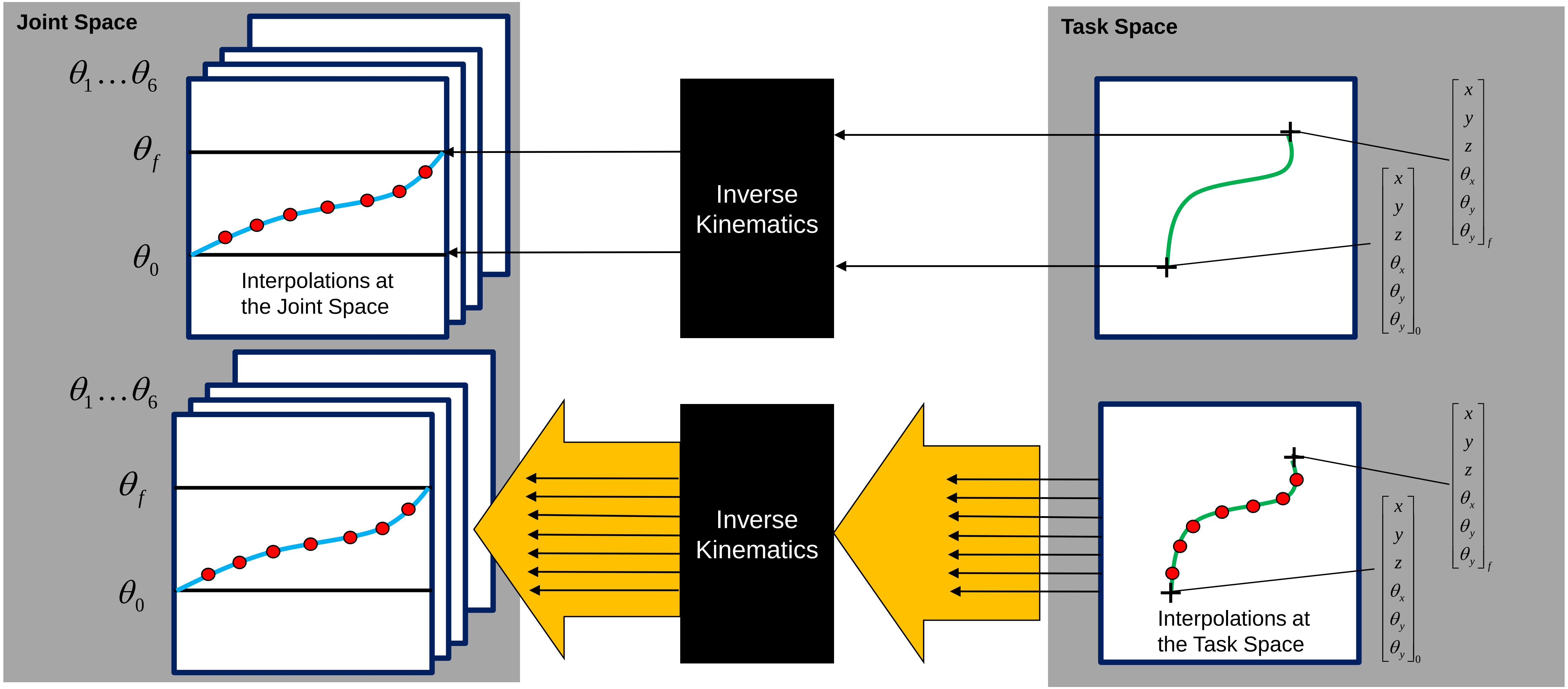
Trajectory Generation (2/2)

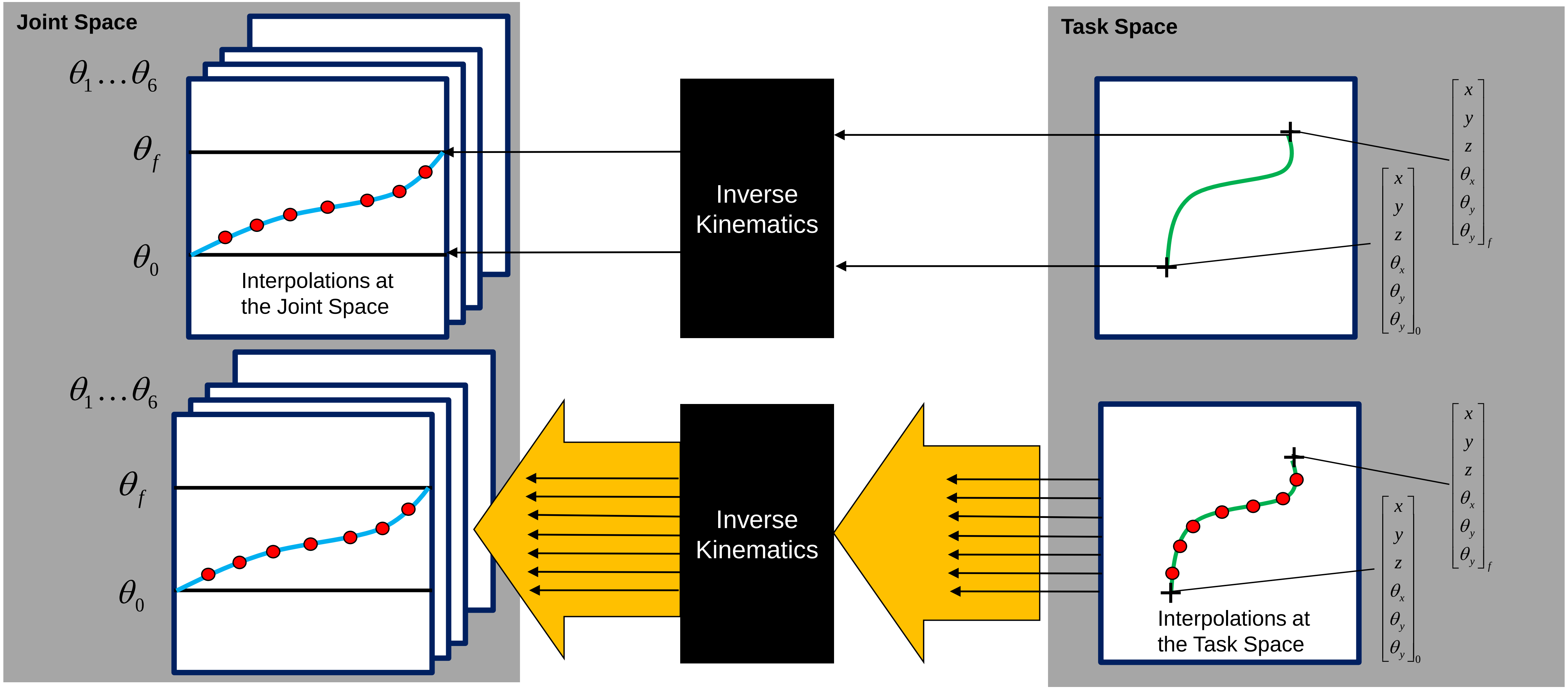
Task Space Versus Joint Space - Interpolations



