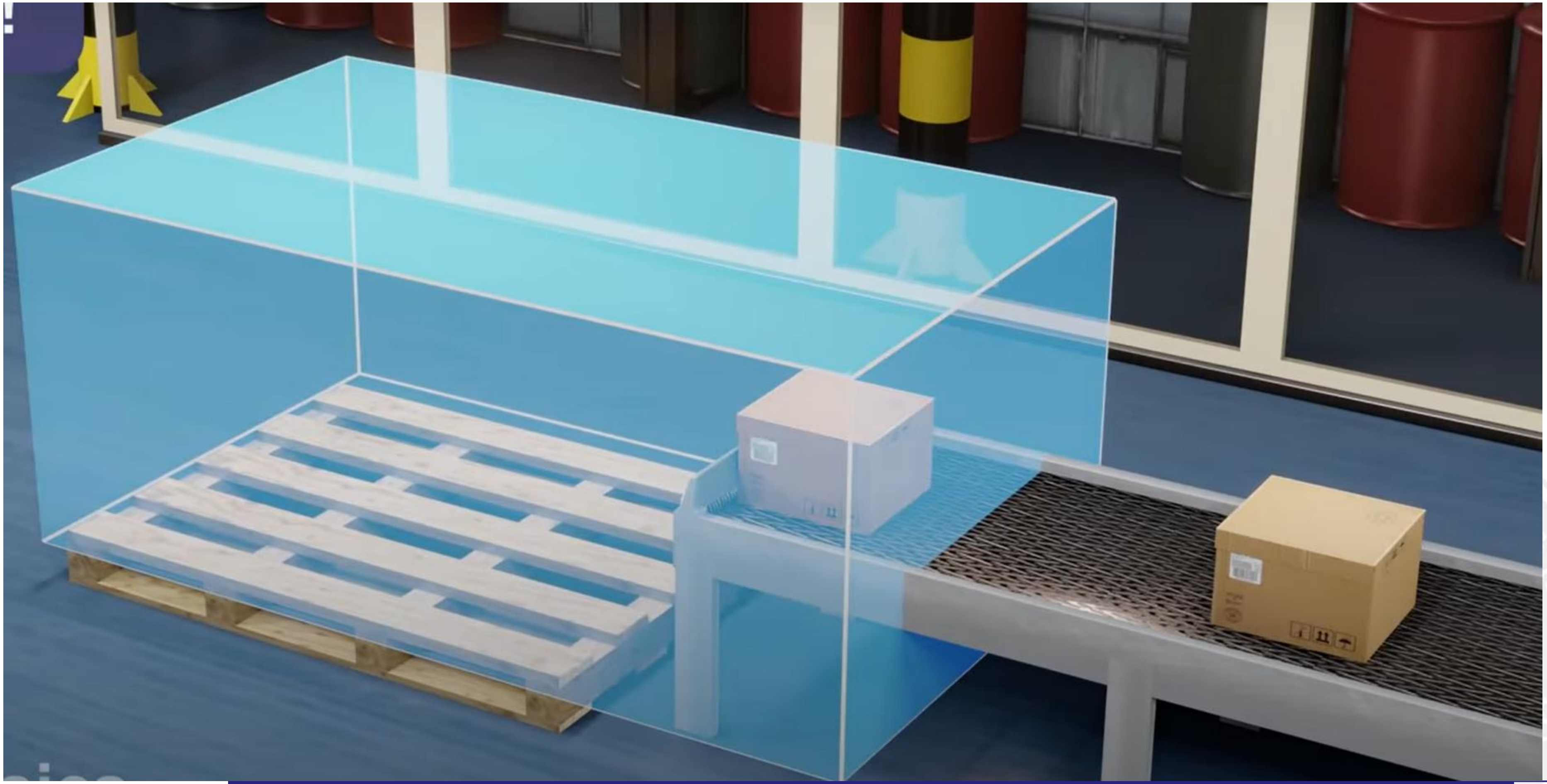
- **Advantages**
  - Simple configuration
  - Equal & constant spatial resolution
  - Use for assembly applications and transfer of material or cargo

Epson Cartesian Robot

- **Disadvantages**
  - Lacks mechanical flexibility
  - Cannot reach objects on the floor
  - Speed of operation in horizontal plane is slower than the robots with rotary base

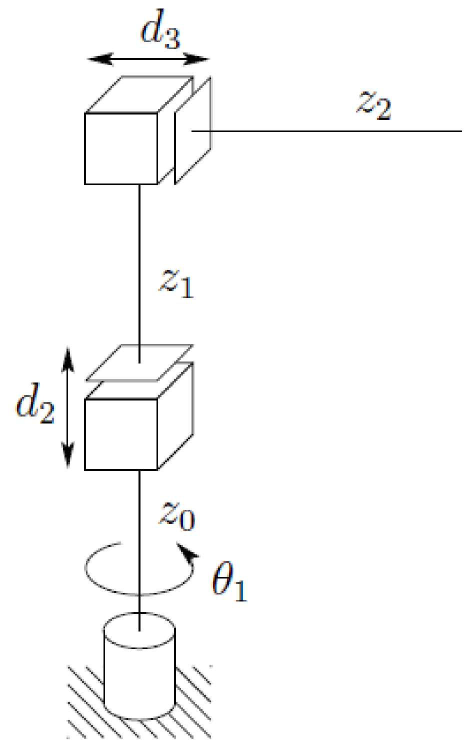
- Guess!

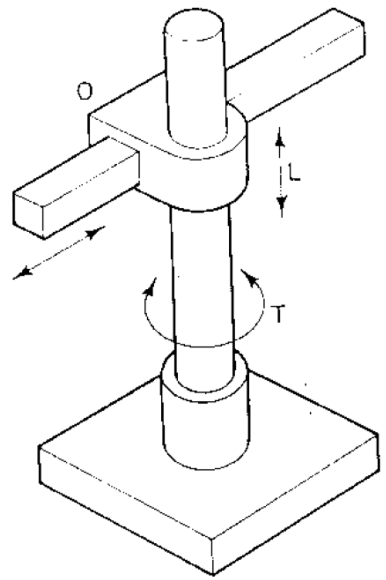
https://www.youtube.com/watch?v=_canCYWZPsc



2-Cylindrical Robot (RPP)

- First joint is revolute and produces a rotation about the base, second and third joints are prismatic

