역운동학의 구현과 응용
Implementation of Inverse Kinematics and Application

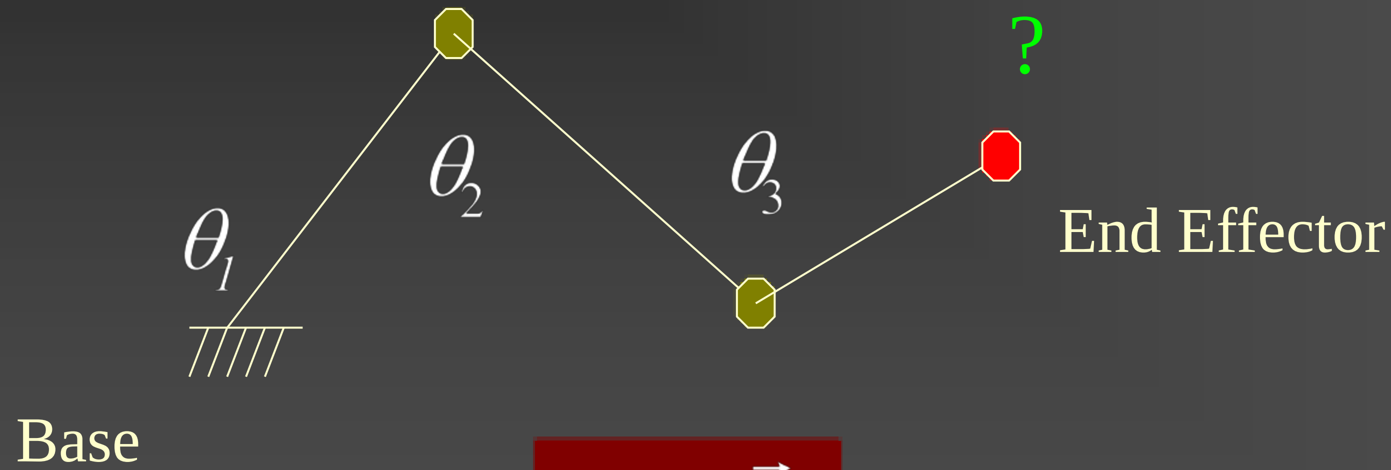
서울대학교
전기공학부
휴먼애니메이션연구단
최광진
kjchoi@graphics.snu.ac.kr

Content

- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

What is Inverse Kinematics?