Task Space Schemes

General Discussion

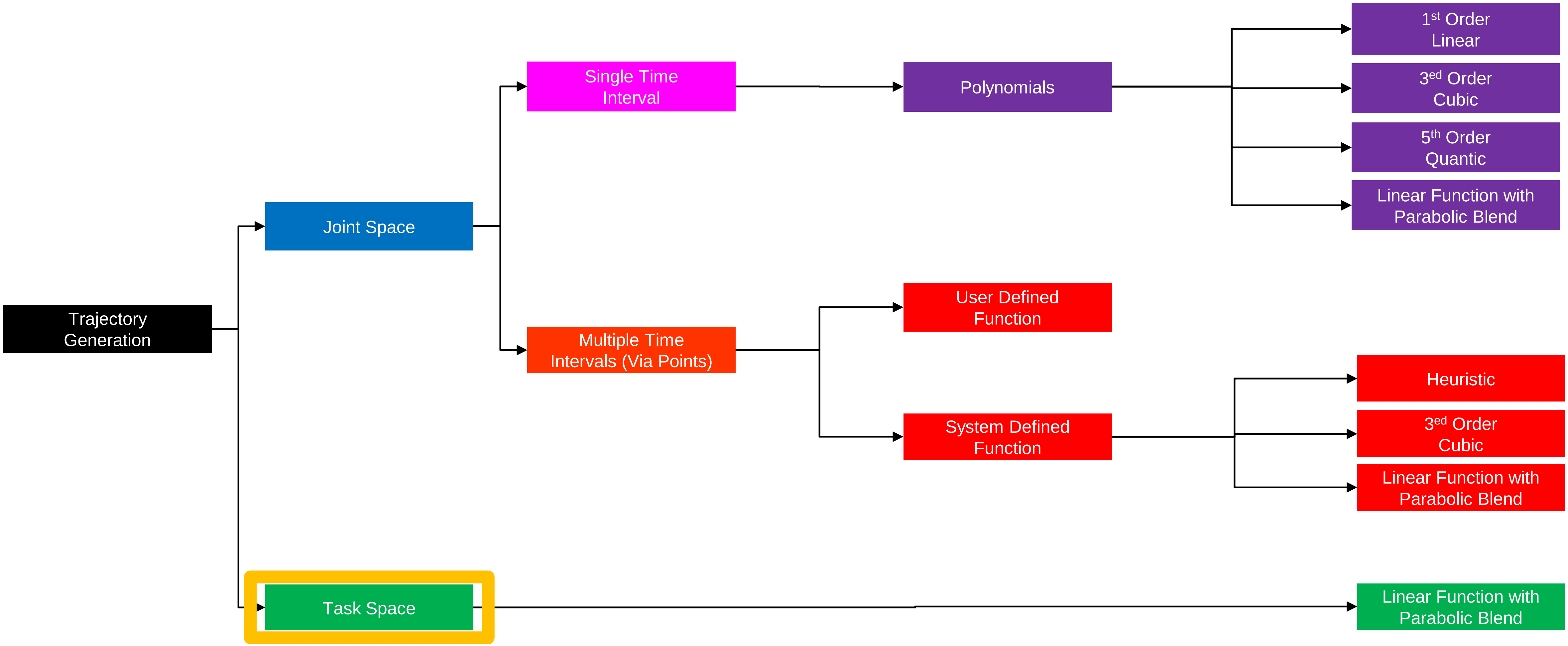
Task Space Versus Joint Space - Interpolations



Join Space Versus Task Space – Comparison

Parameter	Joint Space	Task Space
Interpolation Space intermediate points along the trajectory	Joint Space	Task Space
Tool Trajectory Type / Length	Curved Line / Long	Straight Lines / Short
Invers Kinematics (IK) Usage	Low	High
Computation Expense (IK)	Low (IK for Start/Finish & Via Points)	High (IK for every single point / time steo on the trajectory)
Passing through Via Points	No (Correction by establishing Pseudo Points)	Yes
Via Points Defined in the Task Space	No	Yes
Path Dependency on a Specific Manipulator	Yes	No

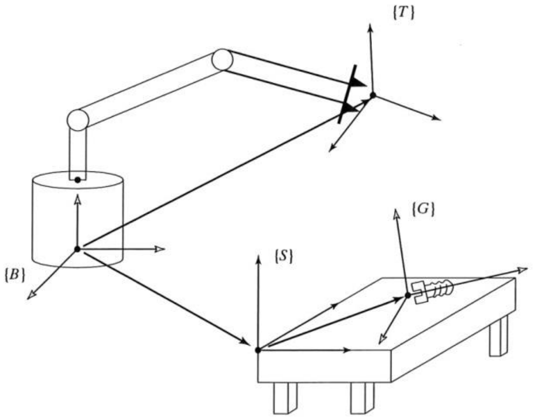
Trajectory Generation – Roadmap Diagram



Task Space Scheme – Problem Definition
Position / Orientation Problem

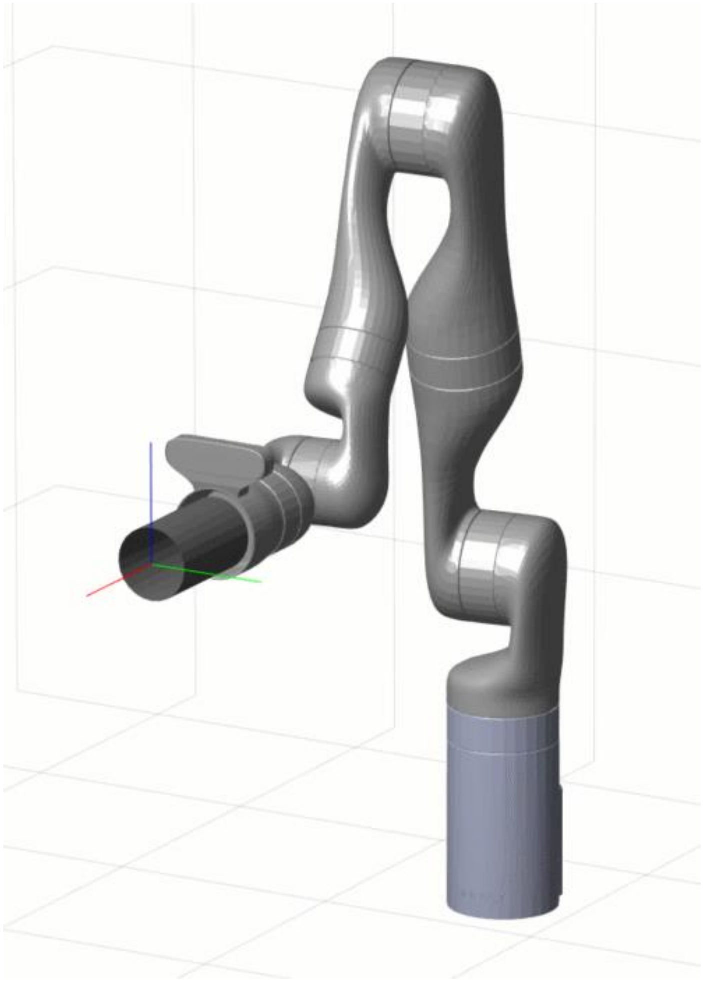
- **General Approach (continue)**
  - Every point along the path is defined by position and orientation of the end effector

$$ {}^{S}_{A}T = \begin{bmatrix} & {}^{S}_{A}R & & {}^{S}P_{AORG} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

