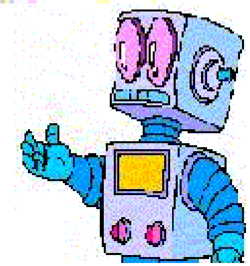
INTRODUCTION TO ROBOTICS

(Kinematics, Dynamics, and Design)

SESSION # 18:

ROBOT TRAJECTORY GENERATION



Ali Meghdari, Professor

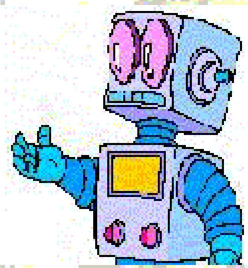
School of Mechanical Engineering

Sharif University of Technology

Tehran, IRAN 11365-9567

Homepage: http://meghdari.sharif.edu

Robot Motion Trajectory Generation



Robot Motion & Trajectory Generation

Motion Planning: refers to the study of generating motion for the robot to accomplish a task. This consists of:

Path Planning; generating a feasible path from an initial position to a final position by describing the geometric position and orientation of the robot during the transition.

Trajectory Generation; attaching a time frame to the paths generates a trajectory. The trajectory not only describes the position of the robot during motion, but also how that position changes with time.

Robot Motion Trajectory Generation

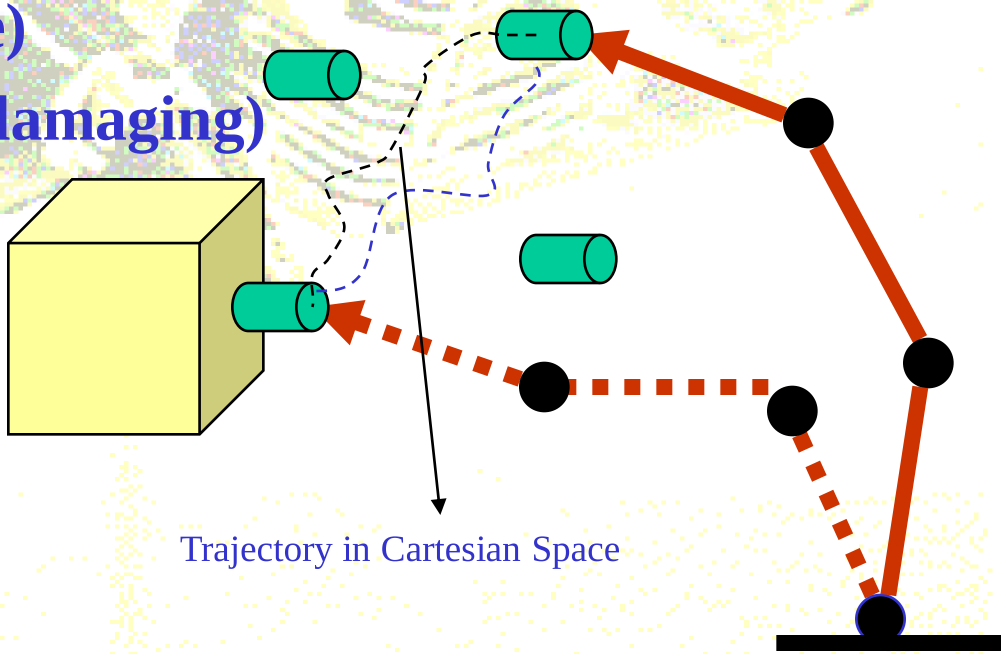
We shall study Methods to:

- Compute a *Trajectory* that describes the desired motion of a manipulator in space…
- *Trajectory* is the time history of position, velocity, and acceleration for each joint (i.e. degrees-of-freedom).
- *Trajectory Planning* means the determination of the actual trajectory, or path, along which the robot end effector will move in Cartesian Space.

Robot Motion Trajectory Generation

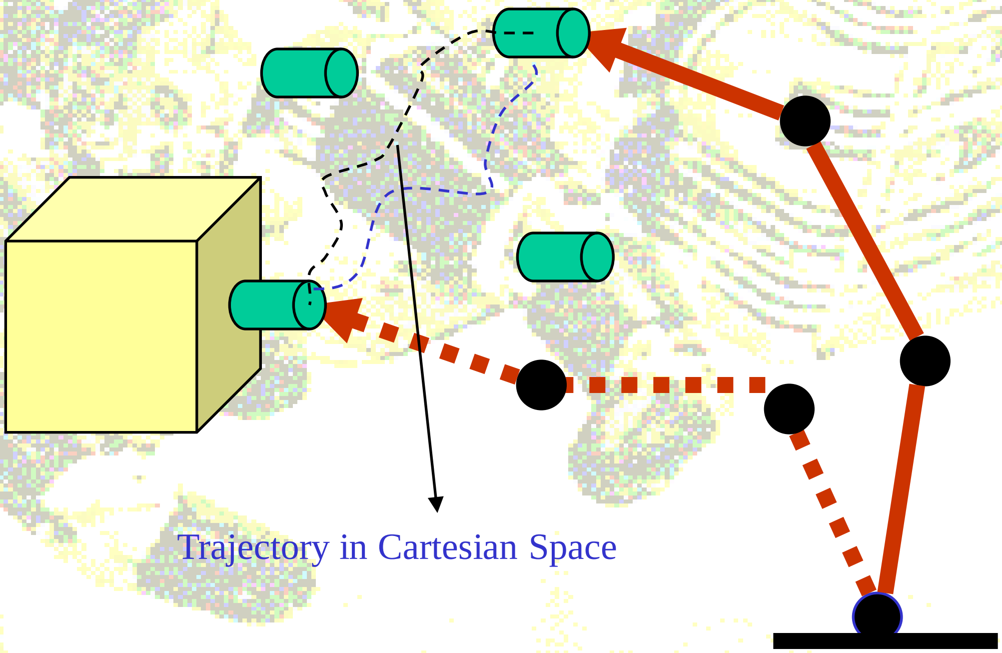
In executing a trajectory, a manipulator moves from its *initial position* to a desired *goal (final) position*. However, the motion may be:

1. ***Smooth* (nice and secure)**
2. ***Rough* (vibrations and damaging)**



Robot Motion Trajectory Generation

In general, the *task* of a robot manipulator is defined by a sequence of points which are denoted as end points and are stored in the robot's control computer.

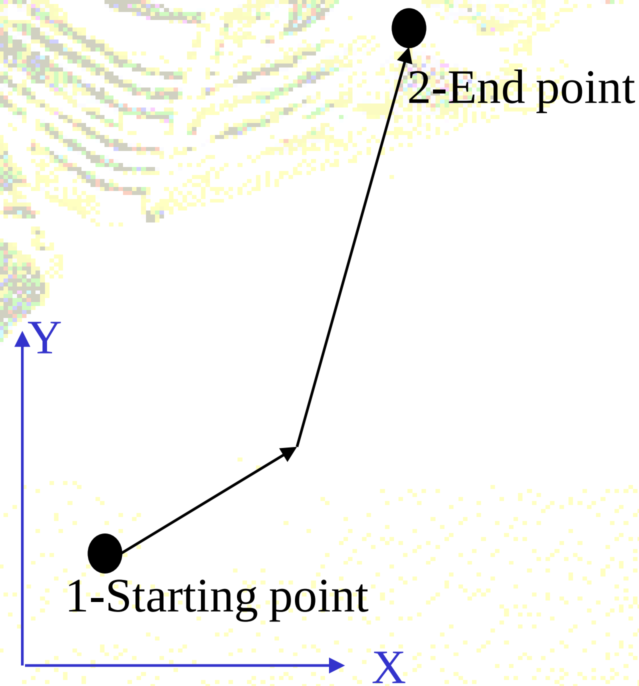


Motions of a robot manipulator is described as the motions of the Tool Frame {T} relative to the Station Frame {S}.

