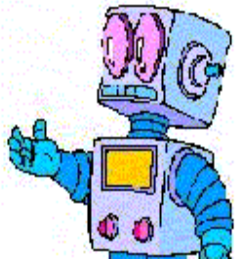
INTRODUCTION TO ROBOTICS

(Kinematics, Dynamics, and Design)

SESSION # 7:

MECHANICS OF MANIPULATORS



Ali Meghdari, Professor

School of Mechanical Engineering

Sharif University of Technology

Tehran, IRAN 11365-9567

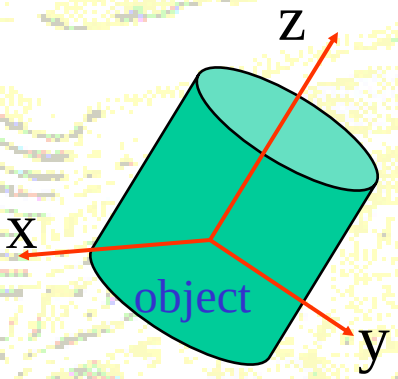
Homepage: http://meghdari.sharif.edu

Mechanics of Manipulators

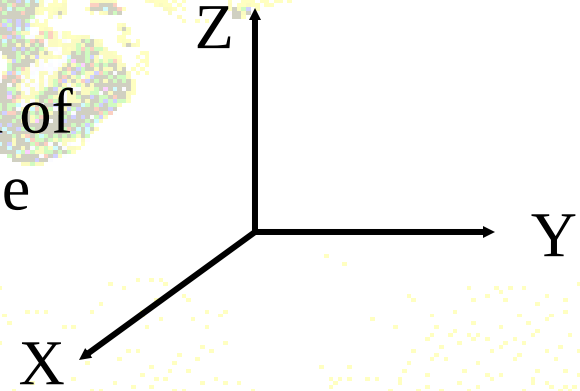
- Robots can be analyzed in terms of their ***mechanical properties*** (*typically used with industrial manipulators*)
  - What is their range of motion?
  - How can they be programmed to optimize a particular motion?
  - How can they be constructed to perform a certain task?
- Requires a thorough understanding of how to specify
  - Position of all components in 3-space
  - Orientation of all components 3-space

Mechanical Considerations

To describe the **Position** and **Orientation** of a body in space:



- Rigidly attach a coordinate system (frame {x, y, z}) to the object, then
- Describe the Position & Orientation of this frame with respect to a reference frame {X, Y, Z}

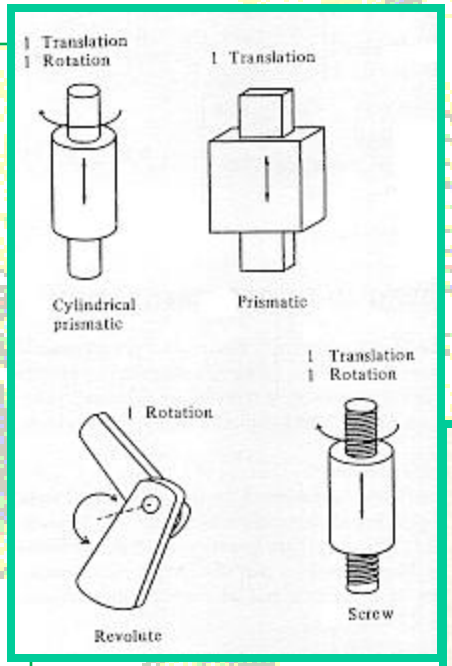


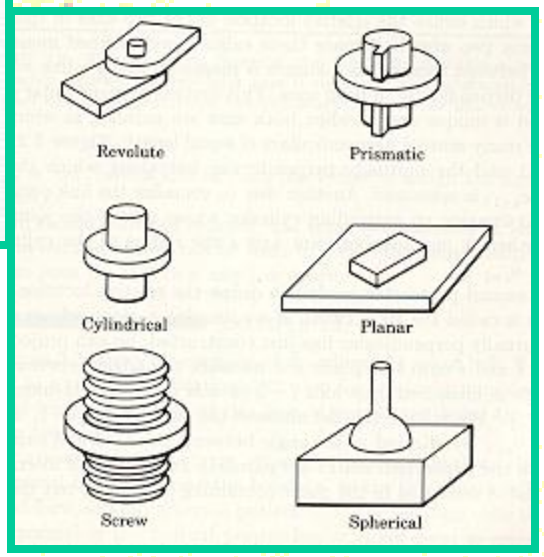
{Reference Coordinate System}

Basic Elements of Manipulator Arms

Manipulators consist of:

- Links (nearly rigid), and
- Joints:
  - Revolute (Displacement = Joint Angle)
  - Prismatic/Sliding (Translation = Joint Offset)





The Six Possible Lower-Pair Joints

Degrees-of-Freedom (Mobility) of a Robot

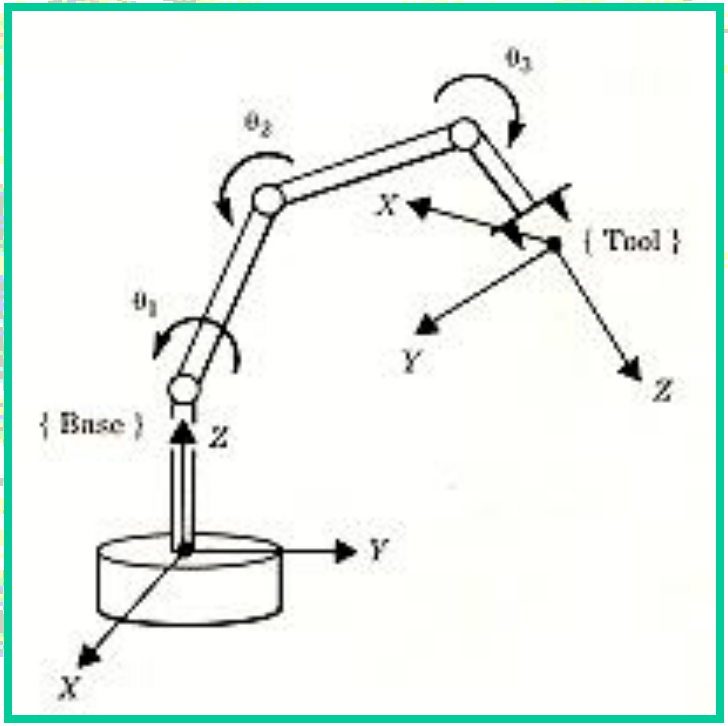
The number of input parameters (i.e. joint variables) which must be independently controlled in order to bring the robotic arm into a particular position/orientation.