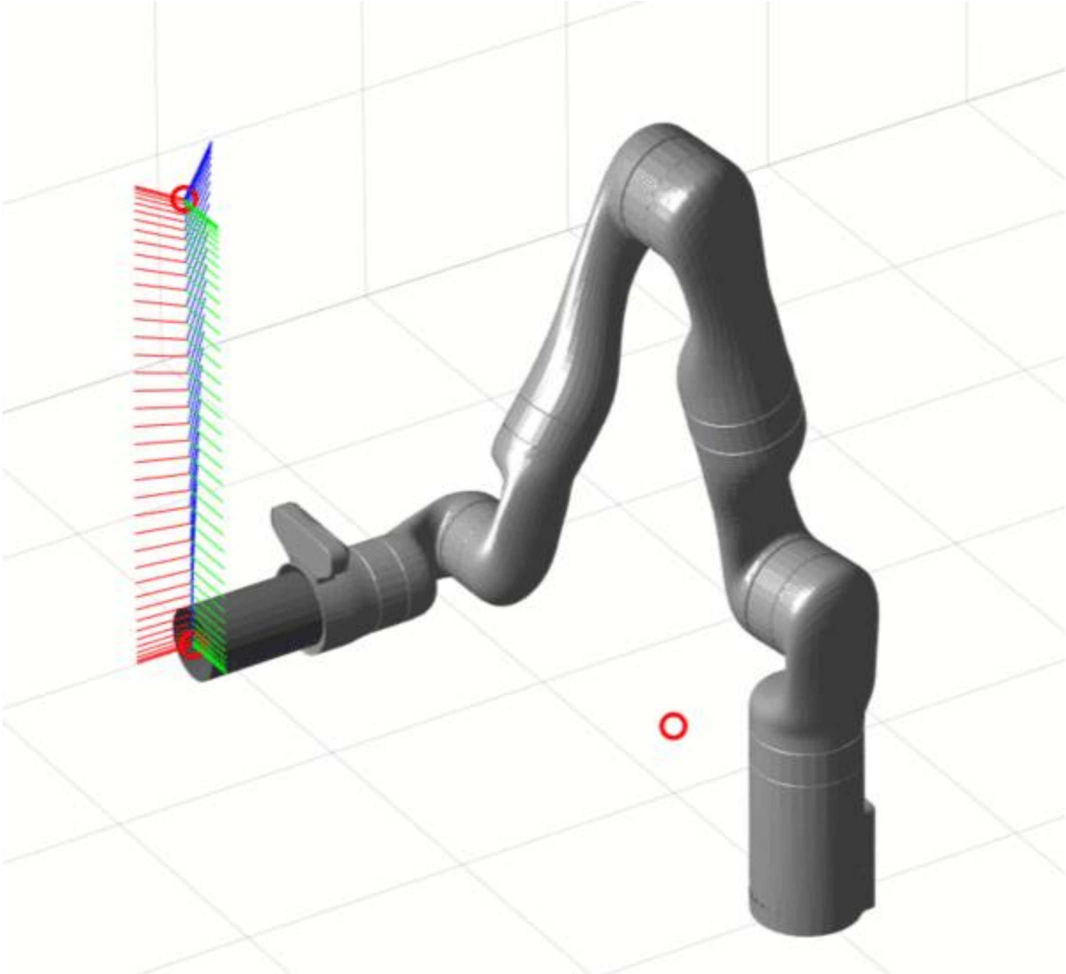


  - **End Effector Position** – Vector – Easy interpolation
  - **End Effector Ordination** – Matrix – Impossible to interpolate (interpolating the individual elements of the matrix violate the requirements that all column of the matrix must be orthogonal)

Task Space Scheme – Problem Definition
Orientation Problem



Task Space Scheme – Problem Definition
Position / Orientation Problem – Trapezoid Velocity



Cartesian Space Schemes – Introduction

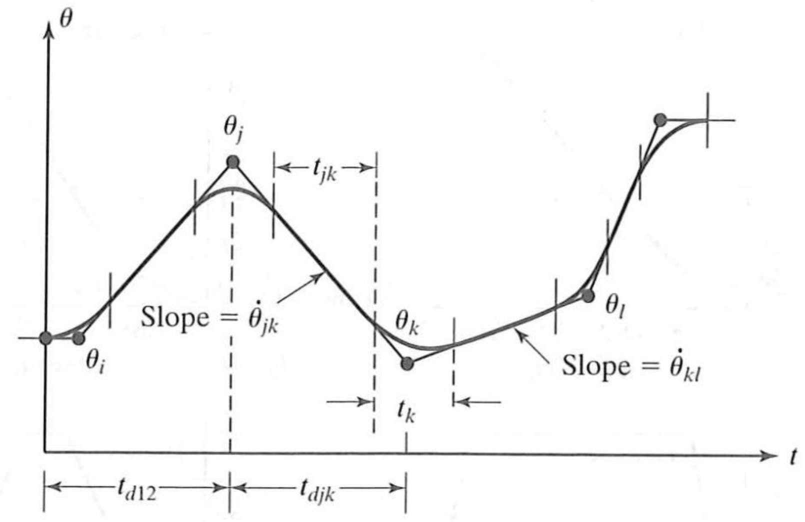
- **Joint Space Schemes**
  - **Advantages –**
    - Path go through all the via and goal points
    - Points can be specified by Cartesian frames.
  - **Disadvantages** -
    - End effector moves along a curved line (not a straight line - shortest distance).
    - Path depends on the particular joint kinematics of the manipulator
-

Cartesian Space Schemes – Introduction

- **Cartesian Space Scheme**
  - **Advantage**
    - Most common path is straight line (shortest). Other shapes can also be used.
  - **Disadvantage**
    - Computationally expansive to execute – At run time the inverse kinematics needs to be solved at path update rate (60-2000 Hz)

Cartesian Space Scheme – Cartesian Straight Line

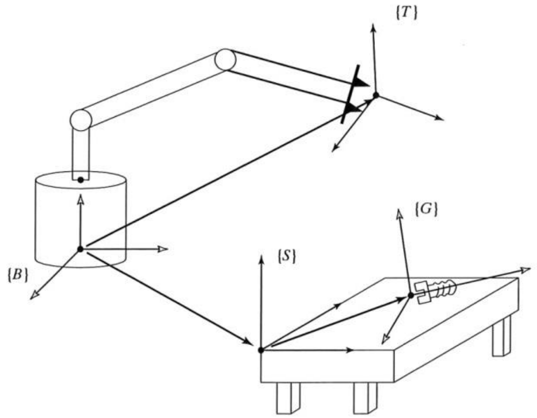
- **General Approach** - Define the path (in the Cartesian space) as
  - straight lines (linear functions)
  - Parabolic lines (blends)



Cartesian Space Scheme – Cartesian Straight Line

- **General Approach (continue)**
  - Every point along the path is defined by position and orientation of the end effector

$$ {}^{S}_{A}T = \begin{bmatrix} & {}^{S}_{A}R & & {}^{S}P_{AORG} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$



  - **End Effector Position** – Vector – Easy interpolation
  - **End Effector Ordination** – Matrix – Impossible to interpolate (interpolating the individual elements of the matrix violate the requirements that all column of the matrix must be orthogonal)

Task Space Scheme – Problem Definition
Orientation Problem - Equivalent Angle – Axis Representation

- Start with the frame coincident with a know frame {A}; then rotate frame {B} about a vector ${}^{A}\hat{K}$ by an angle θ according to the right hand rule.

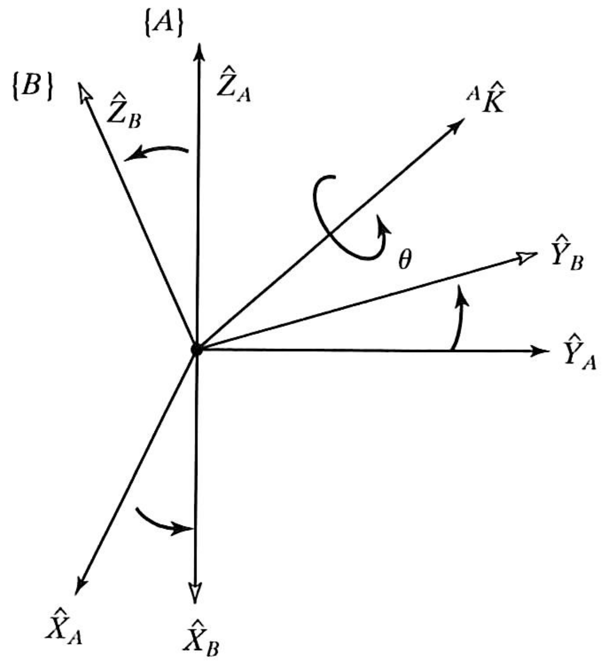
- Equivalent Angle – Axis Representation

$$ {}^{A}_{B}R(\hat{K},\theta) \quad or \quad R_K(\theta) $$

- Vector ${}^{A}\hat{K}$ is called the equivalent axis of a finite rotation.

- The specification of ${}^{A}\hat{K}$ requires two parameters since it length is always 1.