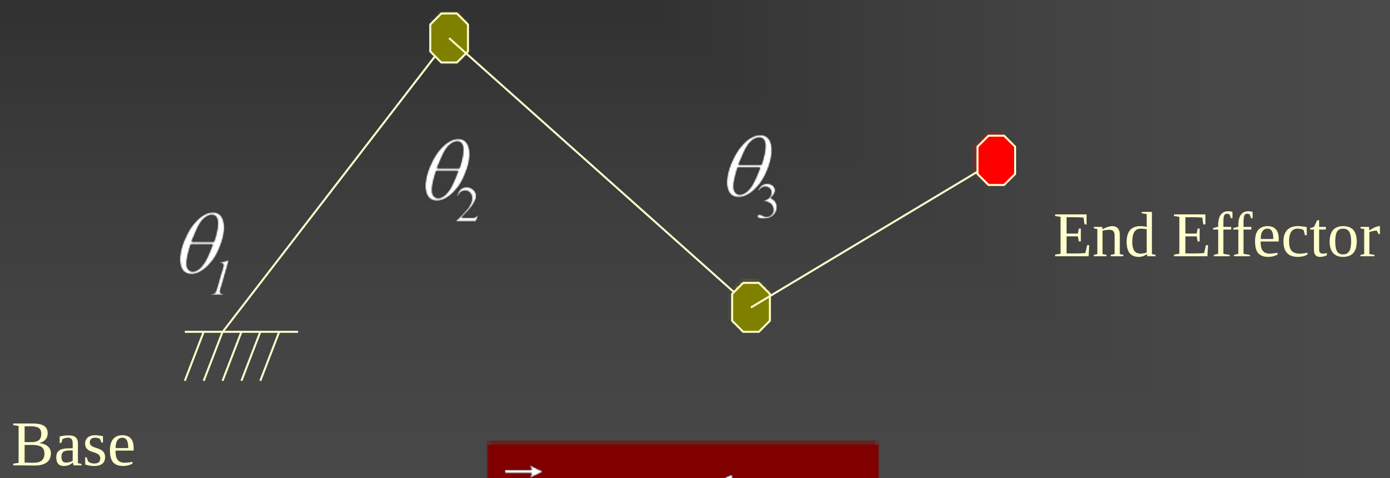
- Forward Kinematics



$$\vec{\mathbf{x}} = \mathrm{f}(\vec{\boldsymbol{\theta}})$$

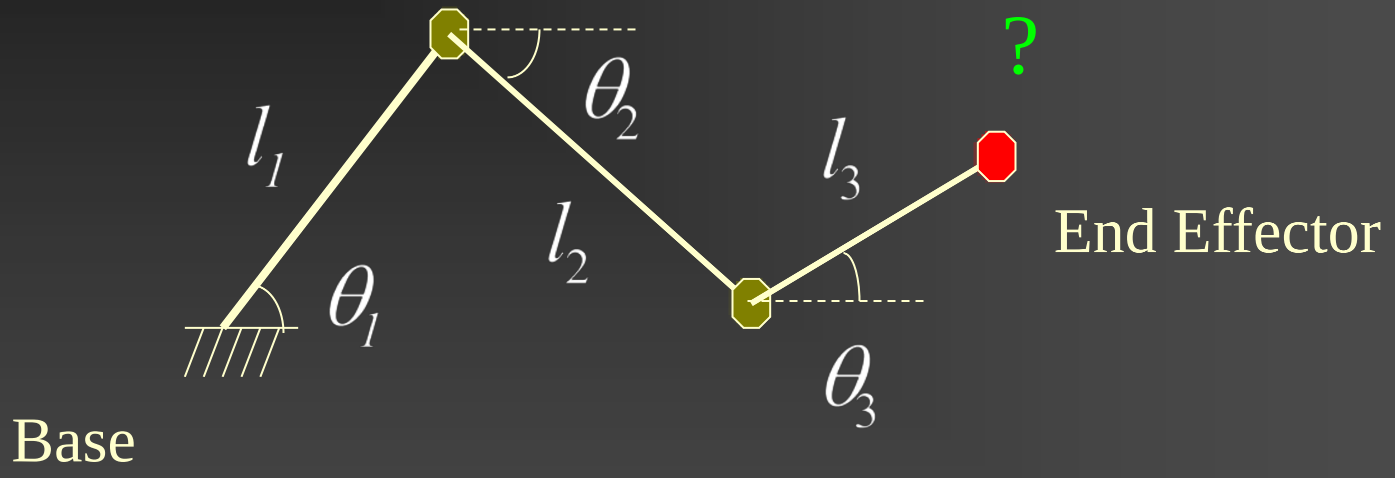
What is Inverse Kinematics?

- Inverse Kinematics



$$\vec{\boldsymbol{\theta}} = f^{-1}(\vec{\mathbf{x}})$$

What does $f(\vec{\boldsymbol{\theta}})$ looks like?



$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3)$$

$$y = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) + l_3 \sin(\theta_3)$$

Solution to $\vec{\boldsymbol{\theta}} = \mathrm{f}^{-1}(\vec{\mathbf{x}})$

- Our example

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3)$$

$$y = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) + l_3 \sin(\theta_3)$$

Number of equation : 2

Unknown variables : 3

$\Rightarrow$ ***Infinite number of solutions !***

Redundancy

- System DOF > End Effector DOF

- Our example

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3)$$

$$y = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) + l_3 \sin(\theta_3)$$

  - System DOF = 3
  - End Effector DOF = 2

Redundancy

- A redundant system has infinite number of solutions
- Human skeleton has 70 DOF
  - Ultra-super redundant
- How to solve highly redundant system?

Content

- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

What is NLP?

