Lecture 20 - Numerical IK

- Last time: Newton's method for IK

$$\theta^{(i+1)} = \theta^{(i)} + \left({}^{\circ}J_t^{N}(\theta^{(i)})\right)^{-1}\left[\,{}^{\circ}p_t^{d} - {}^{\circ}p_t(\theta^{(i)})\,\right]$$

$e^{(i)}$

$\Delta\theta$: Newton Step

- Issues with Newton's method
  1. It doesn't guarantee $\|e^{(i+1)}\| < \|e^{(i)}\|$
  2. It can get stuck at points of zero slope in 1D
     - More generally it has issue when J^{-1} doesn't exist

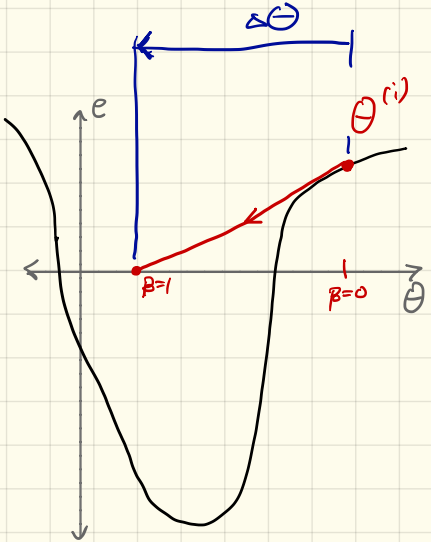
Improvement 1: Step size selection

$0 < \beta \leq 1$ "Step size"

① Consider taking a fraction of the Newton Step: $\theta^{(i+1)} = \theta^{(i)} + \beta \Delta\theta$

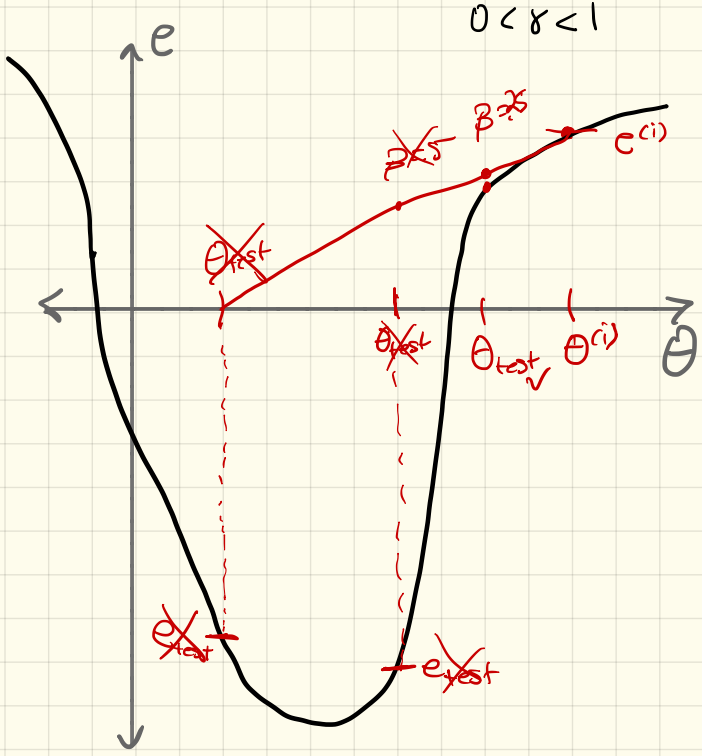
② Observation: If $e(\theta)$ were linear $\underbrace{\|e^{(i)}\| - \|e^{(i+1)}\|}_{\text{reduction of Error}} = \underbrace{\beta \|e^{(i)}\|}_{\text{Expected from linear approx.}}$

③ In practice $e(\theta)$ is nonlinear
we'll be happy if the error is reduced by a fraction γ of what the linearization predicts:



goal:

$$\|e^{(i)}\| - \|e^{(i+1)}\| \geq \gamma \beta \|e^{(i)}\|$$



goal: $\|e^{(i)}\| - \|e^{(i+1)}\| \geq \beta\gamma\|e^{(i)}\|$

Pseudocode: IK Step:

Inputs: $\theta^{(i)}$, p^d

Outputs: $\theta^{(i+1)}$

① $e^{(i)} = p^d - p(\theta^{(i)})$

② $\Delta\theta = J^{-1}(\theta^{(i)})\, e^{(i)}$

③ $\theta_{test} = \theta^{(i)} + \beta\Delta\theta \quad \beta = 1$

reduction $= \|e^{(i)}\| - \|e_{test}\|$

while reduction $< \gamma\beta\|e^{(i)}\|$

$\beta = \beta/2$

$\theta_{test} = \theta^{(i)} + \beta\Delta\theta$

reduction $= \|e^{(i)}\| - \|e_{test}\|$

end

$\theta^{(i+1)} = \theta_{test}$

Challenge In IK: Jacobian Singularities

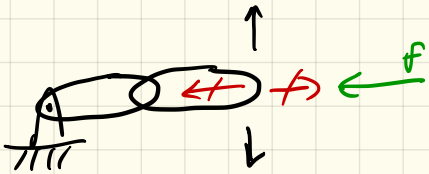
$${}^0p_t = \begin{bmatrix} l_1 c_1 + l_2 c_{12} \\ l_1 s_1 + l_2 s_{12} \end{bmatrix} \qquad {}^0v_t = {}^0J_t^s \dot{\theta} \qquad {}^0J_t^s = \begin{bmatrix} l_1 s_1 + l_2 s_{12} & l_2 s_{12} \\ -l_1 c_1 - l_2 c_{12} & -l_2 c_{12} \end{bmatrix} \Bigg|_{\theta_1=0,\theta_2=0}$$

$$= \begin{bmatrix} 0 & 0 \\ -l_1 - l_2 & -l_2 \end{bmatrix}$$

- What happens when $\theta_2 = 0$?
  - End effector cannot move radially!

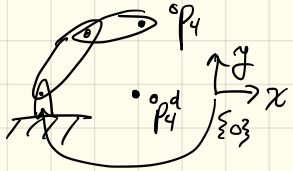
$$\det({}^0J_t^v) = 0 \; ! \quad \Rightarrow \text{No matrix inverse} \; \therefore$$

- For any solution to $\det(J(\theta)) = 0$ — θ is singular configuration



No torque needed to resist f

Improvement #2: Singularity Robustness



- Singularity-Robust Pseudo-inverse

$$J^{-1} \approx J^T \left(JJ^T + \lambda I\right)^{-1} \qquad \text{choose } \lambda \text{ small}$$

when $\lambda = 0$

$$J^T (JJ^T)^{-1} = \cancel{J^T J^{-T}} J^{-1} = J^{-1}$$

- Small change

$$\Delta\theta^{(i)} = J^T \left(JJ^T + \lambda I\right)^{-1} e^{(i)}$$

(Also works when you have more joint variables than task variables)

Including Orientation

Consider an Angle Axis Rotation $\phi, {}^{\circ}\hat{k}$

- ${}^{\circ}R_1(\phi) = R_{{}^{\circ}\hat{k}}(\phi)$ Same as $\{1\}$ rotating at a rate of 1 rad/s with ${}^{\circ}\omega_1 = {}^{\circ}\hat{k}$ for ϕ seconds

- $$\frac{d\,{}^{\circ}R_1(\phi)}{d\phi} = \left[{}^{\circ}\hat{k} \times {}^{\circ}\hat{x}_1 \quad {}^{\circ}\hat{k} \times {}^{\circ}\hat{y}_1 \quad {}^{\circ}k \times {}^{\circ}\hat{z}_1 \right] = S({}^{\circ}\hat{k})\,{}^{\circ}R_1(\phi)$$

$${}^{\circ}R_1 = e^{S({}^{\circ}\hat{k})\phi} \quad \left[\text{From linear systems if Diff eq} \right]$$

