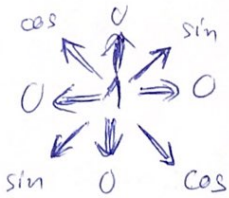
ARKAD

L1) Introduction

Ng Huu Duc

Articulated Robot	Gantry	Scara	Delta	Hexapod
Large workspace			Small workspace	Small workspace
Generally 6 rotational axis's 6 dof	3 linear axis	1 translational 2-3 parallel rotational axes	3 parallel kinematics chains	6 parallel kinematics chains
Market share 68 %	19 %	11 %	1 %	1 %

* Elementary Rotation matrices



; $-1 \leftarrow 1$

L2) Position & Orientation

- Euler Angles: 12 different sets of angles $R_{zy'z''}(\Phi) = R_z(\varphi)\, R_{y'}(\theta)\, R_{z''}(\psi)$
- Roll-Pitch-Yaw: Fixed frame (XYZ): $R(\varphi, \theta, \psi) = R_z(\varphi)\, R_y(\theta)\, R_x(\psi)$
- Gimbal lock: loss of 1 dof in a 3-dimensional, 3 gimbal mechanism when 2/3 gimbals are driven into a parallel configuration

* Exists no matter what rotation sequence you choose

Intuition (unsure) Angle & Axis Representation, Unit Quaternion are ways to avoid dealing with, or deal better with Gimbal lock

Axis & Angle: solves Gimbal lock problem

$R_z(\gamma) R_y(\beta) R_z(\theta) R_y(-\beta) R_z(-\gamma)$ current frame

However, does not guarantee an unique solution when Inverse, has problem ~~when~~ $\sin\theta = 0$

Unit Quaternion: CAN describe any rotation without ambiguities!

L3, Direct Kinematics

- Homogeneous coordinate representation, transformation matrix
- $\vec{e}_{x,i_0} = \vec{e}_{z,i-1_0} \times \vec{e}_{z,i_0}$

L4, Inverse Kinematics

- Kinematic Redundancy: No ⇒ Functional ⇒ Hyper
- 3 Ways to solve:

Analytical	Numerical (Just 1 solution, depending on initial guess)	Algebraic
If analytical solution exists, ALWAYS use it	If closed-form solution does not exist, too hard to find analytically	Solve polynomials equations
non redundant	if redundant	

$$err = r_d - f(q_e)$$
$$q_{k+1} = q_k + \frac{1}{J_A(q)} \mathrm{adj}(J_A(q)) err_k$$

Differential Kinematics

joint velocity $\Rightarrow$ end-effector velocity

Relation $\dot{q} \Longrightarrow$ end-effector $\left\{ \begin{array}{l} \text{linear} \\ \text{angular} \end{array} \right.$ velocity

$$q = [q_1 \dots q_n]^T \Rightarrow \dot{p}_E = [\dot{p}_x \; \dot{p}_y \; \dot{p}_z]^T$$

$$\omega_E = [\omega_x \; \omega_y \; \omega_z]^T$$

$$v_E = J_G(q)\,\dot{q} \Leftarrow v_E = \begin{bmatrix} \dot{p}_E \\ \omega_E \end{bmatrix}$$

$\rightarrow$ $\boxed{\text{Geometric Jacobian}}$ $J_G(q) = \begin{bmatrix} J_{GP}(q) \\ J_{GO}(q) \end{bmatrix}$ $\begin{array}{l} \rightarrow \text{Geometric Position} \\ \rightarrow \text{———— Orientation} \end{array}$

Linear velocity composition rule

$${}^{0}\dot{r}_{P,{}^{0}} = {}^{0}\dot{r}_{i_0,{}^{0}} + {}^{0}R_i\,{}^{i}\dot{r}_{P,i_0} + {}^{0}\omega_i \times {}^{0}r_{P,i_0}$$

Different Approach

$\Rightarrow$ $${}^{0}\dot{r}_{i_0,{}^{0}} = {}^{0}\dot{r}_{i-1_0,{}^{0}} + {}^{0}\dot{r}_{i_0,i-1_0} + {}^{0}\omega_{i-1} \times {}^{0}r_{i_0,i-1_0}$$

Angular Velocity composition rule:

$${}^{0}\omega_i = {}^{0}\omega_{i-1} + {}^{0}R_{i-1}\,{}^{i-1}\omega_{i,i-1} = \boxed{{}^{0}\omega_{i-1} + {}^{0}\omega_{i,i-1}}$$

Strictly follow D-H conventions: then: $\delta \; d \; l \; \alpha$ (θ)

Prismatic Joints:

$${}^{0}\omega_i = {}^{0}\omega_{i-1}$$

$${}^{0}\dot{r}_{i_0,{}^{0}} = {}^{0}\dot{r}_{i-1_0,{}^{0}} + \underset{}{\overset{q_i}{\dot{d}_i}}\,{}^{0}e_{z,i-1_0} + {}^{0}\omega_{i-1} \times {}^{0}r_{i_0,i-1_0}$$

Revolute Joints: $\rightarrow \dot{q}_i$

$${}^{0}\omega_i = {}^{0}\omega_{i-1} + \dot{\delta}_i\,{}^{0}e_{z,i-1_0}$$

$${}^{0}\dot{r}_{i_0,{}^{0}} = {}^{0}\dot{r}_{i-1_0,{}^{0}} + {}^{0}\omega_i \times {}^{0}r_{i_0,i-1_0}$$

short-term and long-term accommodation to you.

$$\begin{bmatrix} {}_jG_{P_i} \\ {}_jG_{O_i} \end{bmatrix} =$$

prismatic $\begin{bmatrix} {}^0e_{z,i-1_0} \\ \vec{0} \end{bmatrix}$

$= {}^0r_{E,i-1_0}$

revolute $\begin{bmatrix} {}^0e_{z,i-1_0} \times \left({}^0r_{E,0_0} - {}^0r_{i-1_0,0_0}\right) \\ {}^0e_{z,i-1_0} \end{bmatrix}$

end effector

Kinematics Singularities

occur if Jacobian is rank - deficient

2 Types
- Boundary singularities : out-stretched or retracted
- Internal singularities : caused by alignment of 2, more axis

When singularity presence: ⇒ may yield infinite solutions

close to a singularities:
small velocities in operational space may
CAUSE large velocity in joint space

Singularities decoupling <u>LOS</u> ☆

L5, Differential Kinematics

- Kinematic Singularities : occur, if the Jacobian is rank-deficient

Boundary Singularities	Internal Singularities
Manipulator is out-stretched/retracted	alignment of (≥ 2) axis of motion
$\Rightarrow$ not drive robot to the edge	

Singularity $\Rightarrow$ Kinematics solutions:
- Mobility of robotic structure is reduced
- Inverse Kin $\Rightarrow$ yields infinite solutions
- Small vel in op space $\Rightarrow$ Large vel joint space

Kinematic Redundancy & Differential Mapping

DOF: n

Op space variables: m

No. op space variables for a task: r

$\Rightarrow r < n \Rightarrow$ might be redundant with $n-r$ redundant DOFs

If: $v_E = J_g(q)\,\dot{q}^*$

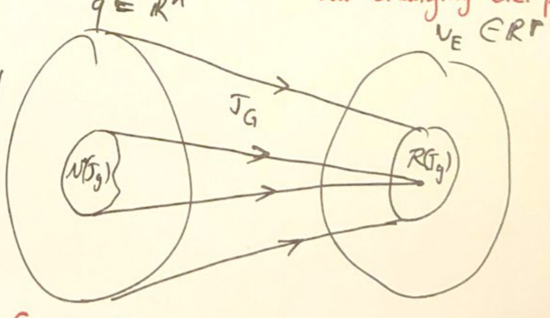
$\mathcal{R}(P) \equiv \mathcal{N}(J_g)$

$\dot{q} = \dot{q}^* + P\dot{q}_0$

$\Rightarrow J_G\dot{q} = J_G\dot{q}^* + J_G P\dot{q}_0 = J_G\dot{q}^* = v_E$

$\Rightarrow J_G P \dot{q}_0 = 0$

$\Rightarrow \dot{q}_0$ generates internal motions without changing e.e. pose



+) range space of J: $\mathcal{R}(J_g)$ is subspace of e.e. space $\mathbb{R}^n$

+) null space of J: $\mathcal{N}(J_g)$ is subspace of joint vel $\mathbb{R}^n$

+) if J full rank, $\dim(\mathcal{R}(J_g)) = m = r$

$\dim(\mathcal{N}(J_g)) = n - r$

+) if J NOT full rank, $\dim(\mathcal{R}(J_g)) + \dim(\mathcal{N}(J_g)) = n$

L6, Inverse Differential Kinematics & Statics

inverse dynamic problem

predetermined path & external forces → velocities & accelerations → inertia forces & torques → generalized actuator forces

direct dynamic problem

generalized actuator forces → inertia forces & torques → equations of motion → resulting robot motion

$$\underbrace{M(q)}_{\text{Mass matrix}}\ddot{q} + \underbrace{G(q,\dot{q})}_{\text{Gyroscopic matrix}}\dot{q} + g(q) = \tau - \underbrace{F_v\dot{q}}_{\text{viscous friction in the joints}} - \underbrace{F_s\,\mathrm{sgn}(\dot{q})}_{\text{Coulomb dry friction in the joint}} - \underbrace{J^T(q)\,h_e}_{\text{e.e generalized forces}}$$

$$v_E = J_G(q)\dot{q} \Rightarrow \dot{q} = J_G^{-1}(q)\cdot v_E \quad ?$$

+) If: J_G is square & full rank $\Rightarrow \exists J_G^{-1} \Rightarrow$ Euler integration method

$$q(t_{k+1}) = q(t_k) + \dot{q}(t_k)\Delta t$$

+) If J_G is NOT $\Rightarrow$ kinematic redundancy $\Rightarrow$ linear optimization problem

$$\min\left(g(\dot{q})\right) = \min\left(\tfrac{1}{2}\dot{q}^T W \dot{q}\right)$$

Use Lagrange method: . . . $\Rightarrow$ Quadratic Programming

$$J^{+} = J^T\left(JJ^T\right)^{-1}$$

$$\dot{q} = J^{+}v \qquad \left(\dot{q} = W^{-1}J^T\left(JW^{-1}J^T\right)^{-1}v\right) \text{ with } W = I$$

L6,

Exploited redundancy ⇒
- avoid singularities
- Joint variables close to center of ranges
- avoid collision

⊗ close to singularities, the determinant yields very small values
⇒ high joint velocities