- The angle specify the third parameter



Task Space Scheme – Problem Definition
Orientation Problem - Equivalent Angle – Axis Representation

- **Conversion 1** - Conversion for single angle axis representation to rotation matrix representation

$$R_K(\theta) = \begin{bmatrix} k_x k_x v\theta + c\theta & k_x k_y v\theta - k_z s\theta & k_x k_z v\theta + k_y s\theta \\ k_x k_y v\theta + k_z s\theta & k_y k_y v\theta + c\theta & k_y k_z v\theta - k_x s\theta \\ k_x k_z v\theta - k_y s\theta & k_y k_z v\theta - k_x s\theta & k_z k_z v\theta + c\theta \end{bmatrix}$$

$$c\theta = \cos\theta$$
$$s\theta = \sin\theta$$
$$v\theta = 1 - \cos\theta$$

- **Conversion 2** – Compute ${}^{A}\hat{K}$ and θ given a rotation a matrix

$$R_K(\theta) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\theta = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right)$$

$$\hat{K} = \frac{1}{2\sin\theta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

Task Space Scheme – Problem Definition
Position / Orientation Problem - Equivalent Angle – Axis Representation

- Combining the angle-axis representation of orientation with the 3x1 Cartesian position representation we have a 6x1 representation of Cartesian position and orientation.

- Consider a via point specified relative to a station point frame as ${}^{S}_{A}T$

$$
{}^{S}_{A}T = \begin{bmatrix} & {}^{S}_{A}R & & {}^{S}P_{AORG} \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

  – Frame {A} specifies a via point
    - Position of the end effector given by ${}^{S}P_{AORG}$
    - Orientation of the end effector given by ${}^{S}_{A}R$

Task Space Scheme – Problem Definition
Position / Orientation Problem - Equivalent Angle – Axis Representation

- Convert the rotation matrix into an angle axis representation

$$ {}^{S}_{A}R = ROT({}^{S}\hat{K}_{A}, \theta_{SA}) = {}^{S}K_{A} $$

- Use the symbol χ to represent 6x1 position and orientation

$$ {}^{S}\chi_{A} = \begin{bmatrix} {}^{S}P_{AORG} \\ {}^{S}K_{A} \end{bmatrix} $$

- Where ${}^{S}K_{A}$ is formed by scaling the unite vector ${}^{S}\hat{K}_{A}$ by the amount of rotation θ_{SA}

Task Space Scheme – Cartesian Straight Line

- Process - For a given trajectory we describe a spline function that smoothly vary these six quantities from path point to path point as a function of time.

$$^{S}\chi_{A} = \begin{bmatrix} ^{S}P_{AORG} \\ ^{S}K_{A} \end{bmatrix}$$

- Linear Spline with parabolic bland
  - Path shape between via points will be linear
  - When via points are passed, the linear and angular velocity of the end effector are changed smoothly

Task Space Scheme – Cartesian Straight Line

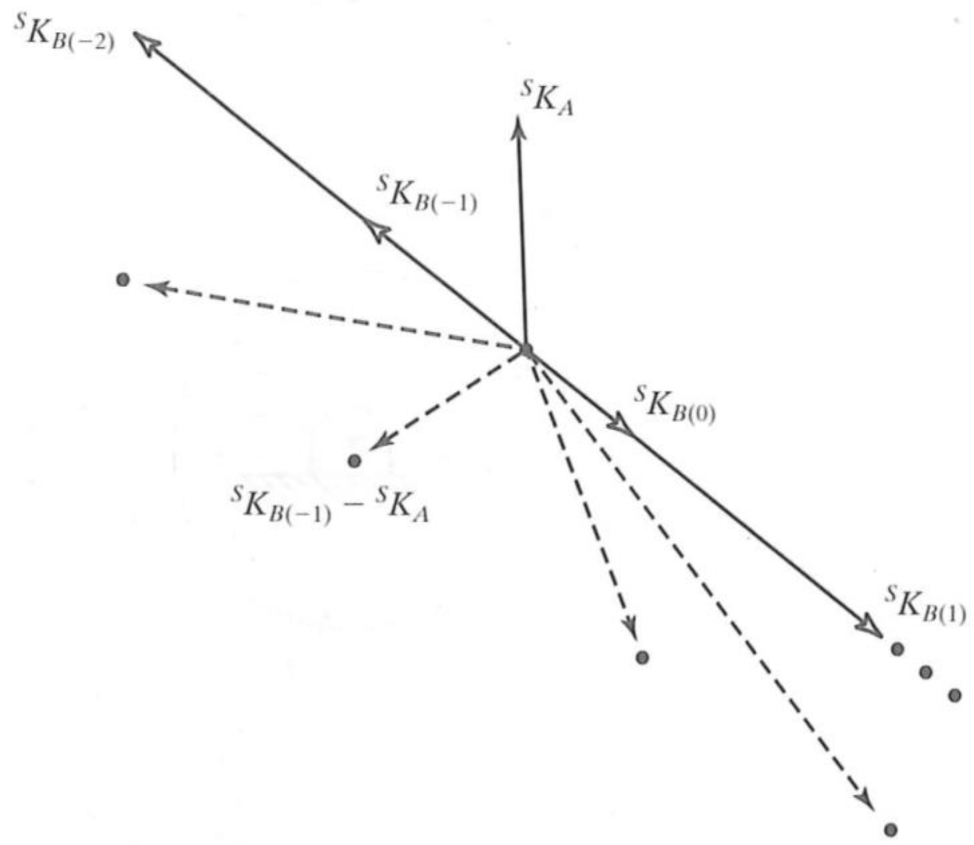
- **Complication –** The angle-axis representation is not unique

$$({}^{S}\hat{K}_{B}, \theta_{SB}) = ({}^{S}\hat{K}_{B}, \theta_{SB} \pm n360)$$

- In going from via point {A} to a via point {B}, the total amount of rotation should be minimized

- Choose ${}^{S}\hat{K}_{B}$ such that

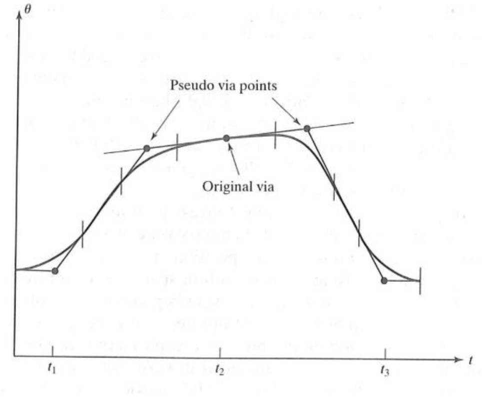
$$\min \left| {}^{S}\hat{K}_{B} - {}^{S}\hat{K}_{A} \right|$$

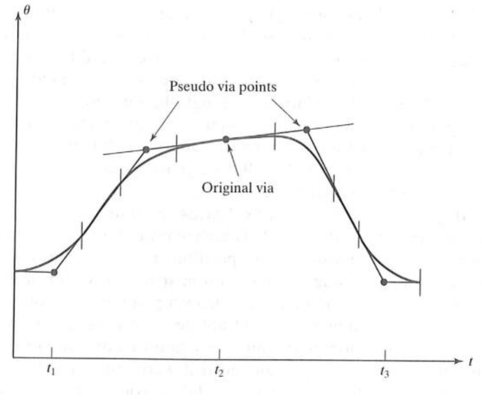


Task Space Scheme – Cartesian Straight Line

- The splines are composed of linear and parabolic blend section

- **Constrain**
  - The transition between the linear segment and the parabolic segment for all the DOF must take place at the same time.