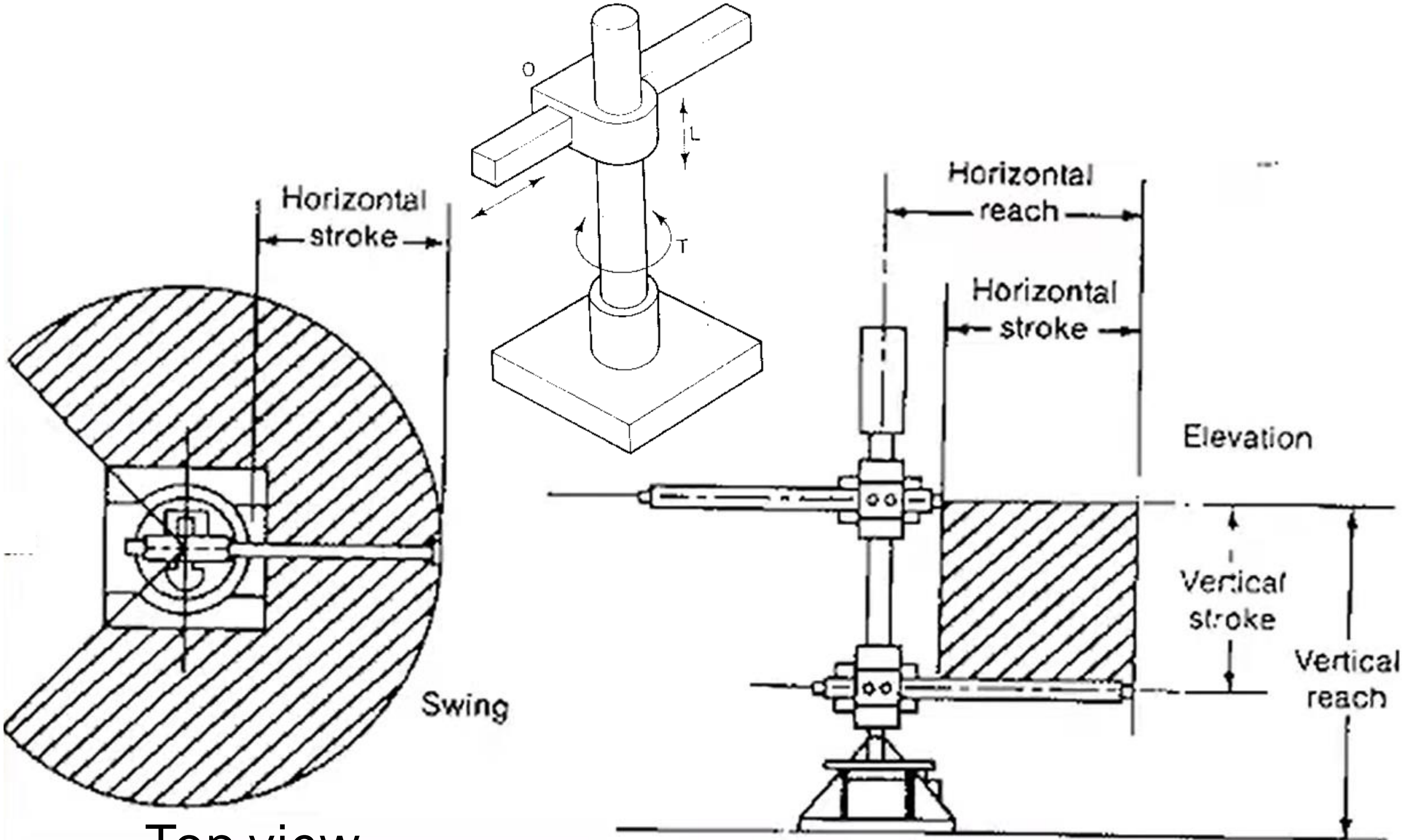




Configuration of Cylindrical Robot

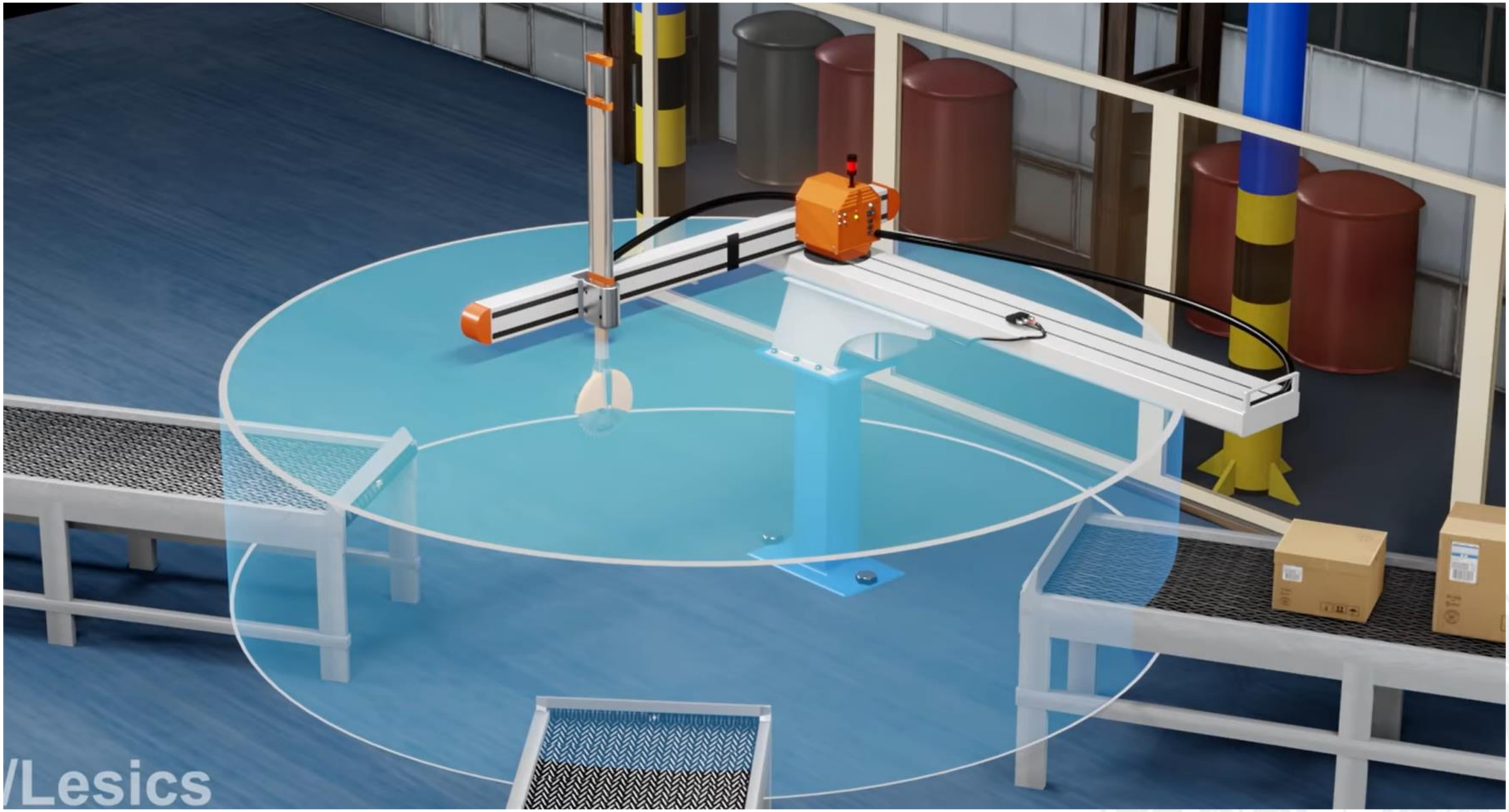
Seiko RT3300 Robot

2-Cylindrical Robot- Work Envelop



Top view

Front view



https://www.youtube.com/watch?v=_canCYWZPsc

2-Cylindrical Robot- Advantages

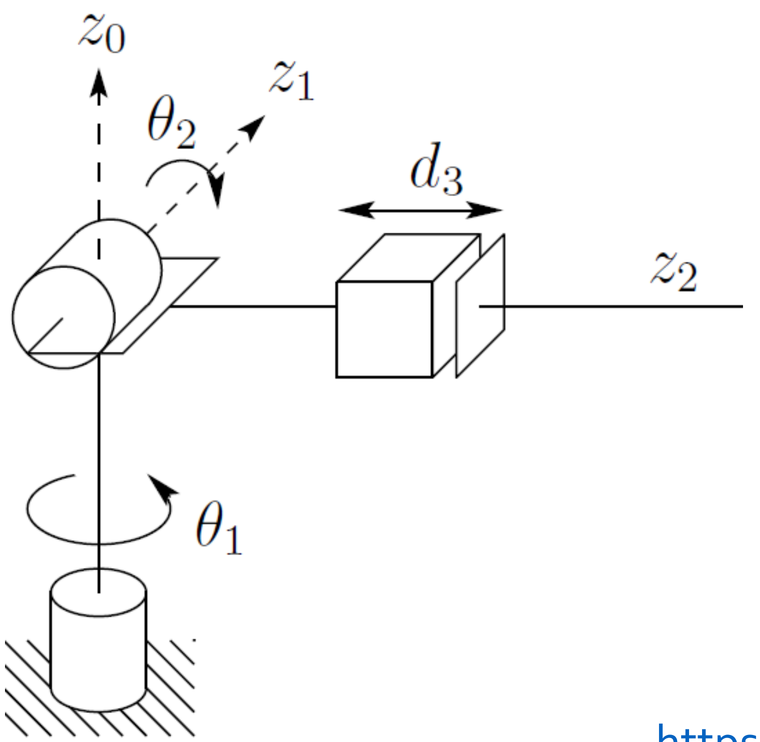
- Results in a larger work envelope than a rectangular robot
- Suited for pick-and-place operations
- Vertical structure preserves the floor space
- Deep horizontal reach is useful for far-reaching operations

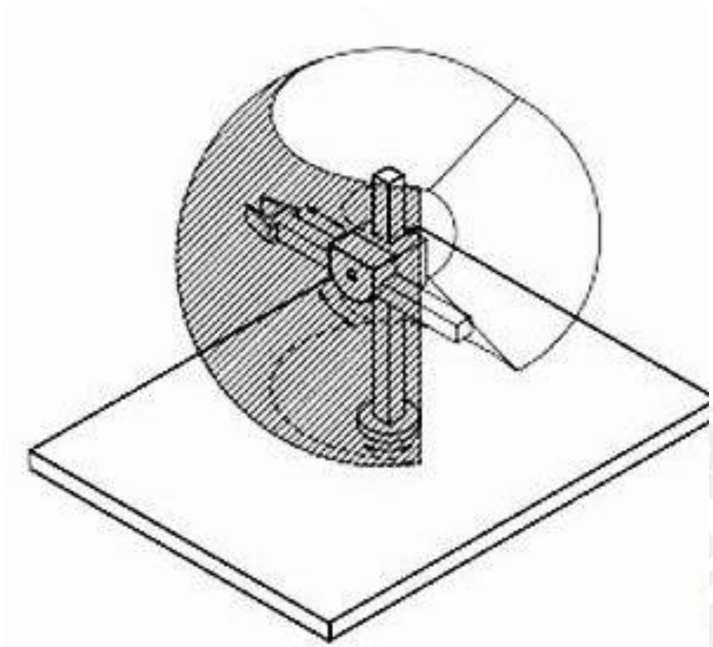
2-Cylindrical Robot- Disadvantages

- Overall mechanical rigidity is lower than that of the rectilinear robots.
- Repeatability and accuracy are also lower in the direction of rotary motion.
- Configuration requires a more sophisticated control system than the rectangular robots.

3-Spherical Robot (RRP)

- Also known as Polar Coordinate Robot
- 2 Revolute and 1 prismatic joint

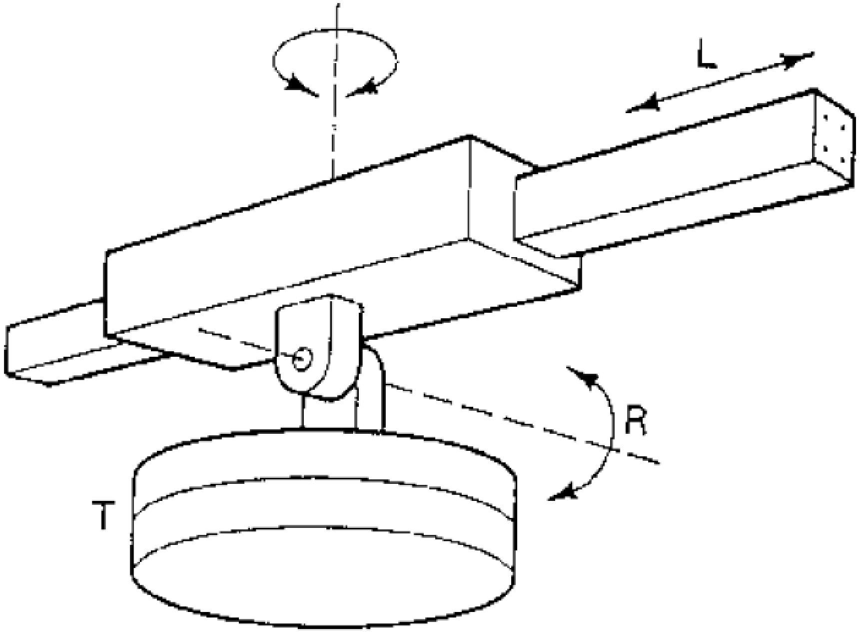




https://www.youtube.com/watch?v=jrF5Dl6ntAc

Configuration of spherical manipulator

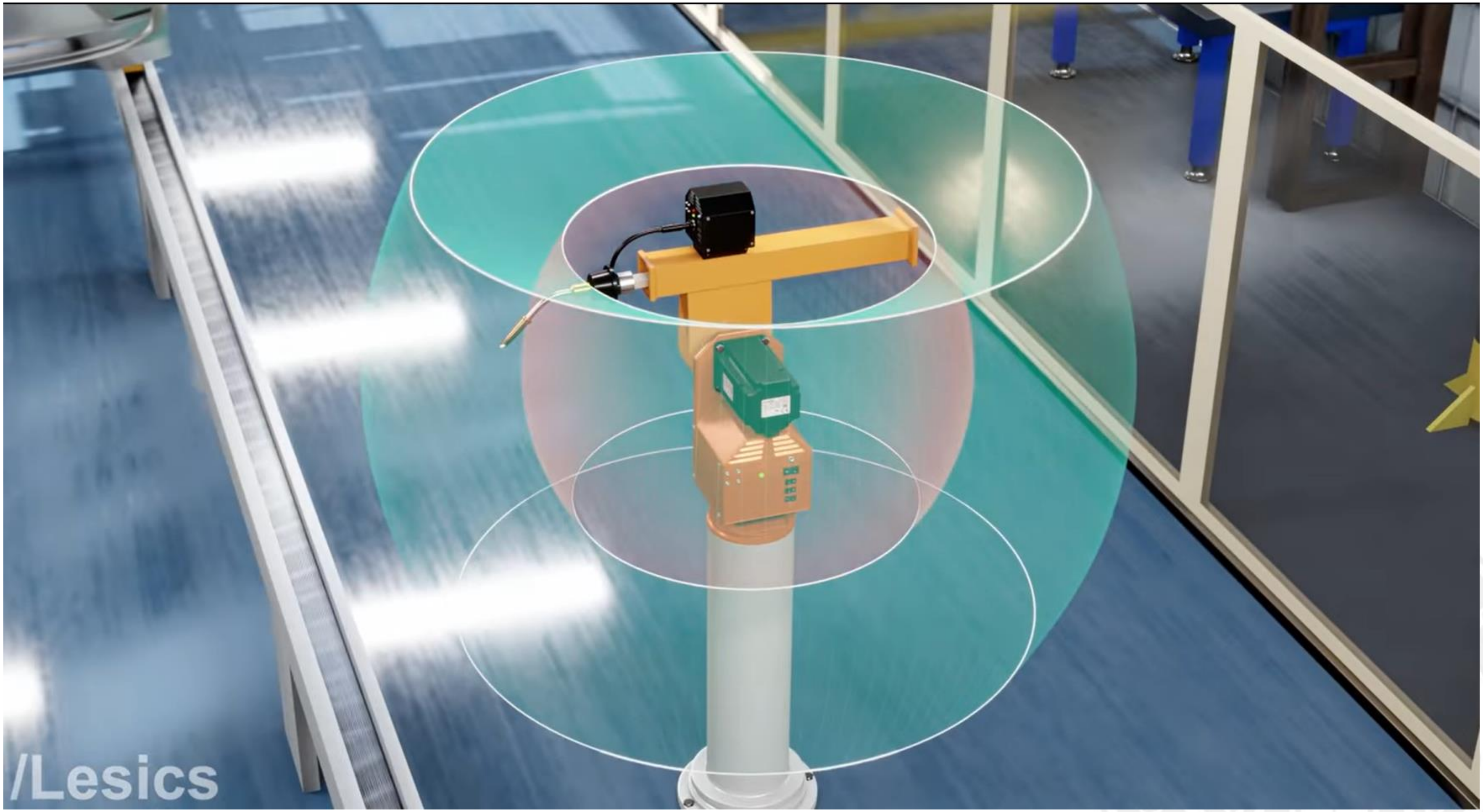
3-Spherical Robot (RRP)