- Non Linear Programming
- Method to optimize a nonlinear function
  - Example

$$\textbf{minimize } \; x^2(y+1)+\sin(x+y)$$

$$\textbf{subject to } \; x \geq 0, y \geq 0$$

  - Objective function
  - Constraint
  - Iterative algorithm

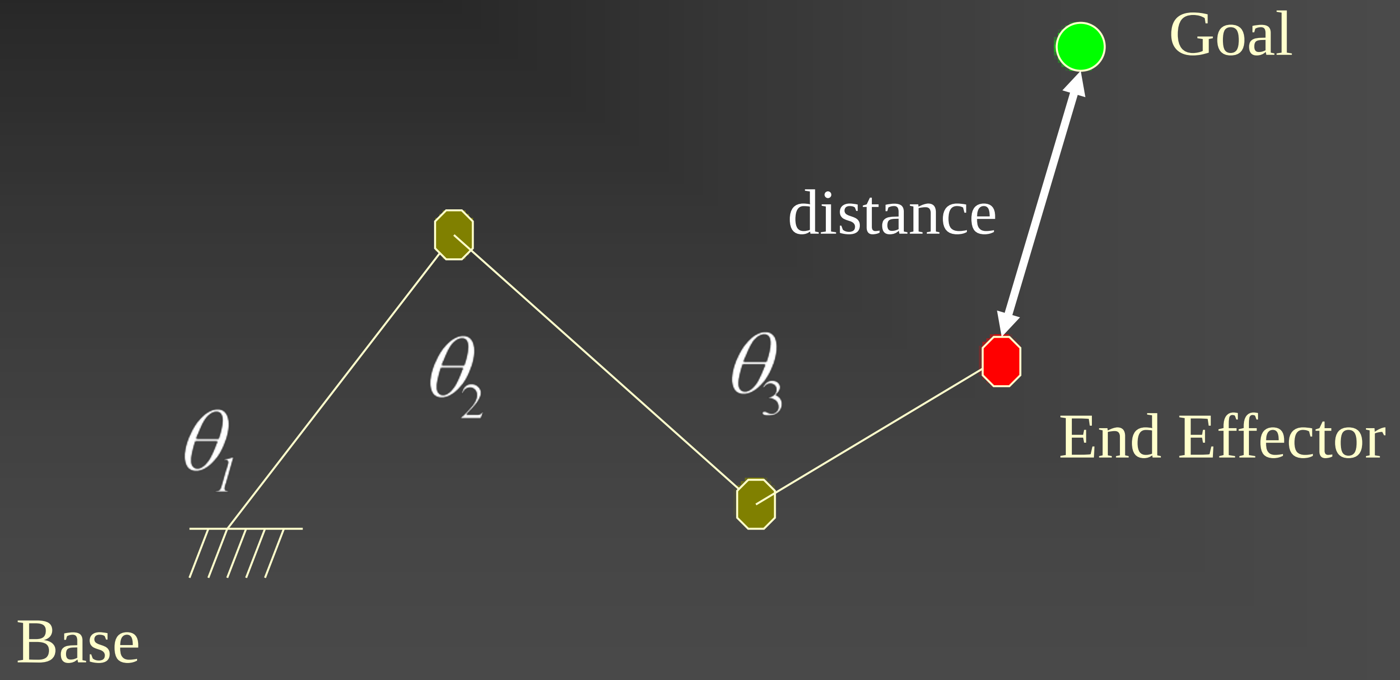
NLP-based Method

- Inverse Kinematics problem as non-linear optimization problem
  - Minimization of Goal Potential Function
- Zhao and Badler, 1994, ACMTOG

Goal Potential Function

- "Distance" from the end effector to the goal
- Function of joint angles : $G(\boldsymbol{\theta})$

Our Example



Goal Potential Function

Position Goal

$$\left\|\mathbf{p}_{\mathrm{g}}-\mathbf{p}_{\mathrm{e}}\right\|^{2}$$

Orientation Goal

$$\left\|\mathbf{r}_{\mathrm{x}}^{\mathrm{g}}-\mathbf{r}_{\mathrm{x}}^{\mathrm{e}}\right\|^{2}+\left\|\mathbf{r}_{\mathrm{y}}^{\mathrm{g}}-\mathbf{r}_{\mathrm{y}}^{\mathrm{e}}\right\|^{2}$$

Position/Orientation Goal

$$\omega\left\|\mathbf{p}_{\mathrm{g}}-\mathbf{p}_{\mathrm{e}}\right\|^{2}+(1-\omega)c\left(\left\|\mathbf{r}_{\mathrm{x}}^{\mathrm{g}}-\mathbf{r}_{\mathrm{x}}^{\mathrm{e}}\right\|^{2}+\left\|\mathbf{r}_{\mathrm{y}}^{\mathrm{g}}-\mathbf{r}_{\mathrm{y}}^{\mathrm{e}}\right\|^{2}\right)$$

Our Example

$$x = l_1 \cos\theta_1 + l_2 \cos\theta_2 + l_3 \cos\theta_3$$
$$y = l_1 \sin\theta_1 + l_2 \sin\theta_2 + l_3 \sin\theta_3$$

- Goal Potential Function

$$\begin{aligned} G(\boldsymbol{\theta}) &= (x_g - x)^2 + (y_g - y)^2 \\ &= (x_g - (l_1 \cos\theta_1 + l_2 \cos\theta_2 + l_3 \cos\theta_3))^2 \\ &\quad + (y_g - (l_1 \sin\theta_1 + l_2 \sin\theta_2 + l_3 \sin\theta_3))^2 \end{aligned}$$

Nonlinear Optimization

- Recasted Constrained Optimization Problem

$$\begin{cases} \text{minimize } G(\boldsymbol{\theta}) \\ \text{subject to } \begin{cases} \mathbf{a}^{\mathrm{T}}\boldsymbol{\theta} = \mathbf{b}_1 \\ \mathbf{a}^{\mathrm{T}}\boldsymbol{\theta} \le \mathbf{b}_2 \end{cases} \end{cases}$$

Nonlinear Optimization

- Available NLP Packages
  - LANCELOT
  - DONLP2
  - MATLAB
  - Etc…

Quiz

- Will $G(\theta)$ be always zero?
  - No : Unreachable Workspace
- Will the solution be always found?
  - No : Local Minima/Singular Configuration
- Will the solution be always unique?
  - No : Redundancy

Handling Singularity

Singular Configuration

- Causes infinite joint velocity
- Occurs when any $\dot{\theta}$ cannot achieve given $\dot{\mathbf{x}}$
- Example
  - Fully stretched limbs

Remedy

- For Jacobian-based method
  - Damped pseudo inverse

    $$\mathbf{J}^{+} = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}} + \lambda^{2}\mathbf{I})^{-1}$$

  - Dexterity measure $\delta_{\max}/\delta_{\min}$
  - Clamping
- For nonlinear optimization method
  - Try again with another initial value

Parametric Singularity

- Gimbal Lock in Euler angle representation
  - When a degree of freedom is lost, the gimbals is said to “lock”
  - Consider a y-roll of 90 degrees which aligns the x and z axis

$$R(\theta_x, 90, \theta_z) = \begin{bmatrix} 0 & 0 & -1 \\ \sin(\theta_x - \theta_z) & \cos(\theta_x - \theta_z) & 0 \\ \cos(\theta_x - \theta_z) & \sin(\theta_x - \theta_z) & 0 \end{bmatrix}$$

- Remedy : Quaternion

Content

- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

Differential Kinematics

$$\mathbf{x} = f(\boldsymbol{\theta}) \qquad \dot{\mathbf{x}} = \mathbf{J}\dot{\boldsymbol{\theta}} \qquad \mathbf{J} \equiv \frac{\partial f}{\partial \boldsymbol{\theta}}$$

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

- $\mathbf{J}$: Jacobian Matrix
  - Linearly relates end-effector change to joint angle change