Robot's Motion Classifications

- Point-to-Point (PTP) Robotic Systems: The robot moves to a numerically defined location and then the motion is stopped. Then, the end-effector performs the required task with the robot being stationary. Upon completion of the task, the robot moves to the next point and the cycle is repeated. (Ex: Spot Welding Operations)

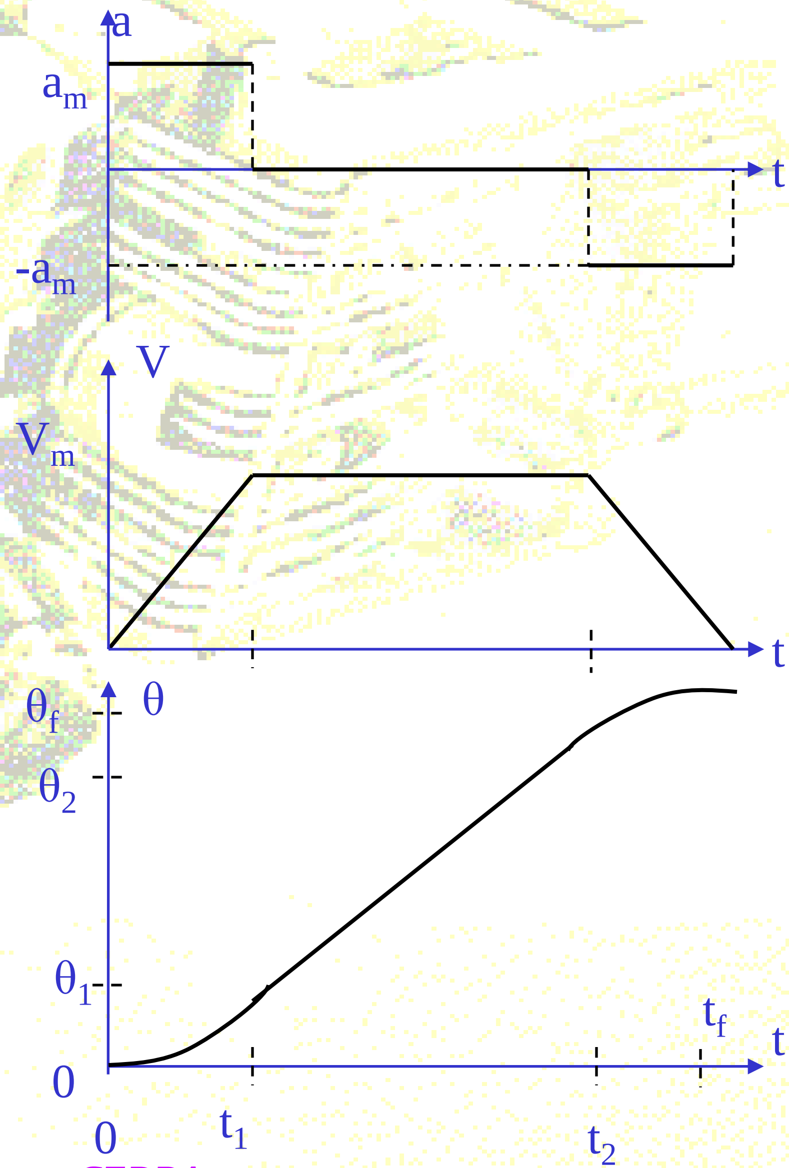
Therefore; in PTP robots, the path of the robot and its velocity while traveling from one point to the next is not important!



Trajectory in PTP Cartesian Robotic System

Robot's Motion Classifications

- **Example:** Consider a simple case in which each joint of a PTP robot must move to its end point θ_f as fast as possible without exceeding a maximum admissible acceleration a_m and a maximum velocity V_m.
- **Desired Joint Trajectory consists of 3-segments:**
  1. $[0, t_1]$; based on *max. acceleration*
  2. $[t_1, t_2]$; based on *max. velocity*
  3. $[t_2, t_f]$; based on *max. deceleration*



Typical acceleration, velocity and position of an axial motion in PTP Robot.

Robot's Motion Classifications

- Lets assume that the time durations of acceleration and deceleration segments are equal:$\{t_1 = t_f - t_2\}$. The control program must calculate the two switching times based on the initial and final joint values ($\theta_0 = 0$ and θ_f) and V_m and a_m.

- **Joint Trajectory during:**

***1st Segment*:** $\theta(t) = \dfrac{a_m t^2}{2}, \quad \theta_1 = \dfrac{a_m t_1^2}{2}, \quad V_m = a_m t_1$

***2nd Segment*:**

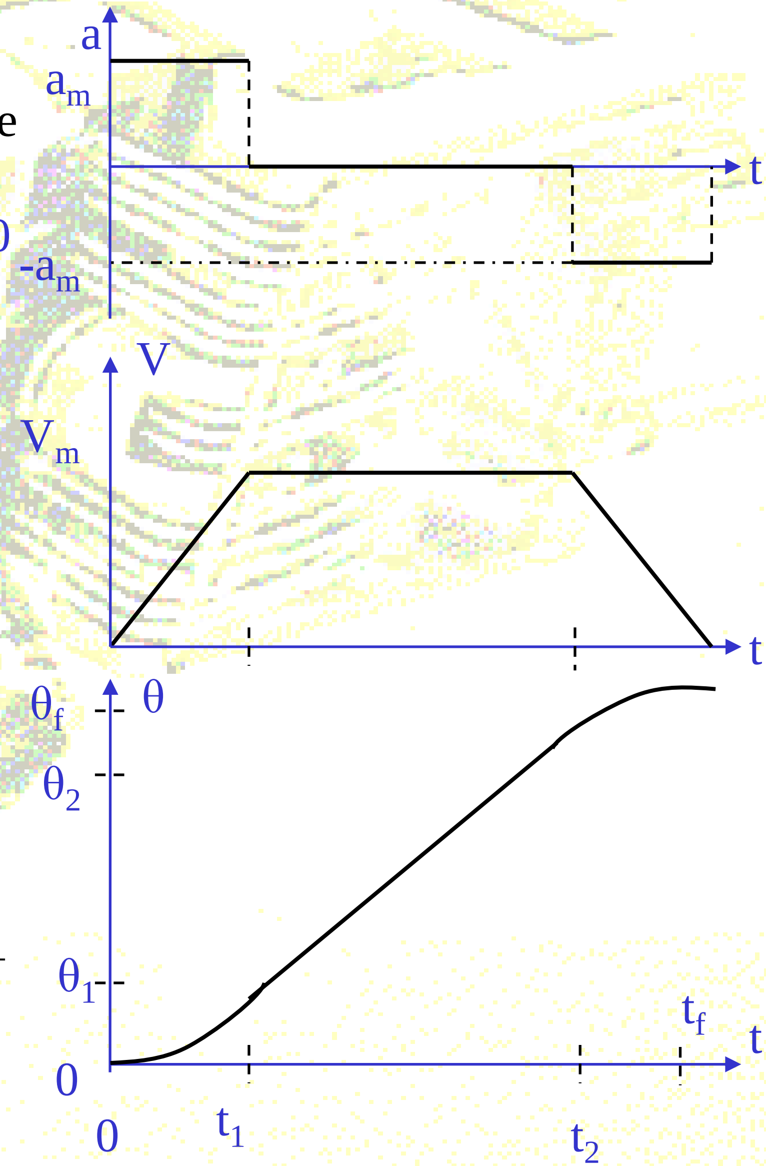
$$\theta(t) = \theta_1 + V_m(t - t_1), \quad \theta_2 = \theta_1 + V_m(t_2 - t_1),$$