➔ In open kinematics chains (i.e. Industrial Manipulators):

{# of D.O.F. = # of Joints}

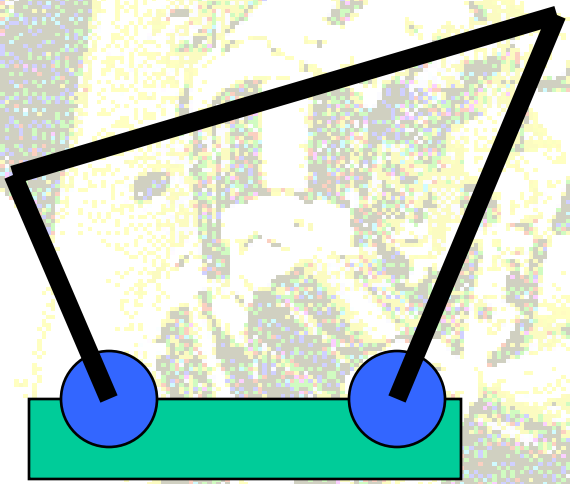
3R = An Arm with 3 successive Revolute Joints.

3P = An Arm with 3 successive Sliding/Prismatic Joints.

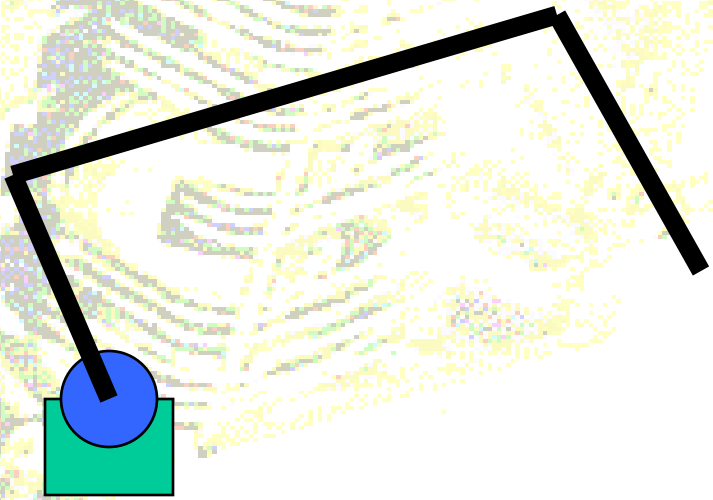


A 3-DOF Manipulator Arm

Simple Examples

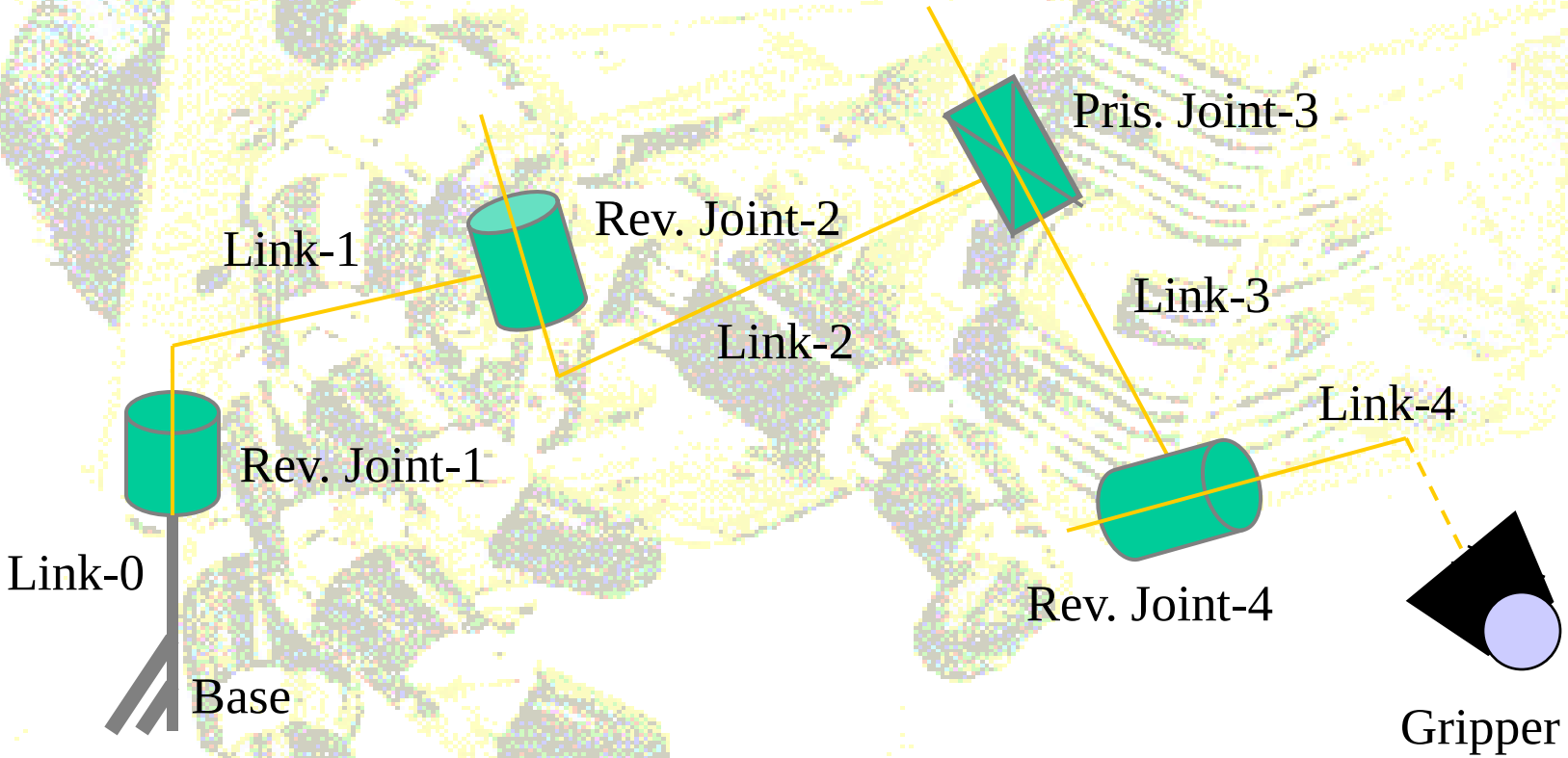


A one degree-of-freedom closed-loop mechanism



A three degrees-of-freedom open-loop mechanism

Degrees-of-Freedom (Mobility) of a Robot



A 2R1P1R Robot Manipulator

Parallel or Closed Chain Manipulators

- **Parallel manipulator:** two or more series chains connect the end-effector to the base (closed-chain).

