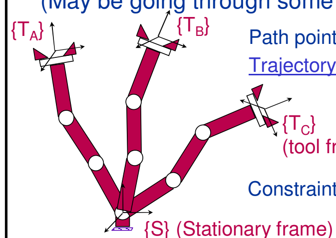
Trajectory Generation

Trajectory Generation

Basic Problem:

Move the manipulator arm from some initial position $\{T_A\}$ to some desired final position $\{T_C\}$.

(May be going through some via point $\{T_B\}$)



Path points : Initial, final and via points

Trajectory : Time history of position, velocity and acceleration for each DOF

Constraints: Spatial, time, smoothness

Solution Spaces :

Joint space

- Easy to go through via points
  (Solve inverse kinematics at all path points and plan)
- No problems with singularities
- Less calculations
- Can not follow straight line

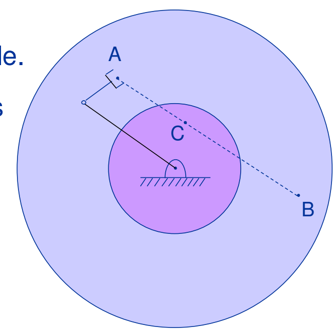
Cartesian space

- We can track a shape
  (for orientation : equivalent axes, Euler angles,…
- More expensive at run time
  (after the path is calculated need joint angles in a lot of points)
- Discontinuity problems

Cartesian planning difficulties :

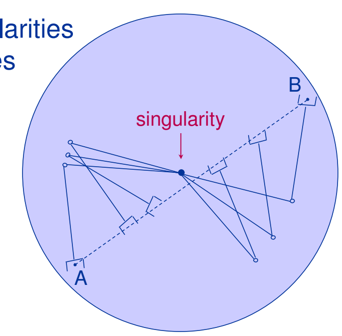
Initial and Goal Points are reachable.

Intermediate points (C) unreachable.



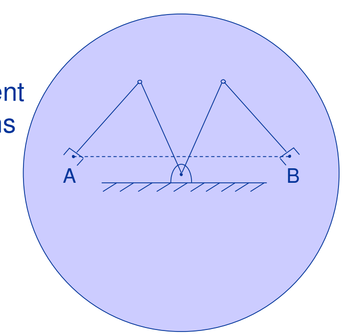
Cartesian planning difficulties :

Approaching singularities some joint velocities go to ∞ causing deviation from the path



Cartesian planning difficulties :

Start point (A) and goal point (B) are reachable in different joint space solutions (The middle points are reachable from below.)



Actual planning in any space:

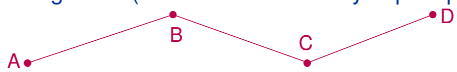
Assume one generic variable u

(can be x, y, z, orientation - α, β, γ)

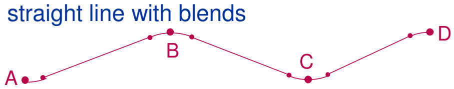
joint variables direction cosines

Candidate curves :

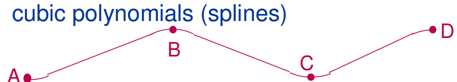
straight line (discontinuous velocity at path points)



straight line with blends

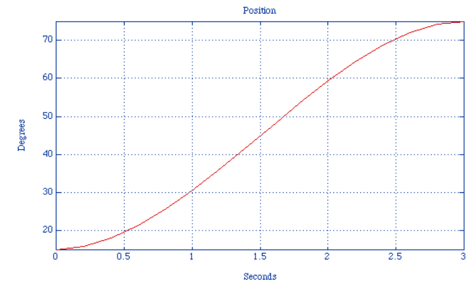


cubic polynomials (splines)