***3rd Segment*:**

$$\theta(t) = \theta_2 + V_m(t - t_2) - \frac{a_m(t - t_2)^2}{2}, \quad \theta_f = \theta_2 + V_m t_1 - \frac{a_m t_1^2}{2},$$

From above equations we can write:

$$\theta_f = V_m t_2 \Rightarrow t_2 = \frac{\theta_f}{V_m} \quad \text{and} \quad t_1 = \frac{V_m}{a_m}, \quad t_f = \frac{V_m}{a_m} + \frac{\theta_f}{V_m}$$



Typical acceleration, velocity and position of an axial motion in PTP Robot.

Robot’s Motion Classifications

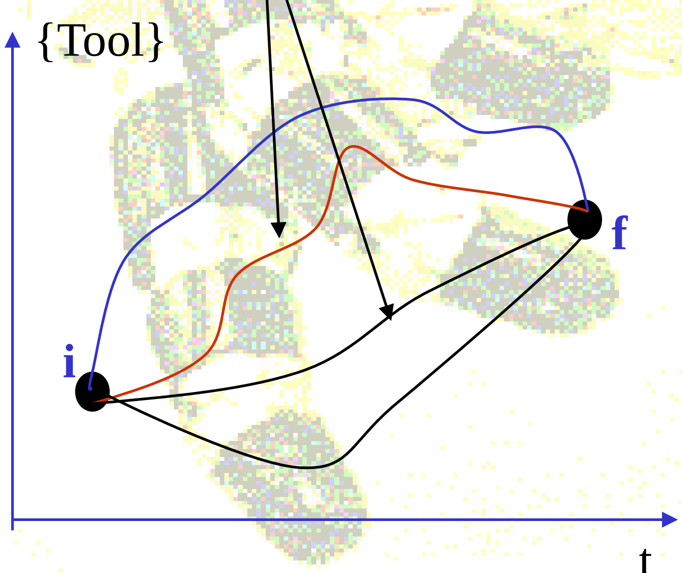
- Continues Path (CP) Robotic Systems: In this system the robot tool performs the task while the axes of motion are moving (both robot and the tool are moving simultaneously, and the speed of each joint can be controlled independently).
  (Ex: Arc Welding Operations)

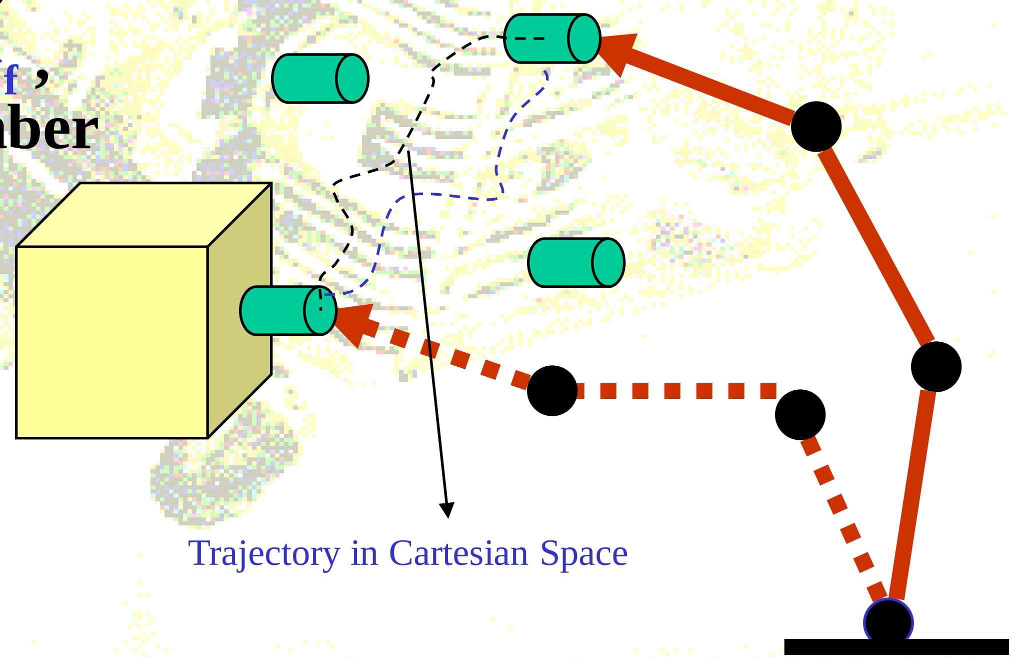
  Therefore; in CP robot operations, the position of the robot’s tool at the end of each segment , together with the ratio of axes velocities, determine the generated trajectory. Ex: variations in the velocity of the arc welding result in a non-uniform weld seam thickness (i.e. an unnecessary metal built-up or even holes)

Robot Motion Trajectory Generation

Motions of a robot manipulator is described as the motions of the Tool Frame {T} relative to the Station Frame {S}.

To move the manipulator's Tool from $\{Tool\}_i$ to $\{Tool\}_f$, there exists an infinite number of *trajectories*.