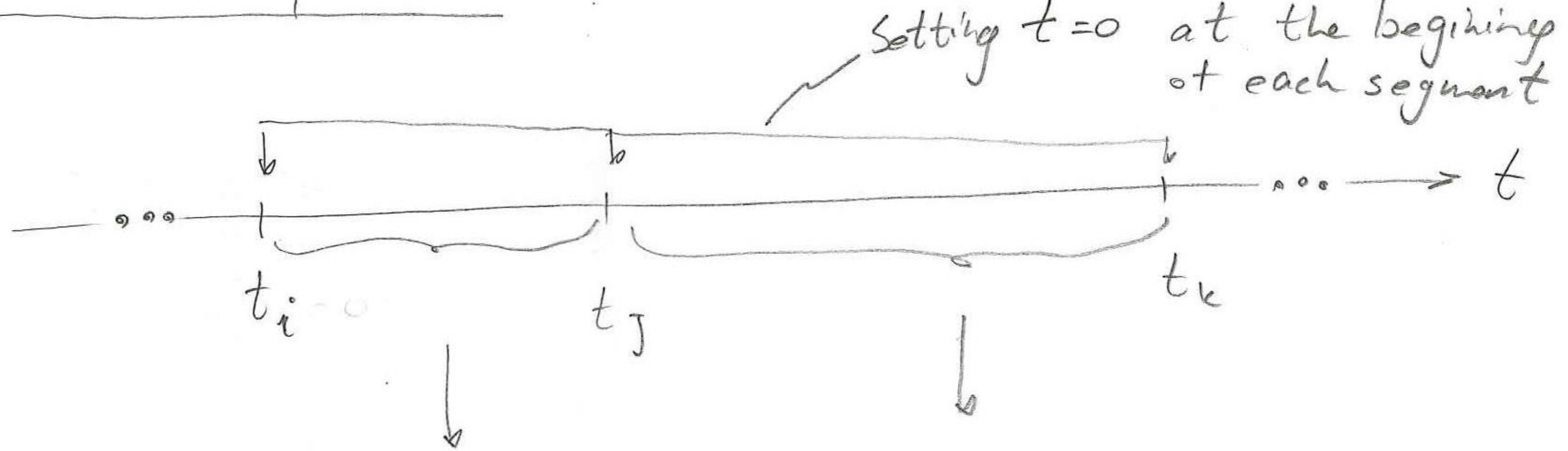
$$ {}^{S}\chi_{A} = \begin{bmatrix} {}^{S}P_{AORG} \\ {}^{S}K_{A} \end{bmatrix} \qquad {}^{S}\chi_{B} = \begin{bmatrix} {}^{S}P_{BORG} \\ {}^{S}K_{B} \end{bmatrix} $$

Path Generation & Run Time – Summary

Joint Space

Task Generation at Run Time – Join Space

Cubic Splines

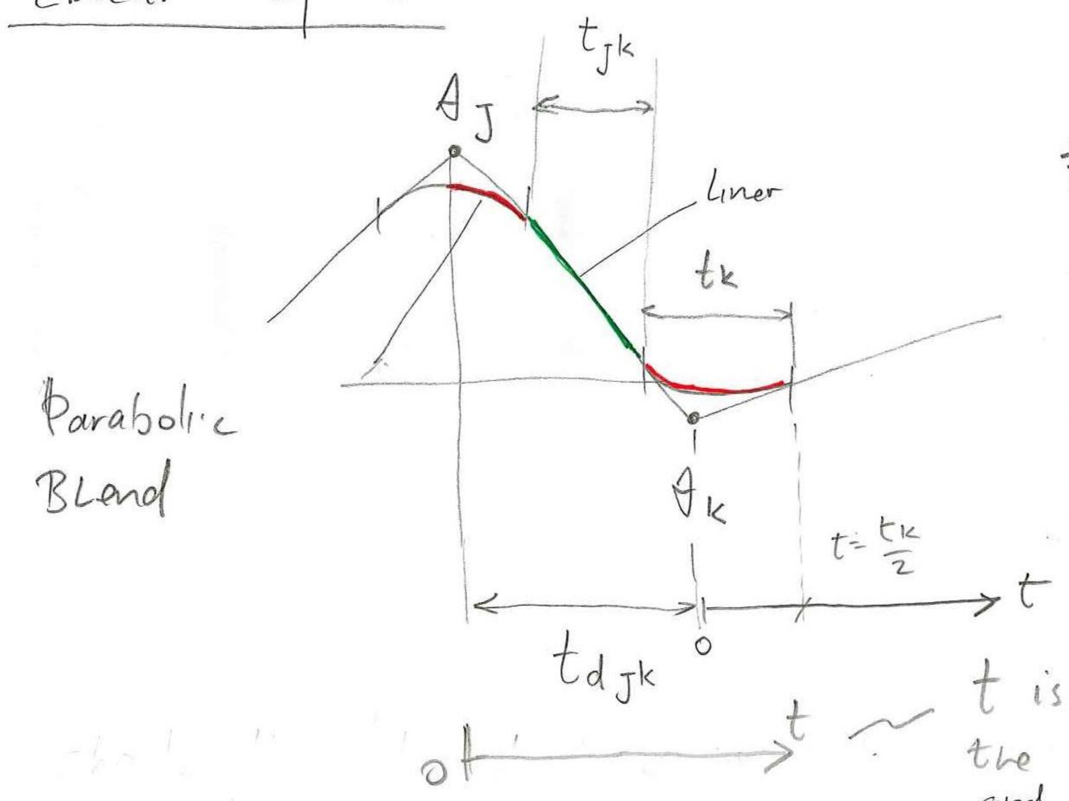


$$\theta(t : t_i \rightarrow t_j) = a_{0ij} + a_{1ij} t + a_{2ij} t^2 + a_{3ij} t^3$$

$$\theta(t : t_j \rightarrow t_k) = a_{0jk} + a_{1jk} t + a_{2jk} t^2 + a_{3jk} t^3$$

Task Generation at Run Time – Join Space

Linear Spline



For each time update

Prabolic Blend

Linear bland

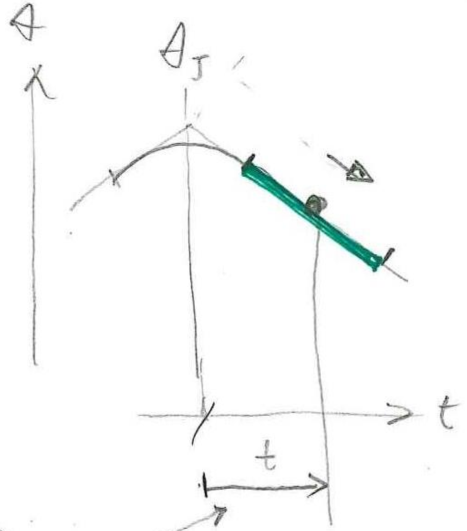
t is measured from the previous via point (J) and reset to tk/2 at the end of the k blending

Task Generation at Run Time – Join Space

Linear

In the linear portion, the trajectory of each joint is calculated as

$$\begin{cases} \theta = \theta_J + \dot{\theta}_{Jk}\, t \\ \dot{\theta} = \dot{\theta}_{Jk} \\ \ddot{\theta} = 0 \end{cases}$$



calculated during the path planing $\dot{\theta}_{Jk} = \dfrac{\theta_k - \theta_J}{t_{dJk}}$

Time since the J via point

Task Generation at Run Time – Join Space

Parabolic Blend

In the blend region the trajectory of each joint is calculated as

$$t_{inb} = t - \left(\frac{1}{2} t_J + t_{Jk}\right)$$

$$\theta = \theta_J + \dot{\theta}_{Jk}(t - t_{inb}) + \frac{1}{2}\ddot{\theta}_k t_{inb}^2$$