+) Orientation error: depends on the chosen orientation representation
- Euler Angles:
  - only Good for spherical wrist
  - NOT real-time processable
- Angle & Axis:
  - unique relationship in range $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$
  - no representation singularities
  - better computational performance
- Unit quarternions: even better ↑

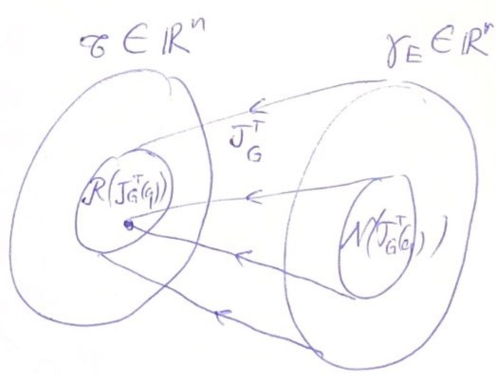
Statics

1, determine $\vec{F}, \vec{M}$ applied on joints
2, generate an equilibrium state with e.e. $\vec{F}$ & $\vec{M}$

$$dW_\tau = \tau^T dq$$
$$dW_\gamma = f_E^T dp_E + \mu_E^T \omega_E dt$$
$$= \gamma_E^T \cdot J_G(q) \cdot dq$$

For static equilibrium: $dW_\tau - dW_\gamma = 0$

$$\Leftrightarrow \tau^T dq = \gamma_E^T J_G(q) dq$$
$$\Leftrightarrow \tau = J_G^T(q) \gamma_E$$



Generalized joint forces (forces & torques at all joints)

Generalized end-effector forces

Manipulability Ellipsoids

Consider a set of generalized joint velocities of unit norm

$$\dot{q}^T \dot{q} = 1$$

Cause $\dot{q} = J_G(q)^{-1} \cdot v_E \Rightarrow \underbrace{v_E^T \left(J_G(q) J_G^T(q)\right)^{-1} v_E = 1}_{\text{a n-dimensional unit ellipsoid}}$

Called: Velocity manipulability ellipsoid

+) The closer the ellipsoid to a sphere $\Rightarrow$ better the isotropic movement

+) Along major axis $\Rightarrow$ ee gain high velocities
   minor ——— low ———

+) if $r < n$ (redundancy) $\Rightarrow$ use $J_G^{\dagger}(q) = J_G^T(q)\left(J_G(q) J_G^T(q)\right)^{-1}$
   Pseudo inverse

+) A scalar measure of the manipulation ability:

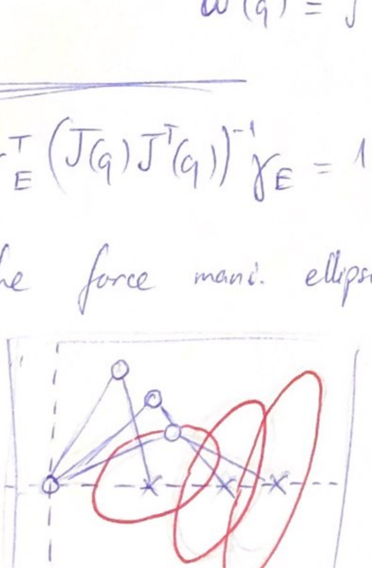
$$w(q) = \sqrt{J_G(q) J_G^T(q)}$$

non-redundant: $w(q) = |J_G(q)|$

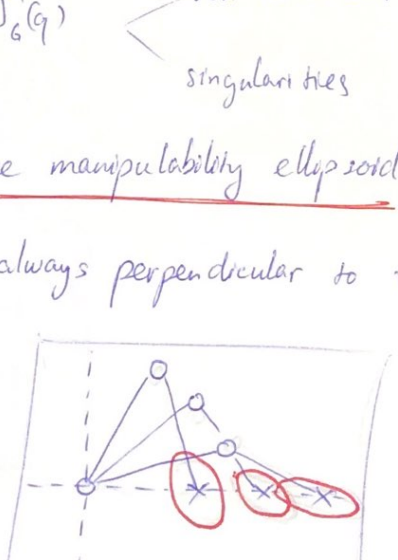
singularities $w(q) = 0$

$$\gamma_E^T \left(J(q) J^T(q)\right)^{-1} \gamma_E = 1$$: Force manipulability ellipsoid

The force mani. ellipsoid is always perpendicular to vel. mani. ellipsoid



Velocity mani. ellipsoid



Force mani. ellipsoid

Dynamic Models

$$M(q)\cdot\ddot{q} + G(q,\dot{q})\dot{q} + g(q) = \tau - F_v\,\dot{q} - F_s\,\mathrm{sgn}(\dot{q}) - J^T_{(q)} h_e$$

Lagrange: $\mathcal{L}(q,\dot{q}) = \underbrace{E_{kin}}_{\text{kinetic}}(q,\dot{q}) - \underbrace{E_{pot}}_{\text{potential (energy)}}(q)$

$$M(q) = \sum_{i=1}^{n}\left(m_{link_i}\, J^T_{GP_{link_i}}\, J_{GP_{link_i}} + J^T_{GO_{link_i}}\, {}^0R_i\, {}^iI_{link_i}\, {}^0R_i^T\, J_{GO_{link_i}}\right)$$

τ: vector of generalized joint forces

$F_v\,\dot{q}$: is the viscous friction in the joints, F_v: diagonal matrix of coefficients

$F_s\,\mathrm{sgn}(\dot{q})$: is the Coulomb friction, F_s: ________

(*) The positional Jacobian from Analytical & Geometric Method are the SAME

→ Geometric: for Control loop, not complex, no derivative
only matrix & vector multiplication
⇒ easy for computer implementation

→ Analytical: less step & less calculation

L8, Dynamic Parameter Identification

+) CAD Modelling: Simple geometry-based parameter identification
Material info available
$\boxed{\downarrow}$ Inaccurate: simplification + lack of dynamic effects

+) Heuristic Approach: Heavy computation schemes & measurements
$\boxed{\downarrow}$ Hard to implement and measure all

+) ☆ Linear Approach: Exploite linearity ⇒ estimate accurately
Based on imposed motion trajectories & measurement of joint torques

---

$$\tau = Y(q, \dot{q}, \ddot{q})\pi$$

(10 dynamic parameters)
11 components

$$\pi_i = [m_i \quad m_i\,{}^ir_{x;CG_i,i_0} \quad m_i\,{}^ir_{y;CG_i,i_0} \quad m_i\,{}^ir_{z;CG_i,i_0} \quad {}^i\hat{I}_{xx;i} \quad {}^i\hat{I}_{xy;i} \quad {}^i\hat{I}_{xz;i} \quad {}^i\hat{I}_{yy;i} \quad {}^i\hat{I}_{yz;i} \quad {}^i\hat{I}_{zz;i} \quad I_{mot_i}]^T$$

link mass, position vector of center of mass, inertia matrix, actuator inertia

N measurement (high enough to avoid ill-condition matrix)

$$\bar{\tau} = \begin{bmatrix} \tau(t_1) \\ \tau(t_2) \\ \vdots \\ \tau(t_n) \end{bmatrix} = \begin{bmatrix} Y\, q(t_1), \dot{q}(t_1), \ddot{q}(t_1) \\ \vdots \\ Y\, q(t_n), \dot{q}(t_n), \ddot{q}(t_n) \end{bmatrix} \pi = \bar{Y}\pi$$

$$\Rightarrow \pi = \underbrace{(\bar{Y}^T\bar{Y})^{-1}\bar{Y}^T}_{\text{left pseudo inverse of } \bar{Y}}\bar{\tau}$$ (least square method)

⊗ ONLY when $\bar{Y}^T\bar{Y}$ has full rank

What if $\bar{Y}^T\bar{Y}$ no full rank? ⇒

Initial Guess: π_0

Attach:

$I\pi = \pi_0 \Rightarrow \tilde{\pi} = \pi - \pi_0$ & $\tilde{y} = y - \bar{Y}\pi_0$

$$\Rightarrow \begin{bmatrix} \bar{Y} \\ \lambda I \end{bmatrix} \tilde{\pi} = \begin{bmatrix} \tilde{y} \\ 0 \end{bmatrix}$$

$$\Rightarrow \tilde{\pi} = [\bar{Y}^T \bar{Y} + \underbrace{\lambda^2}_{\text{damping factor}} I]^{-1} \bar{Y}^T \tilde{y}$$

Damped least-square method

does NOT need $\bar{Y}^T\bar{Y}$ to has full rank

L9/ Direct & Inverse Dynamics

$$M(q)\ddot{q} + G(q,\dot{q})\dot{q} + C_v\dot{q} + F_d\,\mathrm{sgn}(\dot{q}) + g(q) = \tau - J^T(q)h_e$$

$$\Rightarrow \quad \ddot{q} = M^{-1}(q)(\tau - \tau')$$

$$\tau'(q,\dot{q}) = G(q,\dot{q})\dot{q} + C_v\dot{q} + F_d\,\mathrm{sgn}(\dot{q}) + g(q) + J^T(q)h_e$$

(*) $M^{-1}(q)$ & $\tau(q,\dot{q})$ can be computed better using Newton-Euler formulation

⋮

In the end, we have a differential equation $\ddot{q}(t) = f(\dot{q}(t), q(t), t)$ & $q(t_0)$; $\dot{q}(t_0)$

$\Rightarrow$ Aim to find numerical approximation of the solution:

+) Explicit Euler Method:

$$y_{j+1} = y_j + \Delta t \cdot f\ \dot{y}(t_j)$$

The error is proportional to Δt

+) Runge-Kutta Method better

L10) Parallel Kinematics

- Different Approaches to the Direct Kinematics

	Complexity	Stability	Real-time Potential
Elimination	Low	Unstable*	Possible
(Polynomial) Continuation	High	Stable	NONE
Gröbner Bases	Low		
Interval Analysis	Low		

Trajectory Planning

<u>Goal:</u> compute a function of time ⇒ describes robot motions

tasks + time constraints

Trajectory = path (set of points) + time

	Joint space	Operational Space
Task	Pick & Place	Glue, welding
	The actual path <u>should</u> <u>not</u> be of importance	give more control over the path

User → Inv Kin → Jo Space path planning → Jo Space Traj planning → Time sequence of interpolating func → Motion control system

User → Op space path planning → Op space Traj planing → Inv Kin → Time sequence of interpolating func → Motion control system

