CS-184: Computer Graphics

Lecture #17: Forward and Inverse Kinematics

Prof. James O’Brien
University of California, Berkeley
Today

- Forward kinematics
- Inverse kinematics
  - Pin joints
  - Ball joints
  - Prismatic joints
Forward Kinematics

- Articulated skeleton
  - Topology (what’s connected to what)
  - Geometric relations from joints
  - Independent of display geometry
  - Tree structure
    - Loop joints break “tree-ness”
Forward Kinematics

- **Root body**
  - Position set by “global” transformation
- **Root joint**
  - Position
  - Rotation
- **Other bodies relative to root**
  - *Inboard* toward the root
  - *Outboard* away from root
Forward Kinematics

- A joint
  - Joint’s inboard body
  - Joint’s outboard body
Forward Kinematics

- **A body**
  - Body’s inboard joint
  - Body’s outboard joint
    - May have several outboard joints
Forward Kinematics

- A body
  - Body’s inboard joint
  - Body’s outboard joint
    - May have several outboard joints
  - Body’s parent
  - Body’s child
    - May have several children
Forward Kinematics

- **Interior joints**
  - Typically not 6 DOF joints
  - Pin - rotate about one axis
  - Ball - arbitrary rotation
  - Prism - translation along one axis
Forward Kinematics

- Pin Joints
  - Translate inboard joint to local origin
  - Apply rotation about axis
  - Translate origin to location of joint on outboard body
Forward Kinematics

○ Ball Joints
  ○ Translate inboard joint to local origin
  ○ Apply rotation about *arbitrary* axis
  ○ Translate origin to location of joint on outboard body
Forward Kinematics

- **Prismatic Joints**
  - Translate inboard joint to local origin
  - Translate along axis
  - Translate origin to location of joint on outboard body
Forward Kinematics

- Composite transformations up the hierarchy
Forward Kinematics

- Composite transformations up the hierarchy
Forward Kinematics

- Composite transformations up the hierarchy
Forward Kinematics

- Composite transformations up the hierarchy
Forward Kinematics

- Composite transformations up the hierarchy
Inverse Kinematics

- **Given**
  - Root transformation
  - Initial configuration
  - Desired end point location

- **Find**
  - Interior parameter settings
Inverse Kinematics
Inverse Kinematics

- A simple two segment arm in 2D

\[ p_z = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \]
\[ p_x = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \]
Inverse Kinematics

- Direct IK: solve for the parameters

\[ \theta_2 = \cos^{-1} \left( \frac{p_z^2 + p_x^2 - l_1^2 - l_2^2}{2l_1l_2} \right) \]

\[ \theta_1 = \frac{-p_z l_2 \sin(\theta_2) + p_x (l_1 + l_2 \cos(\theta_2))}{p_x l_2 \sin(\theta_2) + p_z (l_1 + l_2 \cos(\theta_2))} \]
Inverse Kinematics

- Why is the problem hard?
  - Multiple solutions separated in configuration space
Inverse Kinematics

- Why is the problem hard?
  - Multiple solutions connected in configuration space
Inverse Kinematics

- Why is the problem hard?
  - Solutions may not always exist
Inverse Kinematics

Numerical Solution

- Start in some initial configuration
- Define an error metric (e.g. goal pos - current pos)
- Compute Jacobian of error w.r.t. inputs
- Apply Newton’s method (or other procedure)
- Iterate...
Inverse Kinematics

- Recall simple two segment arm:

\[
\begin{align*}
    p_z &= l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \\
    p_x &= l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2)
\end{align*}
\]
Inverse Kinematics

- We can write of the derivatives

\[
\frac{\partial p_z}{\partial \theta_1} = -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2)
\]

\[
\frac{\partial p_x}{\partial \theta_1} = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)
\]

\[
\frac{\partial p_z}{\partial \theta_2} = -l_2 \sin(\theta_1 + \theta_2)
\]

\[
\frac{\partial p_x}{\partial \theta_2} = +l_2 \cos(\theta_1 + \theta_2)
\]
Inverse Kinematics

Direction in Config. Space

\[ \theta_1 = c_1 \theta_* \]
\[ \theta_2 = c_2 \theta_* \]

\[ \frac{\partial p_z}{\partial \theta_*} = c_1 \frac{\partial p_z}{\partial \theta_1} + c_2 \frac{\partial p_z}{\partial \theta_2} \]
Inverse Kinematics

The Jacobian (of $p$ w.r.t. $\theta$)

$$J_{ij} = \frac{\partial p_i}{\partial \theta_j}$$

Example for two segment arm

$$J = \begin{bmatrix} \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\ \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \end{bmatrix}$$
Inverse Kinematics