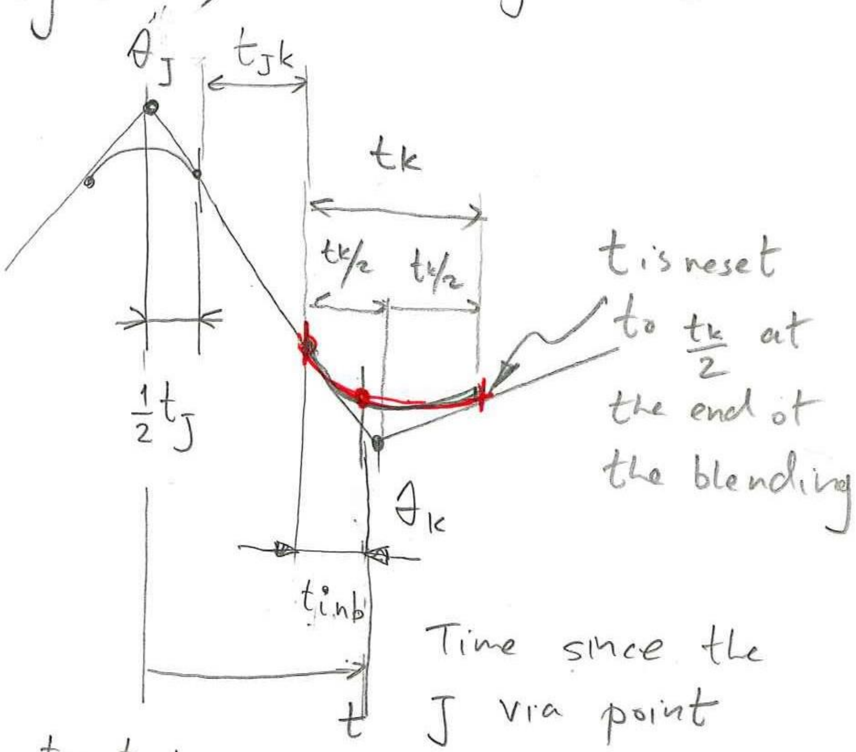
$$\dot{\theta} = \dot{\theta}_{Jk} + \ddot{\theta}_k t_{inb}$$

$$\ddot{\theta} = \ddot{\theta}_k$$

$t - t_{inb} =$

$= t - \left[t - \left(\frac{1}{2} t_J + t_{Jk}\right)\right]$

$= \frac{1}{2} t_J + t_{Jk}$



– See previously eq. for $\dot{\theta}_{Jk}, \ddot{\theta}_k, t_J, t_{Jk}$

Task Generation at Run Time – Join Space

$$\dot{\theta}_{JK} = \frac{\theta_K - \theta_J}{t_{dJK}}$$

user selection

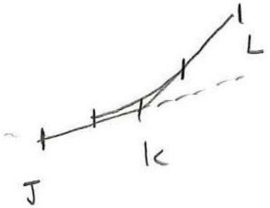
$$\ddot{\theta}_K = SGN\left(\dot{\theta}_{KL} - \dot{\theta}_{JK}\right) \left| \ddot{\theta}_K \right|$$

$$t_K = \frac{\dot{\theta}_{KL} - \dot{\theta}_{JK}}{\ddot{\theta}_K}$$

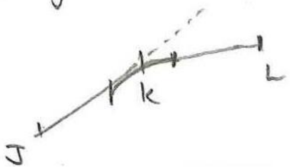
$$t_{JK} = t_{dJK} - \frac{1}{2} t_J - \frac{1}{2} t_K$$