Workspace of spherical manipulator

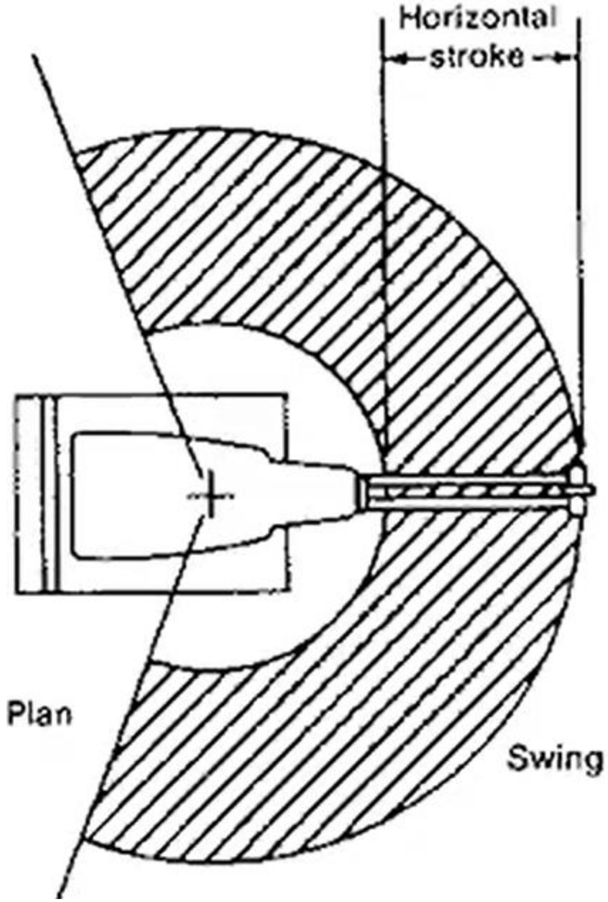
Stanford Arm

3-Spherical Robot (RRP)-Work Envelop

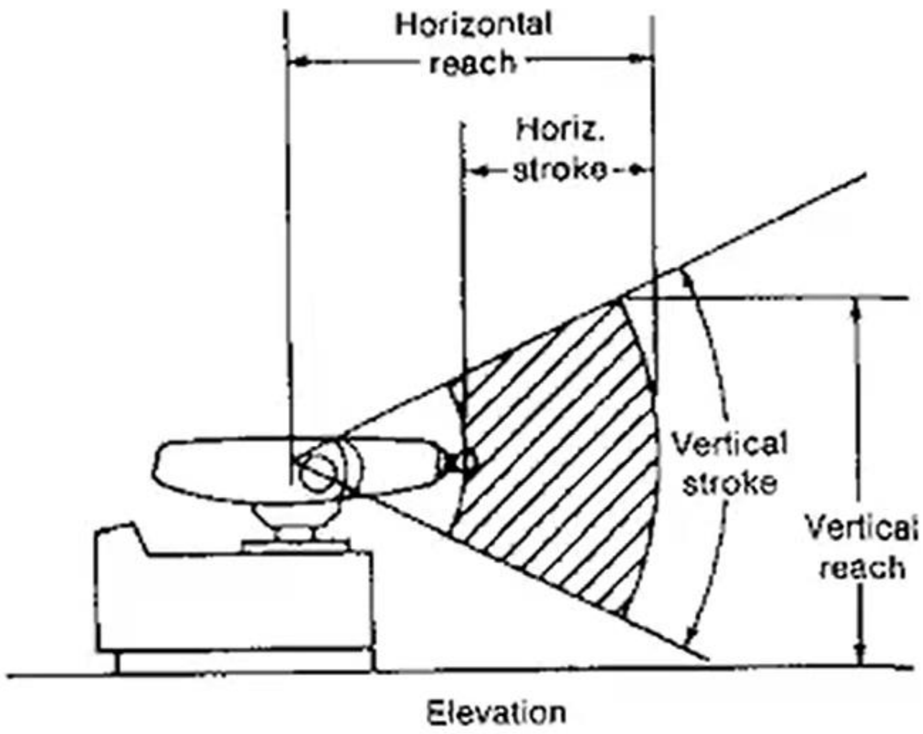


https://www.youtube.com/watch?v=_canCYWZPsc

3-Spherical Robot (RRP)-Work Envelop



Top view



Front view

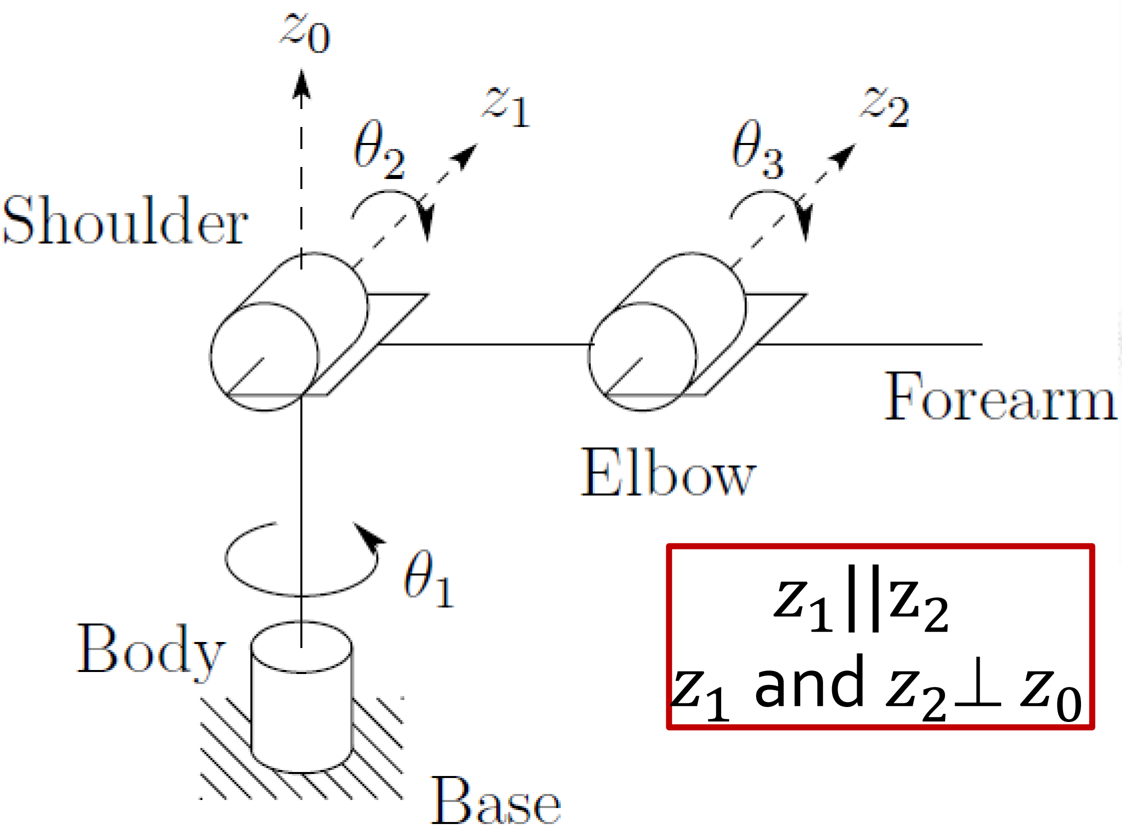
3-Spherical Robot (RRP)

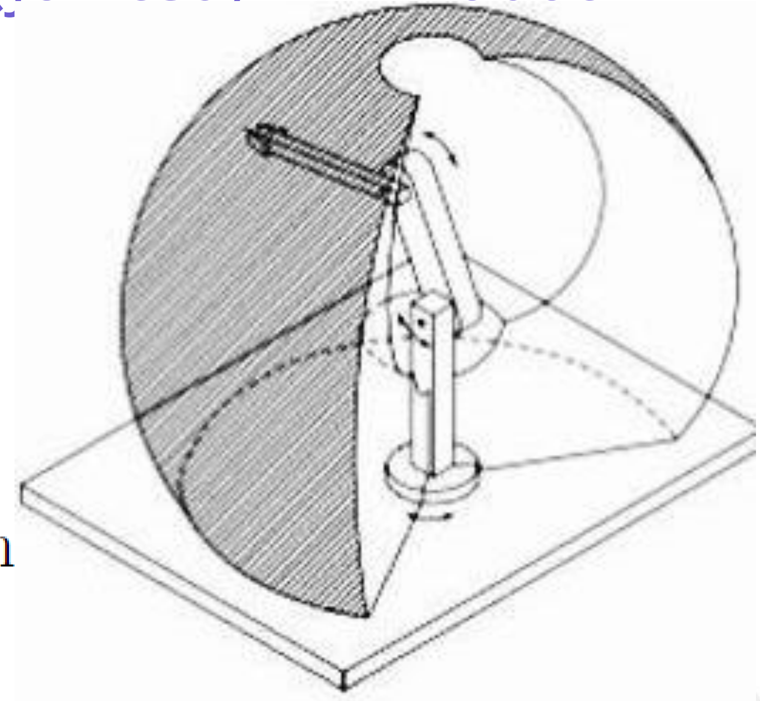
- Provides a larger work envelope than the rectilinear or cylindrical robot
- Design gives weight lifting capabilities
- Advantages and disadvantages same as cylindrical-coordinated robot

https://www.youtube.com/watch?v=jrF5Dl6ntAc

4-Articulated Robot (RRR)

- Also known as anthropomorphic (jointed) Arm Robot
- 3 revolute joints

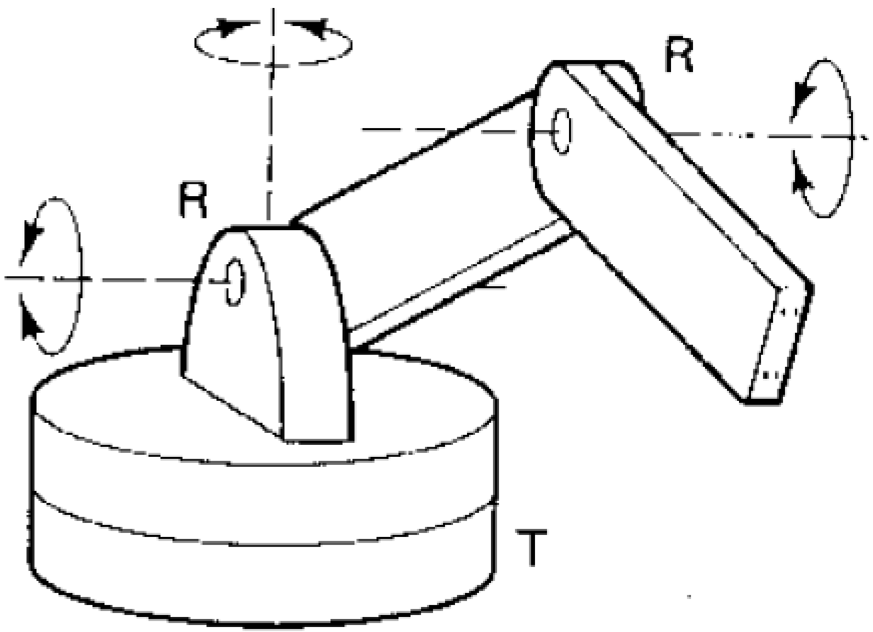




$$\boxed{\begin{array}{c} z_1 || z_2 \\ z_1 \text{ and } z_2 \perp z_0 \end{array}}$$

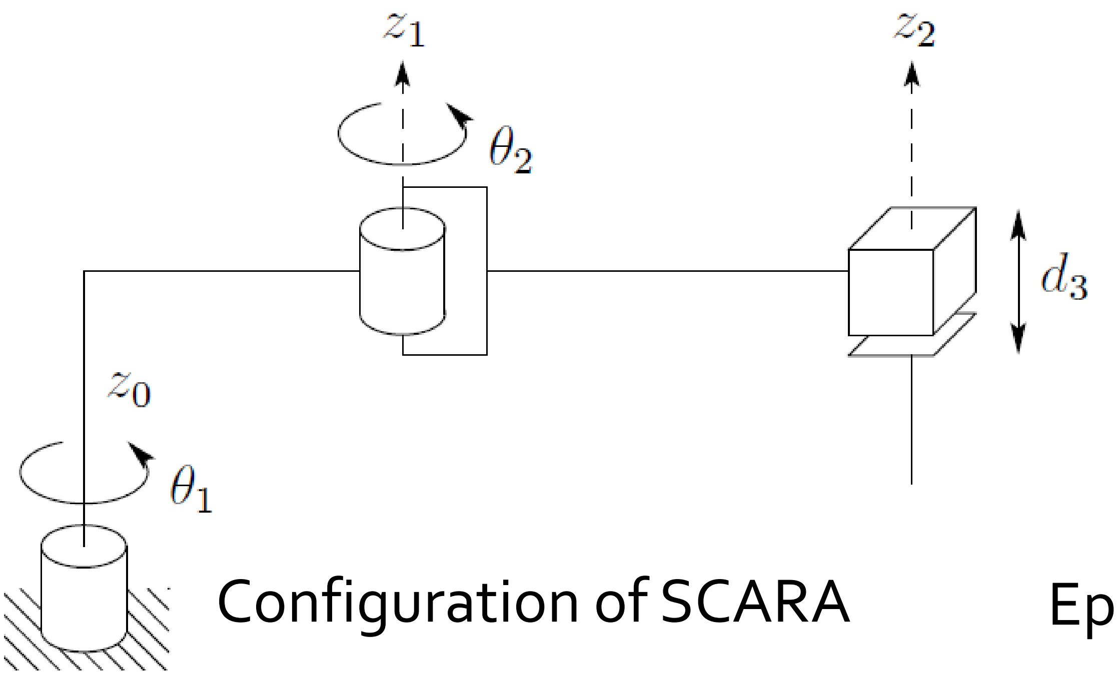
Configuration of articulated robot

4-Articulated Robot (RRR)



5-SCARA (RRP)

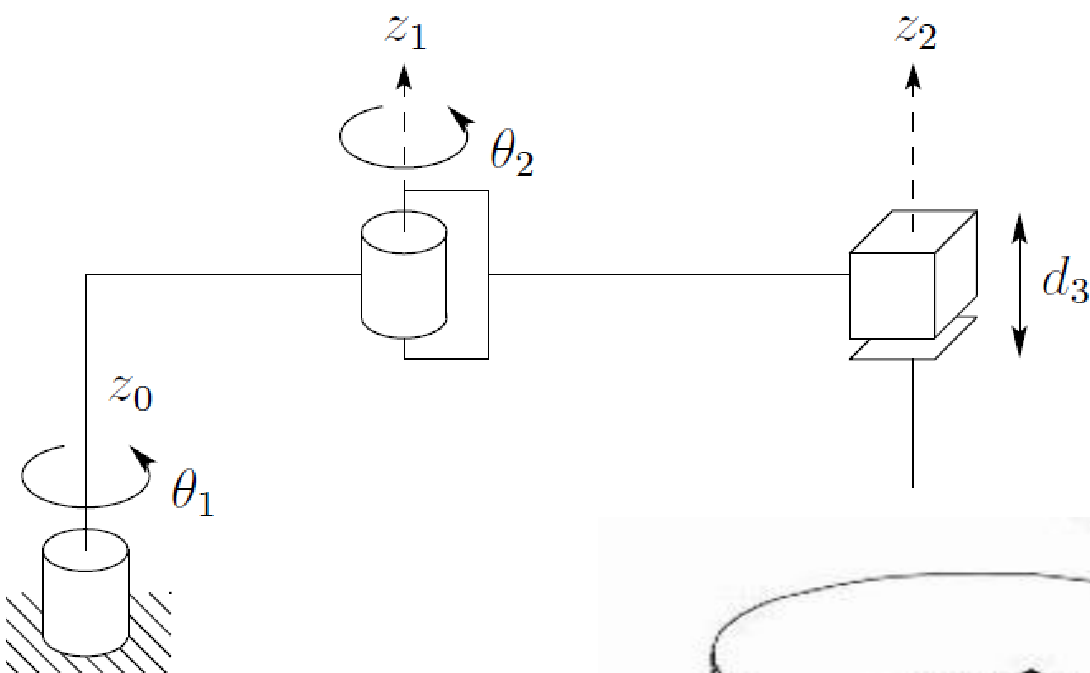
- Selective Compliant Articulated Robot Assembly
- 2 parallel revolute joint that allows the horizontal movement of robot and 1 prismatic that moves vertically
- 4DOF, 3 for Arm and 1 for wrist (roll)

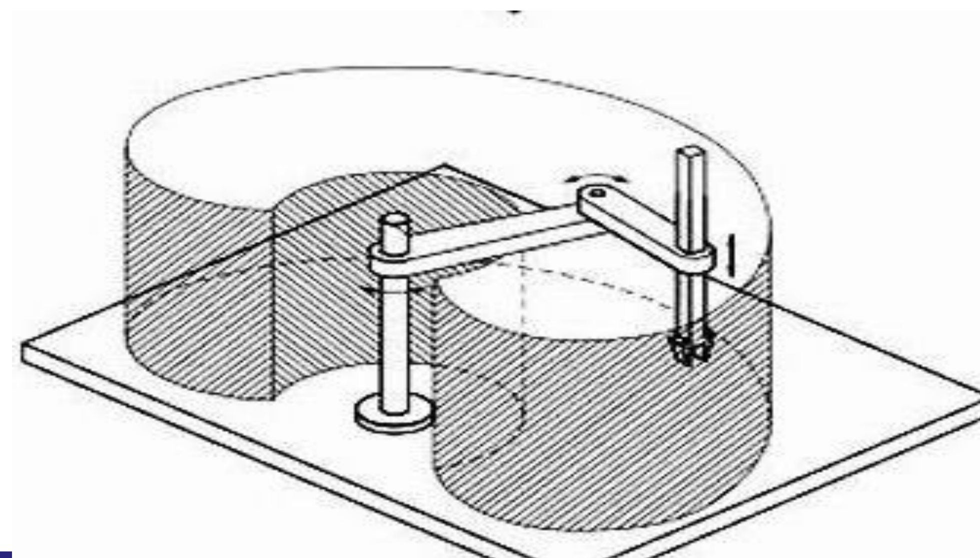


Configuration of SCARA

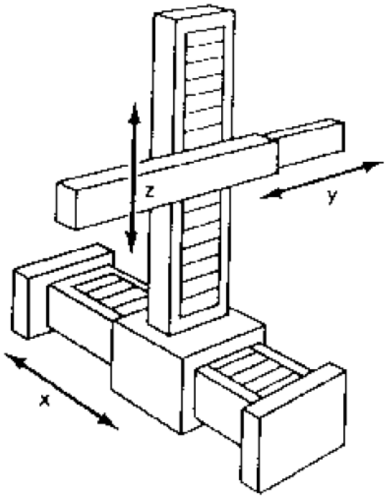
Epson E2L653S SCARA Robot

5-SCARA (RRP)

