Advanced Computer Graphics
CSE 190 [Winter 2016], Lecture 19
Ravi Ramamoorthi
http://www.cs.ucsd.edu/~ravir

To Do

- Assignment 3 due Mar 15
  - Should already be well on way
  - Contact us for difficulties etc
- Please fill out CAPE evaluations (Now!)
- Thu lecture will be short/optional or cancelled

Course Outline

- **3D Graphics Pipeline**

  **Modeling**
  (Creating 3D Geometry)

  **Rendering**
  (Creating, shading images from geometry, lighting, materials)

  Unit 1: Foundations of Signal and Image Processing
  Understanding the way 2D images are formed and displayed, the important concepts and algorithms, and to build an image processing utility like Photoshop
  Weeks 1 – 3. Assignment 1

  Unit 2: Meshes, Modeling
  Weeks 3 – 5. Assignment 2

  Unit 3: Advanced Rendering
  Weeks 6 – 8. (Final Project)

  Unit 4: Animation, Imaging
  Weeks 9, 10. (Final Project)

The Story So Far

Animation

Forward Kinematics

Root body
- Position set by global transform
- Root joint: position, rotation
- Other bodies relative to root
- *Inboard* toward the root
- *Outboard* away from the root
- Tree structure: loop joints break “tree-ness”
Inboard and Outboard Joints

- Inboard body
- Outboard body

Body
- Inboard joint
- Outboard joint (may be several)

Bodies

Bodies arranged in a tree
- For now, assume no loops
- Body’s parent (except root)
- Body’s child (may have many children)

Joints

Interior Joints (typically not 6 DOF)
- Pin – rotate about one axis
- Ball – arbitrary rotation
- Prism – translate along one axis

Pin Joints

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

Ball Joints

- Translate inboard point to local origin
- Apply rotation about arbitrary axis
- Translate origin to location of joint on outboard body
Prism Joint

- Translate inboard joint to local origin
- Translate along axis
- Translate origin to location of joint on outboard

Forward Kinematics

- Composite transformations up the hierarchy

Inverse Kinematics

- Given
  - Root transformation
  - Initial configuration
  - Desired end point location
- Find
  - Interior parameter settings

2 Segment Arm in 2D

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

Direct IK

- Analytically solve for parameters (not general)

\[
\begin{align*}
\theta_2 &= \cos^{-1}\left(\frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1 l_2}\right) \\
\theta_1 &= \frac{-p_x l_2 \sin(\theta_2) + p_y (l_1 + l_2 \cos(\theta_2))}{p_x l_2 \sin(\theta_2) + p_y (l_1 + l_2 \cos(\theta_2))}
\end{align*}
\]
Difficult Issues

- Multiple configurations distinct in config space
- Or connected in config space

Infeasible Regions

Numerical Solution

- Start in some initial config. (previous frame)
- Define error metric (goal pos – current pos)
- Compute Jacobian with respect to inputs
- Use Newton’s or other method to iterate
- General principle of goal optimization

Back to 2 Segment Arm

Solving for Joint Angles

\[
\begin{align*}
\mathbf{c} &= \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \\
\mathbf{dp} &= \begin{bmatrix} dp_1 \\ dp_2 \end{bmatrix} \\
\mathbf{dp} &= \mathbf{J} \cdot \mathbf{c} \\
\mathbf{c} &= \mathbf{J}^{-1} \cdot \mathbf{dp}
\end{align*}
\]
Issues

- Jacobian not always invertible
  - Use an SVD and pseudo-inverse
- Iterative approach, not direct
  - The Jacobian is a linearization, changes
- Practical implementation
  - Analytic forms for prism, ball joints
  - Composing transformations
  - Or quick and dirty: finite differencing
  - Cyclic coordinate descent (each DOF one at a time)

Prism and Ball Joints

- Prism Joints
- Ball Joints

More on Ball Joints

- Ball Joints (moving axis)
- Ball Joints (fixed axis)

Multiple Links

- IK requires Jacobian
  - Need generic method for building one
- Can’t just concatenate matrices

Composing Transformations

- Transformation from body to world
- Rotation from body to world

Inverse Kinematics: Final Form

- Jacobian
- Vector of joint positions
- Jacobian transpose
A Cheap Alternative

- Estimate Jacobian (or parts of it) w. finite diffs.
- Cyclic coordinate descent
  - Solve for each DOF one at a time
  - Iterate till good enough / run out of time

More complex systems

- More complex joints (prism and ball)
- More links
- Other criteria (center of mass or height)
- Hard constraints (e.g., foot plants)
- Unilateral constraints (e.g., joint limits)
- Multiple criteria and multiple chains
- Loops
- Smoothness over time
  - DOF determined by control points of curve (chain rule)

Practical Issues

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

Prior on “good” configurations