SGN - Determine the sign

(+) Positive Acceleration



(-) Negative Acceleration

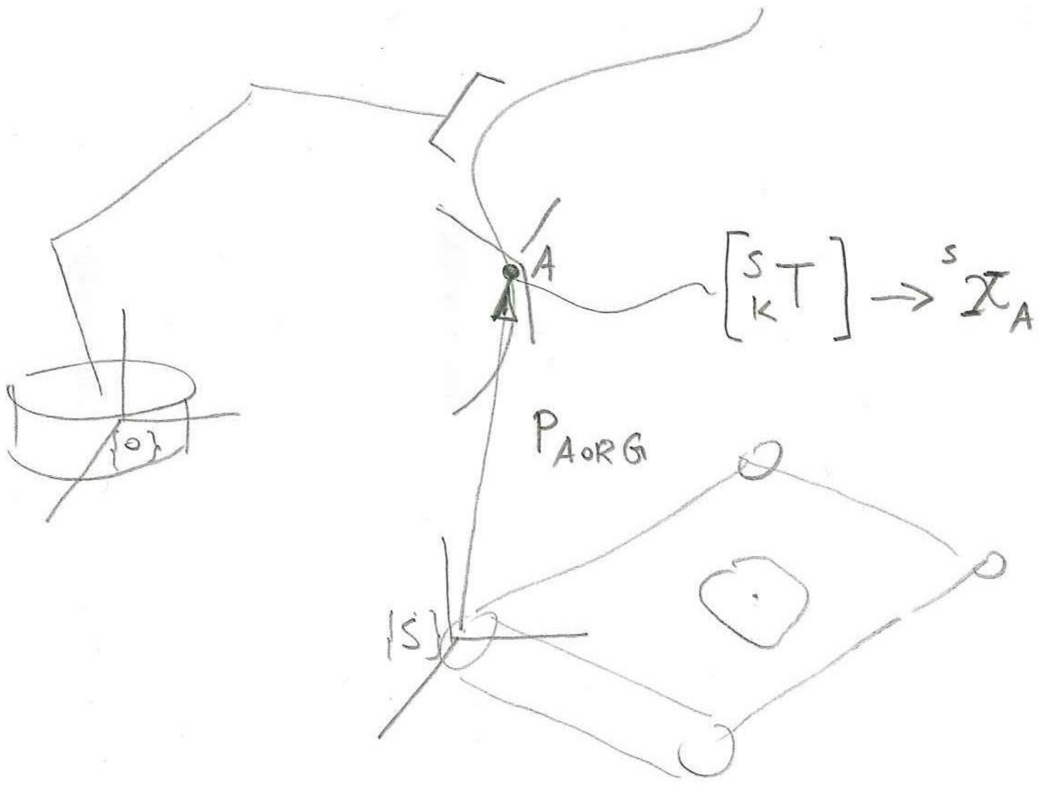


Note: See special case for first/Last segment

Path Generation & Run Time – Summary

Task Space

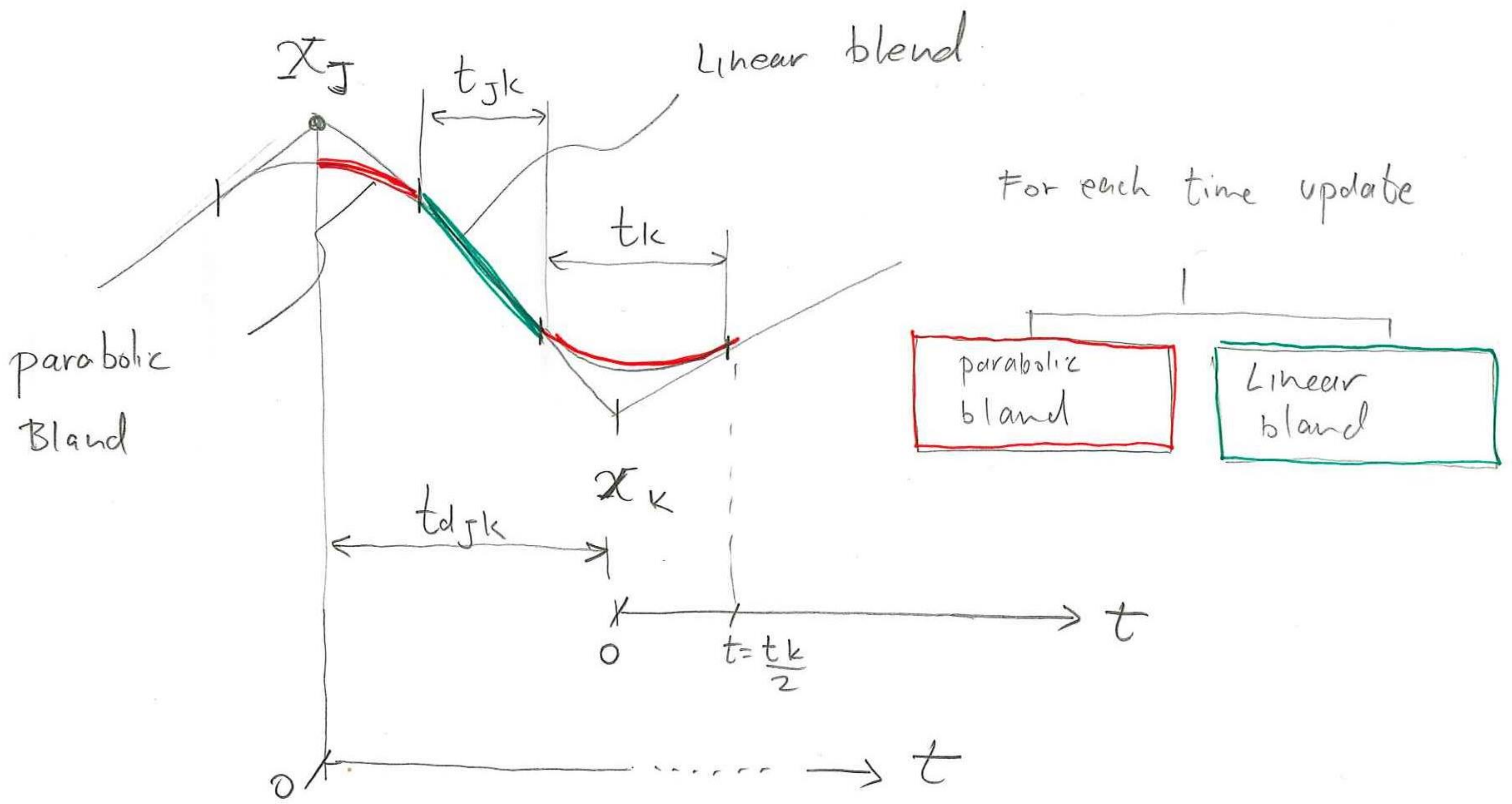
Task Generation at Run Time – Task Space



$$\left[{}^{S}_{K}T\right] \rightarrow {}^{S}\mathcal{X}_A = \begin{bmatrix} P_{AORG} \\ {}^{S}K_A \end{bmatrix} = \overbrace{\begin{bmatrix} P_{AORG\,x} \\ P_{AORG\,y} \\ P_{AOR\,z} \\ {}^{S}K_{Ax} \\ {}^{S}K_{Ay} \\ {}^{S}K_{Ay} \end{bmatrix}}^{6 \times 1}$$

Scaled unit vector by θ of the equ. vector

Task Generation at Run Time – Task Space

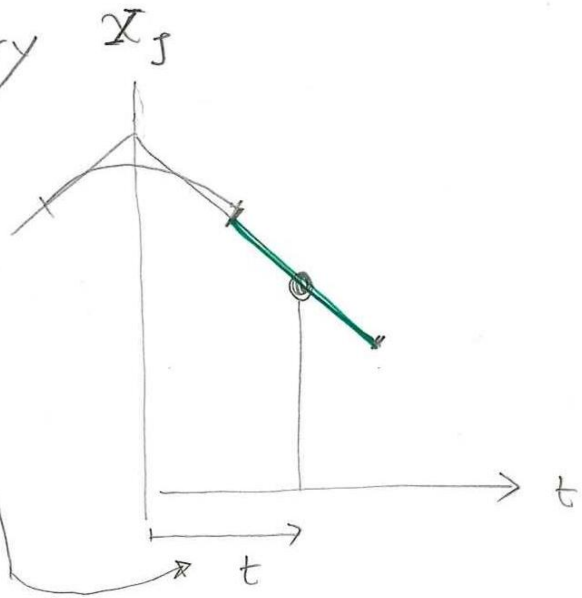


Task Generation at Run Time – Task Space

Linear Blend

In the linear portion, the trajectory of each coordinate of x

$$
\begin{cases}
x = x_J + \dot{x}_{Jk}\, t \\
\dot{x} = x_{Jk} \\
\ddot{x} = 0
\end{cases}
$$



Time since the J via point

calculated during the path planning

$$\dot{P}_{Jk\,ORG}\begin{Bmatrix}x\\y\\z\end{Bmatrix} = \frac{P_{k.ORG}\begin{Bmatrix}x\\y\\z\end{Bmatrix} - P_{J\,ORG}\begin{Bmatrix}x\\y\\z\end{Bmatrix}}{td_{Jk}}$$

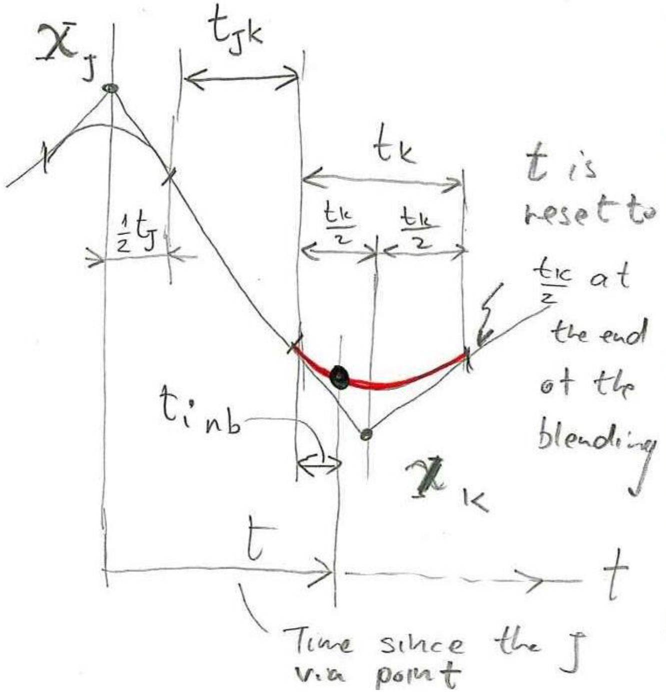
$$\dot{K}_{Jk}\begin{Bmatrix}x\\y\\z\end{Bmatrix} = \frac{K_{k}\begin{Bmatrix}x\\y\\z\end{Bmatrix} - K_{J}\begin{Bmatrix}x\\y\\z\end{Bmatrix}}{td_{Jk}}$$

Task Generation at Run Time – Task Space

Parabolic Blend

In the parabolic blend region the end effector trajectory of each coordinate of the $\bar{x}$

$$
\begin{cases}
t_{inb} = t - \left(\frac{1}{2} t_J + t_{Jk}\right) \\
\bar{x} = x_J + \dot{x}_{Jk}(t - t_{inb}) + \frac{1}{2}\ddot{x}_k t^2 \\
\dot{x} = \dot{x}_{Jk} + \dot{x}_k t_{inb} \\
\ddot{x} = \ddot{x}_k
\end{cases}
$$