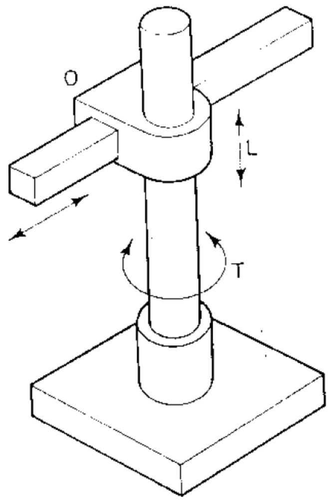




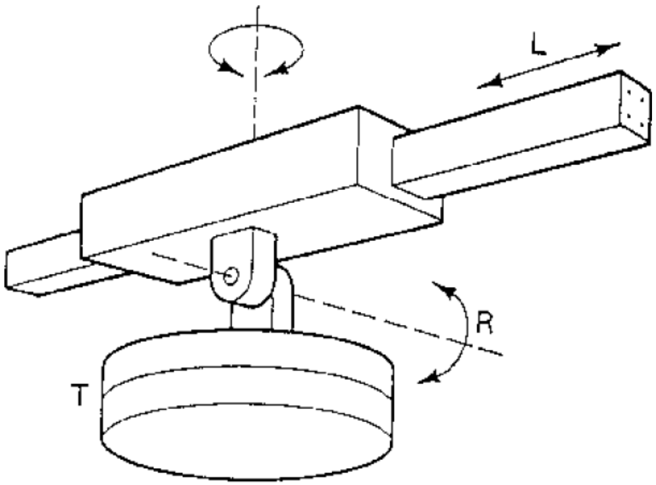
Robot Configurations: Summary



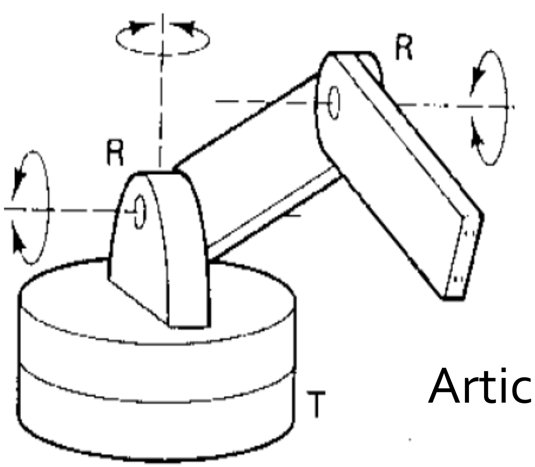
Cartesian



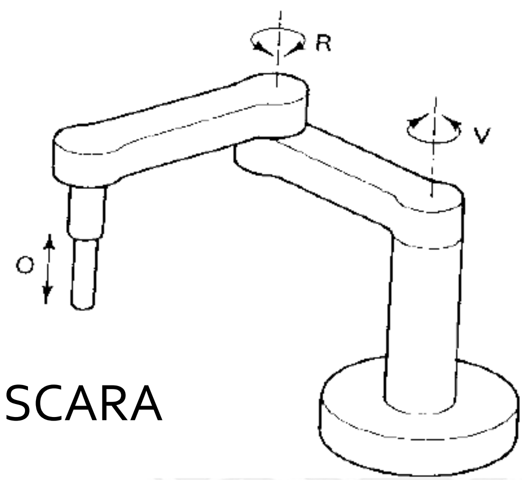
Cylindrical



Spherical



Articulated

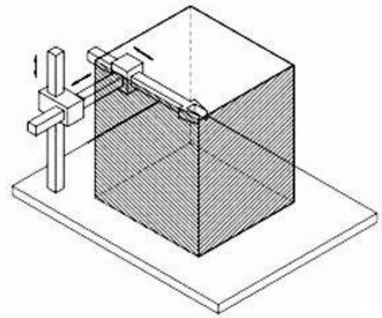


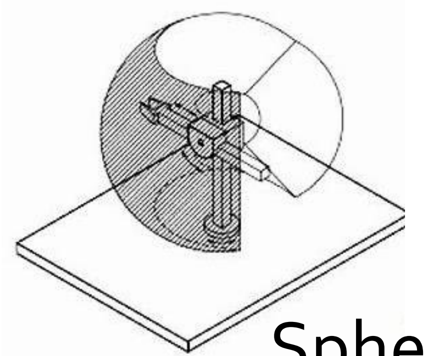
SCARA

Work Space: Summary

- The region in space a robot can fully interact with

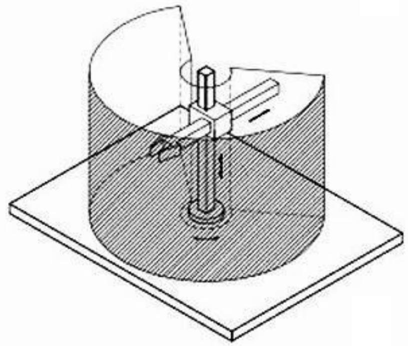
Rectangular/ Cartesian ($3P$)

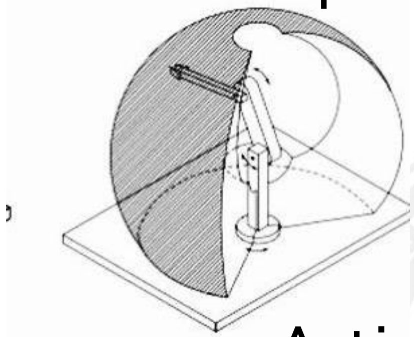




Spherical ($2R1P$)

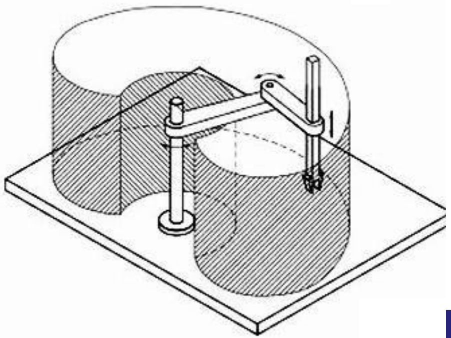
Cylindrical ($1R2P$)





Articulated ($3R$)

SCARA($2R1P$)

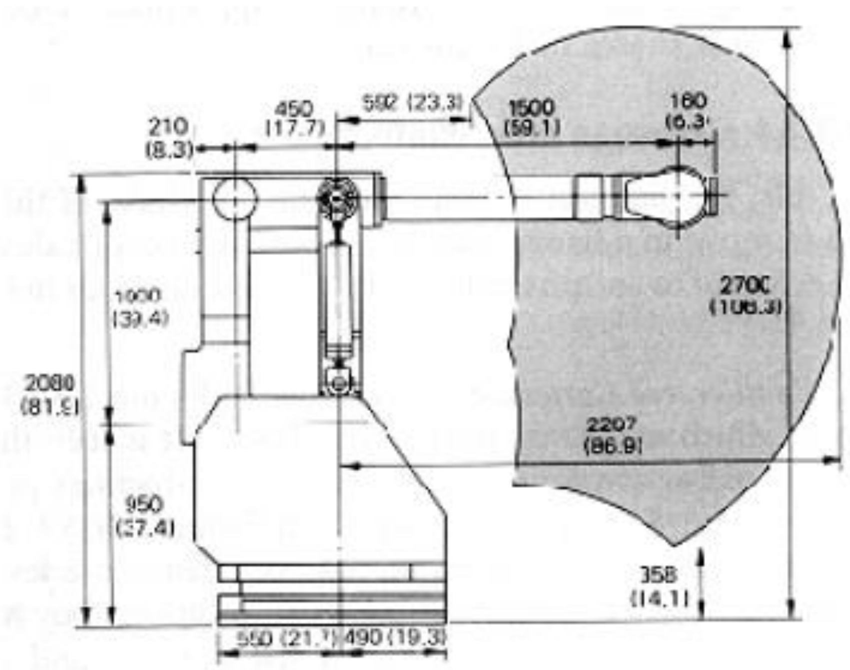


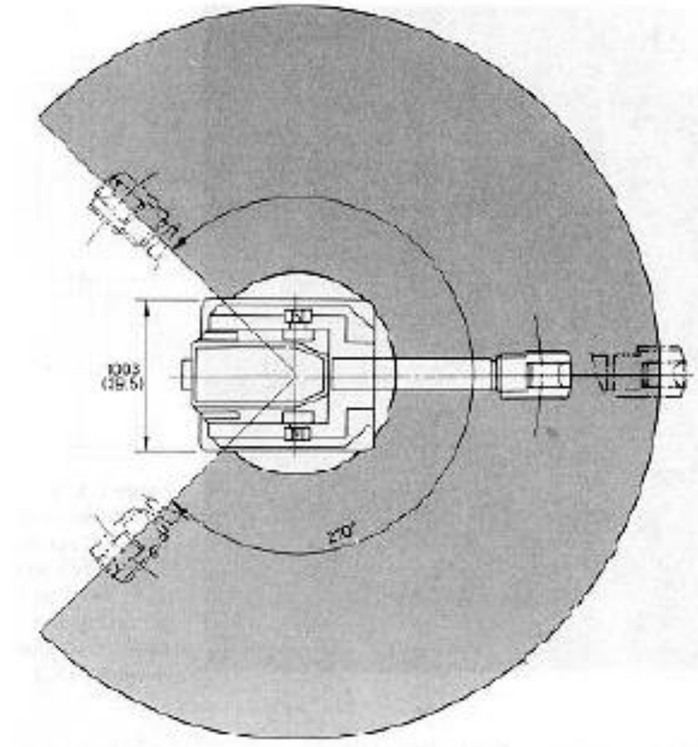
Workspace

- Depending on the configuration and size of the links and wrist joints, robots can reach a collection of points called a **Workspace**.

- Alternately Workspace may be found empirically, by moving each joint through its range of motions and combining all space it can reach and subtracting what space it cannot reach

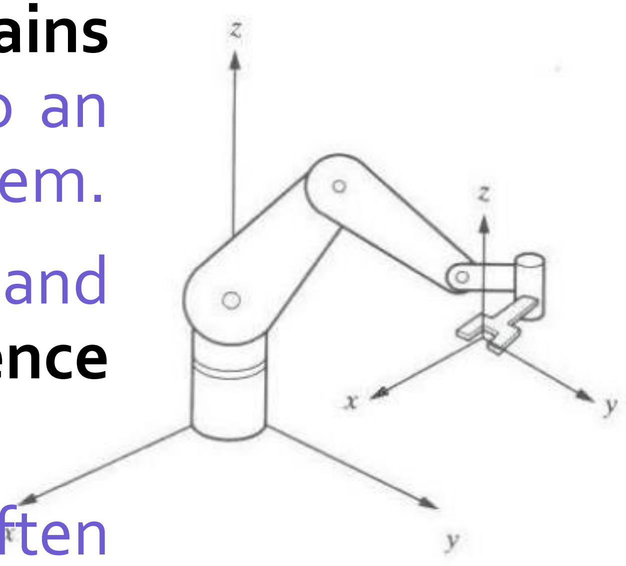
Work Envelope





Reference Frames

- World reference frame aka global reference frame or base reference frame, is an inertial frame of reference fixed in space.
- The world reference frame **remains stationary** and does not move relative to an external reference point or coordinate system.
- All other frames move simultaneously and defined **relative to the world reference frame.**
- In robotics, the world reference frame is often located at the **base of the robot** or at a fixed point in the workspace.



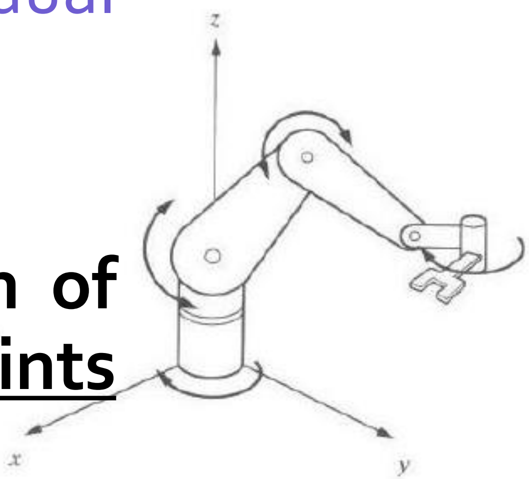
World reference frame

Reference Frames

- Joint reference frame aka local reference frame or joint coordinate frame, is a reference frame attached to each individual joint of the robotic system.

- It defines the **position and orientation of the joint <u>relative to its neighboring joints or links</u>**.

- The joint reference frame moves and rotates as the joint moves during operation.