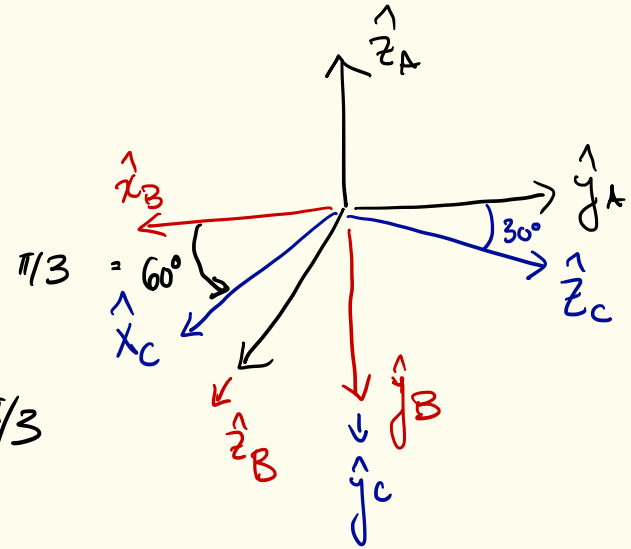
- "log" and "exp" are inveses $\log(e^{\alpha}) = \alpha$. There is also a matrix log, its the inverse of matrix exponential.

MATLAB Commands for Example:

```
% Function for cross-product matrix
S = @(x) [0 -x(3) x(2) ; x(3) 0 -x(1) ; -x(2) x(1) 0];

% Cross product matrix
Smat = S([.1 .2 .3]')

% Rotation matrix from matrix exponential
R = expm(Smat)

% Make sure it's legit
det(R)
R'*R

% Should return the cross product matrix since log and exp are inverse functions
logm(R)
```

Example:

$${}^A R_C = e^{S({}^A\hat{k})\theta}\, {}^A R_B$$



Find: $S({}^A\hat{k})\theta$ $\quad {}^A\hat{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ $\quad \theta = \pi/3$

$$S({}^A\hat{k})\theta = \begin{bmatrix} 0 & -\pi/3 & 0 \\ \pi/3 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

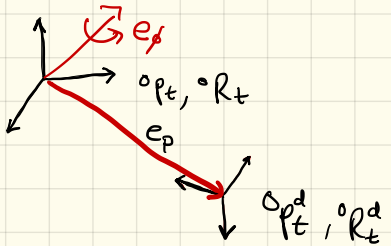
$$e^{S({}^A\hat{k})\theta} = {}^A R_C\, {}^A R_B^T$$

$$S\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}$$

MATLAB Commands for Example:

```
aRb = [0 -1 0 ; 0 0 -1 ; 1 0 0]'

aRc = [sqrt(3)/2 -1/2 0; 0 0 -1 ; 1/2  sqrt(3)/2 0 ]'

% Should give S([0 ; 0 ; 1]) * pi/3
logm(aRc*aRb')
```

Including Orientation



$$e_p = {}^0p_t^d - {}^0p_t(\theta^{(i)})$$

$${}^0R_t^d = e^{S({}^0\hat{R})\phi}\,{}^0R_t(\theta^{(i)})$$

$$S({}^0\hat{R})\phi = \log m\left({}^0R_t^d\;{}^0R_t^T(\theta^{(i)})\right) = \begin{bmatrix} 0 & -\phi k_z & \phi k_y \\ \phi k_z & 0 & -\phi k_x \\ -\phi k_y & \phi k_x & 0 \end{bmatrix}$$

$$e_\phi = \begin{bmatrix} k_x\phi \\ k_y\phi \\ k_z\phi \end{bmatrix}$$

(Angular velocity to follow for one second to get from ${}^0R_t$ to ${}^0R_t^d$)

$$\Delta\theta = {}^0J_t^T\left({}^0J_t\,{}^0J_t^T + \lambda I\right)^{-1}\begin{bmatrix} e_\phi \\ \text{---} \\ e_p \end{bmatrix}$$