Input

Output

Timing Laws

/po li nâu miêu/

+) Cubic Polynomial

Inf jerk $\frac{da}{dt} = \infty$

($\Delta t = 0$) $\Rightarrow$ oscillation

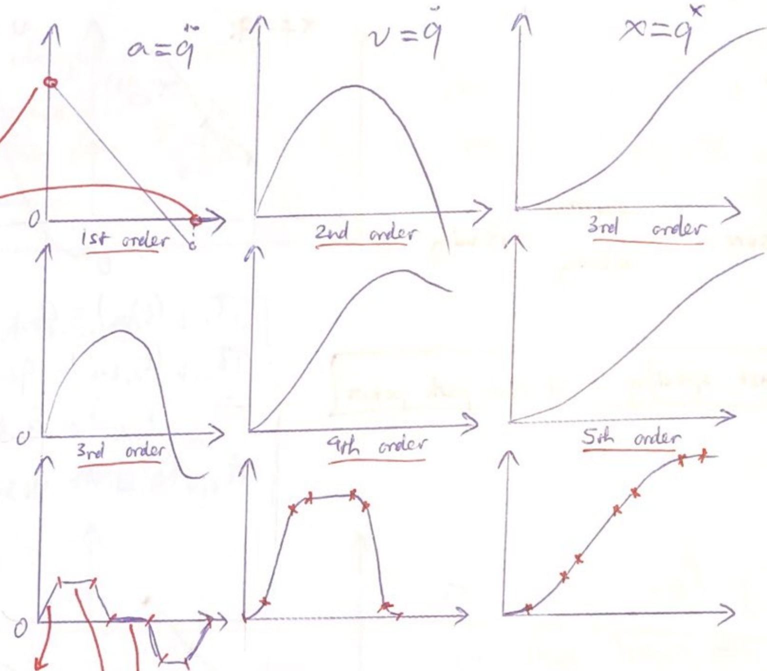
+) 5^{th} order polynomial

finite jerk

/tráp pô zôi đờ/

+) Trapezoidal acceleration profile

trapezoid



$$\begin{cases} \Delta t_{i,I} = T - \dfrac{\dot{q}_{max}}{\ddot{q}_{max}} - \dfrac{q_{max}}{\dot{q}_{max}} \\ \Delta t_{i,II} = 2 \cdot \dfrac{\dot{q}_{max}}{\ddot{q}_{max}} + \dfrac{q_{max}}{\dot{q}_{max}} - T \\ \Delta t_{i,IV} = T - 4\Delta t_{i,I} - 2\Delta t_{i,II} \end{cases}$$

T_i = time period

$\ddot{q}_{i,max}$: max acceleration

$\dot{q}_{i,max}$: max velocity

$q_{i,max}$: max position

These formulars and restrictions :

don't need to know where it comes from and WHY it is so

Just know that it's exist =))

Restriction

$$\frac{q_{max}}{T} \le \dot{q}_{max} \le \frac{2q_{max}}{T}$$

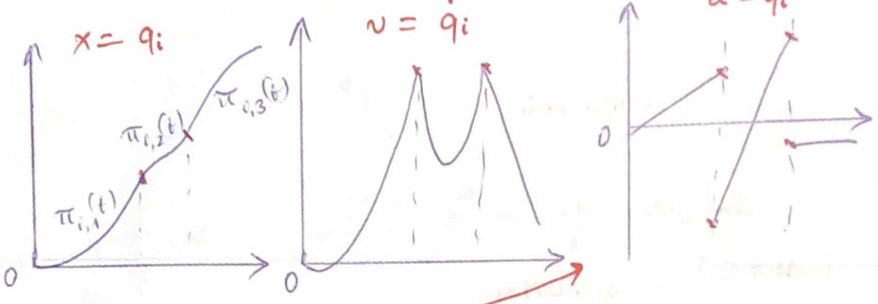
$$\dddot{q}_{max} = \frac{\ddot{q}_{max}}{\Delta t_{i,I}} = \frac{\ddot{q}_{max}}{T - \frac{\dot{q}_{max}}{\ddot{q}_{max}} - \frac{q_{max}}{\dot{q}_{max}}}$$

$$\frac{\dot{q}_{max}}{T - \frac{q_{max}}{\dot{q}_{max}}} \le \ddot{q}_{max} \le \frac{2\dot{q}_{max}}{T - \frac{q_{max}}{\dot{q}_{max}}}$$

Joint space Trajectories for Motion through a Sequece of Points

+) Polynomials with imposed velocities:

Ensure : $\left\langle \begin{array}{l} \text{position} \\ \text{velocity} \end{array} \right.$ continuity



$\boxed{\text{Must specify } v \text{ at each path points}}$

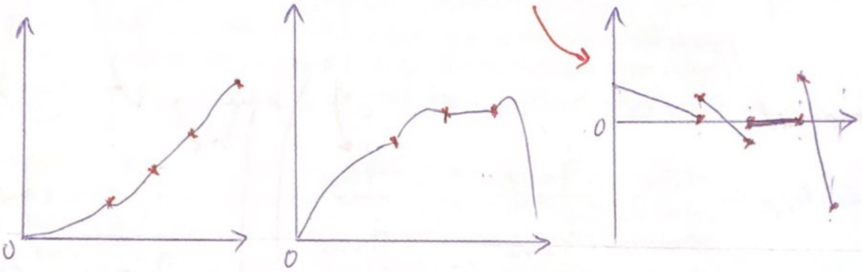
$$\begin{cases} \Pi_{i,k}(t_{i,k}) = q_{i,k} \\ \Pi_{i,k}(t_{i,k+1}) = q_{i,k+1} \\ \dot{\Pi}_{i,k}(t_{i,k}) = \dot{q}_{i,k} \\ \dot{\Pi}_{i,k}(t_{i,k+1}) = \dot{q}_{i,k+1} \end{cases}$$

$\Rightarrow$ N-1 systems of 4 equations with 4 unknowns

$\Rightarrow$ each system: solvable indepen

But: a $(\ddot{q})$ continuity is not guaranteed

+) Polynomials with computed velocities:



Given $q_{i,k}(t_k)$

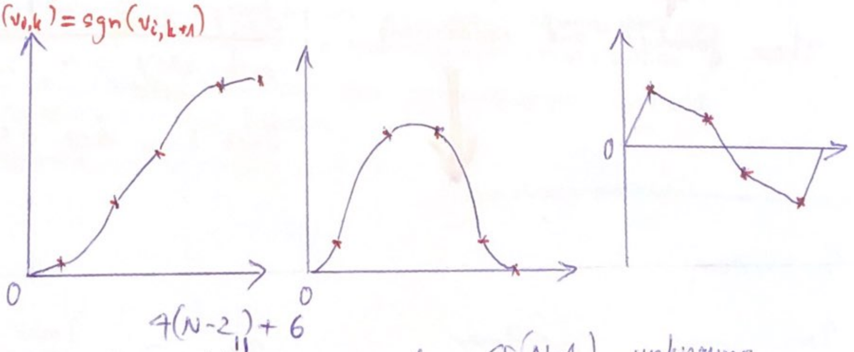
$$\Rightarrow \quad v_{i,k} = \frac{q_{i,k} - q_{i,k-1}}{t_{i,k} - t_{i,k-1}}$$

$\Rightarrow$ $\boxed{\text{Must calculate } \dot{q} \text{ at each path points}}$

$$\dot{q}_{i,1} = \dot{q}_{i,N} = 0$$

$$\dot{q}_{i,k} = \begin{cases} 0 & sgn(v_{i,k}) \neq sgn(v_{i,k+1}) \\ \frac{1}{2}(v_{i,k} + v_{i,k+1}) & sgn(v_{i,k}) = sgn(v_{i,k+1}) \end{cases}$$

+) Splines:



$$\Pi_{i,k-1}(t_{i,k}) = q_{i,k}$$

$$\Pi_{i,k-1}(t_{i,k}) = \Pi_{i,k}(t_{i,k})$$

$$\dot{\Pi}_{i,k-1}(t_{i,k}) = \dot{\Pi}_{i,k}(t_{i,k})$$

$$\ddot{\Pi}_{i,k-1}(t_{i,k}) = \ddot{\Pi}_{i,k}(t_{i,k})$$

$\Rightarrow$ $4.(N-2)$ equations + 6 equations (start, end points)

$\Rightarrow$ $\underbrace{4(N-2)+6}_{} = 4N-2$ equas for $4(N-1)$ unknowns

$\Rightarrow$ + 2 virtial points (locations are irrelevant)

$\Rightarrow$ $4(N+1)$ equations for $4(N+1)$ unknowns

$\Leftarrow$ Use $(N+3)$ order - polynomial has disadvantages

- High order $\rightarrow$ oscillations of high frequency & amplitude
- Numerical accuracy $\downarrow$
- When points are changed $\Rightarrow$ many coefficients must be recalculated

Lecture 11

Efficient algorithm: build $\pi_{i,k}$ from $\frac{\ddot{\pi}_{i,k}(t_k)}{\Delta t}$ and $\frac{\ddot{\pi}_{i,k}(t_{k+1})}{\Delta t}$ $\Rightarrow$ N unknown of $\ddot{q}$

Operational Space Trajectories

- Inv Kine algorithm is needed
- computationally expensive
- Linear micro - interpolation

$\boxed{\text{Path - primitives + timing laws} \Rightarrow \text{trajectory planning}}$

$\downarrow$

analyzed & computed separately

Path Γ /gamma/ with length s, general parameter σ /sigma/ $\in [\sigma_{start}, \sigma_{end}]$

+) Path Γ is closed if $r_p(\sigma_{start}) = r_p(\sigma_{end})$

$r_p(s) = f(s)$ with $\sigma_{start} = 0$, $\sigma_{end} = s_{end}$

Frenet frame

Tangent unit vector $t(s) = \frac{dr_p(s)}{ds}$ the direction

Normal ——— $n(s) = \frac{dt(s)}{ds}$ how the tangent vector is changing

Binormal ——— $b(s) = t(s) \times n(s)$ the plane (of $\vec{t}$ and $\vec{n}$) is twisting how

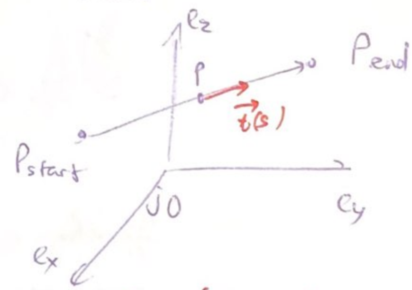
Why unit? Cause $|t(s)| = 1$, but $\frac{dr_p(t)}{dt}$ can $\neq 1$