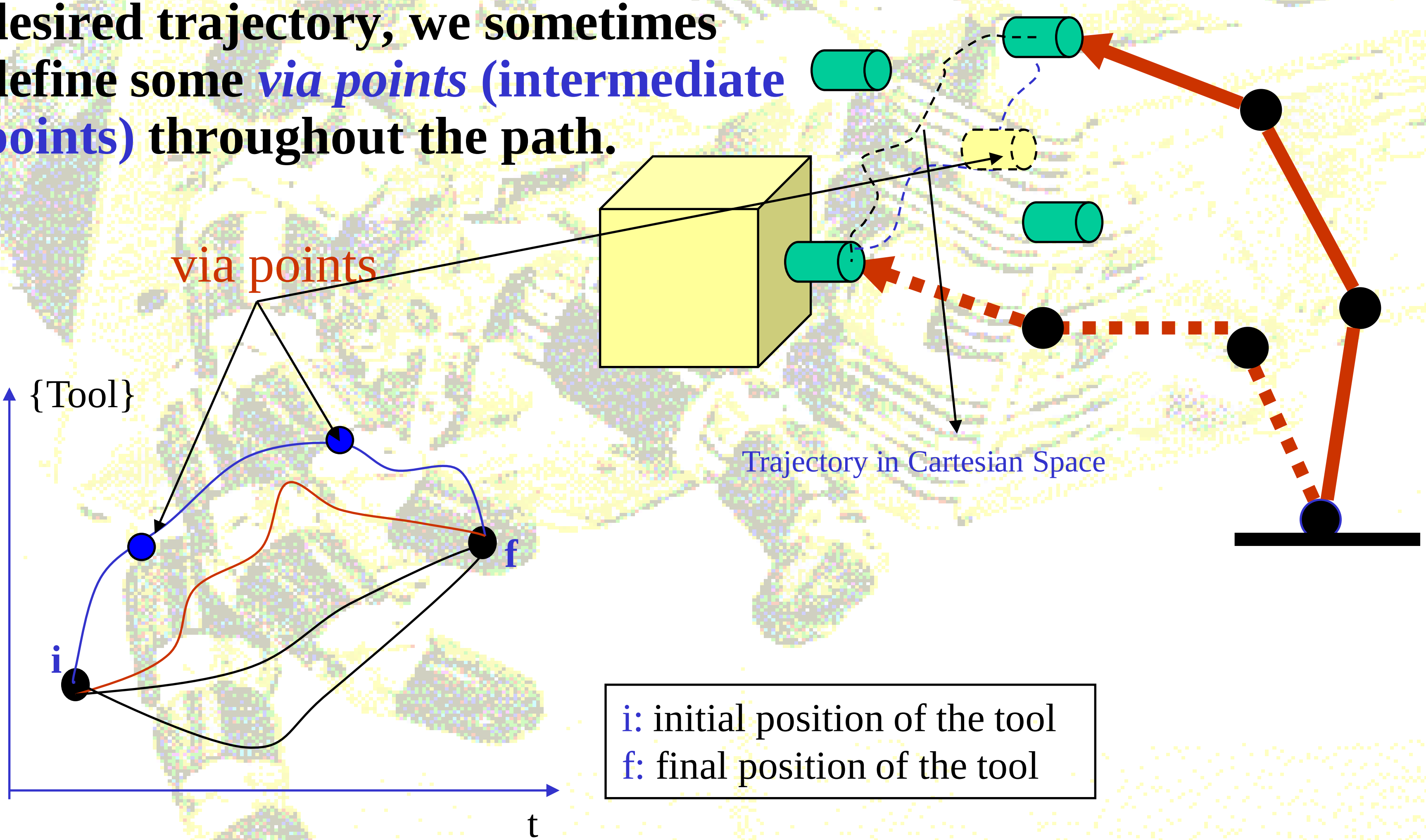




i: initial position of the tool
f: final position of the tool

Robot Motion Trajectory Generation

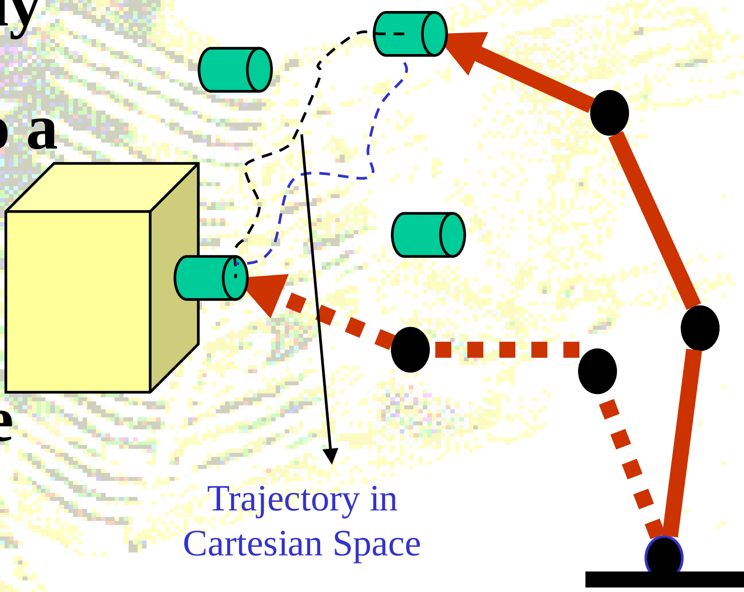
To provide more details about the desired trajectory, we sometimes define some *via points* (intermediate points) throughout the path.



Path Points: a set of Via points plus Initial and Final points.

Robot Motion Trajectory Generation

In executing a trajectory, it is generally desirable to smoothly move the manipulator from its *initial position* to a desired *final position*.



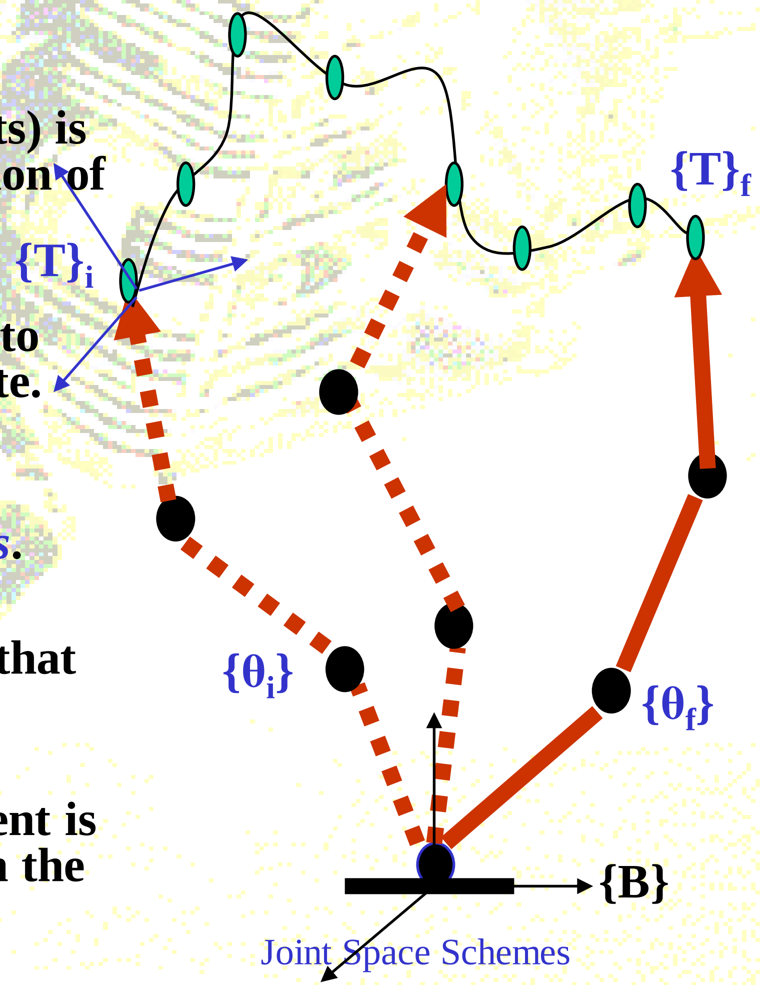
- **For Smooth Motion: We need to have *position* and *velocity* to be *continues functions of time*.**

- **A Smooth Function: A function which is *continues* and has a *continues first derivative*.**

Robot Motion Trajectory Generation

Joint Space Schemes: **Methods of path generation in which path shapes are described in terms of functions of joint angles.**

- **Each *path point* (via points + initial and final points) is specified in terms of a desired position & orientation of the *Tool frame* {T}, relative to the *Base frame* {B}.**
- **In *Cartesian Space* a set of *via* points may be used to take the *Tool frame* from initial state to a final state.**
- **Each via and path point is converted into a set of desired joint angles by applying *inverse kinematics*.**
- **A *Smooth Function* is defined for each of n-joints that pass through the via and path points.**
- **The *time* duration required for each motion segment is the same for each joint, so that all joints will reach the via point at the same time.**



Joint Space Schemes

Robot Motion Trajectory Generation

Joint Space Schemes:

- Each via and path point is converted into a set of desired joint angles by applying *inverse kinematics*.