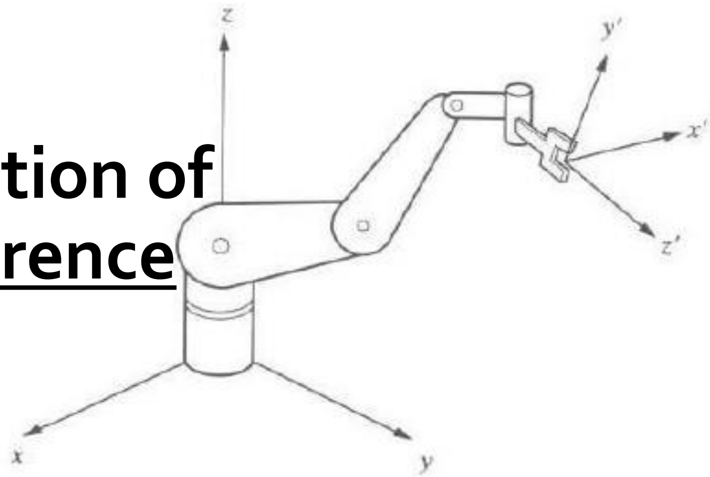


Joint reference frame

Reference Frame

- Tool reference frame aka TCP (Tool Center Point), is a local reference frame attached to the end-effector or tool of the robotic system.

- It defines the **position and orientation of the tool <u>relative to the world reference frame.</u>**



Tool reference frame

- The tool reference frame moves and changes orientation as the end-effector moves and rotates during operation.

Robot Programming/training

Typically performed using one of the following

- Online
  - Teach pendant
  - Lead through programming
- Offline
  - Programming languages
  - Task level programming

Teach Pendant Programming

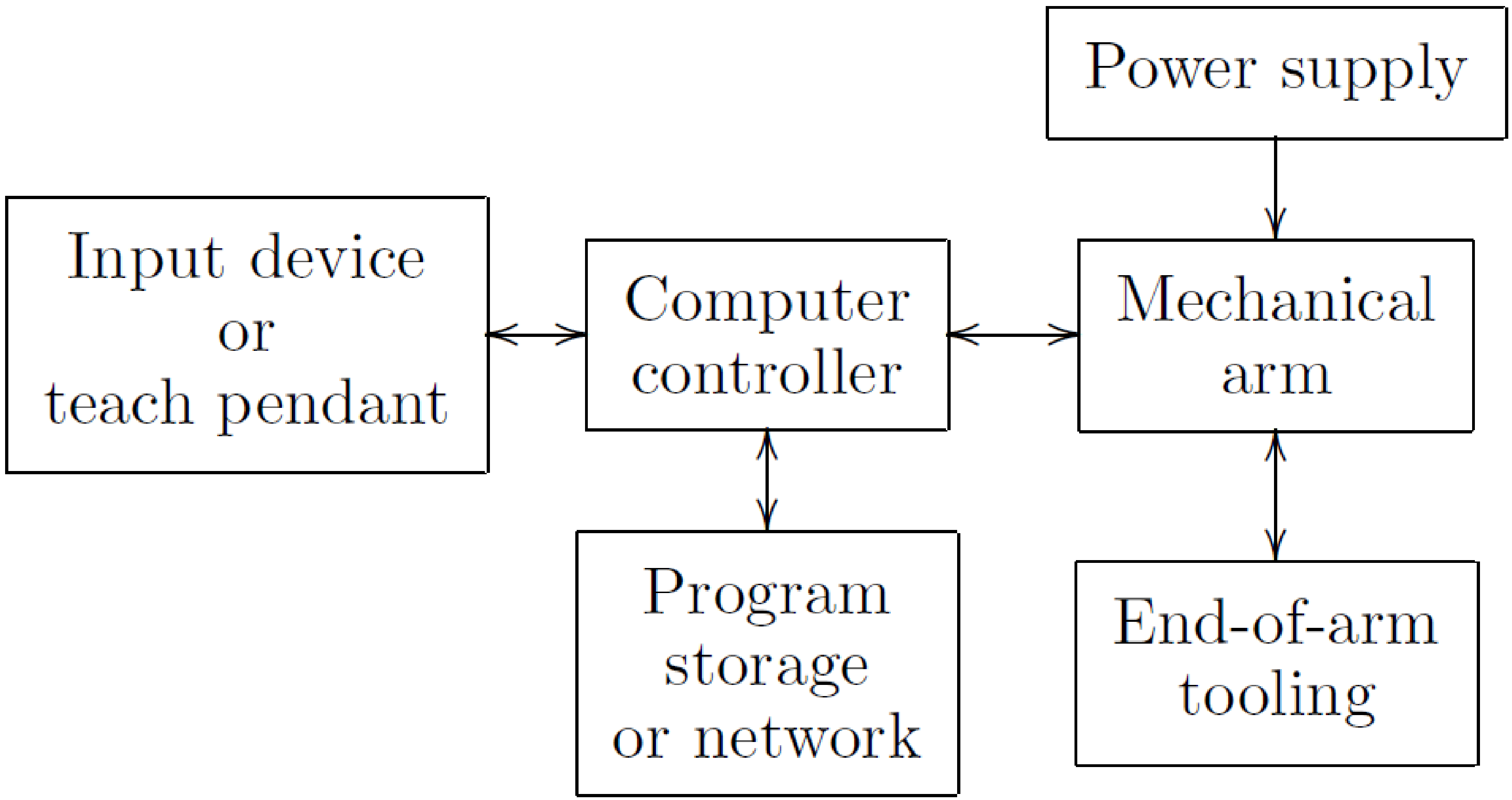
- Hand held device with switches used to control the robot motions
- End points are recorded in controller memory
- Sequentially played back to execute robot actions
- Trajectory determined by robot controller
- Suited for point-to-point control applications

https://www.youtube.com/watch?v=EA6pWwNI_wg

Lead Through Programming

- Lead the robot physically through the required sequences of motions
- Trajectory and endpoints are recorded, using a sampling routine which record points at $60 - 80$ times in a second
- When played back results in a smooth continuous motion
- Large memory requirements

Robotic System



Component of Robotic system

Online Programming

- Advantages
  - Easy to use
  - No special programming skills required
  - Useful when programming the robots for wide range of repetitive tasks for long production runs
- Disadvantages
  - Required production line shutdown
  - Technician programming inside work envelope

Programming Languages

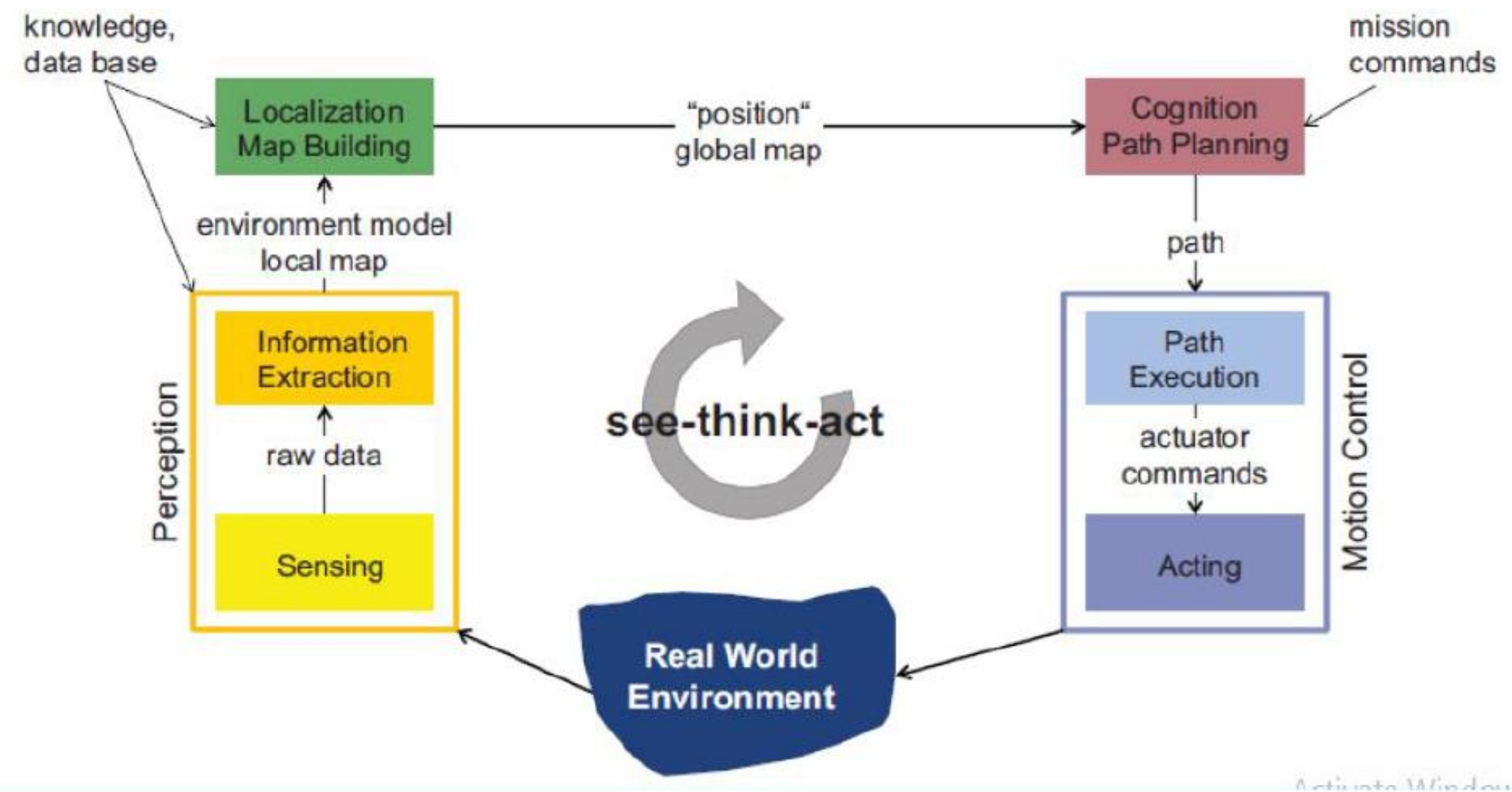
- Motivation:
  - Need to interface robot control system to external sensors to provide "real-time" changes based on **sensory equipment**
  - Commuting based on geometry of environment
  - Ability to interface with CAD/CAM systems
  - Meaningful task descriptions
  - Offline programming capability

Decision Making in Autonomous Mobile Robots

- Sensing
  - **Proprioceptive sensors:** internal measurement of robot; such as, angle of the joints of robotic arm, wheel revolutions, current drawn by an electric motor etc.
  - **Exteroceptive sensors:** measure the external state of the world with respect to the robot; such as, detect collision, distance between robot and the surrounding objects.
- Given an example of exteroceptive sensor
- Perceiving
- Planning

Autonomous Mobile Robots (Vehicles)

THE SEE THINK AND ACT CYCLE!



Programming Languages

- Wide range of robot's programming languages are available such as: AML, VAL, AL, RAIL, RobotStudio (200+)
- Each robot manufacturer has their own robot programming language
- No standards exist
- Portability of programs virtually non-existent

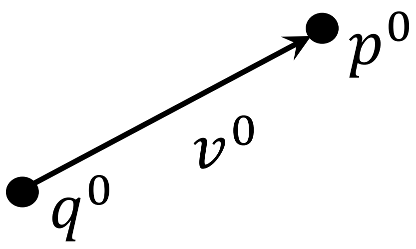
Agenda

- Orientation, Spaces and Transformations (SPONG, chp 2-3)
  - Representing robot position
  - Transforms
  - Mappings
  - Representations of Orientation
  - Joints and Spaces

Point and Vectors

- **Point**: A Point has position in space. The only characteristic that distinguishes one point from another is its position.
  - Draw point as dot

- **Vector**: A Vector has both magnitude and direction, but no fixed position in space.
  - Draw vector as line



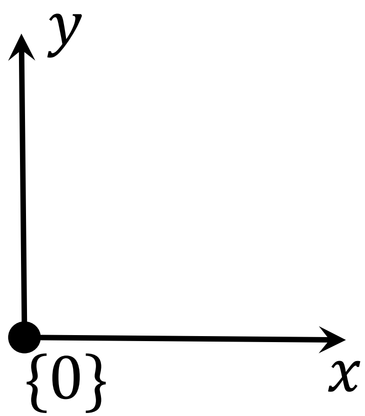
$$p^0 = \begin{bmatrix} 4 \\ 2.5 \end{bmatrix}$$

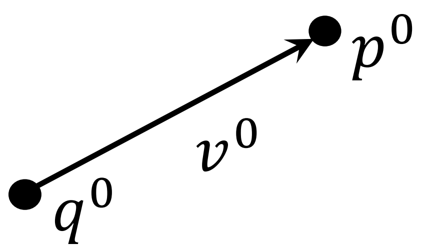
$$q^0 = \begin{bmatrix} 2 \\ 1.5 \end{bmatrix}$$

$$v^0 = p^0 - q^0 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Coordinate Frames

- A **coordinate frame** in two-dimensional space is a set of two vectors having unit length and that are perpendicular to each other.
- Allow us to assign the coordinates to the point





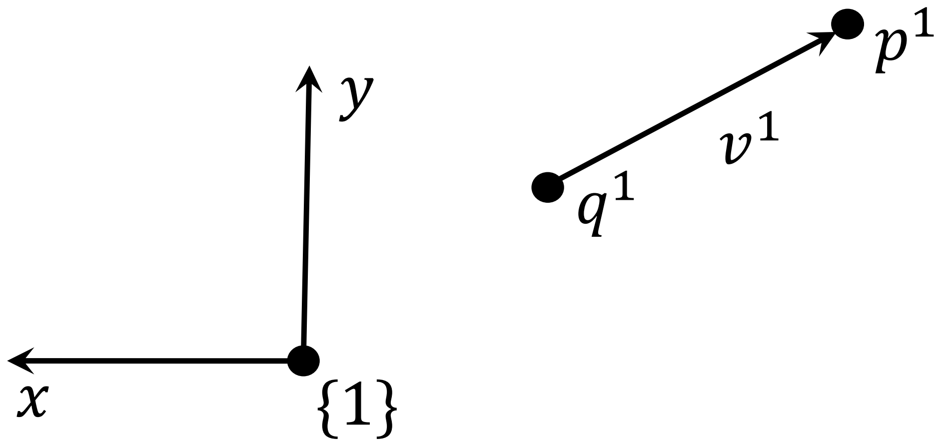
$$p^0 = \begin{bmatrix} 4 \\ 2.5 \end{bmatrix}$$

$$q^0 = \begin{bmatrix} 2 \\ 1.5 \end{bmatrix}$$

$$v^0 = p^0 - q^0 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Choice of Coordinate Frames