It's the derivative on ds, not dt

Path primitive

+) <u>Rectilinear Path:</u> given $\vec{r}_{P_{end},j0}$ & $\vec{r}_{P_{start},j0}$

$$S = \cancel{\frac{r_{end} - r_{start}}{r_{end}}} \quad S = |r_{end} - r_{start}|$$



$$r_{P,j0}(s) = r_{start} + \frac{s}{S} \cdot t(s)$$

$$t(s) = \frac{r_{end} - r_{start}}{S} = const \Rightarrow$$ **CAN NOT** define (t, n, b) uniquely

+) <u>Circular arc:</u> given P_{start}, P_{mid} & P_{end}

$$^{j}e_{z,C_0} = \frac{^{j}r_{P_{mid},P_{start}} \times {}^{j}r_{P_{end},P_{start}}}{|^{j}r_{P_m,P_s} \times {}^{j}r_{P_e,P_s}|}$$

$$^{j}r_{H_1} = h_1 \frac{^{j}e_{z,C_0} \times {}^{j}r_{P_m,P_s}}{|^{j}e_{z,C_0} \times {}^{j}r_{P_m,P_s}|}$$

$$^{j}r_{H_2} = h_2 \frac{^{j}e_{z,C_0} \times {}^{j}r_{P_e,P_s}}{| \; . \; . \; . \; |} \Rightarrow \begin{cases} \underline{^{j}r_{C_0,j0}} = {}^{j}r_{P_s,j0} + \frac{1}{2}{}^{j}r_{P_m,P_s} + h_1 {}^{j}e_{H_1} \\ \quad = {}^{j}r_{P_s,j0} + \frac{1}{2}{}^{j}r_{P_e,P_s} + h_2 {}^{j}e_{H_2} \\ h_1 {}^{j}e_{H_1} - h_2 {}^{j}e_{H_2} = \frac{1}{2}{}^{j}r_{P_e,P_m} \end{cases}$$

$$^{j}r_{C_0,j0} = \dots \text{ from } h_1 \text{ or } h_2$$

$\Rightarrow$ ~~Solte~~ Solve: $\Rightarrow h_1, h_2$

$$^{j}e_{x,C_0} = \frac{r_{P_s} - r_{C_0}}{| \; . \; . \; . \; |}$$

$$^{j}e_{y,C_0} = e_z \times e_x$$

$$r_P = r_{C_0} + R_C {}^{C}r_{P,C_0} \Rightarrow \begin{cases} n(s) = {}^{j}R_C \dfrac{-{}^{C}r_{P,C_0}(s)}{|^{C}r_{P,C_0}(s)|} & (2) \\ b(s) = e_{z,C_0} & (1) \\ t(s) = n(s) \times b(s) & (3) \end{cases}$$

order of calculation

$$r_c = |^{c}r_{P_s,C_0}|$$

$$s = K_P \cdot r_c$$

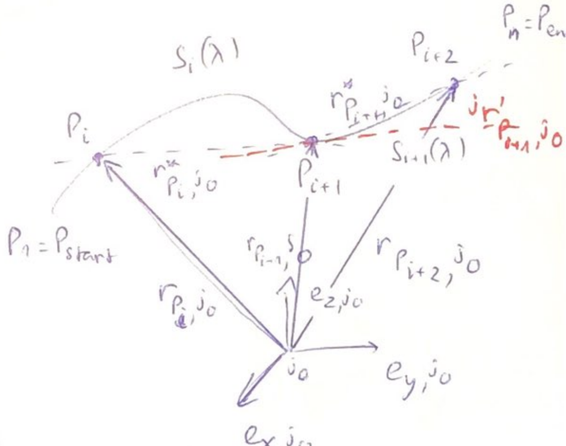
+) <u>Spline Path:</u> use 5th order polynomials + boundary conditions

We have to define the 1st and 2nd derivatives of the path at path points are not given:

$$ {}^{j}r^{*}_{P_i,j_0} = \frac{{}^{j}r_{P_{i+1},j_0} - {}^{j}r_{P_i,j_0}}{\lambda_i} $$

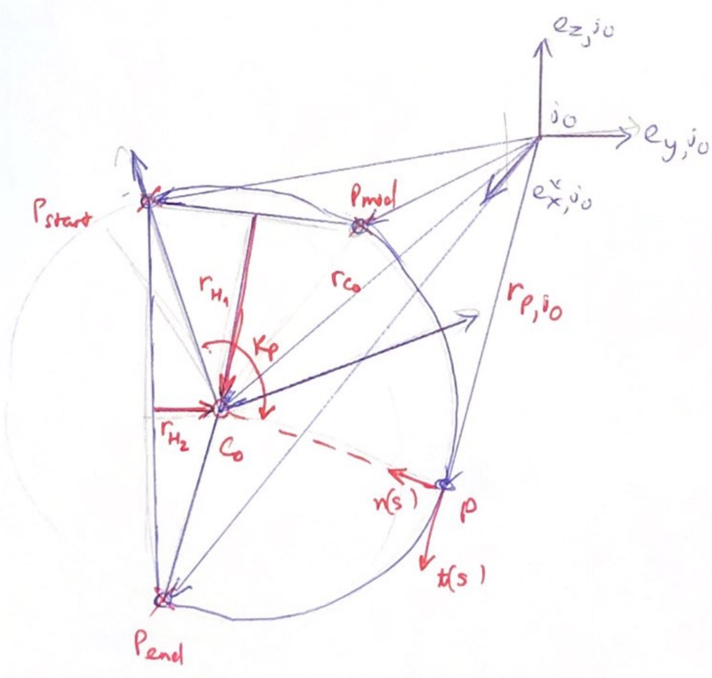
λ_i: length of $S_i(\lambda)$

$$ {}^{j}r_{P_{i+1},j_0} = \frac{{}^{j}r_{P_{i+2},j_0} - {}^{j}r_{P_{i+1},j_0}}{\lambda_{i+1}} $$



Final

$$ \Rightarrow \begin{cases} {}^{j}r'_{P_{i+1},j_0} = \dfrac{\lambda_{i+1}\,{}^{j}r_{P_i,j_0} + \lambda_i\,{}^{j}r_{P_{i+1},j_0}}{\lambda_i + \lambda_{i+1}} \\[2ex] {}^{j}r''_{P_{i+1},j_0} = \dfrac{{}^{j}r^{*}_{P_{i+1},j_0} - {}^{j}r_{P_i,j_0}}{\dfrac{\lambda_i + \lambda_{i+1}}{2}} \end{cases} $$



E. Ex5:

Task: Compute Geometric Jacobian:

$\theta \quad d \quad l \quad \alpha$

$\delta \quad d \quad l \quad \lambda$

D-H Convention:

Link	③ $l_{j,i}$	④ $\lambda_{j,i}$	② $d_{j,i}$	① $\delta_{j,i}$
1,0				
2,1				
3,2				

$i, i-1$

$${}^{i-1}R_i$$

$$\begin{bmatrix} \boxed{\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 \; 0 \; 0 & 1 \end{bmatrix} \Rightarrow {}^{i-1}T_i \Rightarrow {}^{0}T_i = \begin{bmatrix} \cdot & \cdot & \boxed{\cdot} & \boxed{\cdot} \\ \cdot & \cdot & \boxed{\cdot} & \boxed{\cdot} \\ \cdot & \cdot & \boxed{\cdot} & \boxed{\cdot} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

${}^{0}T_1, {}^{1}T_2 \dots \quad {}^{0}T_n = {}^{0}T_1 \cdot {}^{1}T_2 \cdot {}^{2}T_3 \dots$

$${}^{0}R_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\|$$

${}^{0}e_{z,i-1_0}$ (${}^{0}e_{z,0_0}$ from z_0, ${}^{0}e_{z,1_0}$ from ${}^{0}R_1$

${}^{0}r_{E_0,0_0}$ from ${}^{0}T_N = \begin{bmatrix} \cdot & \cdot & \cdot & \boxed{\cdot} \\ \cdot & \cdot & \cdot & \boxed{\cdot} \\ \cdot & \cdot & \cdot & \boxed{\cdot} \\ 0 & 0 & 0 & 1 \end{bmatrix}$

${}^{0}r_{i-1_0,0_0}$ from ${}^{0}T_i = \begin{bmatrix} \cdot & \cdot & \cdot & \boxed{\cdot} \\ \cdot & \cdot & \cdot & \boxed{\cdot} \\ \cdot & \cdot & \cdot & \boxed{\cdot} \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$$\Rightarrow J_G = \begin{bmatrix} j_{GP_1} & j_{GP_2} & j_{GP_3} & \dots \\ j_{GO_1} & j_{GO_2} & j_{GO_3} & \dots \end{bmatrix}$$

size $(6 \times n)$

Revolute joint $\begin{cases} j_{GP_i} = {}^{0}e_{z,i-1_0} \times ({}^{0}r_{E,0_0} - {}^{0}r_{i-1_0,0_0}) \\ j_{GO_i} = {}^{0}e_{z,i-1_0} \end{cases}$

Example: $j_{GP_1} = {}^{0}e_{z,0_0} \times (\rho_E - {}^{0}r_{0_0,0_0})$

Prismatic joint $\begin{cases} j_{GP_i} = {}^{0}e_{z,i-1_0} \\ j_{GO_i} = \vec{0} \end{cases}$

Example: $j_{GP_3} = {}^{0}e_{z,2_0}$

Task: Singularities Consider when $\det|J| = 0$

$\det|J_{GP}| \overset{?}{=}$

Ex 6: Task 1 Analytical Jacobian:

$$\dot{x}_E = \begin{bmatrix} \dot{p}_E \\ \dot{\Phi}_E \end{bmatrix} = J_A(q) \cdot \dot{q}$$

DH Convention $\Rightarrow {}^0T_n = \begin{bmatrix} \boxed{\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}} & \boxed{\begin{matrix} 0 \\ 0 \\ 0 \end{matrix}} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$

$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = p_E \Rightarrow J_A = \begin{bmatrix} J_{AP}(q) \\ J_{A\Phi}(q) \end{bmatrix} = \begin{bmatrix} \frac{\partial J_{AP}}{\partial q_1} & \cdots & \frac{\partial J_{AP}}{\partial q_n} \\ \frac{\partial J_{A\Phi}}{\partial q_1} & \cdots & \frac{\partial J_{A\Phi}}{\partial q_n} \end{bmatrix}$ or $\begin{bmatrix} \frac{\partial p_x}{\partial q_1} & \cdots \\ \frac{\partial p_y}{\partial q_1} & \cdots \\ \vdots & \\ \frac{\partial \psi}{\partial q_1} & \cdots \end{bmatrix}$