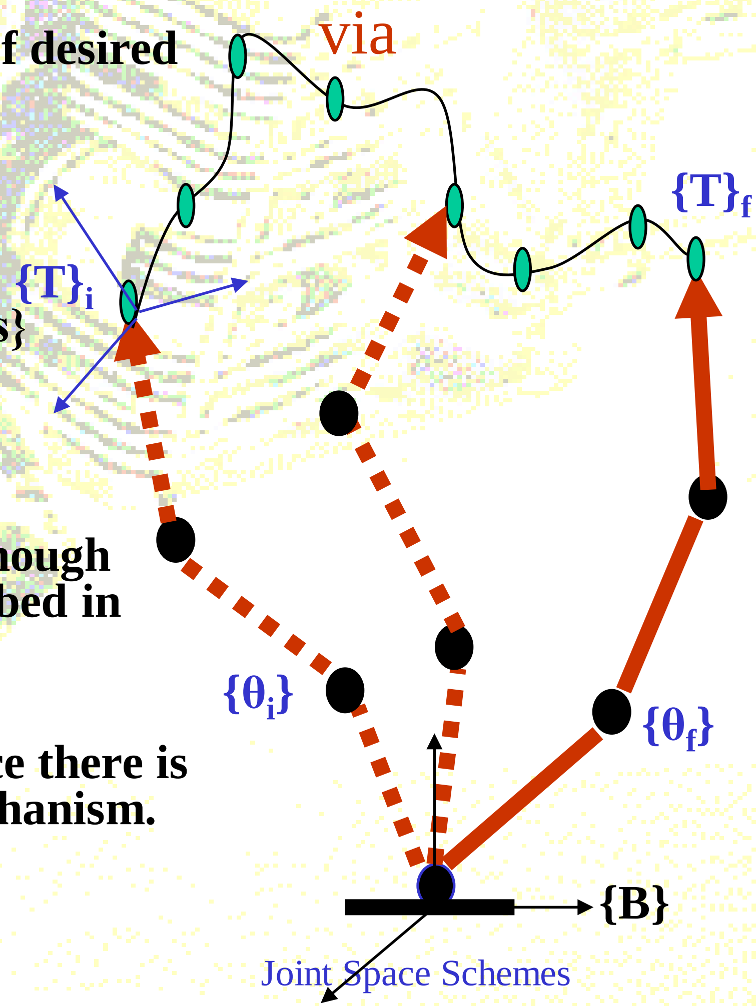
  {Tool} frame =➢ $\{\theta_{vector}\}$

  {by Inverse Kinematics} =➢a set of $\{\theta_i$'s and θ_f's}

- Generate a *Smooth Function* through all points.

- In between via points, the shape of the path, although rather simple in joint space, is complex if described in Cartesian space.

- Joint Space schemes are easiest to compute, since there is usually no problem with *singularities* of the mechanism.



Joint Space Schemes

Robot Motion Trajectory Generation

Cubic Polynomials:

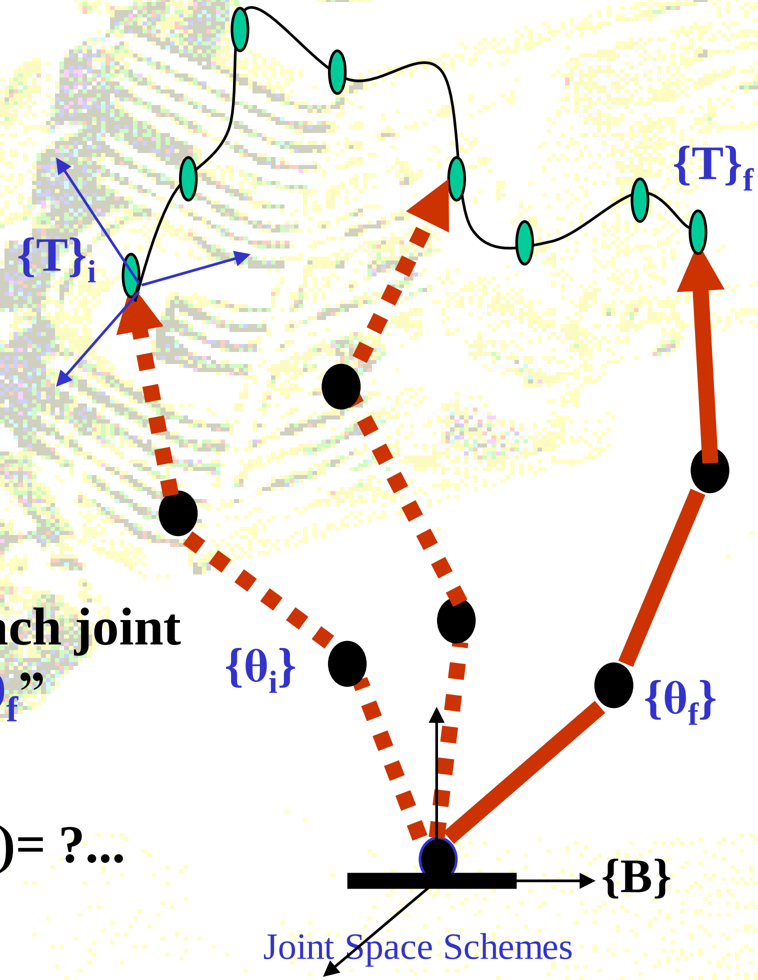
- By applying ***inverse kinematics***.

  $\{Tool\}_i$ =➢ $\{\theta_i$'s $\}$

  $\{Tool\}_f$ =➢ $\{\theta_f$'s $\}$

  We need to define a *smooth function* for each joint whose value at "t_i" is "θ_i", and at "t_f" is "θ_f"

- **There exist many smooth functions for $\theta(t)$= ?...**



Robot Motion Trajectory Generation

Cubic Polynomials:

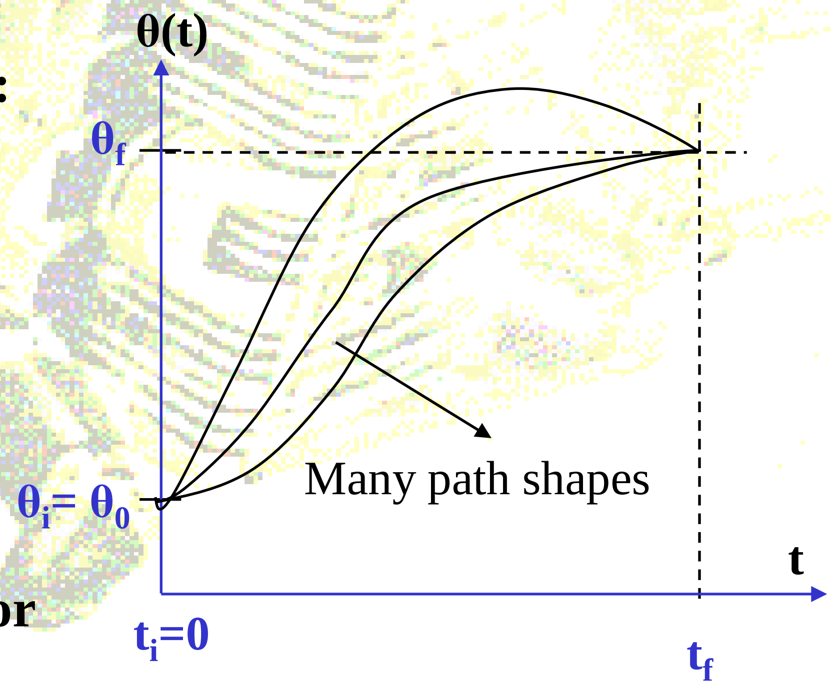
- For Smooth joint motion, we need to impose at least *4-constraints* on θ(t) as:

$$\theta(0) = \theta_i = \theta_0$$

$$\theta(t_f) = \theta_f$$

$$\left.\begin{aligned}\dot{\theta}(0) &= 0\\ \dot{\theta}(t_f) &= 0\end{aligned}\right\}$$