- **# of DOF** for a parallel manipulator determined by taking the total DOFs for all links and subtracting the number of constraints imposed by the closed-chain configuration.

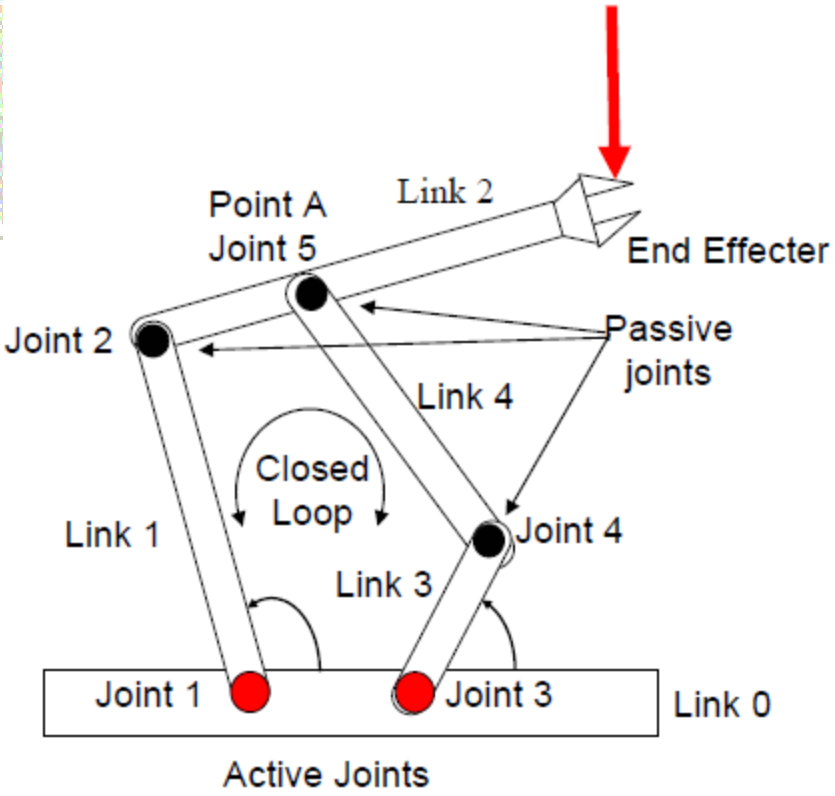
***Gruebler's formula* (3D):** $\#\text{DOF} = 6\left(n_L - n_j\right) + \sum_{i=1}^{n_j} f_i$

number of links (without the base)

Number of joints

#DOF for joint *i*

Parallel or Closed Chain Manipulators



A 2 DOF Closed-Loop Robot Kinematic Chain

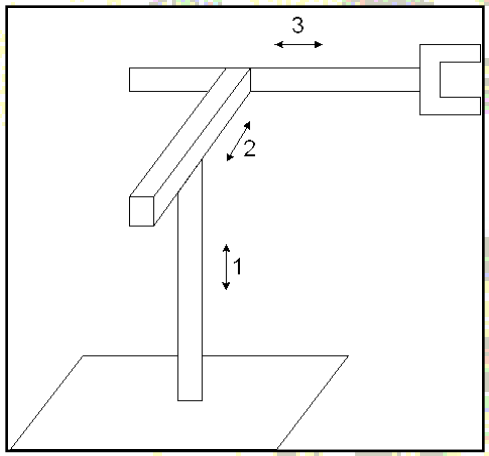
number of links (without base)

#DOF for joint *i*

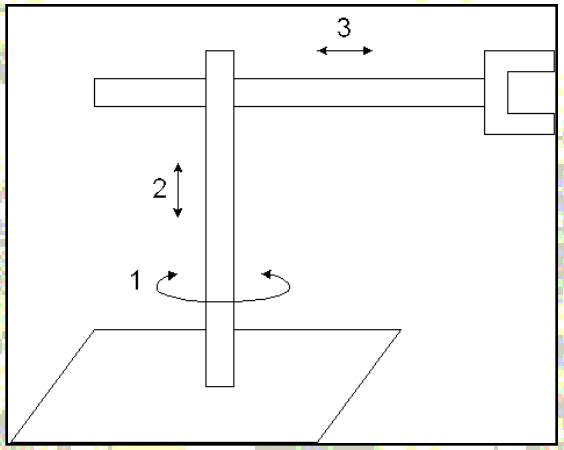
***Gruebler's formula* (2D):** $\#\mathbf{DOF} = 3\left(n_L - n_j\right) + \sum_{i=1}^{n_j} f_i$