Differential Kinematics

- Our Example

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f_1(\boldsymbol{\theta}) \\ f_2(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} l_1 \cos\theta_1 + l_2 \cos\theta_2 + l_3 \cos\theta \\ l_1 \sin\theta_1 + l_2 \sin\theta_2 + l_3 \sin\theta_3 \end{bmatrix} \qquad \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_1} & \dfrac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_2} & \dfrac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_3} \\ \dfrac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_1} & \dfrac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_2} & \dfrac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_3} \end{bmatrix} = \begin{bmatrix} -l_1 \sin\theta_1 & -l_2 \sin\theta & -l_3 \sin\theta_3 \\ l_1 \cos\theta_1 & l_2 \cos\theta & l_3 \cos\theta \end{bmatrix}$$

Differential Kinematics

- Is J always invertible?   No!
  - Remedy : Pseudo Inverse

$$\mathbf{J}^{+} = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$$

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}} \quad \text{(minimal norm solution)}$$

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\,\dot{\boldsymbol{\varphi}} \quad \text{(general solution)}$$

Null space

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\,\dot{\boldsymbol{\varphi}} \quad \text{(general solution)}$$

- The null space of J is the set of vectors which have no influence on the constraints

$$\theta \in nullspace(J) \Leftrightarrow J\theta = 0$$

- The pseudoinverse provides an operator which projects any vector to the null space of J

$$J\Delta\theta = \Delta x$$

$$\Delta\theta = J^{+}\Delta x + (I - J^{+}J)z$$

Utility of Null Space

- The null space can be used to reach secondary goals

$$\Delta\theta = J^{+}\Delta x + (I - J^{+}J)z$$

$$\min_{z} f(\theta)$$

- Or to find comfortable positions

$$f(\theta) = \sum_{i} (\theta_{comfort}(i) - \theta(i))^2$$

Null Space

$$\dot{\theta} = (J^+J - I)z$$

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

$$V = J(J^+J - I)z$$

$$V = (JJ^+J - J)z$$

$$V = (J - J)z$$

$$V = 0 \cdot z$$

$$V = 0$$

Calculating Pseudo Inverse

- Gaussian Elimination

$$\mathbf{J}^{+} = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$$

- Singular Value Decomposition

$$\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}$$

$$\mathbf{J}^{+} = \mathbf{V}\mathbf{S}^{-1}\mathbf{U}^{\mathrm{T}}$$

How Can We Get $\boldsymbol{\theta}(t)$ From $\mathbf{x}(t)$?

$$\boldsymbol{\theta}(t) = \int_t \dot{\boldsymbol{\theta}}(t) = \int_t \mathbf{J}^+(t)\dot{\mathbf{x}}(t)$$

Integrating $\dot{\boldsymbol{\theta}}(t) = \mathbf{J}^{+}\dot{\mathbf{x}}(t)$

- Problems
  - Initial tracking error
  - Numerical drift
- Remedy
  - Error feedback

$$\dot{\boldsymbol{\theta}}(t) = \mathbf{J}^{+}(\dot{\mathbf{x}}(t) + \mathbf{K}\mathbf{e}(t))$$

$$\mathbf{e}(t) = \mathbf{x}(t) - f(\boldsymbol{\theta}(t))$$

Discretization

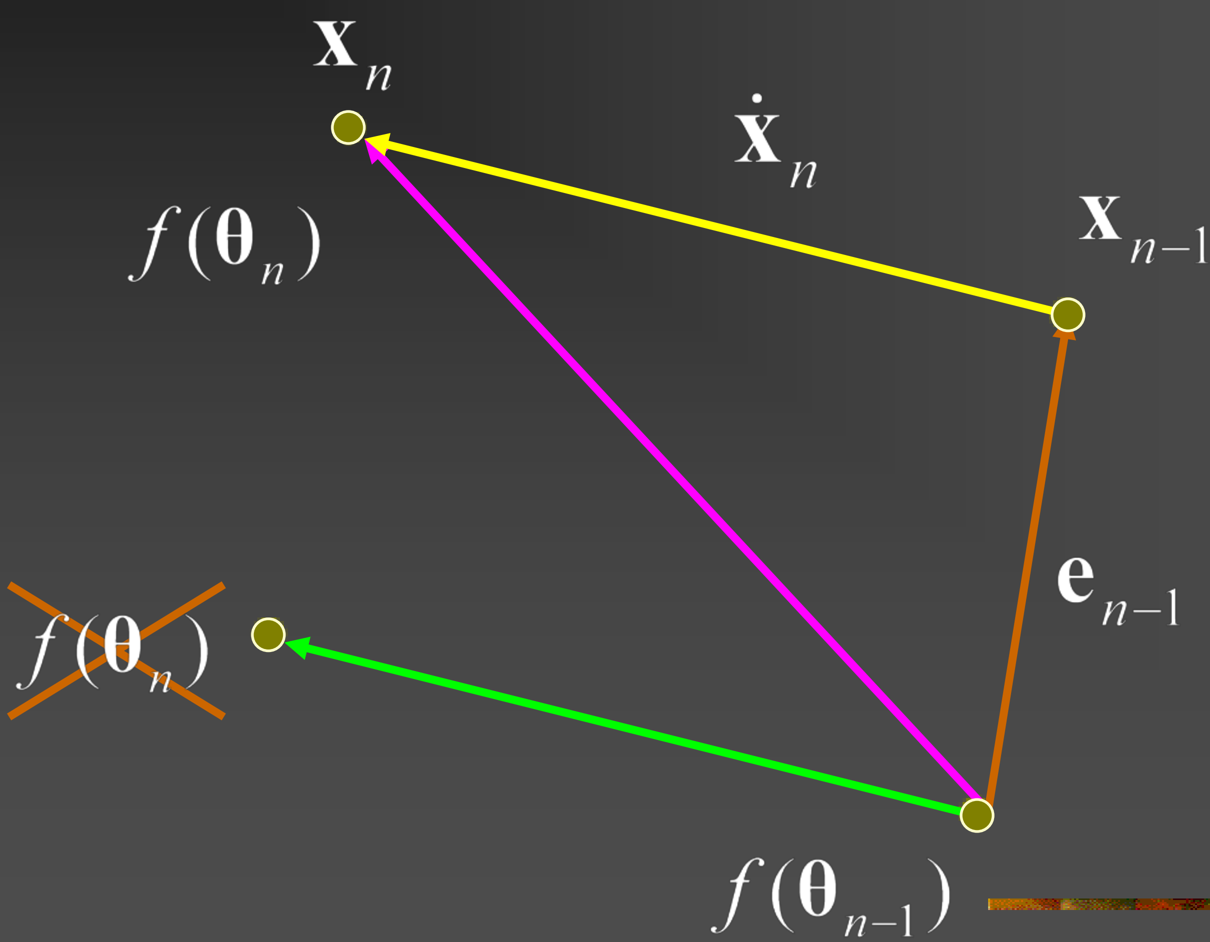
$$\dot{\boldsymbol{\theta}}(t) = \mathbf{J}^{+}(\dot{\mathbf{x}}(t) + \mathbf{K}\mathbf{e}(t))$$