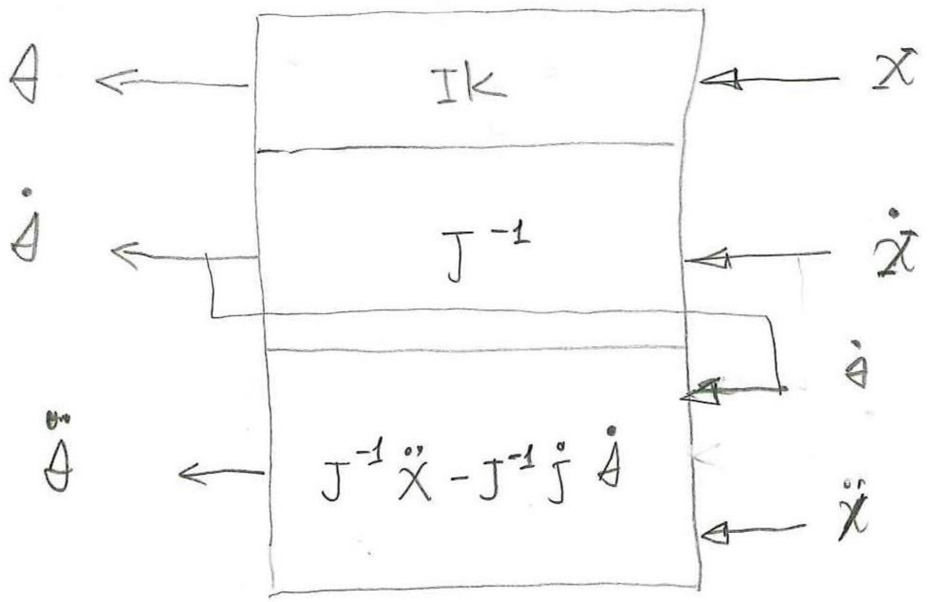


where $\dot{x}_{Jk}$, $\ddot{x}_k$, t_J and t_{Jk} are computed at plan time, just as in the joint space

Task Generation at Run Time – Task/Joint Space Mapping

Mapping $x, \dot{x}, \ddot{x}$ from the task space to the joint space

<u>Option 1</u> – Time derivative are done at the task space



$$\dot{\theta} = J^{-1}\dot{x}$$

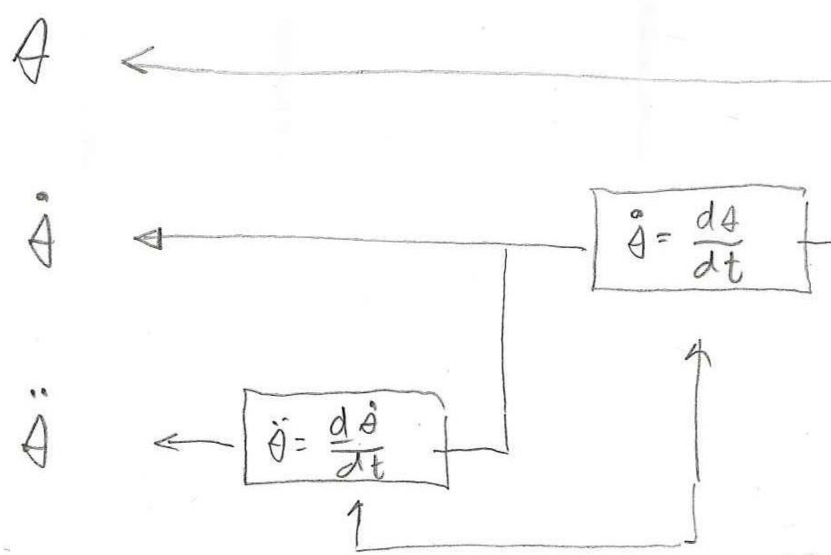
$$\dot{x} = J\dot{\theta}$$

$$\ddot{x} = \dot{J}\dot{\theta} + J\ddot{\theta}$$

$$\ddot{\theta} = J^{-1}\ddot{x} - J^{-1}\dot{J}\dot{\theta}$$

Task Generation at Run Time – Task/Joint Space Mapping

Option 2 – Time derivative of $\dot{\theta}, \ddot{\theta}$ are done at the joint space



$$\dot{\theta} = \frac{\theta(t) - \theta(t - \delta t)}{\delta t}$$

$$\ddot{\theta} = \frac{\dot{\theta}(t) - \dot{\theta}(t - \delta t)}{\delta t}$$

- Note + This differentiation can be done off-line resulting in better quality of $\dot{\theta}, \ddot{\theta}$
- \+ Many control systems do not require a $\ddot{\theta}$ input

Task Space Schemes

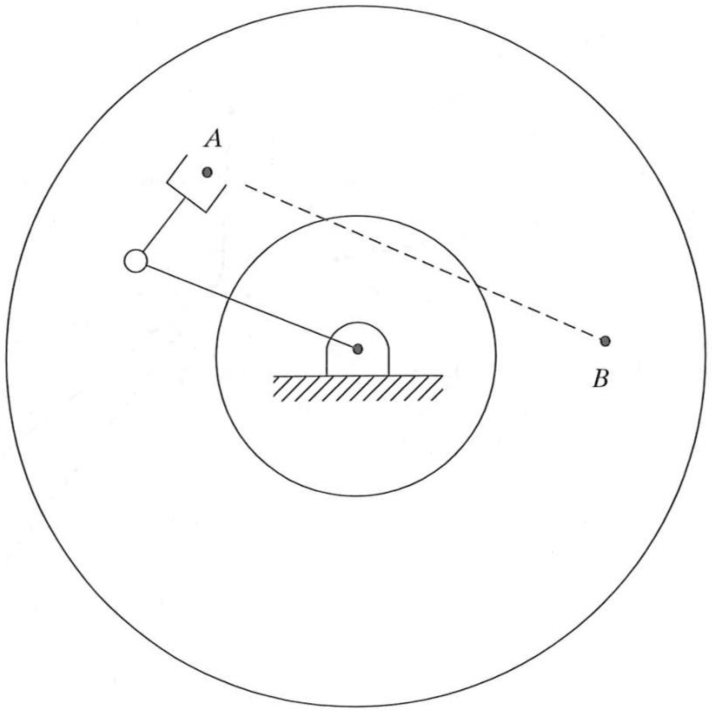
Geometric Problems with Paths in Task Space

Geometric Problems – Cartesian Paths

- **Problem Type 1 – Unreachable Intermediate Points**

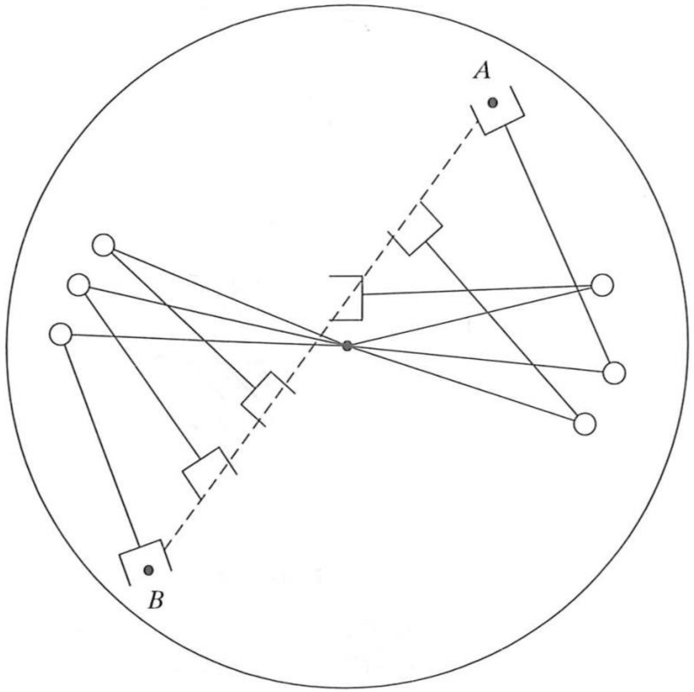
- The initial and the final point are in the reachable workspace however some point along the path may be out of the workspace.

- **Solution**
  – Joint space path – unreachable
  – Cartesian straight Path – reachable



Geometric Problems – Cartesian Paths

- **Problem Type 2 – High Joint Rate Near Singularity.**
- In singularity the velocity of one or more joint approach infinity.
- The velocity of the mechanism are upper bounded, approaching singularity results in the manipulator's deviation form the desired path.
- **Solution**
  - Slow down the velocity such that all the joint velocities will remain in their bounded velocities



Geometric Problems – Cartesian Paths

- **Problem Type 3 – Start and Goal reachable in different solutions**

- Joint limits may restrict the number of solutions that the manipulator may use given a goal point.

- Solution
  - Switch between joint space (default) and Cartesian space trajectories (used only if needed)