$\begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} = \Phi_E$

Analytical Jacobian of ORientation $\neq$ Rotational Jacobian of given sequence

Task 2 Rotational Jacobian: current frame

Example: $ZY'X''$

$$\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

Elemental Rotational metrices $\Rightarrow$ R_x, R_y, R_z

$$\frac{\partial R_z}{\partial \varphi} \cdot R_z^T \Rightarrow R_z \frac{\partial R_y}{\partial \theta} R_y^T R_z^T \Rightarrow R_z R_y \frac{\partial R_x}{\partial \psi} R_x^T R_y^T R_z^T$$

$\Downarrow$ cột 1: sẽ là $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ vì quay quanh z

$\Downarrow$ cột 2: sẽ chỉ chứa φ

$\Downarrow$ cột 3: sẽ chỉ chứa φ và θ

fixed frame: doing back ward: Example: XYZ $\Rightarrow$ do $Z \Rightarrow Y, \Rightarrow X$ như trên

Task 3:

Intuition: $J_R(\Phi_E)$ has each column is the unit vector of the rotation axis

Example: $J_R(\Phi_E) = \begin{bmatrix} 0 & -s\varphi & c\varphi s\theta \\ 0 & c\varphi & s\varphi c\theta \\ 1 & 0 & -s\theta \end{bmatrix}$ for $ZY'X''$ rotation

$\begin{bmatrix} c\varphi s\theta \\ s\varphi c\theta \\ -s\theta \end{bmatrix}$ is the X'' axis after 2 rotations

$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ $\begin{bmatrix} -s\varphi \\ c\varphi \\ 0 \end{bmatrix}$ is the Y' axis after 1st rotation

For both current & fixed frame (Fixed frame just reverse the order)

$$J_{G0} = J_R \cdot J_{A0} \quad \begin{cases} \omega_E = J_R(\Phi_E) \cdot \dot{\Phi}_E \\ \omega_E = J_{G0} \cdot \dot{q} \\ \dot{\Phi}_E = J_{A0} \cdot \dot{q} \end{cases}$$

Inverse Differential Kinematics

1, Direct Kinematics

$$p_{k-1,E} = k(q_{k-1})$$

2, Weighted error vector

$$e_{k-1,wP} = K(p_d - p_{k-1,E})$$

3, Joint velocities:

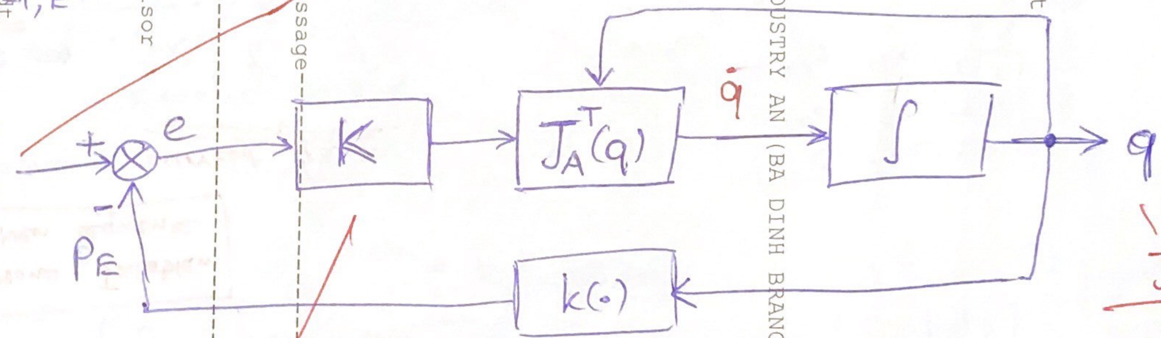
$$\dot{q}_k = J_A^T(q_{k-1})\, e_{k-1,wP}$$

4, Joint Configuration

$$q_k = q_{k-1} + \Delta t \cdot \dot{q}_k$$

5, Repeat

desired end-effector position



Joint coordinates

Optimizing weighting parameters yields faster convergence

$k_i < 1 \Rightarrow$ damping behavior

$k_i > 1 \Rightarrow$ (maybe) oscillation

Ex 2: Lagrange

DH notation $\Rightarrow {}^{i}T_{i+1} \Rightarrow {}^{0}T_n \Rightarrow \begin{cases} {}^{0}r_{i_0,0_0} \\ {}^{0}R_i \\ {}^{0}e_{z,i_0} \end{cases}$

$$j_{GPj\,link\,i} = \begin{cases} {}^{0}e_{z,j-1_0} & \text{prismatic} \\ {}^{0}e_{z,j-1_0} \times \left({}^{0}r_{CG\,link_i,0_0} - {}^{0}r_{j-1_0,0_0}\right) & \text{revolute} \end{cases}$$

$\boxed{\begin{array}{l} i \text{ link} \\ j \text{ from } 1 \to i \end{array}}$

$$\otimes \quad {}^{0}r_{CG\,link\,i,0_0} = {}^{0}r_{i_0,0_0} + {}^{0}R_i \cdot {}^{i}r_{CG\,link\,i,i_0}$$

$$J_{GO\,link\,i} = [\underbrace{j_{GO1\,link\,i}}_{j=1} \quad \underbrace{j_{GO2\,link\,i}}_{j=2} \dots 0]$$

$$j_{GOj\,link\,i} = \begin{cases} 0 & \text{prismatic} \\ {}^{0}e_{z,j-1_0} & \text{revolute} \end{cases}$$

$$\underbrace{J_{GP\,link\,1}}_{i=1} = [\underbrace{j_{GP_1\,link_1}}_{j=1} \quad 0 \quad 0]$$

$$\underbrace{J_{GP\,link\,2}}_{i=2} = [\underbrace{j_{GP_1\,link_2}}_{j=1,\,i=2} \quad \underbrace{j_{GP_2\,link\,2}}_{j=2,\,i=2} \quad 0]$$

$$M(q) = \sum_{i=1}^{n} \left(m_{link_i} J_{GP_{link_i}}^{T} J_{GP_{link_i}} + J_{GO_{link_i}}^{T}\, {}^{0}R_i\, {}^{i}I_{link_i}\, {}^{0}R_i^{T} J_{GO\,link_i} \right)$$

size $n \times n$

n = no. of joints

Ex 2,

$$g_{ij} = \sum_{k=1}^{n} g_{ijk}\,\dot{q}_k \quad ; \quad g_{ijk} = \frac{1}{2}\left(\frac{\partial m_{ij}}{\partial q_k} + \frac{\partial m_{ik}}{\partial q_j} - \frac{\partial m_{jk}}{\partial q_i}\right) = g_{ikj}$$

$$\Rightarrow G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ \vdots & & \ddots & \vdots \\ g_{n1} & \cdots & \cdots & g_{nn} \end{bmatrix}$$

$$g_i(q) = -\sum_{j=1}^{n}\left(m_{link_j}\,{}^{0}g_0^T\,\frac{\partial\, {}^{0}r_{CG\,link_j,\,0}}{\partial q_i}\right)$$

$$\rightarrow \quad M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

and for Control loop

Note: The positional Jacobian from Analytical Method and Geometric Method are the same

But Analytical is easy, less calculation

But Geometric is not complex, no derivative, only matrix, vector multiplication $\Rightarrow$ easy for computer to implement
& has physical meaning (in operational space)

Newton - Euler Algorithm

+) Given: $l, d \ldots$ Initial conditions: ${}^{0}\ddot{r}_{0,0}$, ${}^{0}\omega_0$, ${}^{0}\dot{\omega}_0$

$m_{link\ i} = \ldots$ ${}^{N}f_{N,N-1}$, ${}^{N}\mu_{N,N-1}$

${}^{i}I_{link_i} = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$ (force) (Moment)

+) D-H notation

DH parameters

Transformation matrices

Link	3 a $l_{i,i-1}$	4 α $\lambda_{i,i-1}$	2 d $d_{i,i-1}$	1 - θ $\delta_{i,i-1}$
1				
⋮				
N				

$$\Rightarrow {}^{i-1}T_i = \begin{bmatrix} \boxed{\begin{matrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{matrix}} & \begin{matrix} \cdot \\ \cdot \\ \cdot \end{matrix} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \Rightarrow {}^{0}T_i = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$({}^{0}T_1, {}^{1}T_2, \ldots)$ $({}^{0}T_n = {}^{0}T_1 \cdot {}^{1}T_2 \cdot {}^{2}T_3 \ldots)$

${}^{i-1}R_i$

+) <u>Angular velocities of frames</u>

acceleration

${}^{0}\omega_i$

$$ {}^{i}\omega_i = \begin{cases} {}^{i-1}R_i^T \cdot {}^{i-1}\omega_{i-1} & \text{, prismatic} \\ {}^{i-1}R_i^T \left({}^{i-1}\omega_{i-1} + \dot{\delta}_i \, \underline{{}^{i-1}e_{z,{}^{i-1}0}} \right) & \text{, revolute} \end{cases} $$

Example: $ {}^{1}\omega_1 = {}^{0}R_1^T \left({}^{0}\omega_0 + \dot{q}_1 \, {}^{0}e_{z,{}^{0}0} \right) $

$ {}^{2}\omega_2 = {}^{1}R_2^T \left({}^{1}\omega_1 + \dot{q}_2 \, {}^{1}e_{z,{}^{1}0} \right) $

$ {}^{0}\omega_0 $ is given, $\Rightarrow {}^{1}\omega_1 = \dots \Rightarrow {}^{2}\omega_2$

of the base

$$ \boxed{ {}^{i-1}e_{z,{}^{i-1}0} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} } $$

${}^{0}\dot{\omega}_i$

$$ {}^{i}\dot{\omega}_i = \begin{cases} {}^{i-1}R_i^T \cdot {}^{i-1}\dot{\omega}_{i-1} & \text{, prismatic} \\ {}^{i-1}R_i^T \left({}^{i-1}\dot{\omega}_{i-1} + \ddot{\delta}_i \, \underline{{}^{i-1}e_{z,{}^{i-1}0}} + \dot{\delta}_i \, {}^{i-1}\omega_{i-1} \times \underline{{}^{i-1}e_{z,{}^{i-1}0}} \right) & \text{, revolute} \end{cases} $$

$ {}^{0}\dot{\omega}_0 $ is given, $\Rightarrow {}^{1}\dot{\omega}_1 = \dots \Rightarrow {}^{2}\dot{\omega}_2 = \dots$

of the base

Example: $ {}^{1}\dot{\omega}_1 = {}^{0}R_1^T \left({}^{0}\dot{\omega}_0 + \ddot{q}_1 \, {}^{0}e_{z,{}^{0}0} + \dot{q}_1 \, {}^{0}\omega_0 \times {}^{0}e_{z,{}^{0}0} \right) $