number of joints

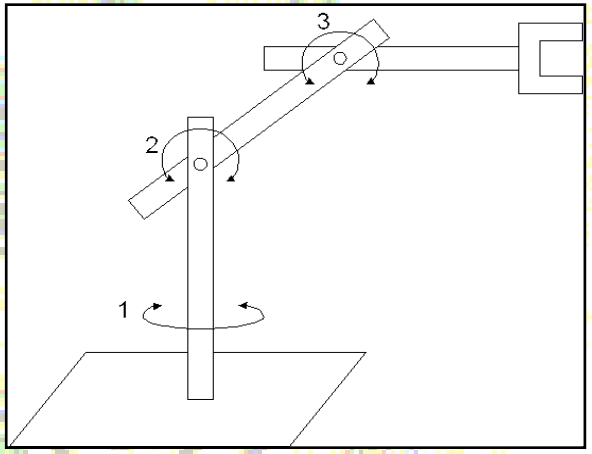
Manipulators



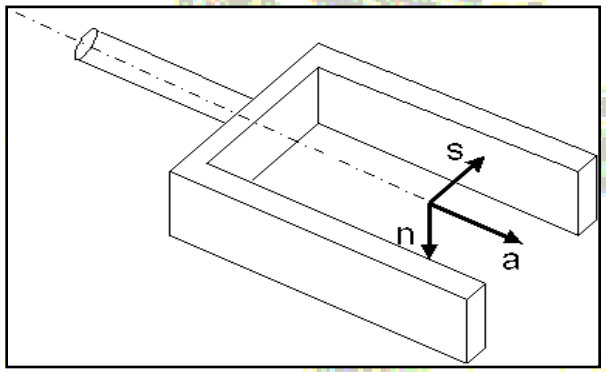
Cartesian



Cylindrical



Articulated



Hand coordinate:

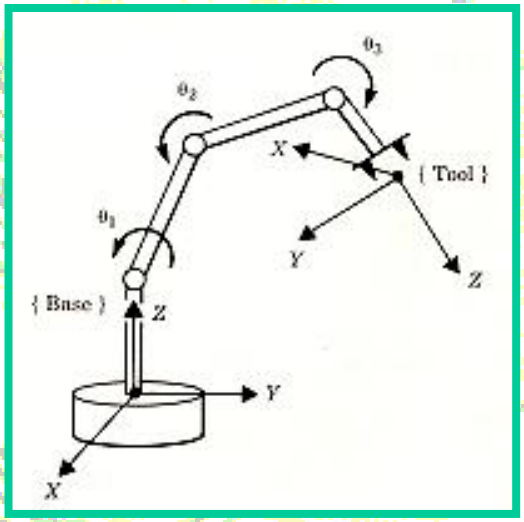
n: normal vector,

s/o: sliding/orientation vector,

a: approach vector; normal to the tool mounting plate

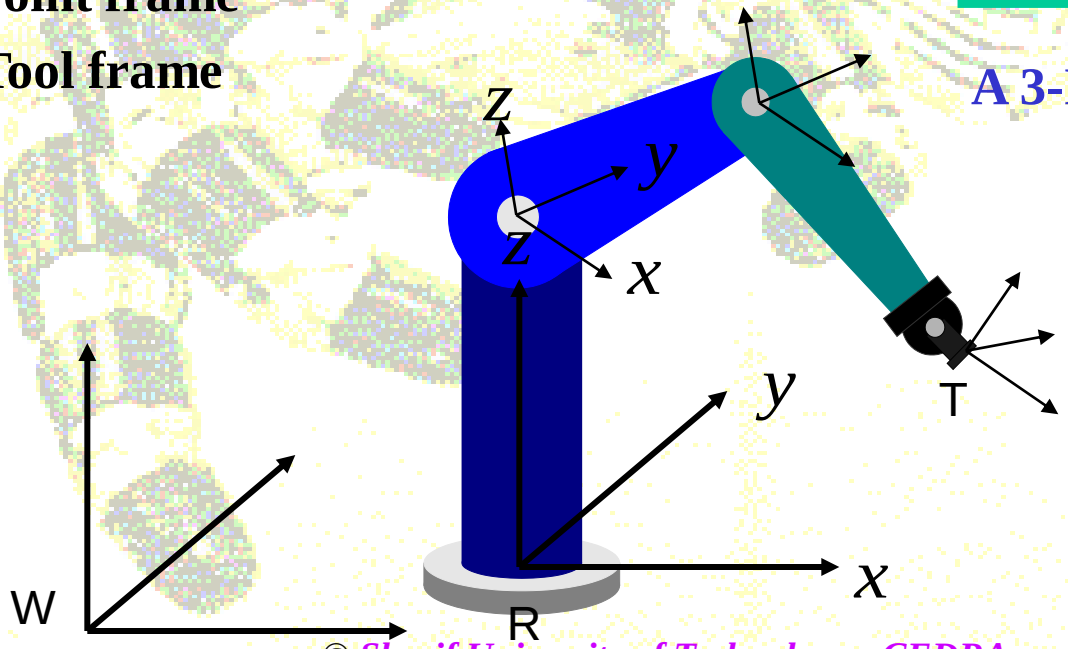
Robot Manipulator Kinematics

Position & Orientation of the manipulator is described by a description of the **{Tool Frame}** relative to the **{Base Frame}.**



A 3-DOF Manipulator Arm

- Robot Reference Frames
  - **World frame**
  - **Joint frame**
  - **Tool frame**



Kinematics

Forward Kinematics: Given a set of joint angles (joint offsets), compute the position and orientation of the tool frame (end-effector) relative to the base frame.

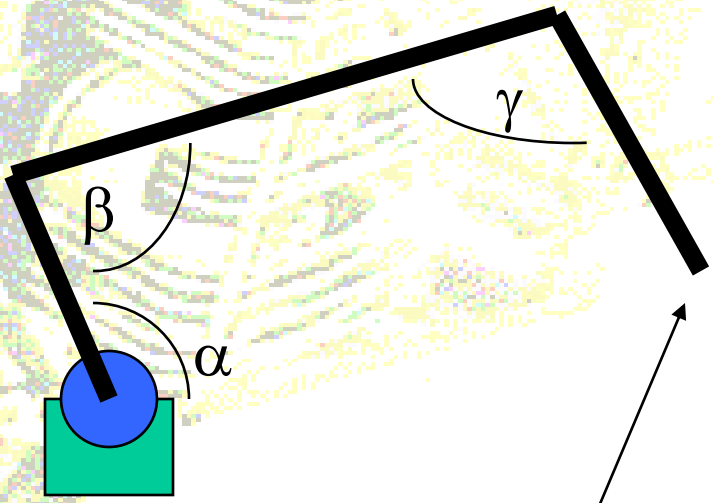
Inverse Kinematics: Given the position and orientation of the end-effector, compute all possible sets of joint angles/offsets which could be used to attain the given position and orientation.