- Coordinates change depending on the choice of frame



$$p^1 = \begin{bmatrix} -0.5 \\ 4 \end{bmatrix}$$

$$q^1 = \begin{bmatrix} 0.5 \\ 2 \end{bmatrix}$$

$$v^1 = p^1 - q^1 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

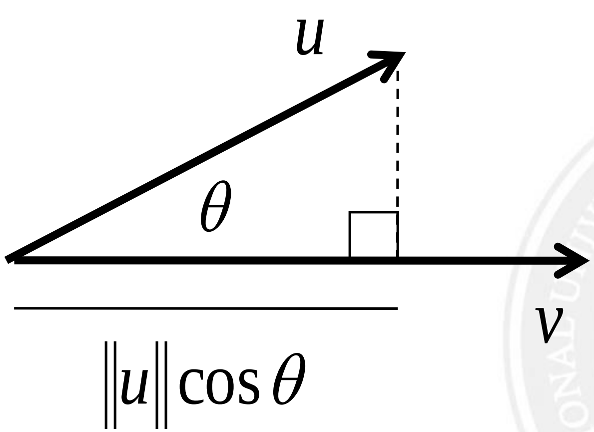
Dot Product

- Dot product of two vectors gives the projection of one onto the other

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

$$u \cdot v = u_1 v_1 + u_2 v_2 + \ldots + u_n v_n = u^T v$$

$$u \cdot v = \|u\| \|v\| \cos\theta$$



$$\boxed{\begin{array}{c}\text{Orthogonal vectors} \\ u.v = 0\end{array}}$$

Home Work!

- List the application of dot product in AI, gaming etc. with examples.

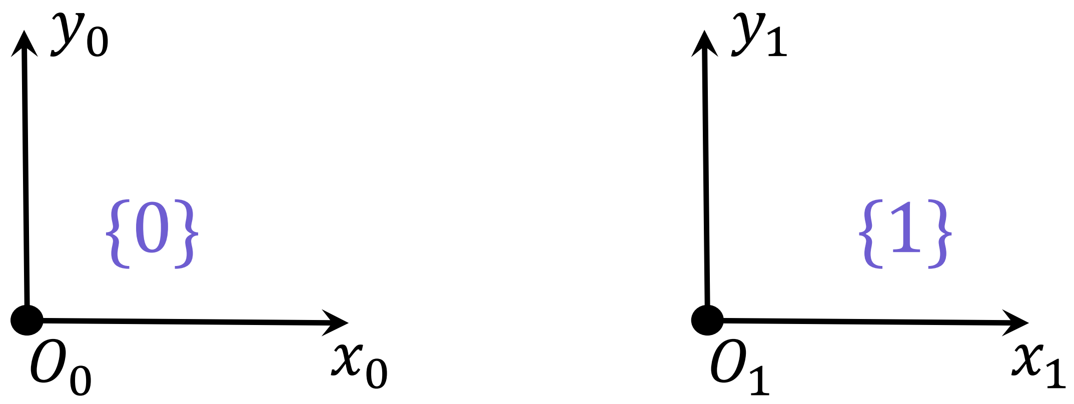
Transformations

Translation

1. The translation vector d_j^i can be interpreted as the **location** of frame {j} expressed in frame {i}.

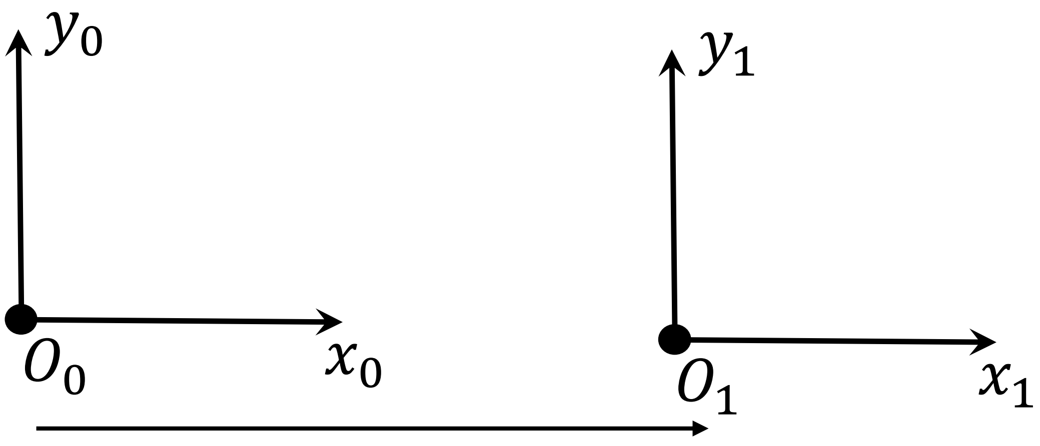
Translation

- **Example 1**



- Suppose O_0 is **zero vector** and $O_1 = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$

Translation



- The location of {1} is expressed in {0}

$$d_1^0 = O_1 - O_0 = \begin{bmatrix} 3 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

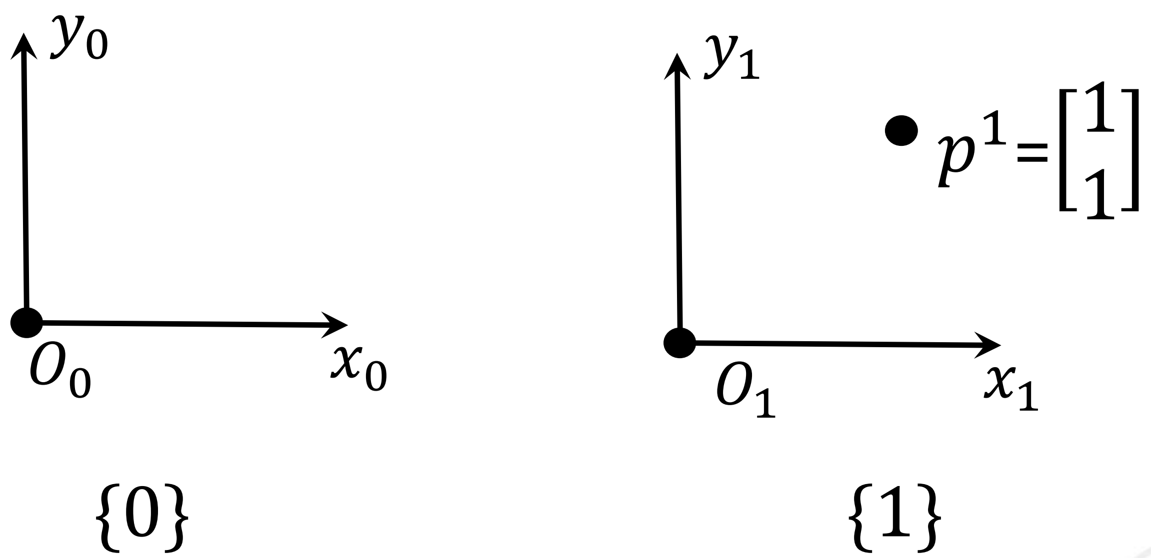
Translation

1. The translation vector d_j^i can be interpreted as the **location** of frame {j} expressed in frame {i}.

2. The translation vector d_j^i can be interpreted as a **coordinate transformation of a point** from frame {j} to frame {i}.

Translation: Example 2

- A point expressed in frame $\{0\}$ when $d_1^0 = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$.



- p^1 is expressed in $\{0\}$

$$p^0 = d_1^0 + p^1 = \begin{bmatrix} 3 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$$

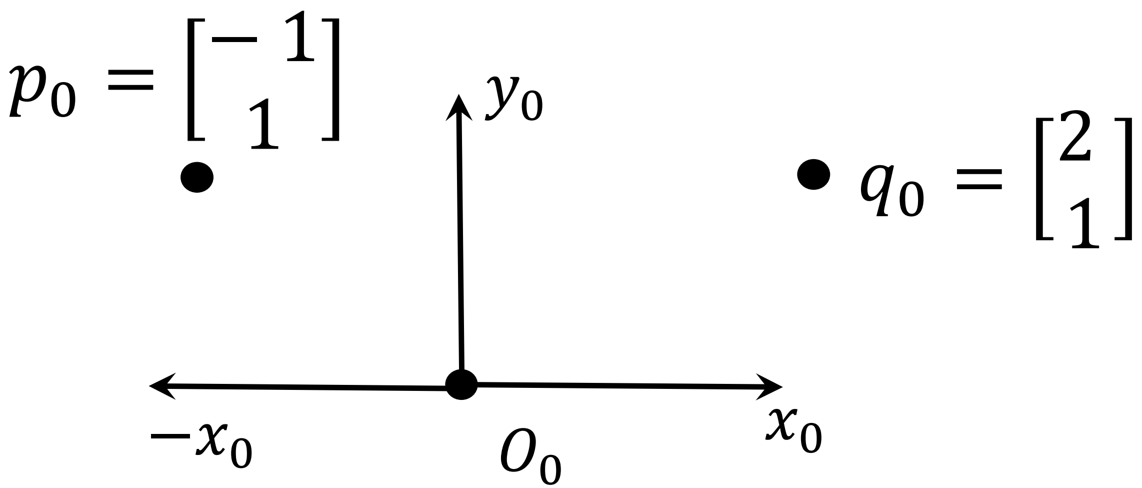
Translation

1. The translation vector d_j^i can be interpreted as the **location** of frame {j} expressed in frame {i}.

2. The translation vector d_j^i can be interpreted as a **coordinate transformation of a point** from frame {j} to frame {i}.

3. Translation vector d can be interpreted as an operator that takes a point and **moves it to a new point in the same frame.**

Translation: Example 3



- q_0 expressed in $\{0\}$, given is $d = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$

$$q_0 = d + p_0 = \begin{bmatrix} 3 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Translation: Example 3

- $p_0 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ and $p_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$

- The location of p_1 is expressed in p_0

$$d_1^0 = p_1 - p_0 = \begin{bmatrix} 2 \\ 1 \end{bmatrix} - \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix}$$

Rotation

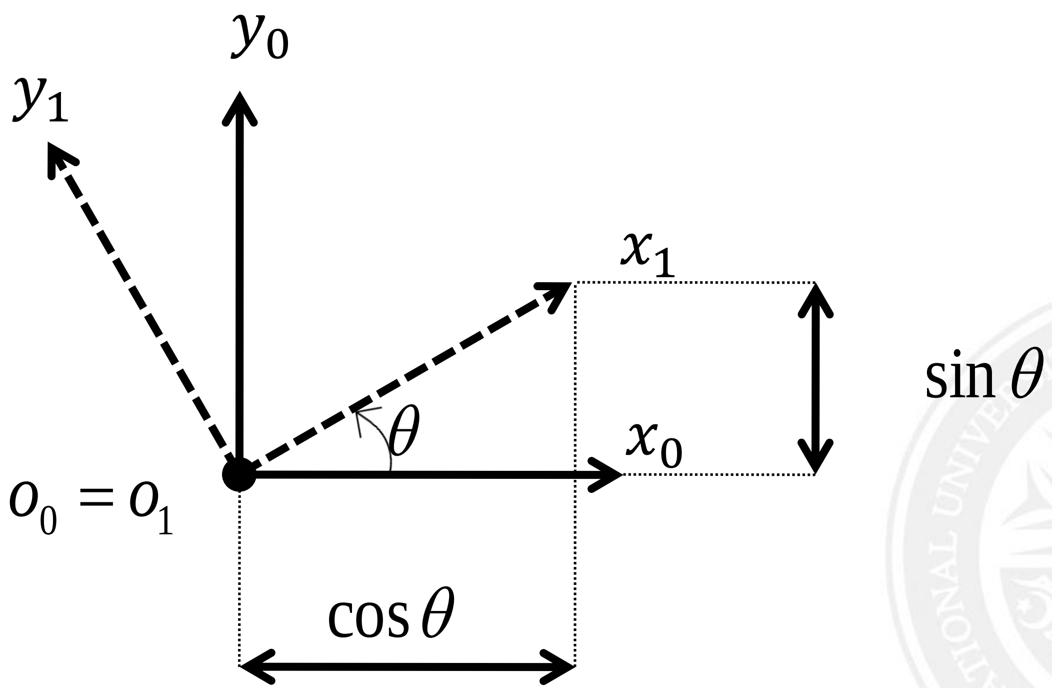
1. The rotation matrix R_j^i can be interpreted as the **orientation** of frame {j} expressed in frame {i}.