higher order polynomials (quintic,…) or other curves

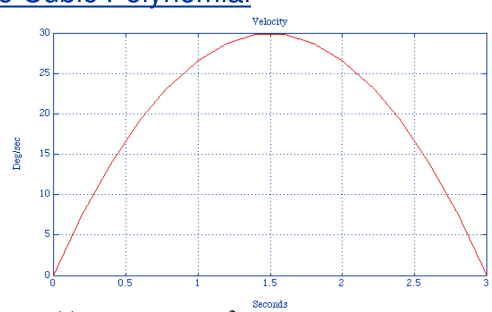
Single Cubic Polynomial



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

Initial Conditions: $u(0)=u_0$; $u(t_f)=u_f$

Single Cubic Polynomial

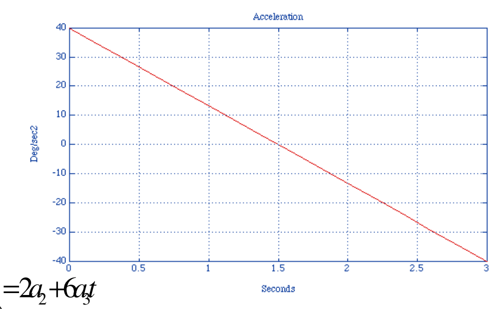


$$\dot{u}(t)=a_1+2a_2t+3a_3t^2$$

Initial Conditions: $\dot{u}(0)=0$; $\dot{u}(t_f)=0$

Starts and ends at rest

Single Cubic Polynomial



$\ddot{u}(t)=2a_2+6a_3t$

$\dddot{u}(t)=6a_3$ (constant)

Solution : $u(t)=u_0+\frac{3}{t_f^2}(u_f-u_0)t^2+\left(-\frac{2}{t_f^3}\right)(u_f-u_0)t^3$

Cubic Polynomials with via points

- If we come to rest at each point
  use formula from previous slide
- For continuous motion (no stops)
  need velocities at intermediate points:

$\dot{u}(0)=\dot{u}_0$

$\dot{u}(t_f)=\dot{u}_f$ Initial Conditions

Solution :

$$a_0=u_0$$

$$a_1=\dot{u}_0$$

$$a_2=\frac{3}{t_f^2}(u_f-u_0)-\frac{2}{t_f}\dot{u}_0-\frac{1}{t_f}\dot{u}_f$$

$$a_3=-\frac{2}{t_f^3}(u_f-u_0)+\frac{1}{t_f^2}(\dot{u}_f+\dot{u}_0)$$

How to find $\dot{u}_0$, $\dot{u}_f$,…(velocities at via points)

- if we know Cartesian linear and angular velocities

$$\rightarrow \text{use } J^{-1}: \ \dot{u}=J^{-1}\begin{pmatrix}\mathbf{v}\\ \omega\end{pmatrix}$$

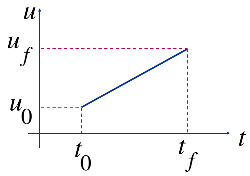
- the system chooses reasonable velocities using heuristics (average of 2 sides etc.)
- the system chooses them for continuous

velocity $\dot{u}_1(t_f)=\dot{u}_2(0)$ and

acceleration $\ddot{u}_1(t_f)=\ddot{u}_2(0)$

Linear interpolation:

Straight line



$$u(t)=a_0+a_1t$$

2 conditions : $u(t_0)=u_0$, $u(t_f)=u_f$

Discontinuous velocity - can not be controlled

Linear interpolation:

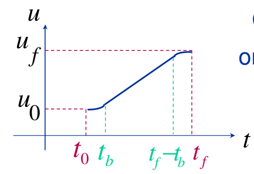
Parabolic blend

$$u(t)=\frac{1}{2}at^2$$

at blend regions

Linear velocity $\dot{u}(t)=at$

Constant acceleration $\ddot{u}(t)=a$



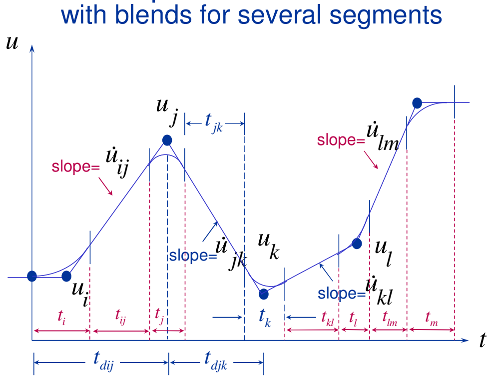
or $u(t)=\frac{1}{2}\ddot{u}t^2$ at blend regions

From continuous velocity:

$$t_b=\frac{t}{2}-\frac{\sqrt{\ddot{u}t^2-4\ddot{u}(u_f-u_0)}}{2\ddot{u}}$$

where $t=t_f-t_0$ desired duration of motion

Linear Interpolation
with blends for several segments



Given:

- positions u_i, u_j, u_k, u_l, u_m
- desired time durations $t_{dij}, t_{djk}, t_{dkl}, t_{dlm}$
- the magnitudes of the accelerations: $|\ddot{u}_i|, |\ddot{u}_j|, |\ddot{u}_k|, |\ddot{u}_l|$

Compute:

- blends times t_i, t_j, t_k, t_l, t_m
- straight segment times $t_{ij}, t_{jk}, t_{kl}, t_{lm}$
- slopes (velocities) $\dot{u}_{ij}, \dot{u}_{jk}, \dot{u}_{kl}, \dot{u}_{lm}$
- signed accelerations

Formulas (7.24), (7.26) and (7.28)

System usually calculates or uses default values for accelerations. The system can also calculate desired time durations based on default velocities.

First segment

$$\ddot{u}_1 = sign(u_2-u_1)|\ddot{u}_1|$$

$$t_1 = t_{d12} - \sqrt{t_{d12}^2 - \frac{2(u_2-u_1)}{\ddot{u}_1}}$$

$$\dot{u}_{12} = \frac{u_2-u_1}{t_{d12}-\frac{1}{2}t_1}$$

$$t_{12} = t_{d12} - t_1 - \frac{1}{2}t_2$$

Inside segments

$$\dot{u}_{jk} = \frac{u_k-u_j}{t_{djk}}$$

$$\ddot{u}_k = sign(\dot{u}_{kl}-\dot{u}_{jk})|\ddot{u}_k|$$

$$t_k = \frac{\dot{u}_{kl}-\dot{u}_{jk}}{\ddot{u}_k}$$

$$t_{jk} = t_{djk} - \frac{1}{2}t_j - \frac{1}{2}t_k$$

Last segment

$$\ddot{u}_n = sign(u_{n-1} - u_n)|\ddot{u}_n|$$

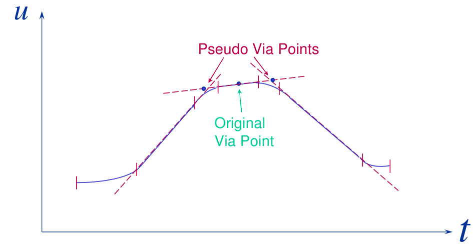
$$t_n = t_{d(n-1)n} - \sqrt{t_{d(n-1)n}^2 - \frac{2(u_n - u_{n-1})}{\ddot{u}_n}}$$

$$\dot{u}_{(n-1)n} = \frac{u_n - u_{n-1}}{t_{d(n-1)n} - \frac{1}{2}t_n}$$

$$t_{(n-1)n} = t_{d(n-1)n} - t_n - \frac{1}{2}t_{n-1}$$

To go through the actual via points:

- Introduce "Pseudo Via Points"



- Use sufficiently high acceleration
- If we want to stop there, simply repeat the via point

Higher Order Polynomials

- For example if given:

6 conditions:
- position (initial u_0, final u_f)
- velocity $(\dot{u}_0, \dot{u}_f)$
- acceleration $(\ddot{u}_0, \ddot{u}_f)$

Use quintic: $u(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$

and find a_i (i=0 to 5) (formulas (7.18) in the book)

Use different functions (exponential, trigonometric,…)

Run Time Path Generation

- trajectory in terms of $\Theta, \dot{\Theta}, \ddot{\Theta}$ fed to the control system
- Path generator computes at path update rate
- In joint space directly:
  - cubic splines -- change set of coefficients at the end of each segment
  - linear with parabolic blends -- check on each update if you are in linear or blend portion and use appropriate formulas for **u**
- In Cartesian space:
  - calculate Cartesian position and orientation at each update point using same formulas
  - convert into joint space using inverse Jacobian and derivatives

    or

    find equivalent frame representation and use inverse kinematics function to find $\Theta, \dot{\Theta}, \ddot{\Theta}$

Trajectory Planning with Obstacles

- Path planning for the whole manipulator
  - Local vs. Global Motion Planning
    - Gross motion planning for relatively uncluttered environments
    - Fine motion planning for the end-effector frame
  - Configuration space (C-space) approach
- Planning for a point robot
  - graph representation of the free space, quadtree
  - Artificial Potential Field method
- Multiple robots, moving robots and/or obstacles