- **More difficult than forward kinematics**
- **Non-linear solutions**
- **Multiple solutions typically exist**

{**No solution** usually means that the manipulator cannot reach the desired position & orientation, because it lies outside of its workspace.}

Forward Kinematics Example

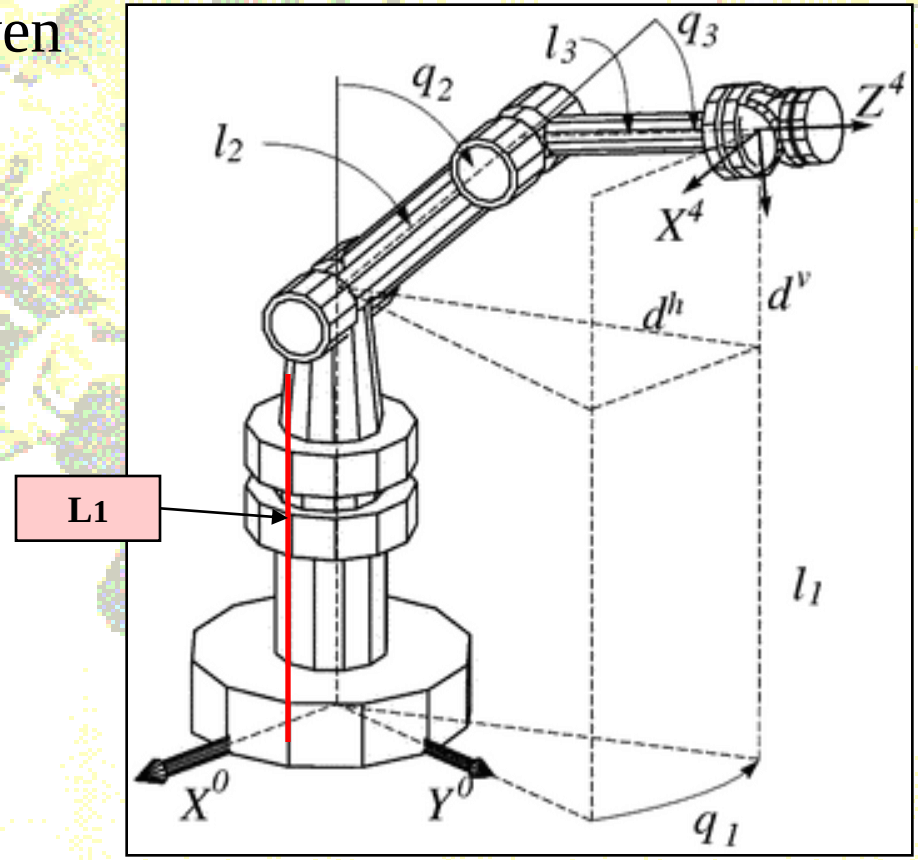
- Given the three angles α, β, and γ, compute the position of the end effector



Where is this in 3 space?

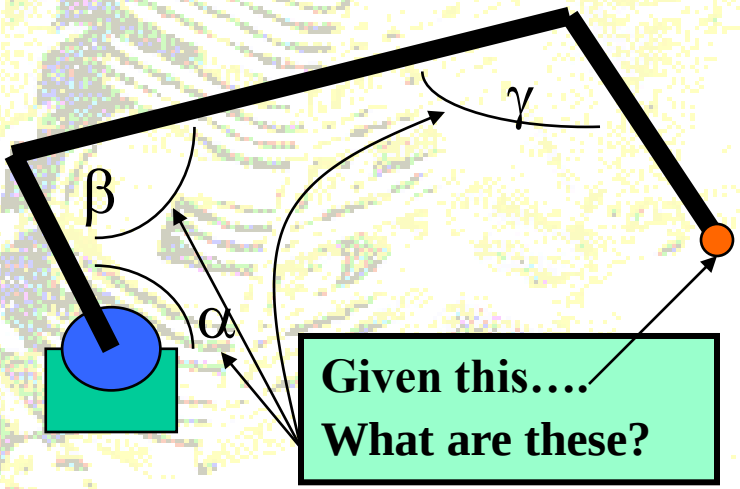
Forward Kinematics Example

- Given the following robot
  - Where is the end effecter given the position of each motor?
- Given the following:
  - L1 = .5 meters
  - L2 = .3 meters
  - L3 = .2 meters
  - Q1 = +30 degrees
  - Q2 = -45 degrees
  - Q3 = -45 degrees

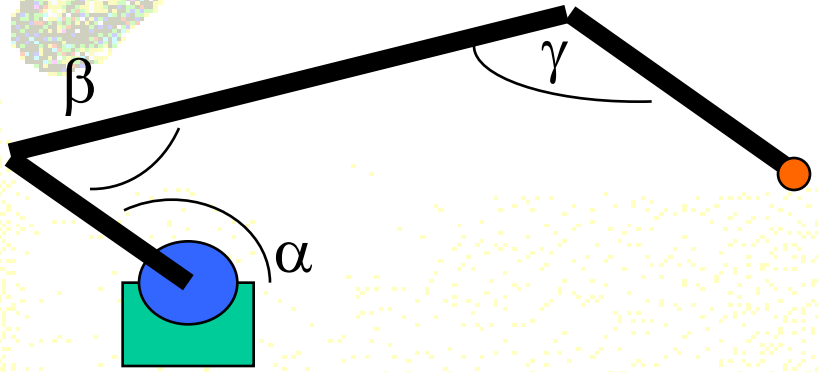


Inverse Kinematics Example

- Given the position of the end effector, what are angles α, β, and γ?