$ {}^{2}\dot{\omega}_2 = {}^{1}R_2^T \left({}^{1}\dot{\omega}_1 + \ddot{q}_2 \, {}^{1}e_{z,{}^{1}0} + \dot{q}_2 \, {}^{1}\omega_1 \times {}^{1}e_{z,{}^{1}0} \right) $

+) *Linear acceleration of frames*

—— centers of mass:

$$ {}^0\ddot r_{i_0,0_0} $$

$$ {}^i\ddot r_{i_0,0_0} = \begin{cases} {}^{i-1}R_i^T \cdot {}^{i-1}\ddot r_{i-1_0,0_0} + {}^i\dot\omega_i \times {}^i r_{i_0,i-1_0} + {}^i\omega_i \times ({}^i\omega_i \times {}^i r_{i_0,i-1_0}) & \text{revolute} \\ {}^{i-1}R_i^T \left({}^{i-1}\ddot r_{i-1_0,0_0} + \ddot d_i\, {}^{i-1}e_{z,i-1_0}\right) + 2\dot d_i\, {}^i\omega_i \times \left({}^{i-1}R_i^T \cdot {}^{i-1}e_{z,i-1_0}\right) + {}^i\dot\omega_i \times {}^i r_{i_0,i-1_0} + {}^i\omega_i \times ({}^i\omega_i \times {}^i r_{i_0,i-1_0}) & \text{prismatic} \end{cases} $$

(*) +) Find from graph (if possible)

+) ${}^i r_{i_0,i-1_0} = {}^{i-1}R_i^T \cdot {}^{i-1} r_{i_0,i-1_0}$

$$ {}^0\ddot r_{0_0,0_0} = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \text{ is given, } \Rightarrow {}^1\ddot r_{1_0,0_0} = \dots \Rightarrow {}^2\ddot r_{2_0,0_0} = \dots $$

Example:

$$ {}^1\ddot r_{1_0,0_0} = {}^0R_1^T \cdot {}^0\ddot r_{0_0,0_0} + {}^1\dot\omega_1 \times {}^1 r_{1_0,0_0} + {}^1\omega_1 \times ({}^1\omega_1 \times {}^1 r_{1_0,0_0}) $$

$$ {}^0\ddot r_{CG\,i,0_0} $$

$$ {}^i\ddot r_{CG\,link_i,0_0} = {}^i\ddot r_{i_0,0_0} + {}^i\dot\omega_i \times {}^i r_{CG\,link_i,i_0} + {}^i\omega_i \times ({}^i\omega_i \times {}^i r_{CG\,link_i,i_0}) $$

Example:

$$ {}^1\ddot r_{CG\,link_1,0_0} = {}^1\ddot r_{1_0,0_0} + {}^1\dot\omega_1 \times {}^1 r_{CG\,link_1,1_0} + {}^1\omega_1 \times ({}^1\omega_1 \times {}^1 r_{CG\,link_1,1_0}) $$

In the lecture, all vectors were referred to frame0

(*) But to achieve computationally efficient algorithm

They need to be in the current frame i

+) Forces at frames:

Moments ————:

$$ {}^{i}f_{i,i-1} = {}^{i}R_{i+1}\, {}^{i+1}f_{i+1,i} + m_{link\,i}\, {}^{i}\ddot{r}_{CG\,link\,i,\,{}^{0}0} $$

$$ {}^{N}f_{N,N-1} \Rightarrow {}^{N-1}f_{N-1,N-2} \quad \dots $$

Example: $ {}^{3}f_{3,2} = {}^{3}R_{4}\, {}^{4}f_{4,3} + m_{link\,3}\, {}^{3}\ddot{r}_{CG\,link\,3,\,{}^{0}0} $

...

$$ {}^{i}\mu_{i,i-1} = -{}^{i}f_{i,i-1} \times \left({}^{i}r_{i_0,i-1_0} + {}^{i}r_{CG\,link_i,\,i_0}\right) + {}^{i}R_{i+1}\, {}^{i+1}\mu_{i+1,i} + \left({}^{i}R_{i+1}\, {}^{i+1}f_{i+1,i}\right) \times {}^{i}r_{CG\,link_i,\,i_0} $$
$$ + {}^{i}I_{link_i} \cdot {}^{i}\dot{\omega}_i + {}^{i}\omega_i \times \left({}^{i}I_{link\,i}\, {}^{i}\omega_i\right) $$
$$ \left(+ k_{mot_{i+1}}\, \ddot{q}_{i+1}\, I_{mot_{i+1}}\, {}^{0}e_{z,mot_{i+1}} + k_{mot_{i+1}}\, \dot{q}_{i+1}\, I_{mot_{i+1}}\, {}^{0}\omega_i \times {}^{0}e_{z,mot_{i+1}} \right) $$

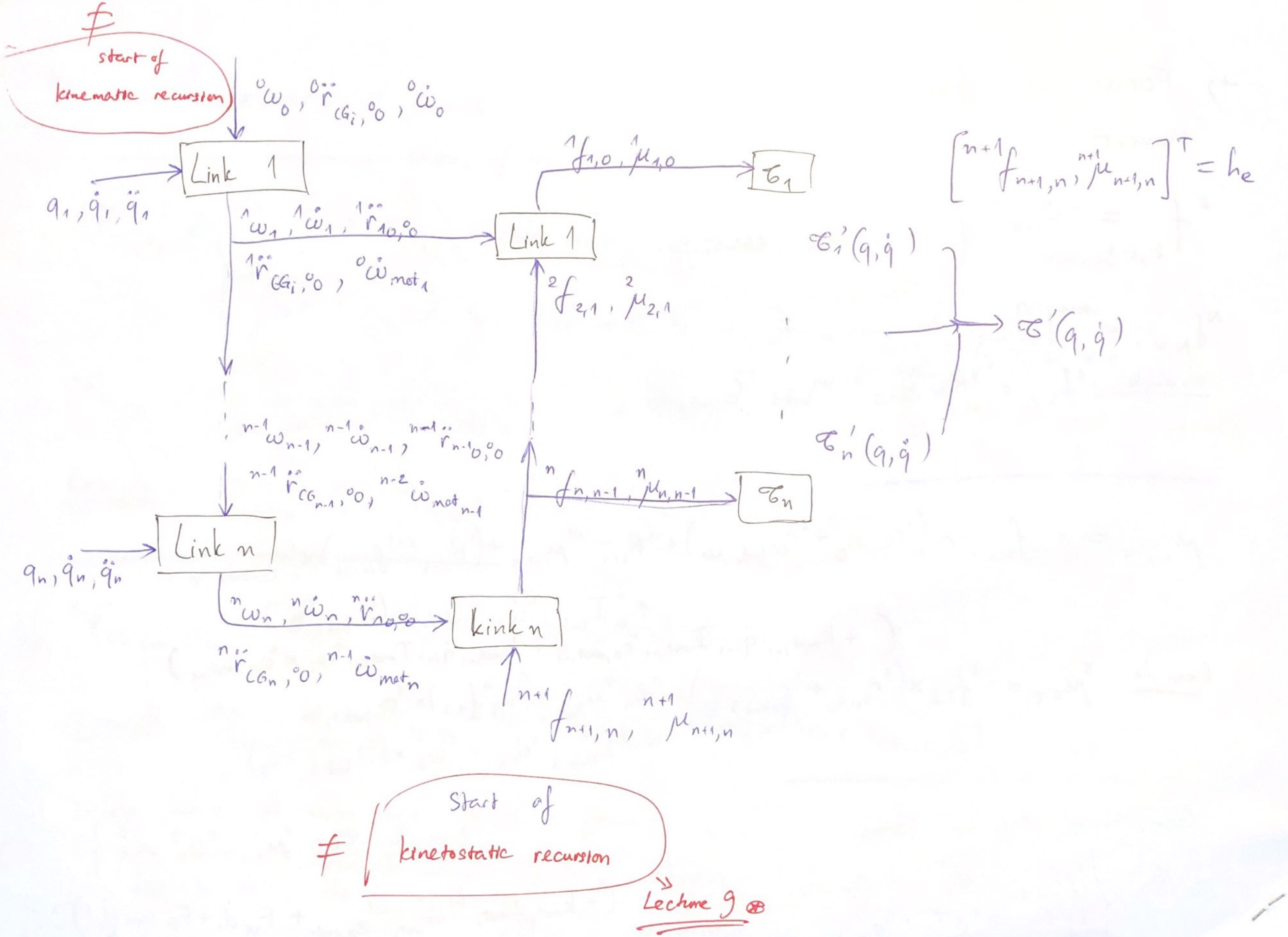
Example: $ {}^{3}\mu_{3,2} = -{}^{3}f_{3,2} \times \left({}^{3}r_{3_0,2_0} + {}^{3}r_{CG\,link\,3,\,{}^{3}0}\right) + {}^{3}R_{4}\, {}^{4}\mu_{4,3} + \left({}^{3}R_{4}\, {}^{4}f_{4,3}\right) \times {}^{3}r_{CG\,link\,3,\,{}^{3}0} $
$$ + {}^{3}I_{link\,3}\, {}^{3}\dot{\omega}_3 + {}^{3}\omega_3 \times \left({}^{3}I_{link\,3}\, {}^{3}\omega_3\right) $$

Generalized Actuator Forces

Example: $ \tau_1 = {}^{1}\mu_{1,0}^{T} \cdot {}^{0}R_{1}^{T} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} $

$$ \tau_i = \begin{cases} {}^{i}\mu_{i,i-1}^{T} \cdot {}^{i-1}R_{i}^{T} \cdot {}^{i-1}e_{z,i-1_0} & \text{, revolute} \\ {}^{i}f_{i,i-1}^{T} \cdot {}^{i-1}R_{i}^{T} \cdot {}^{i-1}e_{z,i-1_0} & \text{, prismatic} \end{cases} $$

$$ \left(+ k_{mot_i}\, I_{mot_i}\, {}^{i-1}\dot{\omega}_{mot_i}^{T} \cdot {}^{i-1}e_{z,mot_i} + F_{vi}\, \dot{q}_i + F_{si}\, \mathrm{sgn}(\dot{q}_i) \right) \xrightarrow{\text{or}} \dot{\theta}_i $$


start of kinematic recursion
$^0\omega_0$, $^0\ddot{r}_{CG_i,{}^0O}$, $^0\dot{\omega}_0$
Link 1
$q_1, \dot{q}_1, \ddot{q}_1$
$^1\omega_1$, $^1\dot{\omega}_1$, $^1\ddot{r}_{1O,{}^0O}$
$^1\ddot{r}_{CG_i,{}^0O}$, $^0\dot{\omega}_{mot_1}$
$^{n-1}\omega_{n-1}$, $^{n-1}\dot{\omega}_{n-1}$, $^{n-1}\ddot{r}_{n-1O,{}^0O}$
$^{n-1}\ddot{r}_{CG_{n-1},{}^0O}$, $^{n-2}\dot{\omega}_{mot_{n-1}}$
Link n
$q_n, \dot{q}_n, \ddot{q}_n$
$^n\omega_n$, $^n\dot{\omega}_n$, $^n\ddot{r}_{nO,{}^0O}$
$^n\ddot{r}_{CG_n,{}^0O}$, $^{n-1}\dot{\omega}_{mot_n}$
$^1f_{1,0}$, $^1\mu_{1,0}$
τ_1
Link 1
$^2f_{2,1}$, $^2\mu_{2,1}$
$^nf_{n,n-1}$, $^n\mu_{n,n-1}$
τ_n
Link n
$^{n+1}f_{n+1,n}$, $^{n+1}\mu_{n+1,n}$
$\left[{}^{n+1}f_{n+1,n}, {}^{n+1}\mu_{n+1,n}\right]^T = h_e$
$\tau_1'(q,\dot{q})$
$\tau_n'(q,\dot{q})$
$\tau'(q,\dot{q})$
Start of kinetostatic recursion
Lecture 9