$$\dot{\boldsymbol{\theta}}_n = \mathbf{J}^{+}(\dot{\mathbf{x}}_n + \mathbf{K}\mathbf{e}_n)$$

$$\Rightarrow \frac{\boldsymbol{\theta}_n - \boldsymbol{\theta}_{n-1}}{h} = \mathbf{J}^{+}\left(\frac{\mathbf{x}_n - \mathbf{x}_{n-1}}{h} + \mathbf{K}\mathbf{e}_n\right)$$

$$\Rightarrow \boldsymbol{\theta}_n - \boldsymbol{\theta}_{n-1} = \mathbf{J}^{+}(\mathbf{x}_n - f(\boldsymbol{\theta}_{n-1}))$$

Open-loop vs. Closed-loop scheme



Content

- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

Redundancy Is Evil

- Multiple choices for one goal
  - What happens if we pick any of them?

Redundancy Is Good

- We can exploit redundancy
- Additional objective
  - Minimal Change
  - Similarity to Given Example
  - Naturalness

Minimal Change

- Pseudo Inverse Solution
  - Minimal velocity norm solution

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\boldsymbol{\theta}} \;\rightarrow\; \dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}}$$

  - Minimal acceleration norm solution

$$\ddot{\mathbf{x}} = \mathbf{J}\ddot{\boldsymbol{\theta}} + \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} \;\rightarrow\; \ddot{\boldsymbol{\theta}} = \mathbf{J}^{+}(\ddot{\mathbf{x}} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}})$$

- Penalty Term in Goal Potential Function

$$G_{new}(\boldsymbol{\theta}) = G(\boldsymbol{\theta}) + \boldsymbol{\theta}^{T}\mathbf{M}_{P}\boldsymbol{\theta} + \dot{\boldsymbol{\theta}}^{T}\mathbf{M}_{V}\dot{\boldsymbol{\theta}} + \ddot{\boldsymbol{\theta}}^{T}\mathbf{M}_{A}\ddot{\boldsymbol{\theta}}$$

Similarity to Given Example

- Adding homogeneous solution term to the pseudo inverse solution

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{K}(\boldsymbol{\theta}_e - \boldsymbol{\theta})$$

- New Goal Potential Function

$$G_{new}(\boldsymbol{\theta}) = G(\boldsymbol{\theta}) + k\left\|\boldsymbol{\theta}_e - \boldsymbol{\theta}\right\|^2$$

