Notes

- Handing assignment 0 back (at the front of the room)
- Read the newsgroup!
- Planning to put 16mm films on the web soon (possibly tomorrow)

Forward Kinematics

- A simple layered approach
  - Get the root link moving first (e.g. the pelvis)
  - Fix the pose outward, link by link (the back and the legs next, then the head, the arms, the hands, the fingers, …)
- Great for certain types of motion
  - General acting, moving in free space, …
- Problems when interacting with other objects
  - How do you make sure fingers are exactly in contact with a doorknob throughout a scene, as the arm moves, as the body moves, …
  - Even simpler: how do you make sure the feet stay planted on the ground? (avoiding "foot-skate")

Inverse Kinematics (IK)

- Keep the same tree structure,
  fix the root,
  get the body into roughly the right pose
- Specify a target position (maybe orientation too) for an "end-effector"
- Solve system of equations for the joint angles that achieve the target
- This is really just a user interface for the computer to help you set the joint angles to what you want

IK Applications

- Obvious application in robotics:
  robot arm needs to figure out how to reach something
- Used in graphics as an efficient UI for animating grasping, contact, foot plants, etc.
- Also used in graphics for automatic corrections
  (later in the term: footskate cleanup)

IK issues 1

- Too many joints: there are many solutions, how do you pick the best one?
  - Often add constraints---e.g. keep the arm in the specified plane so it doesn't rotate wildly
  - Minimize some quantity---e.g. the sum of squares of differences from original rough pose (or previous frame)
  - Try to keep poses close to a set of known good poses (e.g. from motion capture data)

IK issues 2

- Target impossible (or just barely possible) --- not enough degrees of freedom
  - Try to get as close as possible to the target in some sense
  - Penalize bad joint angles, huge differences from initial pose
    - So if pose is only just possible, via some weird unrealistic angles, we don't do it

How do you do IK?

- Simple skeletons: analytic solutions may exist
  - Look up formulas from papers (or textbook)
  - Not fun, not flexible
- More generally: numerical solution of optimization problem
  - Minimize difference between actual end-effort pose and the target
  - Possibly include constraints
    - E.g. joint angle limits, collisions

IK as optimization

- Unconstrained optimization problem:

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

- In our case:
  - theta is a vector of all the joint angles
  - f(.) is a measure of the distance between the pose of the end-effector and the target
- Start from an initial guess, use algorithms to steadily make better guesses until close enough

Line Search Approach

- Say we have guess θ_k
- ♣ Pick a "direction" d_k to search along
- ♣ Look for new guess as $\theta_{k+1}= \theta_k+\alpha d_k$
- ♣ Line search: figure out a scalar α (the "step size") that will make $f(\theta_{k+1})$ smaller than $f(\theta_k)$
- ♣ Assuming d_k is scaled properly, simplest algorithm:
  - ♣ Check $\alpha=1$: if it's an improvement stop
  - ♣ Otherwise halve α and try again
- ♣ If d_k is "downhill" $d_k \cdot \nabla f<0$ then guaranteed to improve if α is small enough

Picking the direction

- Cyclic Coordinate Descent: in each direction, pick a different axis to move along (adjust just one angle at a time)
  - Problem 1: scaling isn't clear
  - Problem 2: extremely inefficient, might not converge at all
- Steepest Descent: use the direction that f(.) is decreasing most rapidly in: $-\nabla f$
  - Problem 1: scaling isn't clear
  - Problem 2: inefficient for difficult f

More on Steepest Descent

- Typically we have a nonlinear least-squares problem:

$$f(\theta)=1/2|x(\theta)-x_{target}|^2$$

- ♣ In this case, $-\nabla f=J^T(x_{target} -x(\theta))$ where $J=\partial x/\partial\theta$ is the Jacobian
- ♣ Then sometimes this is called the "Jacobian method"