Continues in Velocity: Start and Stop at Zero Velocities



We need to define a *smooth function* for each joint whose value at "t_i" is "θ_i", and at "t_f" is "θ_f"

- **There exist many smooth functions for θ(t)= ?...**

Robot Motion Trajectory Generation

Cubic Polynomials:

- **A Cubic Polynomials satisfies the above constraints.**

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

(4-Coefficients)

$$\left.\begin{aligned}
\theta(0) &= \theta_0 = a_0 \\
\theta(t_f) &= \theta_f = a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \\
\dot{\theta}(0) &= 0 = a_1 \\
\dot{\theta}(t_f) &= 0 = a_1 + 2a_2 t_f + 3a_3 t_f^2
\end{aligned}\right\} \Rightarrow
\begin{aligned}
a_0 &= \theta_0 \\
a_1 &= 0 \\
a_2 &= \frac{3}{t_f^2}(\theta_f - \theta_0) \\
a_3 &= -\frac{2}{t_f^3}(\theta_f - \theta_0)
\end{aligned}$$

Robot Motion Trajectory Generation

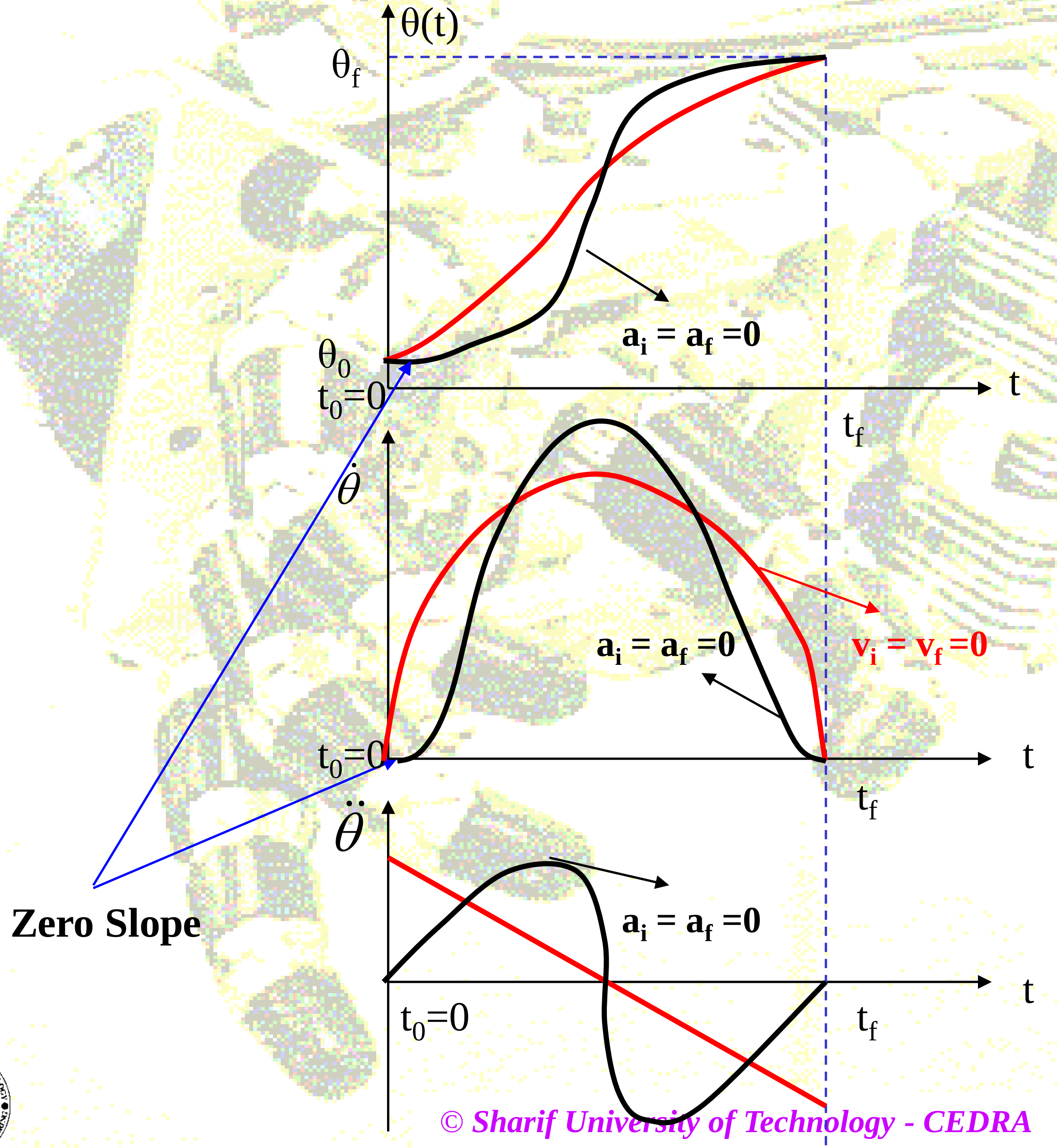
5th Order Polynomials:

- If we add the *acceleration constraints* at the beginning and end of the path; then we have 6-Constraints.
- If acceleration is continues, then vibration is diminished (i.e. if $a_i = a_f = 0$) at the *start* and the *stopping* points.
- Hence, a 5th Order Polynomial would be sufficient to define the joint motion.

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

(6-Coefficients)