Naturalness

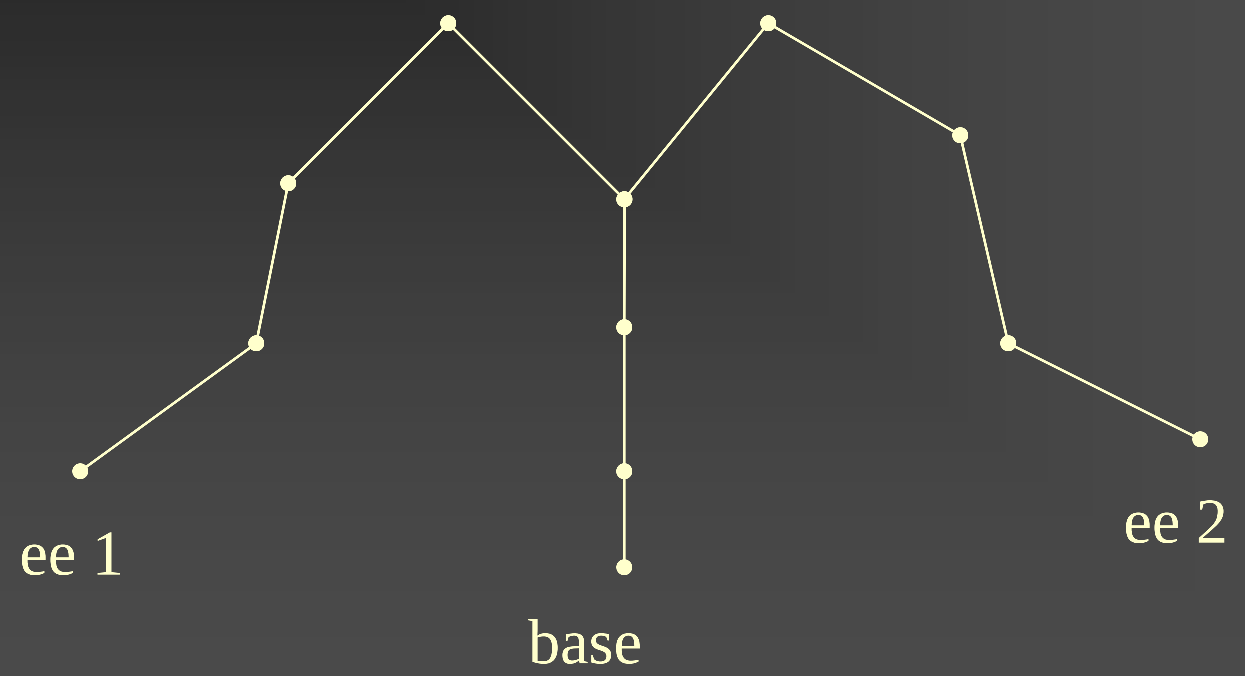
- Based on observation of natural human posture
- Neurophysiological experiments
- Example
  - Pointing task with pen stylus
    - Linear mapping between shoulder joint and pen stylus in spherical coordinate system
    - Analytic solution for human arm
  - endgame.mov

Content

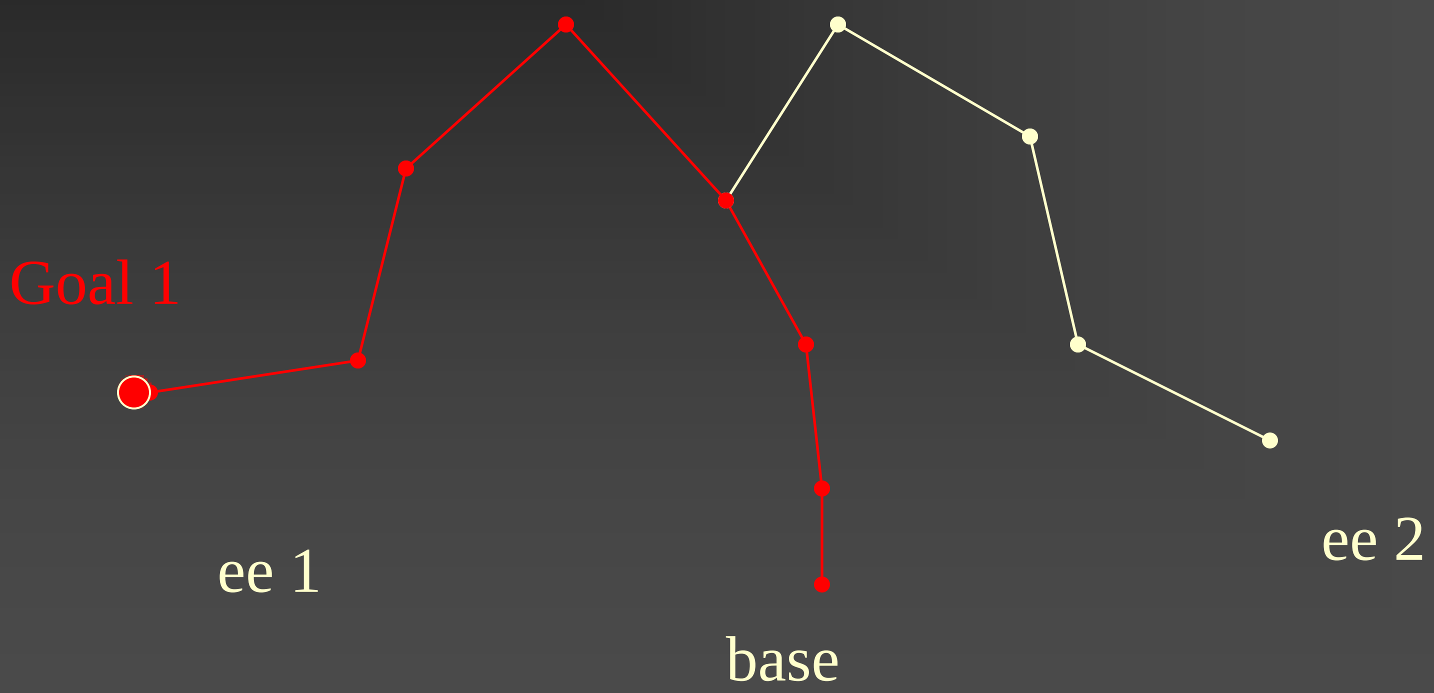
- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