Orientation	
RPY Rotation Matrix	$\mathbf{R}_{RPY}(\boldsymbol{\Phi}) = \begin{bmatrix} c_\varphi c_\theta & c_\varphi s_\theta s_\psi - s_\varphi c_\psi & c_\varphi s_\theta c_\psi + s_\varphi s_\psi \\ s_\varphi c_\theta & s_\varphi s_\theta s_\psi + c_\varphi c_\psi & s_\varphi s_\theta c_\psi - c_\varphi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}$
Axis and Angle Rotation Matrix	$\mathbf{R}(\theta, \mathbf{r}) = \begin{bmatrix} r_x^2(1-c_\theta)+c_\theta & r_x r_y(1-c_\theta) - r_z s_\theta & r_x r_z(1-c_\theta) + r_y s_\theta \\ r_x r_y(1-c_\theta) + r_z s_\theta & r_y^2(1-c_\theta)+c_\theta & r_y r_z(1-c_\theta) - r_x s_\theta \\ r_x r_z(1-c_\theta) - r_y s_\theta & r_y r_z(1-c_\theta) + r_x s_\theta & r_z^2(1-c_\theta)+c_\theta \end{bmatrix}$
Unit Quaternion: Rotation Matrix	$\mathbf{R}(\eta, \boldsymbol{\epsilon}) = \begin{bmatrix} 2(\eta^2+\epsilon_x^2)-1 & 2(\epsilon_x\epsilon_y - \eta\epsilon_z) & 2(\epsilon_x\epsilon_z + \eta\epsilon_y) \\ 2(\epsilon_x\epsilon_y + \eta\epsilon_z) & 2(\eta^2+\epsilon_y^2)-1 & 2(\epsilon_y\epsilon_z - \eta\epsilon_x) \\ 2(\epsilon_x\epsilon_z - \eta\epsilon_y) & 2(\epsilon_y\epsilon_z + \eta\epsilon_x) & 2(\eta^2+\epsilon_z^2)-1 \end{bmatrix}$
Unit Quaternion: Inverse Problem	$\eta = \frac{1}{2}\sqrt{r_{11}+r_{22}+r_{33}+1}$ $\boldsymbol{\epsilon} = \frac{1}{2}\begin{bmatrix} \mathrm{sgn}(r_{32}-r_{23})\sqrt{r_{11}-r_{22}-r_{33}+1} \\ \mathrm{sgn}(r_{13}-r_{31})\sqrt{r_{22}-r_{33}-r_{11}+1} \\ \mathrm{sgn}(r_{21}-r_{12})\sqrt{r_{33}-r_{11}-r_{22}+1} \end{bmatrix}$
DH Convention	
Transformation Matrix	${}^{i-1}\mathbf{T}_i = \begin{bmatrix} c_{\delta_{j,i}} & -s_{\delta_{j,i}} c_{\lambda_{j,i}} & s_{\delta_{j,i}} s_{\lambda_{j,i}} & l_{j,i} c_{\delta_{j,i}} \\ s_{\delta_{j,i}} & c_{\delta_{j,i}} c_{\lambda_{j,i}} & -c_{\delta_{j,i}} s_{\lambda_{j,i}} & l_{j,i} s_{\delta_{j,i}} \\ 0 & s_{\lambda_{j,i}} & c_{\lambda_{j,i}} & d_{j,i} \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Inverse Kinematics	
Matrix Inverse	$\lvert\mathbf{R}\rvert = r_{11}(r_{22}r_{33} - r_{23}r_{32}) - r_{21}(r_{12}r_{33} - r_{13}r_{32}) + r_{31}(r_{12}r_{23} - r_{13}r_{22})$ $\mathbf{R}^{-1} = \frac{1}{\lvert\mathbf{R}\rvert}\begin{bmatrix} r_{22}r_{33} - r_{23}r_{32} & r_{13}r_{32} - r_{12}r_{33} & r_{12}r_{23} - r_{13}r_{22} \\ r_{23}r_{31} - r_{21}r_{33} & r_{11}r_{33} - r_{13}r_{31} & r_{13}r_{21} - r_{11}r_{23} \\ r_{21}r_{32} - r_{22}r_{31} & r_{12}r_{31} - r_{11}r_{32} & r_{11}r_{22} - r_{12}r_{21} \end{bmatrix}$
Differential Kinematics	
Geometric Jacobian of a Prismatic Joint	$\begin{bmatrix} \mathbf{j}_{GPi} \\ \mathbf{j}_{GOi} \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{e}_{z,{}^{i-1}o} \\ \mathbf{0} \end{bmatrix}$
Geometric Jacobian of a Revolute Joint	$\begin{bmatrix} \mathbf{j}_{GPi} \\ \mathbf{j}_{GOi} \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{e}_{z,{}^{i-1}o} \times \left(\mathbf{p}_E - {}^0\mathbf{r}_{{}^{i-1}o,{}^0o}\right) \\ {}^0\mathbf{e}_{z,{}^{i-1}o} \end{bmatrix}$
Angular Velocity	${}^0\boldsymbol{\omega}_i = {}^0\boldsymbol{\omega}_{i-1} + (1-\sigma_i)\,{}^0\mathbf{R}_{i-1}[0 \quad 0 \quad \dot{q}_i]^T$
Linear Velocity	${}^0\dot{\mathbf{r}}_{{}^io,{}^0o} = {}^0\boldsymbol{\omega}_i \times {}^0\mathbf{r}_{{}^io,{}^{i-1}o} + {}^0\dot{\mathbf{r}}_{{}^{i-1}o,{}^0o} + \sigma_i\,{}^0\mathbf{R}_{i-1}[0 \quad 0 \quad \dot{q}_i]^T$
Skew Matrix	$\mathbf{S}_E(t) = \dot{\mathbf{R}}_E(t)\mathbf{R}_E^T(t) = \begin{bmatrix} 0 & -\omega_z(t) & \omega_y(t) \\ \omega_z(t) & 0 & -\omega_x(t) \\ -\omega_y(t) & \omega_x(t) & 0 \end{bmatrix}$

Inverse Differential Kinematics	
Pseudo Inverse	$\mathbf{J}^{\dagger} = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$
Inverse Velocities	$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{x}}_{\mathrm{E}} + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\dot{\mathbf{q}}_{0}$, with $\dot{\mathbf{q}}_0 = \mathrm{k}_0 \left(\frac{\partial \mathrm{w}(\mathrm{q})}{\partial \mathrm{q}}\right)^{\mathrm{T}}$