Rotation

- Suppose that frame {1} is rotated relative to frame {0}

$$R_1^0 = [x_1^0 | y_1^0]$$

- In 2D case



Rotation-I

- Project frame {1} onto the frame {0} (Dot product)

$$R_1^0 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

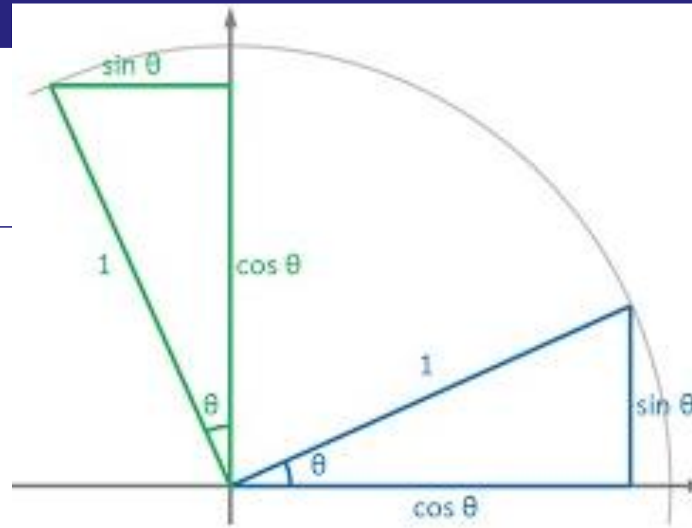
$$\Rightarrow \begin{bmatrix} x_1 . x_0 \\ x_1 . y_0 \end{bmatrix}, \begin{bmatrix} y_1 . x_0 \\ y_1 . y_0 \end{bmatrix}$$

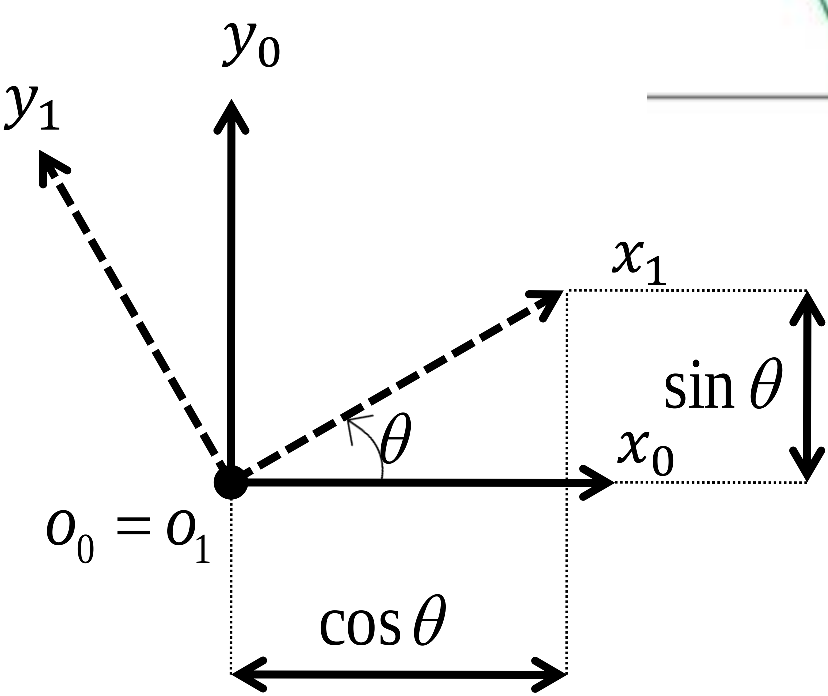
Projection of {1} over {0} $\Rightarrow$ $\boxed{x_1 \qquad y_1}$

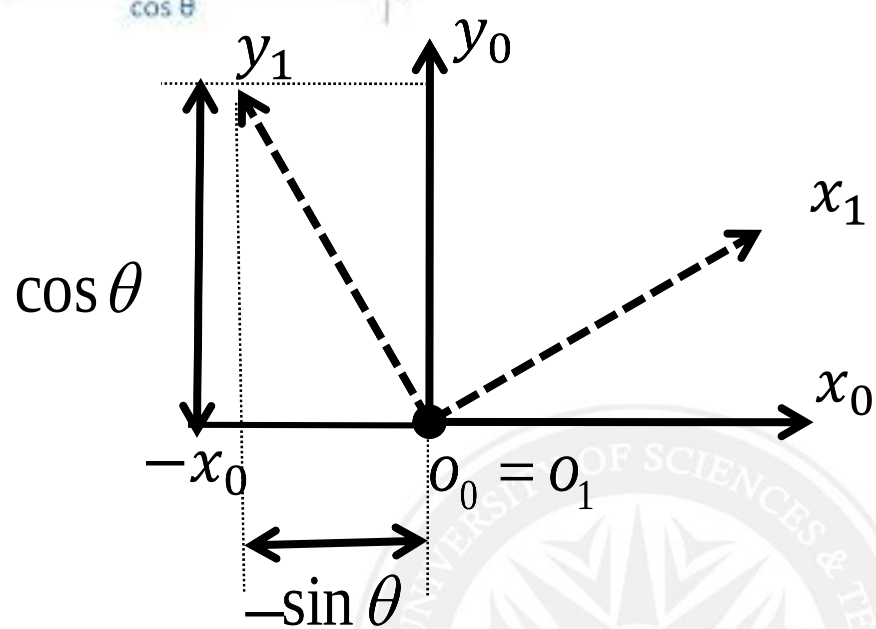
$$R_1^0 = \begin{bmatrix} x_1 . x_0 & y_1 . x_0 \\ x_1 . y_0 & y_1 . y_0 \end{bmatrix} \boxed{\begin{matrix} x_0 \\ y_0 \end{matrix}}$$

Rotation-II

- Alternate approach







$$x_1^0 = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \qquad y_1^0 = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$$

Rotation-III

- The orientation of frame {1} is expressed in frame {0}

$$x_1^0 = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \qquad y_1^0 = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$$

$$\Downarrow \quad R_1^0 = [x_1^0 | y_1^0] \quad \Downarrow$$

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

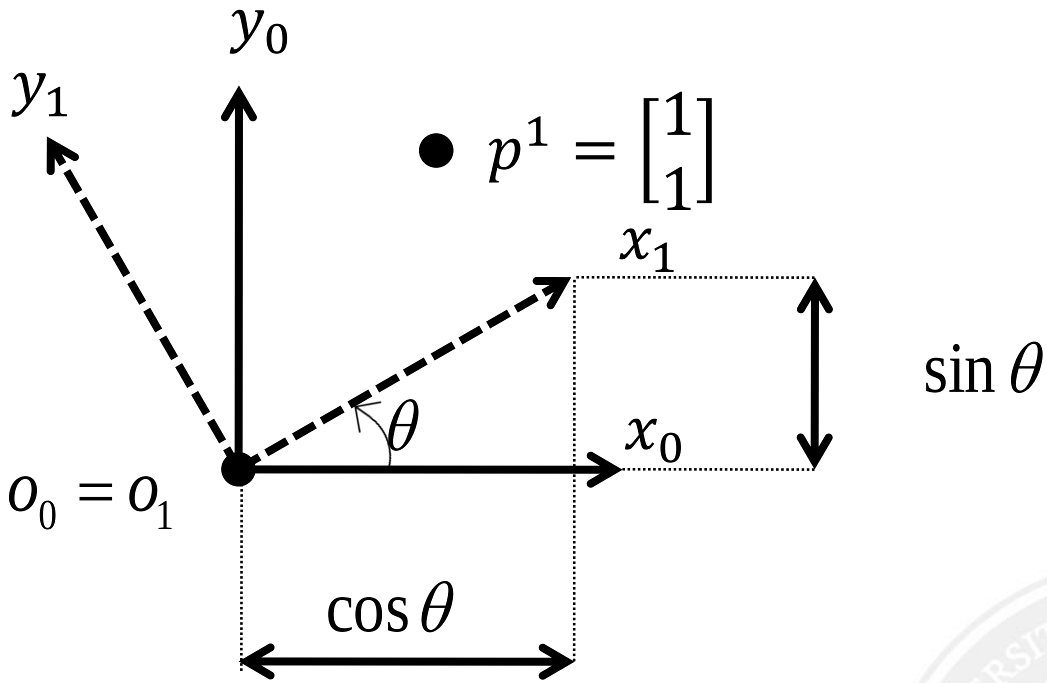
Rotation

1. The rotation matrix R_j^i can be interpreted as the **orientation** of frame {j} expressed in frame {i}.

2. The rotation matrix R_j^i can be interpreted as a **coordinate transformation of a point** from frame {j} to frame {i}.

Rotation: Example

- p^1 expressed in {0}



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

Rotation

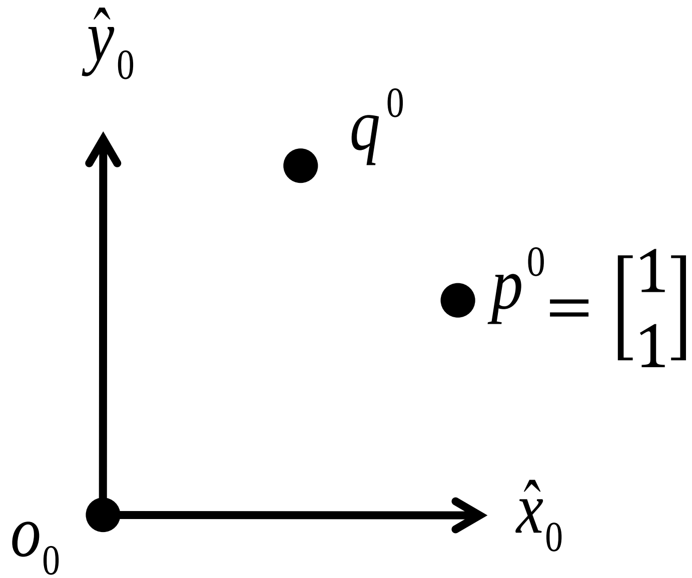
1. The rotation matrix R_j^i can be interpreted as the **orientation** of frame {j} expressed in frame {i}.

2. The rotation matrix R_j^i can be interpreted as a **coordinate transformation of a point** from frame {j} to frame {i}.

3. Rotation matrix R can be interpreted as an operator that takes a point and **moves it to a new point in the same frame.**

Rotation: Example

- q^0 expressed in frame {0}



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

Properties of Rotation Matrix

Properties of rotation matrix

- $R_j^i = \left(R_i^j\right)^T$
- $\left(R_j^i\right)^T = \left(R_j^i\right)^{-1}$
- $R(\theta_1)R(\theta_2) = R(\theta_1 + \theta_2)$
- $R(-\theta) = R(\theta)^T$
- Columns/rows of R are mutually orthogonal
- Each column/row of R is a unit vector
- Determinant of R is equal to 1 ($\det(R) = 1$)

Rotation in 3D

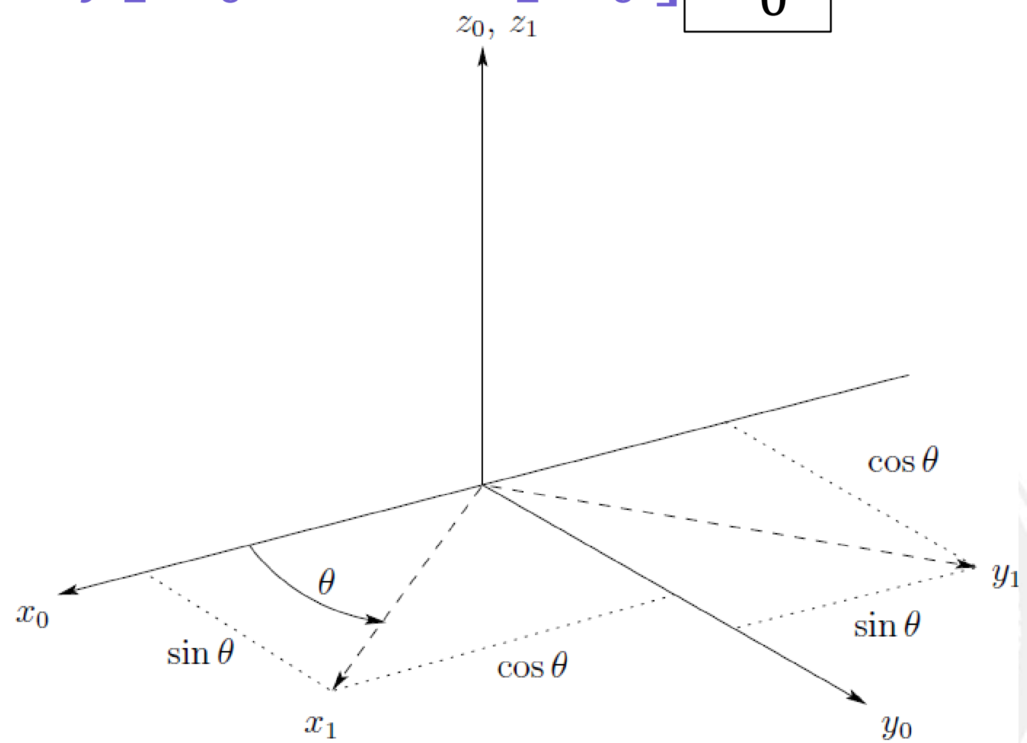
Projection of {1} over {0} $\boxed{x_1 \qquad y_1 \qquad z_1}$

$$R_1^0 = \begin{bmatrix} x_1 . x_0 & y_1 . x_0 & z_1 . x_0 \\ x_1 . y_0 & y_1 . y_0 & z_1 . y_0 \\ x_1 . z_0 & y_1 . z_0 & z_1 . z_0 \end{bmatrix} \boxed{\begin{matrix} x_0 \\ y_0 \\ z_0 \end{matrix}}$$

Example 2. 1: $(R_{z,\theta})$

Rotation of {1} about z-axis

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

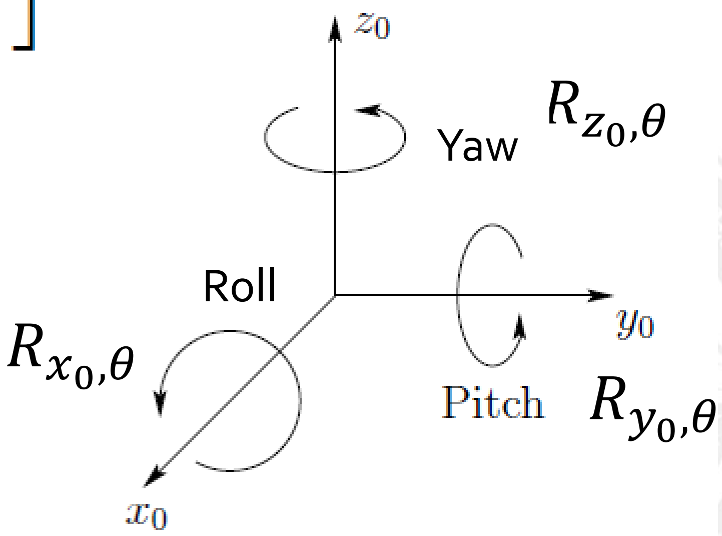


Rotation in 3D

Rotation of {1} about x and y-axis are as;

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$



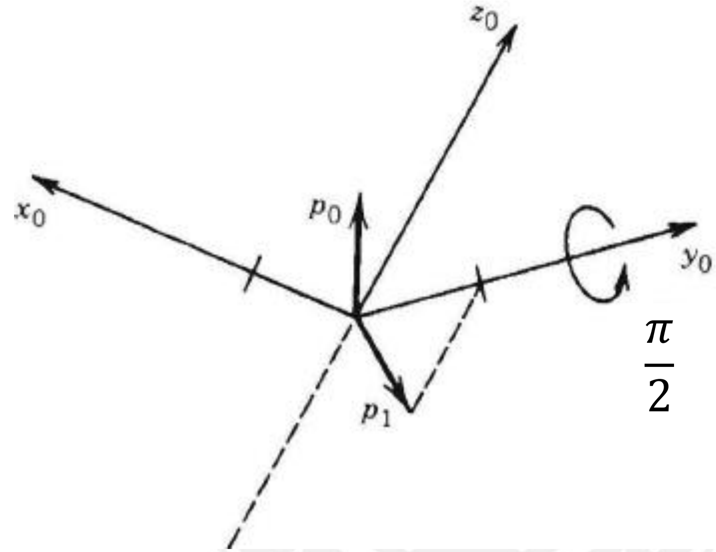
Rotation in 3D

Example: Vector $p_0 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$ is rotated about y_0-axis by $\frac{\pi}{2}$ as shown in figure. What will be the resulting vector p_1?