Position, Velocity, and Acceleration Curves

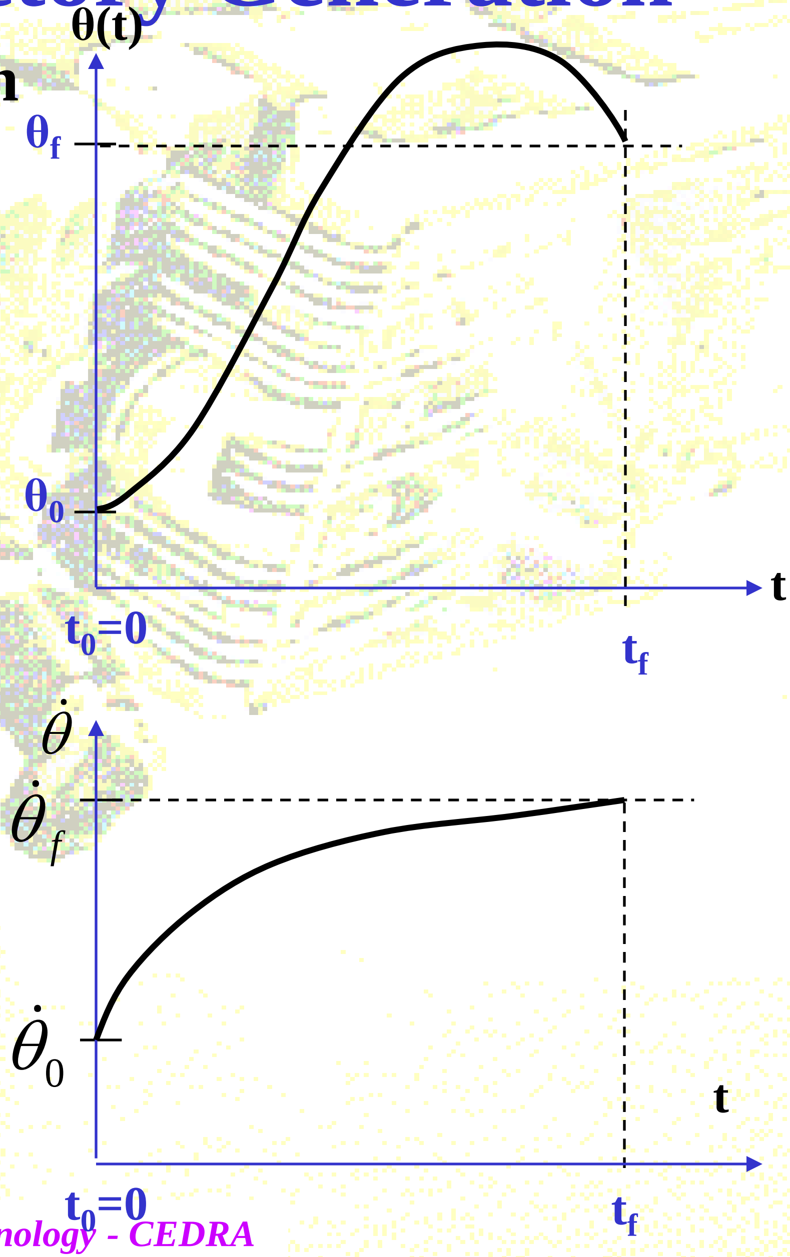


Robot Motion Trajectory Generation

For *intermediate (via) points* at which you don't want the robot to stop, velocity constraints at the end of cubic polynomial is no longer zero.

$$\left.\begin{aligned} \theta(0) &= \theta_0 \\ \theta(t_f) &= \theta_f \\ \dot{\theta}(0) &= \dot{\theta}_0 \\ \dot{\theta}(t_f) &= \dot{\theta}_f \end{aligned}\right\}$$ **4-Constraints**

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$



Robot Motion Trajectory Generation

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

4-Coefficients

$$\begin{cases} a_0 = \theta_0 \\ a_1 = \dot{\theta}_0 \\ a_2 = \dfrac{3}{t_f^2}(\theta_f - \theta_0) - \dfrac{2}{t_f}\dot{\theta}_0 - \dfrac{1}{t_f}\dot{\theta}_f \\ a_3 = -\dfrac{2}{t_f^3}(\theta_f - \theta_0) + \dfrac{1}{t_f^2}(\dot{\theta}_f + \dot{\theta}_0) \end{cases}$$

➢ **Several Ways to specify the desired velocities at the *via points*:**

- ❖ **By the user in terms of Cartesian linear and angular velocities (mapped into joint velocities by Jacobian Inverse)**
- ❖ **The system automatically choose…**
- ❖ **The system automatically choose such that the acceleration is continues…**

Robot Motion Trajectory Generation

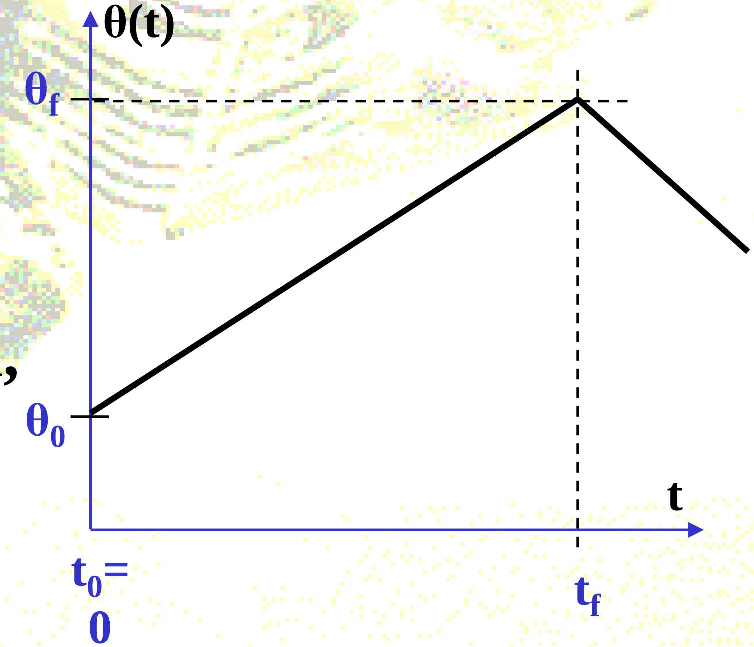
Linear Function with Parabolic Blends:

- **Another choice of path shape is to move *Linearly* from *initial* joint position "θ_0" to the *final* joint position "θ_f".**

- **Linear Function Results in:**

  1. Velocity to be *discontinuous* at the beginning and end of motion,

  2. Acceleration to be *discontinuous*



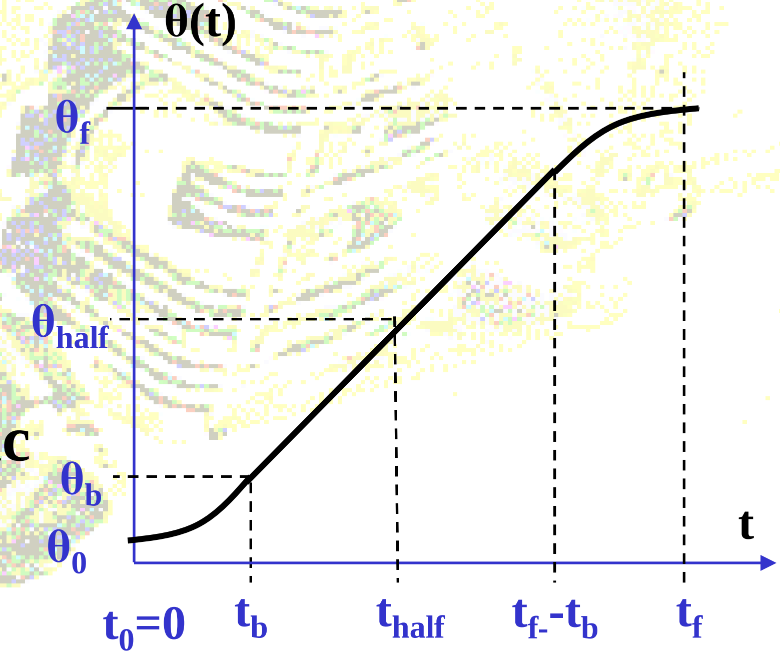
(Linear Function)

- **Undesirable (*Rough Motion*)**

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

- **To avoid rough motion, we can *add a Parabolic Blend* to the linear function at each end.**
- **Adding *Parabolic blends* with *constant acceleration* $(\ddot{\theta}_b)$ results in smooth change in velocity.**
- **Linear function and two parabolic functions are splined together so that the entire path is continuous in *Position* and *Velocity*.**



(Linear Function with Parabolic Blends)

$$\theta(t) = a_0 + a_1 t + a_2 t^2$$

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

- **Assume both blends have the same time duration, and note that velocity at the end of the blend must be equal to the velocity of the linear segment.**

For the *Linear* segment:

$$\dot{\theta}(t_b) = \ddot{\theta}_b t_b = \frac{\theta_h - \theta_b}{t_h - t_b} = (slope - speed)$$