Dynamics		
Position and Orientation of Geometric Jacobi Matrices for Link i	$\begin{bmatrix} \mathbf{J}_{GP_{link_i}} \\ \mathbf{J}_{GO_{link_i}} \end{bmatrix} = \begin{bmatrix} \mathbf{j}_{GP1_{link_i}} & \cdots & \mathbf{j}_{GPi_{link_i}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{j}_{GO1_{link_i}} & \cdots & \mathbf{j}_{GOi_{link_i}} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$ $\mathbf{j}_{GPj_{link_i}} = \begin{cases} {}^0\mathbf{e}_{z,{}^{j-1}o} & \text{Prismatic Joint} \\ {}^0\mathbf{e}_{z,{}^{j-1}o} \times \left({}^0\mathbf{r}_{CG_{link_i},{}^0o} - {}^0\mathbf{r}_{{}^{j-1}o,{}^0o}\right) & \text{Revolute Joint} \end{cases}$ $\mathbf{j}_{GOj_{link_i}} = \begin{cases} \mathbf{0} & \text{Prismatic Joint} \\ {}^0\mathbf{e}_{z,{}^{j-1}o} & \text{Revolute Joint} \end{cases}$	
Angular velocity of frame i	${}^i\boldsymbol{\omega}_i = {}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\boldsymbol{\omega}_{i-1}$	Prismatic
	${}^i\boldsymbol{\omega}_i = {}^{i-1}\mathbf{R}_i^T\left({}^{i-1}\boldsymbol{\omega}_{i-1} + \dot{\delta}_i\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o}\right)$	Revolute
Angular acceleration of frame i	${}^i\dot{\boldsymbol{\omega}}_i = {}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\dot{\boldsymbol{\omega}}_{i-1}$	Prismatic
	${}^i\dot{\boldsymbol{\omega}}_i = {}^{i-1}\mathbf{R}_i^T\left({}^{i-1}\dot{\boldsymbol{\omega}}_{i-1} + \ddot{\delta}_i\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o} + \dot{\delta}_i\,{}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{e}_{z,{}^{i-1}o}\right)$	Revolute
Linear acceleration of frame i	${}^i\ddot{\mathbf{r}}_{{}^io,{}^0o} = {}^{i-1}\mathbf{R}_i^T\left({}^{i-1}\ddot{\mathbf{r}}_{{}^{i-1}o,{}^0o} + \ddot{d}_i\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o}\right) + 2\dot{d}_i\,{}^i\boldsymbol{\omega}_i \times \left({}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o}\right) + {}^i\dot{\boldsymbol{\omega}}_i \times {}^i\mathbf{r}_{{}^io,{}^{i-1}o} + {}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{r}_{{}^io,{}^{i-1}o}\right)$	Prismatic
	${}^i\ddot{\mathbf{r}}_{{}^io,{}^0o} = {}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\ddot{\mathbf{r}}_{{}^{i-1}o,{}^0o} + {}^i\dot{\boldsymbol{\omega}}_i \times {}^i\mathbf{r}_{{}^io,{}^{i-1}o} + {}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{r}_{{}^io,{}^{i-1}o}\right)$	Revolute
Linear acceleration of CG of the augmented link i	${}^i\ddot{\mathbf{r}}_{CG_i,{}^0o} = {}^i\ddot{\mathbf{r}}_{{}^io,{}^0o} + {}^i\dot{\boldsymbol{\omega}}_i \times {}^i\mathbf{r}_{CG_i,{}^io} + {}^i\boldsymbol{\omega}_i \times \left({}^i\boldsymbol{\omega}_i \times {}^i\mathbf{r}_{CG_i,{}^io}\right)$	
Angular acceleration of motor i	${}^{i-1}\dot{\boldsymbol{\omega}}_{mot_i} = {}^{i-1}\dot{\boldsymbol{\omega}}_{i-1} + i_{mot_i}\ddot{q}_i\,{}^{i-1}\mathbf{e}_{z,mot_i} + i_{mot_i}\dot{q}_i\,{}^{i-1}\boldsymbol{\omega}_{i-1} \times {}^{i-1}\mathbf{e}_{z,mot_i}$	
Forces on frame i	${}^i\mathbf{f}_{i,i-1} = {}^i\mathbf{R}_{i+1}\,{}^{i+1}\mathbf{f}_{i+1,i} + m_i\,{}^i\ddot{\mathbf{r}}_{CG_i,{}^0o}$	
Moments on frame i	${}^i\boldsymbol{\mu}_{i,i-1} = -{}^i\mathbf{f}_{i,i-1} \times \left({}^i\mathbf{r}_{{}^io,{}^{i-1}o} + {}^i\mathbf{r}_{CG_i,{}^io}\right) + {}^i\mathbf{R}_{i+1}\,{}^{i+1}\boldsymbol{\mu}_{i+1} + \left({}^i\mathbf{R}_{i+1}\,{}^{i+1}\mathbf{f}_{i+1,i}\right) \times {}^i\mathbf{r}_{CG_i,{}^io} + {}^i\bar{\mathbf{I}}_i\,{}^i\dot{\boldsymbol{\omega}}_i + {}^i\boldsymbol{\omega}_i \times \left({}^i\bar{\mathbf{I}}_i\,{}^i\boldsymbol{\omega}_i\right) + i_{mot_{i+1}}\ddot{q}_{i+1}I_{mot_{i+1}}\,{}^i\mathbf{e}_{z,mot_{i+1}} + i_{mot_{i+1}}\dot{q}_{i+1}I_{mot_{i+1}}\,{}^i\boldsymbol{\omega}_i \times {}^i\mathbf{e}_{z,mot_{i+1}}$	
Torque joint i	$\tau_i = {}^i\mathbf{f}_{i,i-1}^T\,{}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o} + i_{mot_i}I_{mot_i}\,{}^{i-1}\dot{\boldsymbol{\omega}}_{mot_i}^T\,{}^{i-1}\mathbf{e}_{z,mot_i} + c_{vi}\dot{d}_i + F_{di}\,\mathrm{sgn}(\dot{d}_i)$	Prismatic
	$\tau_i = {}^i\boldsymbol{\mu}_{i,i-1}^T\,{}^{i-1}\mathbf{R}_i^T\,{}^{i-1}\mathbf{e}_{z,{}^{i-1}o} + i_{mot_i}I_{mot_i}\,{}^{i-1}\dot{\boldsymbol{\omega}}_{mot_i}^T\,{}^{i-1}\mathbf{e}_{z,mot_i} + c_{\omega i}\dot{\delta}_i + T_{di}\,\mathrm{sgn}(\dot{\delta}_i)$	Revolute

Task: Coating Handling
Joining Testing

⊛ Gimbal lock: axes of 2/3 are parallel
The first and last axis coincide when the second is $\pm\frac{\pi}{2}$

Structure
- Serial kinematics topology — large workspace
- +
- Parallel — small drive ⇒ high payload, precision, stiffness
- +
- Hybrid

Homogeneous

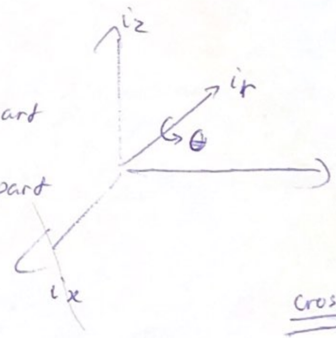
Unit ~~quarte~~ quaternion

$$Q = \{\eta, \varepsilon\}$$

$\eta = \cos\frac{\theta}{2}$ scalar part

$\varepsilon = \sin\frac{\theta}{2} r$ vector part

$$\eta^2 + \varepsilon_x^2 + \varepsilon_y^2 + \varepsilon_z^2 = 1$$



cross product

$$Q_1 Q_2 = \{\eta_1\eta_2 - \varepsilon_1^T\varepsilon_2,\ \eta_1\varepsilon_2 + \eta_2\varepsilon_1 + \varepsilon_1 \times \varepsilon_2\}$$

$$c = a \times b = \begin{bmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}$$

$$(a_2 b_3 - a_3 b_2)\vec{i} + \dots$$

$$R(\eta, \varepsilon) = \begin{bmatrix} 2(\eta^2 + \varepsilon_x^2) - 1 & 2(\varepsilon_x\varepsilon_y - \eta\varepsilon_z) & 2(\varepsilon_x\varepsilon_z + \eta\varepsilon_y) \\ 2(\varepsilon_x\varepsilon_y + \eta\varepsilon_z) & 2(\eta^2 + \varepsilon_y^2) - 1 & 2(\varepsilon_y\varepsilon_z + \eta\varepsilon_x) \\ 2(\varepsilon_x\varepsilon_z - \eta\varepsilon_y) & 2(\varepsilon_y\varepsilon_z - \eta\varepsilon_x) & 2(\eta^2 + \varepsilon_z^2) - 1 \end{bmatrix}$$

Books: Robotics Modelling, Planning & Control — Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo

Handbooks of Robotics — Siciliano & Khatib

Laws of Robotics by Asimov

0, A robot may not injure humanity or through inaction, allow humanity to come to harm

1, A robot may not injure a human being or ______, allow a human being to come to harm, unless this would violate a higher order law

2, A robot must obey orders given to it by human beings, except where such orders would conflict with a higher order law

3, ______ protect its own existance as long as such protection.. does not conflict with a higher order law

---

World Robotics Market: grows rate 2017 30%, 2018 10%, avg 14%

4D environments. dangerous, dirty, difficult, dull

4A tasks automation, augmentation, autonomous, assistance

In Europe 50 Million industry workers

500.000 Robots 1%

each years + 70.000 units

5 years ⇒ 2%

ARKaD

L1) Intro, Annotation $\quad {}^{i}r_{j_0, i_0}$

L2) Position & Orientation $\quad {}^{j}R_i = ({}^{i}R_j)^T = ({}^{i}R_j)^{-1}$

Euler $zy'z''(\varphi, \theta, \psi)$

4 quadrant inverse tangent atan2

⊗ Gimbal lock — happen when doing $\boxed{\text{inverse}}$

Angle and axis (singularity when $\sin\theta = 0$)

$$r = \frac{1}{2\sin\theta}\begin{bmatrix} \vdots \end{bmatrix}$$

⊗ Quarternion

$\eta \geq 0$, $\theta \in [-\pi, \pi]$ can describe any rotation without ambiguities!

L3) Direct Kinematics: Homogenous Transformation

$${}^{i}\tilde{r}_{p,i_0} = \begin{bmatrix} {}^{i}r_{p,i_0} \\ 1 \end{bmatrix}; \quad {}^{i}T_j = \begin{bmatrix} {}^{i}R_j & {}^{i}r_{j_0,i_0} \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow {}^{j}T_i = {}^{i}T_j^{-1} = \begin{bmatrix} {}^{i}R_j^T & -{}^{i}R_j^T \, {}^{i}r_{j_0 i_0} \\ 0 & 1 \end{bmatrix}$$

⊗ EE - Tool center Point (TCP)

$[r_1 \dots r_m]^T, m \leq 6$ $\quad$ Operation Space - Task level - EE pose

$[q_1 \dots q_n]^T, n = DOF$ $\quad$ Joint Space - Execution level -

DH - Convention (z axis first)

L4) Inverse Kinematics: Consider $\begin{cases} \text{Solution existance} \\ \text{Numbers of solution} \\ \text{Computation method} \end{cases}$

Analytical Solution: $N_q < N_r$, 3 joint axes intersects / parallel

Numerical ——: $N_q > N_r$ (redundancy) or closed form sol $\begin{cases} \text{doesnot exist} \\ \text{too hard to find analytical} \end{cases}$

(desired) $r_d = [x_d \ y_d \ z_d \ \varphi_d \ \theta_d \ \psi_d]^T \Rightarrow e = r_d - f(q_e)$

$q_e = [\delta_1, \delta_2 \dots \delta_n]^T$

$\Rightarrow q_{k+1} = q_k + \underset{\text{Gradient}}{\alpha} J_r^{-1}(q) e_k$ (or J_r^{+})

Algebraic Solution

Solve polynomial equations

L5) Differential Kinematics $\quad$ Geometric Jacobian

$$J_G(q) = \begin{bmatrix} J_{GP}(q) \\ J_{GO}(q) \end{bmatrix} \begin{matrix} \text{position} \\ \text{orientation} \end{matrix} \qquad v_E = \begin{bmatrix} \dot{p}_E \\ \omega_E \end{bmatrix} = J_G(q)\dot{q}$$

Kinematics Singularities

Analysis of Redundancy

Newton's method

$$q_{k+1} = q_k - err_k'^{-1} err_k$$

RSE

L1) History

Laws of Robotics by Asimov

World Robotics Market

4D & 4A
environment $\quad$ tasks

~~Dangerous~~ ~~D..~~ Evolution of robots

2 Kinematics structure (+1) topology

Componentes

L2) Robot in Application

Space, food, Medical

Home cleaning, mobile

Intelligent Vehicles,

DOF Formular

Type of Joints ⊗

L3)

⊗ Work space $<$ Reachable / Dexterous

Kin. Redundancy $<$ Intrinsic / Functional

DH. Conventions: Common normal

L4) Identification of Serial Kinematics

Rotation CBA, Trans ZYX

Intersection Distance (int^0, ^{1}) ⊗

L5) None

Gradient:

$$q_{k+1} = q_k + \alpha J_A^T(q_k) e_k$$