The Jacobian (of $p$ w.r.t. $\theta$)

\[
J = \begin{bmatrix}
\frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\
\frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2}
\end{bmatrix}
\]

\[
\frac{\partial p}{\partial \theta^*_*} = J \cdot \begin{bmatrix}
\frac{\partial \theta_1}{\partial \theta^*_*} \\
\frac{\partial \theta_2}{\partial \theta^*_*}
\end{bmatrix} = J \cdot \begin{bmatrix} c_1 \\
c_2 \end{bmatrix}
\]
Inverse Kinematics

Solving for $c_1$ and $c_2$

\[ c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \]
\[ dp = \begin{bmatrix} dp_z \\ dp_x \end{bmatrix} \]

\[ dp = J \cdot c \]
\[ c = J^{-1} \cdot dp \]
Inverse Kinematics

Solving for $c_1$ and $c_2$

Is the Jacobian invertible?
Inverse Kinematics

- **Problems**
  - Jacobian may (will!) not always be invertible
    - Use pseudo inverse (SVD)
    - Robust iterative method
  - Jacobian is not constant
    
    \[
    J = \begin{bmatrix}
    \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\
    \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2}
    \end{bmatrix} = J(\theta)
    \]
    
    - Nonlinear optimization, but problem is (mostly) well behaved
Inverse Kinematics

- More complex systems
  - More complex joints (prism and ball)
  - More links
  - Other criteria (COM or height)
  - Hard constraints (joint limits)
  - Multiple criteria and multiple chains
Inverse Kinematics

- Some issues
  - How to pick from multiple solutions?
  - Robustness when no solutions
  - Contradictory solutions
  - Smooth interpolation
    - Interpolation aware of constraints
Inverse Kinematics

Prism Joints

\[ p_z = l_1 \]
\[ p_x = d \]

\[ p_z = l_1 + d \]
\[ p_x = 0 \]
Inverse Kinematics

Ball Joints

\[ p = \hat{r} (\hat{r} \cdot \mathbf{x}) + \sin(\|\mathbf{r}\|)(\hat{r} \times \mathbf{x}) - \cos(\|\mathbf{r}\|)(\hat{r} \times (\hat{r} \times \mathbf{x}))) \]
Inverse Kinematics

Ball Joints (moving axis)

\[ dp = [dr] \cdot e^r \cdot x = [dr] \cdot p = -[p] \cdot dr \]

That is the Jacobian for this joint

\[ \begin{bmatrix} r \end{bmatrix} = \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix} \]

\[ [r] \cdot x = r \times x \]
Inverse Kinematics

Ball Joints (fixed axis)

\[ dp = (d\theta)[\hat{r}] \cdot x = -[x] \cdot \hat{r} d\theta \]

That is the Jacobian for this joint
Inverse Kinematics

- Many links / joints
  - Need a generic method for building Jacobian
Inverse Kinematics

- Can’t just concatenate individual matrices

\[ \begin{aligned}
\tilde{J} & = \begin{bmatrix} J_3 & J_{2b} & J_{2a} & J_{1b} \end{bmatrix} \\
\end{aligned} \]

\[ \begin{aligned}
d & = \begin{bmatrix} d_3 \\ d_{2b} \\ d_{2a} \\ d_{1b} \end{bmatrix} \\
\end{aligned} \]

\[ \begin{aligned}
dp \neq \tilde{J} \cdot dd \\
\end{aligned} \]
Inverse Kinematics

Transformation from body to world

\[ X_{0\leftarrow i} = \prod_{j=1}^{i} X_{(j-1)\leftarrow j} = X_{0\leftarrow 1} \cdot X_{1\leftarrow 2} \cdots \]

Rotation from body to world

\[ R_{0\leftarrow i} = \prod_{j=1}^{i} R_{(j-1)\leftarrow j} = R_{0\leftarrow 1} \cdot R_{1\leftarrow 2} \cdots \]
Inverse Kinematics

Need to transform Jacobians to common coordinate system (WORLD)

\[ J_{i,WORLD} = R_{0\leftarrow(i-1)} \cdot J_i \]
Inverse Kinematics

\[ J = \begin{bmatrix}
R_{0\leftarrow 2b} \cdot J_3(\theta_3, p_3) \\
R_{0\leftarrow 2a} \cdot J_{2b}(\theta_{2b}, X_{2b\leftarrow 3} \cdot p_3) \\
R_{0\leftarrow 1} \cdot J_{2a}(\theta_{2a}, X_{2a\leftarrow 3} \cdot p_3) \\
J_1(\theta_1, X_{1\leftarrow 3} \cdot p_3)
\end{bmatrix}^T \]

\[ d = \begin{bmatrix}
d_3 \\
d_{2b} \\
d_{2a} \\
d_{1b}
\end{bmatrix} \]

Note: Each row in the above should be transposed....

\[ dp = J \cdot dd \]
Suggested Reading

- Advanced Animation and Rendering Techniques by Watt and Watt
  - Chapters 15 and 16