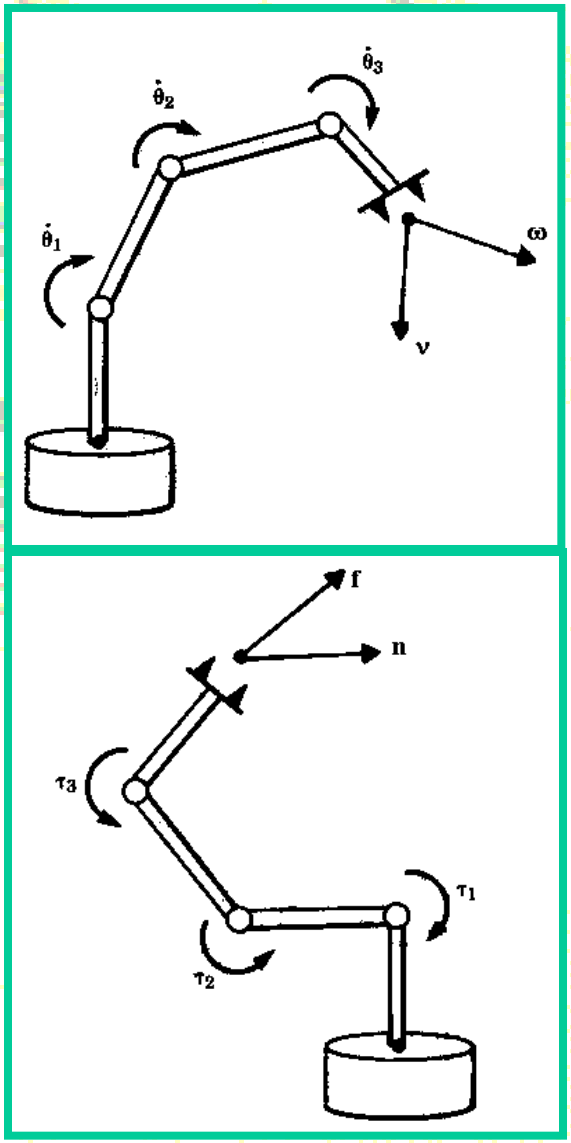
- May be multiple solutions!



Jacobian in Robotics Sense

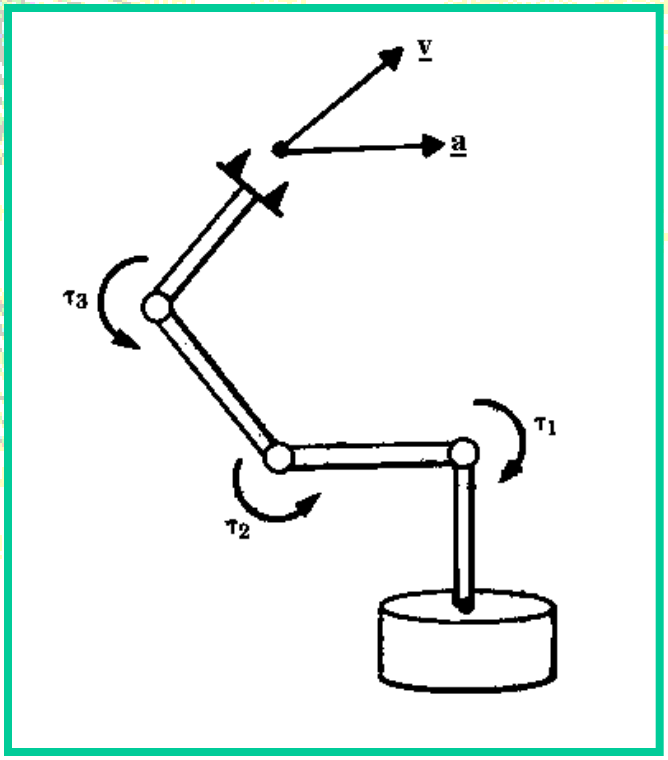
- A matrix quantity called the **Jacobian** specifies a mapping from velocities in {Joint Space} to velocities in {Cartesian space}.

- For a desired contact "static" {force and moment}, **Jacobian** can also be used to compute the set of {Joint Torques} required to generate them.



Robot Manipulator Dynamics

Study of joint forces/torques required to cause the motion in manipulators:



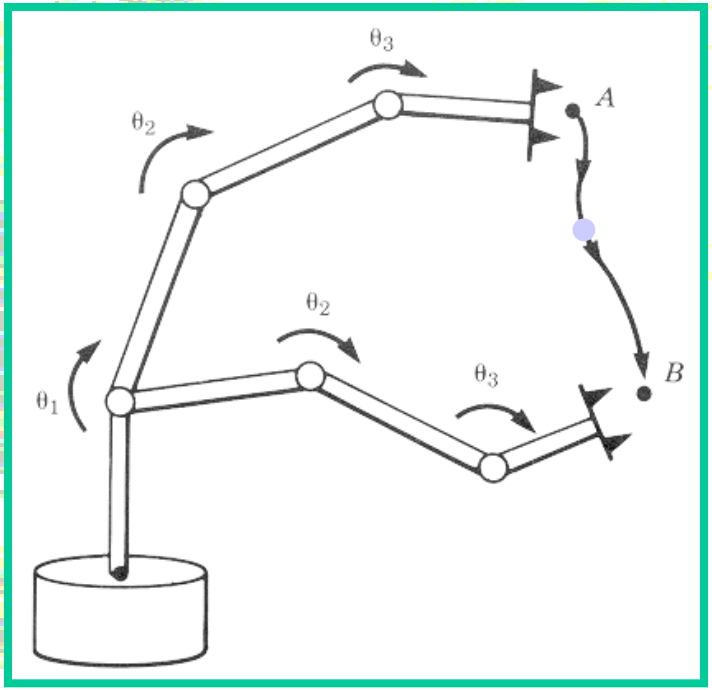
- **Forward Dynamics:** Given a set of joint torques/forces $\{\tau, f\}$, compute the resulting motion of the manipulator ($\theta, \dot{\theta}, \ddot{\theta}$, or $\underline{v}, \underline{a}$ of the end-effector). (Useful for Simulation of the Manipulator)

- **Inverse Dynamics:** Given the desired motion of the manipulator/end-effector, compute the complex set of torque/force functions which must be applied by the joint actuators in order to generate the desired motion. (i.e. to accelerate an arm from rest, glide at a constant end-effector velocity, and finally decelerate to a stop). (Useful for Controlling the Manipulator)

Robot Trajectory Generation

To move a manipulator from point A to point B in a smooth and controlled fashion, we need to cause each joint to move as specified by a smooth function of time. (commonly, each joint starts and stops at the same time so that the arm's motion appears coordinated).