Multiple Goals

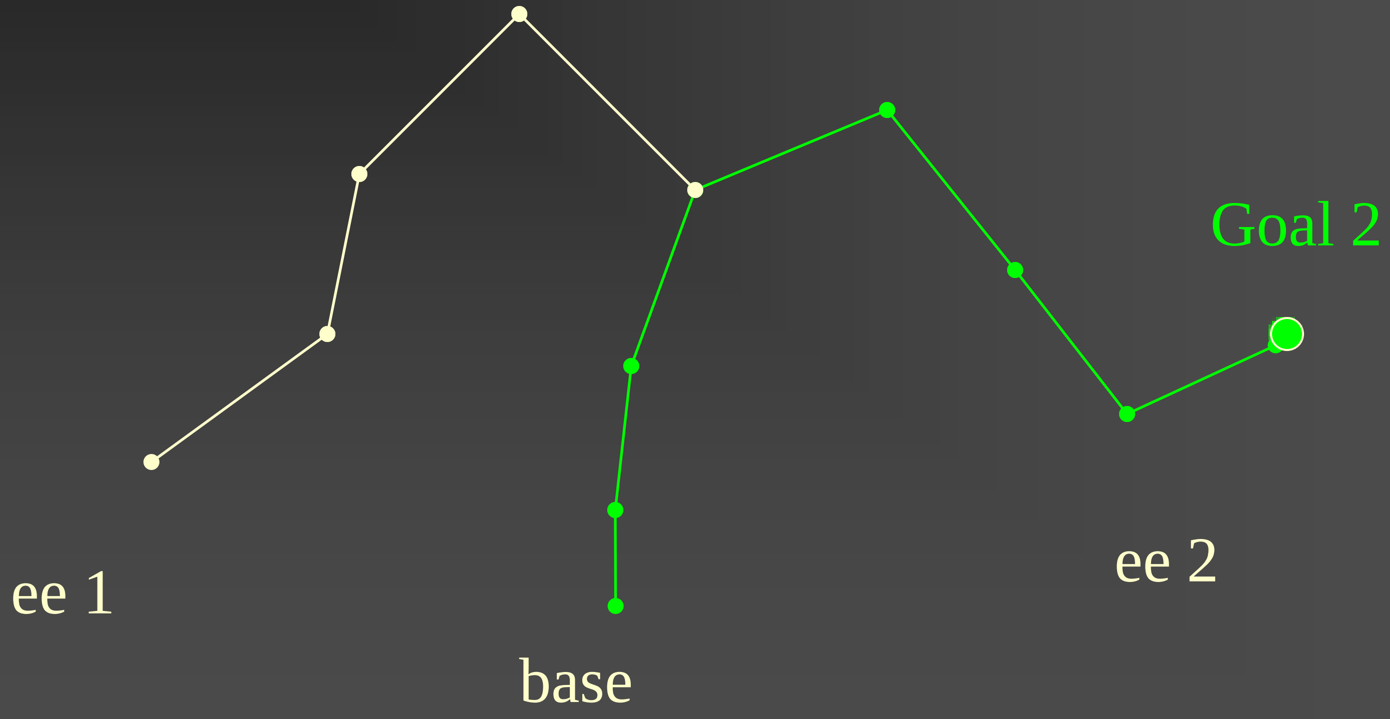
Conflict Between Goals



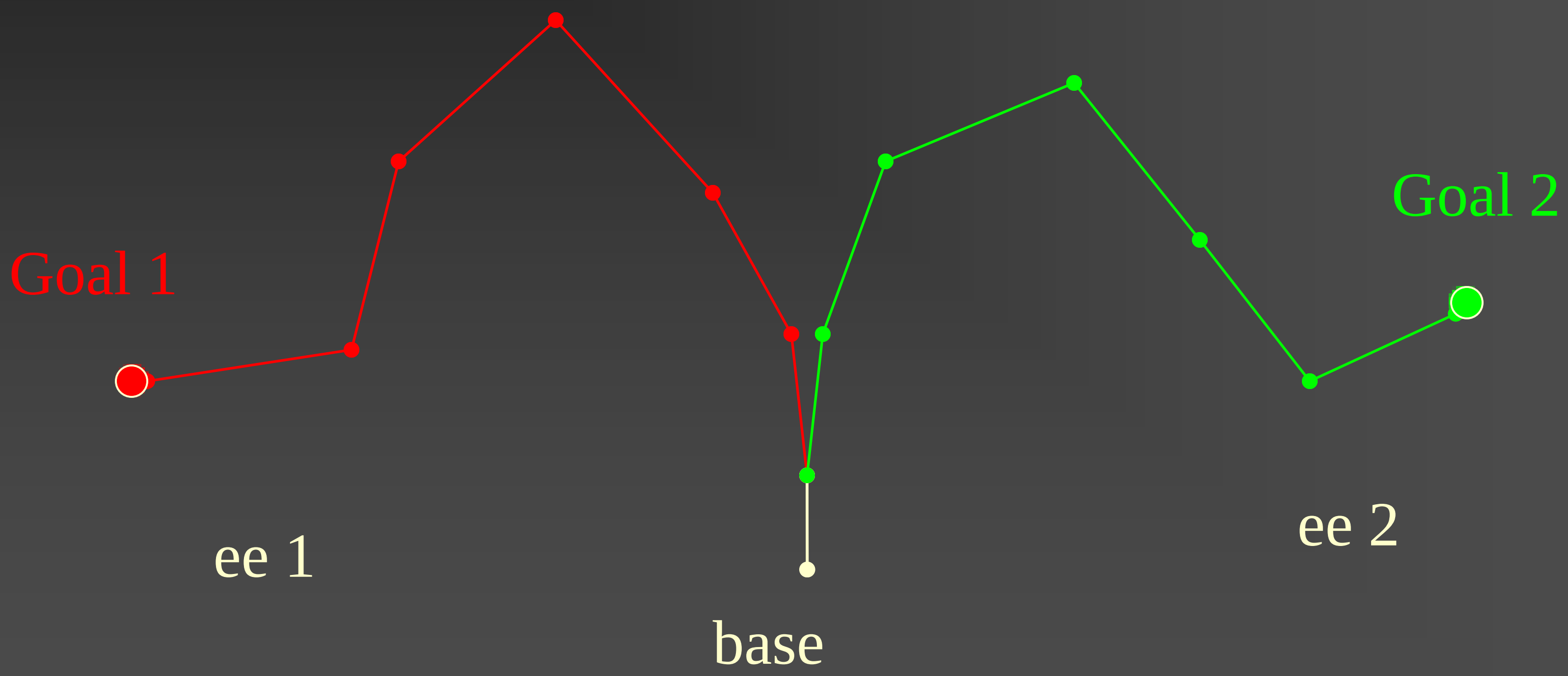
Conflict Between Goals



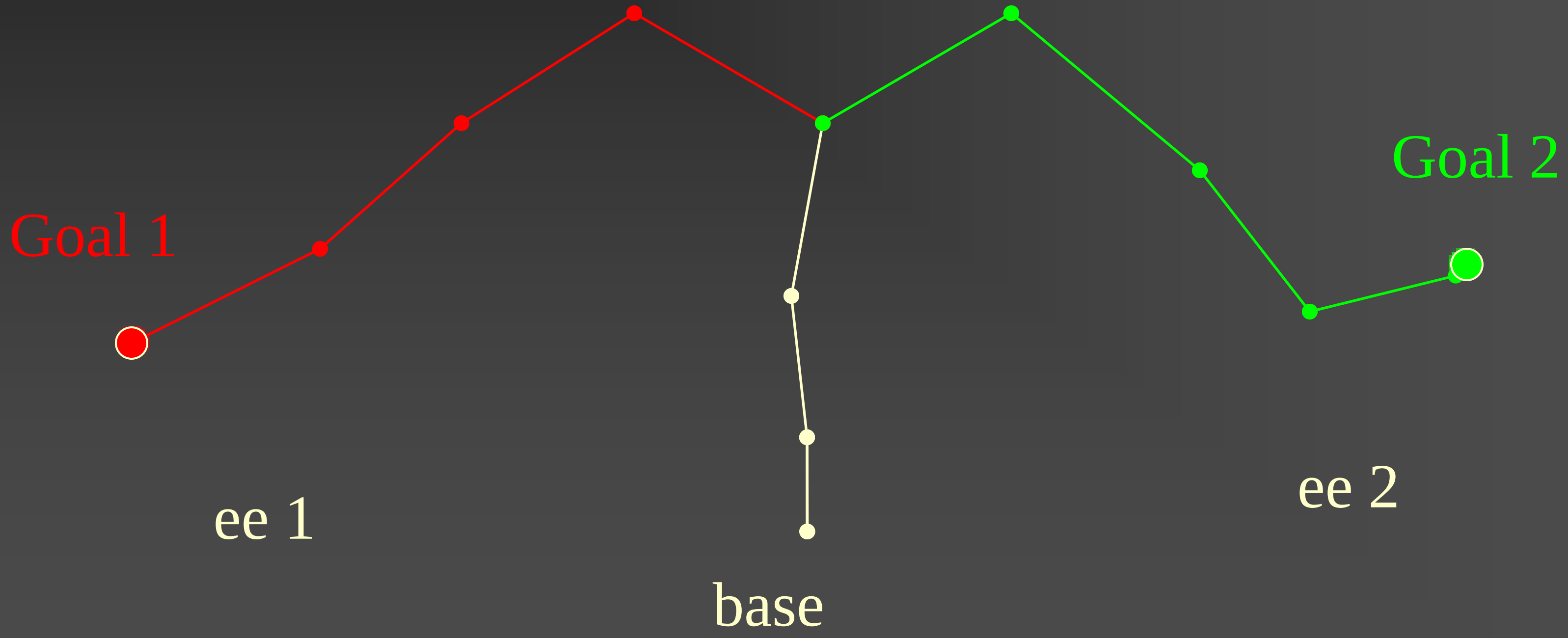
Conflict Between Goals



Conflict Between Goals



Conflict Between Goals



Handling Multiple Goals

- Weighted sum of each goal potential function

$$G^{all}(\boldsymbol{\theta}) = \sum_{i=1}^{m} \omega_i G_i(\boldsymbol{\theta})$$

- Jacobian-based method
  - Same formulation $\dot{\boldsymbol{\theta}}(t) = \mathbf{J}^{+}\dot{\mathbf{x}}(t)$
  - No weighting

Content

- What is Inverse Kinematics?
- Redundancy
- Basic Method
  - NLP-based method
  - Jacobian-based method
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Application : Motion Retargetting

Summary

- Inverse Kinematics
- Solver
  - NLP-based Solver
  - Jacobian-based Solver
- Issues
  - Resolving Redundancy
  - Multiple Goals
- Motion Retargetting

Thank You