- **Solution**:

$$p_1 = R_{y_0,\frac{\pi}{2}} p_0$$

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$



$\frac{\pi}{2}$

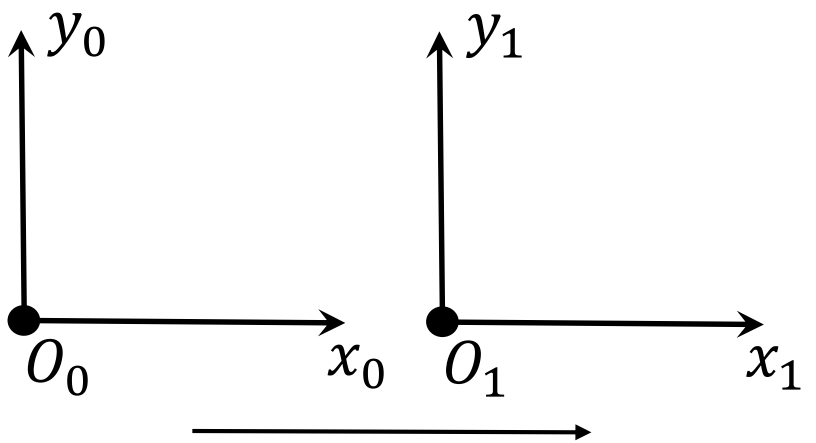
Properties of Rotation Matrix

Properties of rotation matrix

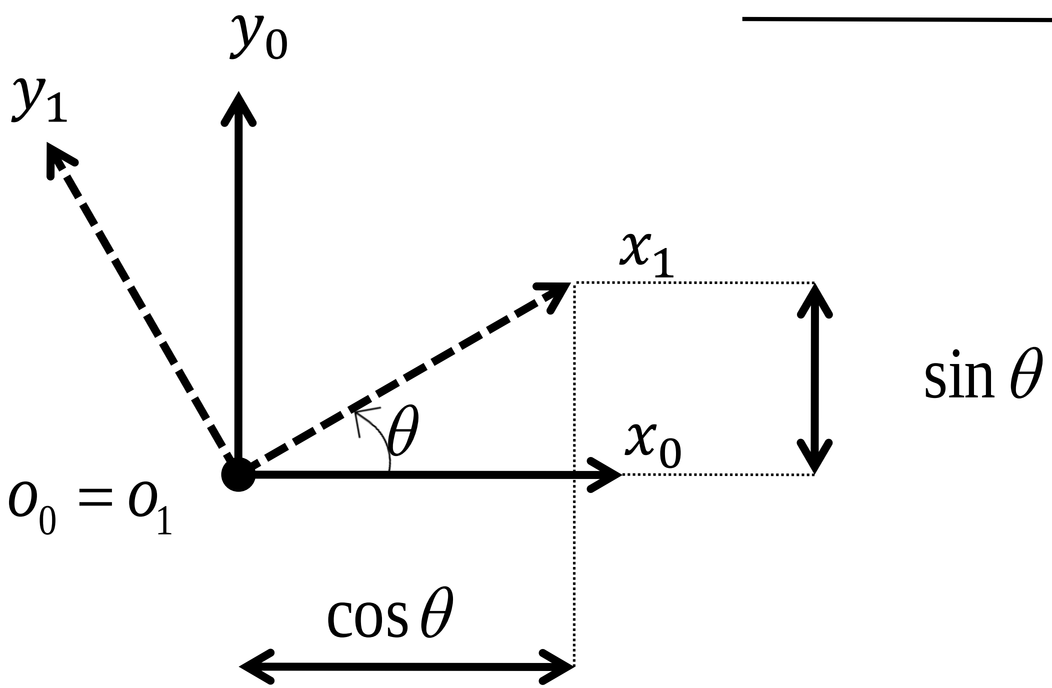
- $R_{z,0} = R_{y,0} = R_{x,0} = I$
- $R_{z,\theta} R_{z,\phi} = \mathrm{R}_{\mathrm{z},\theta+\phi}$
- $R_{z,\theta}^{-1} = R_{z,-\theta}$

Summary

- Translation



- Rotation

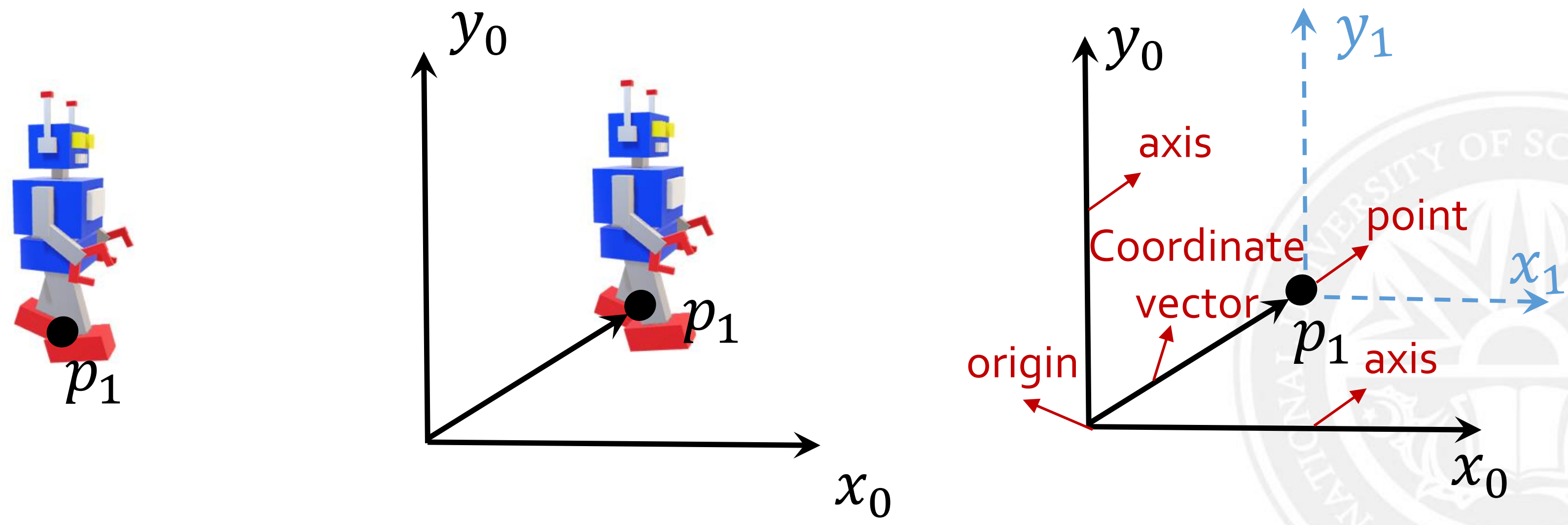


Pose

- How to define the pose of an object in space?
- **Pose**: combination of **position and orientation**
- A point in space?
- Coordinate frame/ Cartesian coordinate system?

Pose

- ***Convention***: Attach the coordinate frame to the object. It enables us to describe the pose of the object with respect to reference/universal coordinate frame.
- **Assumption**: Object has rigid body
- What should be the required dimension to define the pose of an object?



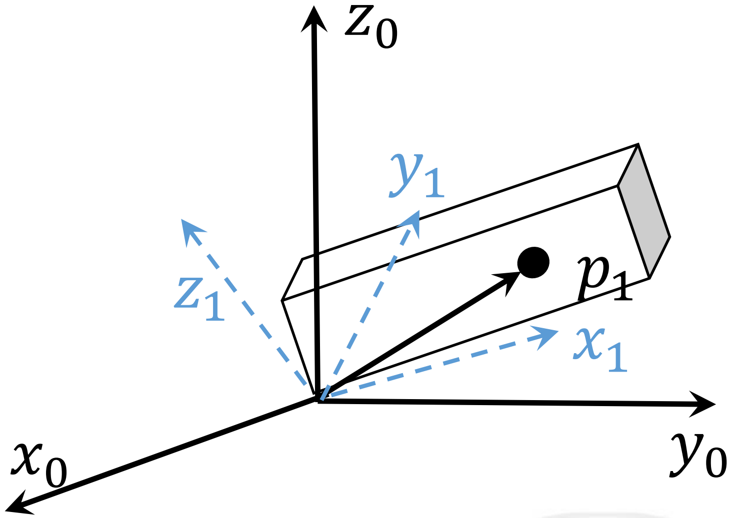
Pose: Position

- **Position**: we can locate any point in space with 3D position vector

$$p_1 = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

or

$$p_1 = u x_1 + v y_1 + w z_1$$



Pose: Position

- Project the point p_1 on reference frame $\{0\}$

$$p_0 = (ux_1 + vy_1 + wz_1).\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} (ux_1 + vy_1 + wz_1).x_0 \\ (ux_1 + vy_1 + wz_1).y_0 \\ (ux_1 + vy_1 + wz_1).z_0 \end{bmatrix}$$

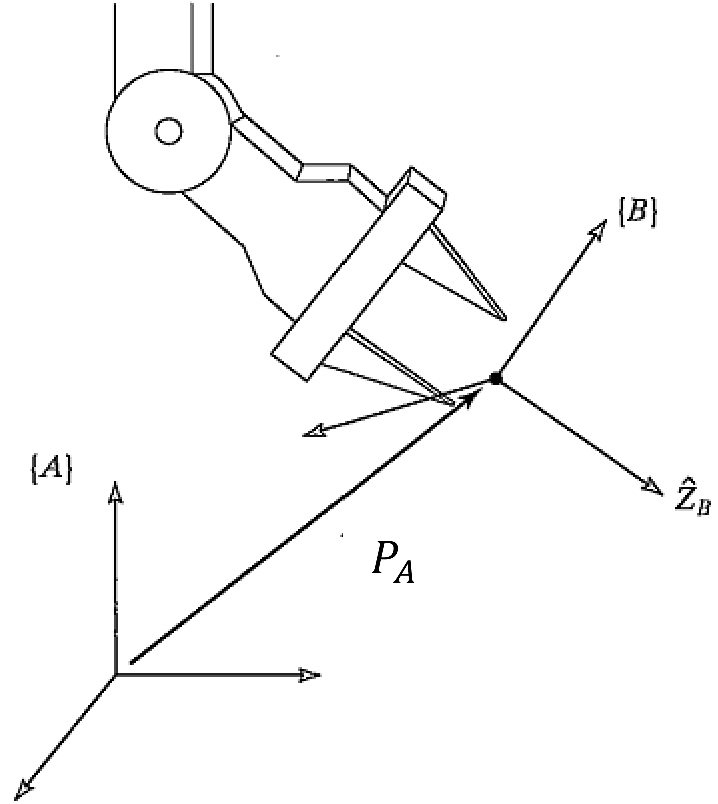
$$p_0 = \begin{bmatrix} ux_1.x_0 + vy_1.x_0 + wz_1.x_0 \\ ux_1.y_0 + vy_1.y_0 + wz_1.y_0 \\ ux_1.z_0 + vy_1.z_0 + wz_1.z_0 \end{bmatrix}$$

$$p_0 = \underbrace{\begin{bmatrix} x_1.x_0 + y_1.x_0 + z_1.x_0 \\ x_1.y_0 + y_1.y_0 + z_1.y_0 \\ x_1.z_0 + y_1.z_0 + z_1.z_0 \end{bmatrix}}_{R_1^0} \underbrace{\begin{bmatrix} u \\ v \\ w \end{bmatrix}}_{p_1}$$

$$p_0 = R_1^0 p_1$$

Pose: Rotation

- To describe the orientation of a body, we attach a coordinate system to the body and then give a description of this coordinate system relative to the reference system.



- $R_B^A = \begin{bmatrix} x_B.x_A & y_B.x_A & z_B.x_A \\ x_B.y_A & y_B.y_A & z_B.y_A \\ x_B.z_A & y_B.z_A & z_B.z_A \end{bmatrix}$

Summary: Pose

- **Position** of point are described with vectors
- **Orientation** of bodies are described with an attached coordinate system using **Rotation matrix**