➔ **Via-Points:** Sometimes a path is described not only by a desired destination but also by some intermediate points, through which the arm must pass to reach the destination.



(To move the arm from A to B, we must compute a trajectory function for each joint to follow).

Robotic Sensors

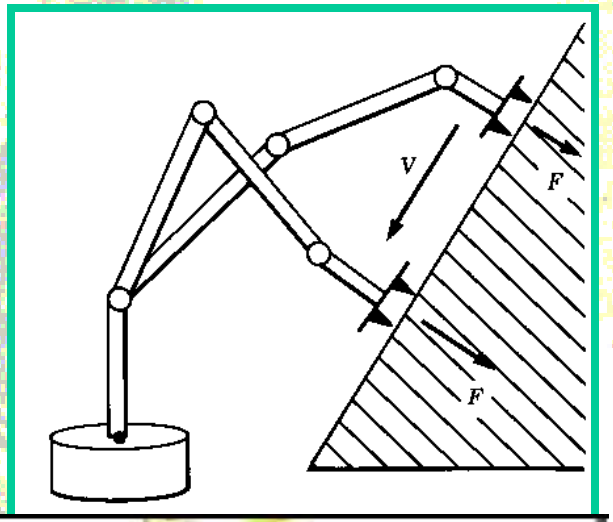
- **Internal Sensors: Are used to control robot joints motion (i.e. Potentiometers, Tachometers, Strain Gages, Micro-switches).**
- **External Sensors: Are used to monitor and control robots motion with the environment.**
  - **Tactile Sensors: They respond to contact forces with another object.**
  - **Proximity/Range Sensors: A device that indicates when an object is close to another object before contact has been made, and is useful in non-contact tasks such as spray painting (i.e. ultrasonic sensors, SONAR systems, etc.).**
  - **Miscellaneous Sensors: Other sensors like temperature, pressure, etc.**
  - **Machine Vision: A device (Camera) capable of viewing the robots workspace to perform inspection, parts recognition, etc.**

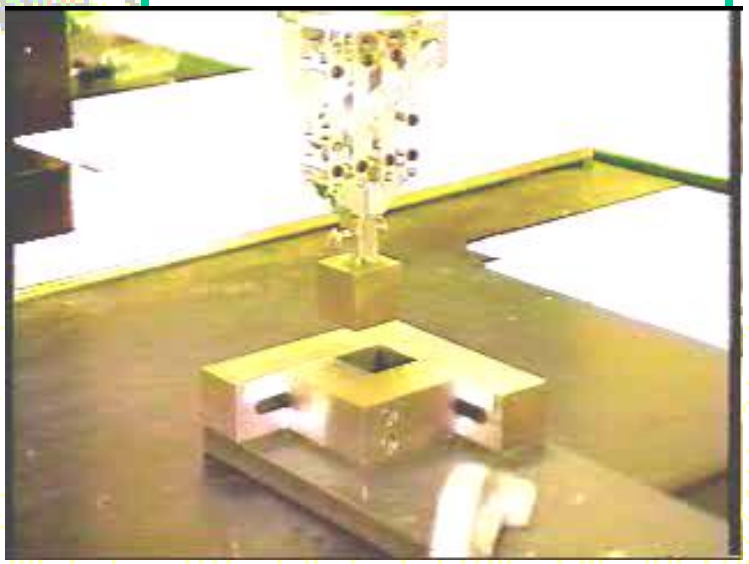
Robot Position and Force Control

- **Position Control System**: It is implemented to cause the robot arm to follow a desired trajectory. This system uses feedback from joint sensors (position & velocity) to keep the manipulator on course, when the arm moves in free space.

- **Force Control System**: This is complementary to the position control system , and becomes necessary when the arm is touching (in contact with) a rigid surface, and applies a constant force.

- **Hybrid (Position-Force) Control System**: Since manipulators are rarely constrained by reaction surfaces in all directions simultaneously, a hybrid (mixed) control system is generally applied, with some directions controlled by a position control law and remaining directions controlled by a force control law.