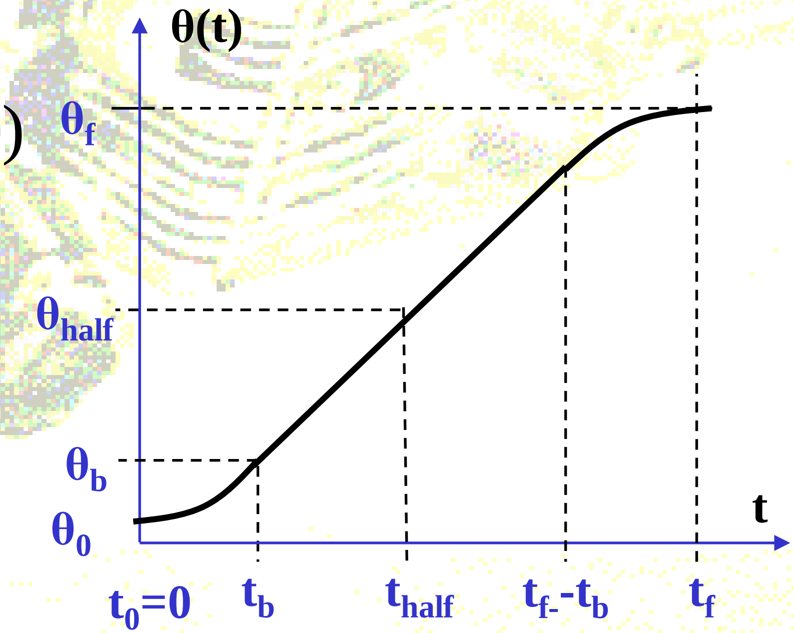
For the *Blend* region:

$$\ddot{\theta}_b = Cons\tan t$$

$$\text{Integrating using I.C.:} \begin{cases} \dot{\theta}_b(0) = 0 \\ \theta_b(0) = \theta_0 \end{cases}$$

$$\dot{\theta}_b = \ddot{\theta}_b t + \cancelto{0}{C_1}$$

$$\theta_b = \frac{1}{2}\ddot{\theta}_b t^2 + \cancelto{0}{C_1} t + \cancelto{\theta_0}{C_2}$$



(Linear Function with Parabolic Blends)

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

$$\theta_b = \frac{1}{2}\ddot{\theta}_b t_b{}^2 + \theta_0$$ at t_b, and let: $t=2t_h$

Combining the above relations, we get:

$$\ddot{\theta}_b t_b = \frac{\theta_h - \theta_b}{t_h - t_b} = \frac{\frac{\theta_f + \theta_0}{2} - \frac{1}{2}\ddot{\theta}_b t_b^2 - \theta_0}{\frac{1}{2}t - t_b} \Longrightarrow \boxed{\ddot{\theta}_b t_b^2 - \ddot{\theta}_b t t_b + (\theta_f - \theta_0) = 0}$$

Where:

$$\begin{cases} t: & \text{Desired duration of motion} \\ \ddot{\theta}_b: & \text{Acceleration acting during the blend region} \end{cases}$$

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

Given any "θ_f", "θ_0", and t, we can follow any of the paths given by choices of $\ddot{\theta}_b$ and t_b which satisfy equation "*".

$$\boxed{\ddot{\theta}_b t_b^2 - \ddot{\theta}_b t t_b + (\theta_f - \theta_0) = 0}$$

You may pick $\ddot{\theta}_b$ and solve for t_b ?

$$t_b = \frac{t}{2} - \frac{\sqrt{\ddot{\theta}_b^2 t^2 - 4\ddot{\theta}_b(\theta_f - \theta_0)}}{2\ddot{\theta}_b} \geq 0$$

$$\ddot{\theta}_b^2 t^2 \geq 4\ddot{\theta}_b(\theta_f - \theta_0) \Rightarrow \ddot{\theta}_b \geq \frac{4(\theta_f - \theta_0)}{t^2}$$

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

You may pick $\ddot{\theta}_b$ and solve for t_b ?

$$t_b = \frac{t}{2} - \frac{\sqrt{\ddot{\theta}_b^2 t^2 - 4\ddot{\theta}_b(\theta_f - \theta_0)}}{2\ddot{\theta}_b} \geq 0$$

$$\ddot{\theta}_b^2 t^2 \geq 4\ddot{\theta}_b(\theta_f - \theta_0) \Rightarrow \ddot{\theta}_b \geq \frac{4(\theta_f - \theta_0)}{t^2}$$

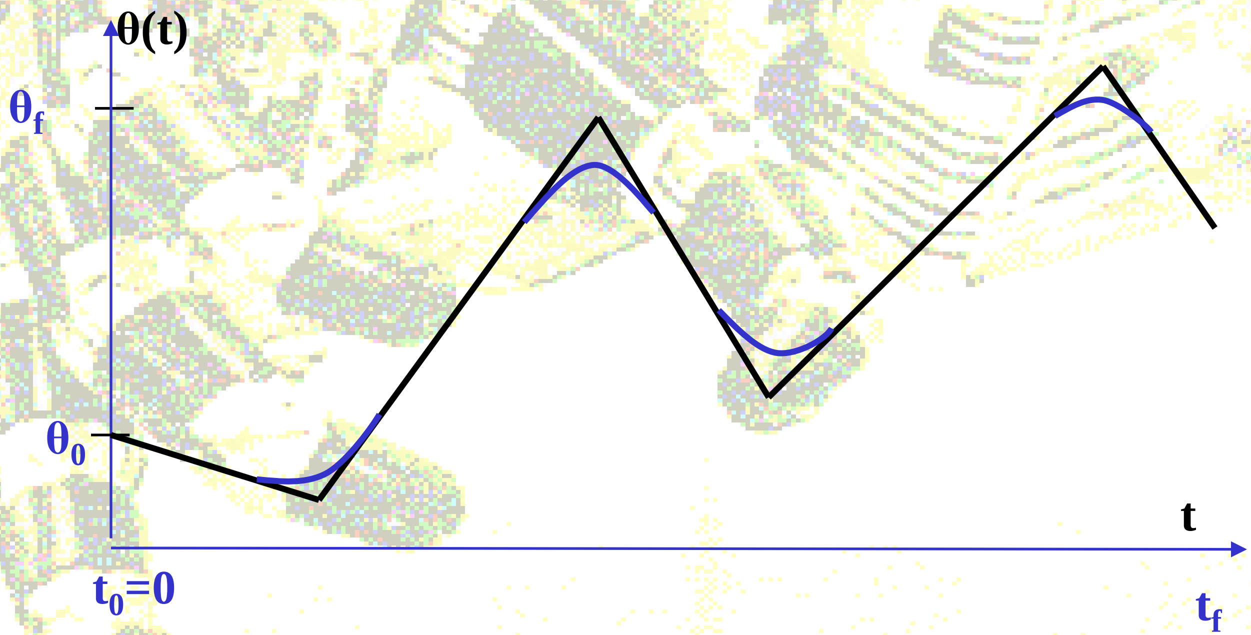
= Equality means linear segment has zero length (Path is composed of two blends with equivalent slope).

> As the acceleration $\ddot{\theta}_b$ increases, the blend region becomes shorter and shorter. In the limit when $\ddot{\theta}_b \longrightarrow \infty$, we are back to the simple linear interpolation case.

Robot Motion Trajectory Generation

Linear Function with Parabolic Blends:

- **Adding via points:**



(Linear Function with Parabolic Blends)

Chapter 7 Exercises:

7.3, 7.7, 7.8, 7.9, 7.11, 7.12

Chapter 8 Exercises:

8.1, 8.2, 8.3, 8.4, 8.8