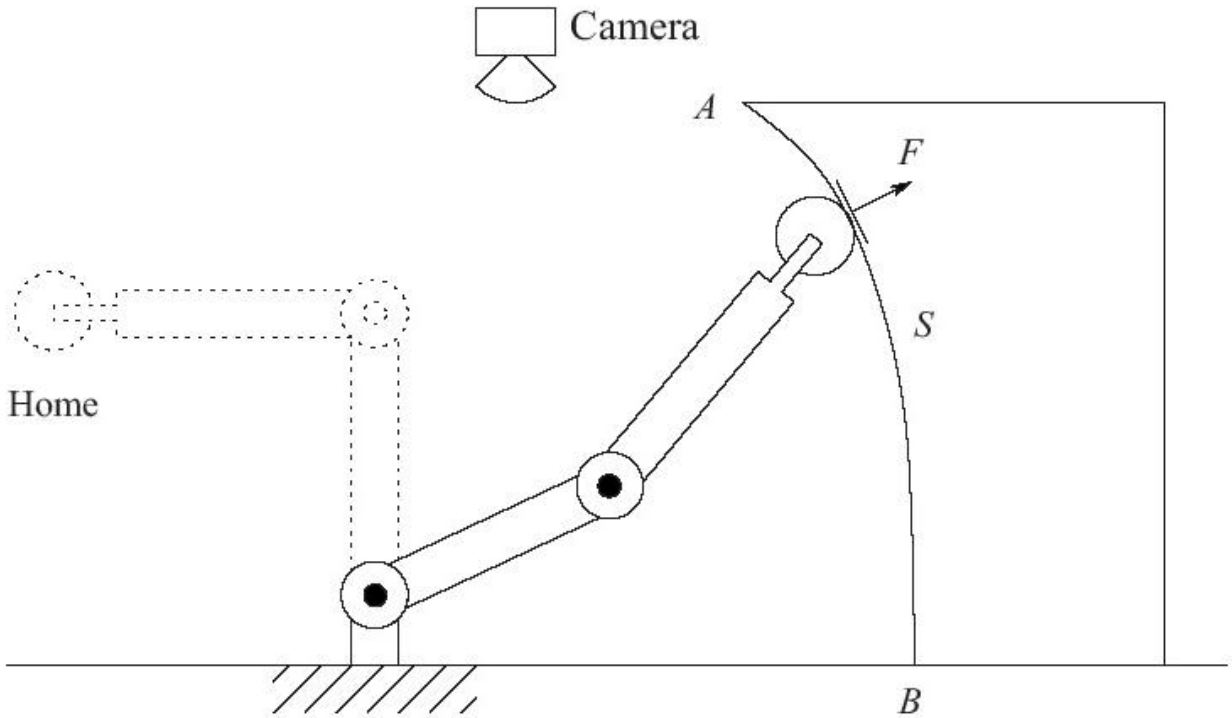




Robot Position and Force Control

Example: control of a 2DOF planar manipulator

Move from “home” position and follow the path **AB** with a constant contact force ***F*** all using visual feedback.

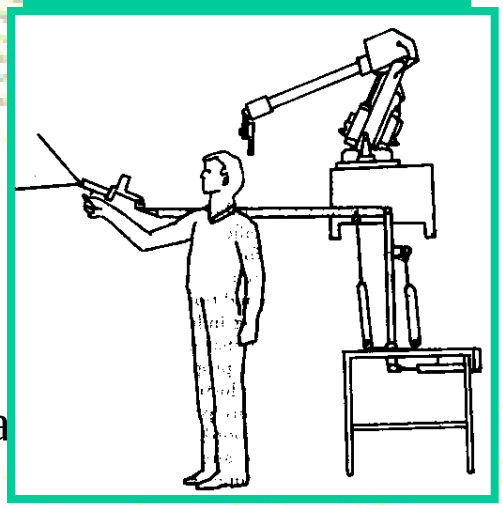


Robot Programming

Four programming methods are employed in robotics:

- **Manual Teaching/Show-and-Teach:** This is done by a teach box (teach pendant) on which a number of buttons and joysticks enable the programmer to move the robot around and save every move in the computer memory, and creating a file that can be replayed by the robot.

- **Lead-and-Teach/Guiding:** This is done by physically grasping the robot end-effector and leading it through any desired path at the required speed/acceleration while simultaneously storing the continuous positions and orientations, and creating a file for robot to replay.

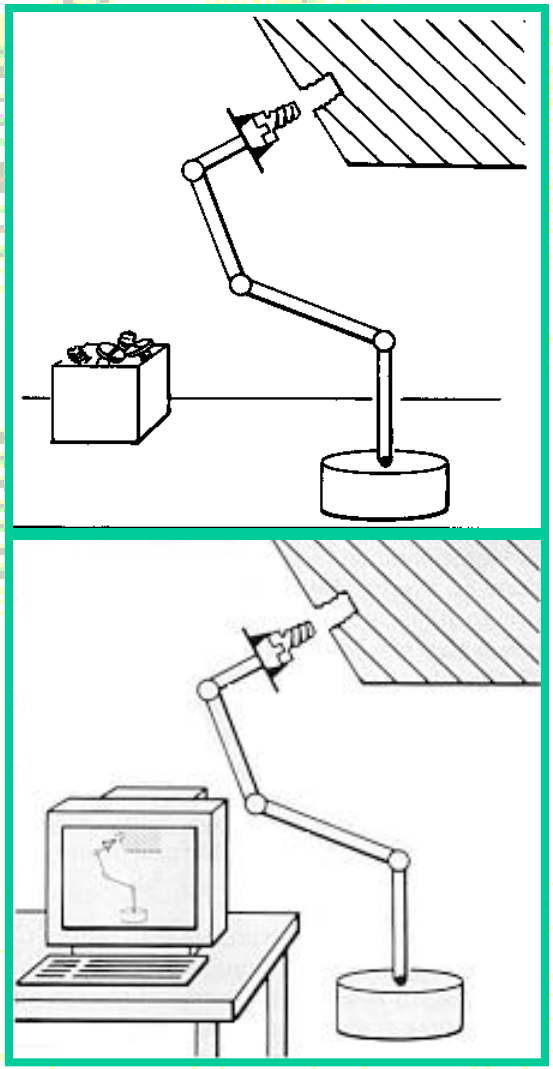


A Robot Simulator for Lead-Through Teaching

Robot Programming

- **Robot Programming Language (RPL): It serves as the interface between the human user and the industrial robot. The desired motions of the arm and end-effector, desired contact forces, and complex manipulation strategies can all be described in a RPL (i.e. VAL-II, AL, AML, AR-Basic, etc. see chapter-12).**

- **Off-Line Programming (OLP) System: This technique is generally extended by means of computer graphics (the development of robot programs takes place without access to robot itself). It becomes important when the robot needs to be reprogrammed without being tied up.**



Book Notations

- **Uppercase Variables: Vectors/Matrices (i.e. A, J, M, V)**
- **Lowercase Variables: Scalars (m, a, b, c)**
- **Leading Sub/Super Scripts: Coordinate system quantity written in.**

  Ex: ${}^{A}P$: Position Vector written in Coord. Sys. {A}

- **Trailing Subscripts: Vector Components (a_x , a_y)**
- **Trailing Superscripts: Inverse/Transpose of a Matrix**

  Ex: (A^{-1}, R^{T})

- **Trigonometric Functions:**

  Ex: Cos θ_1= Cθ_1= C_1
  Sin θ_1= Sθ_1= S_1