Announcements

Assignment 1 on the web page:
www.cs.cmu.edu/~jkh/anim_class.html

Test login procedure NOW!

Forward and Inverse Kinematics

Parent: Chapter 4.2

Girard and Maciejewski 1985

Zhao and Badler 1994

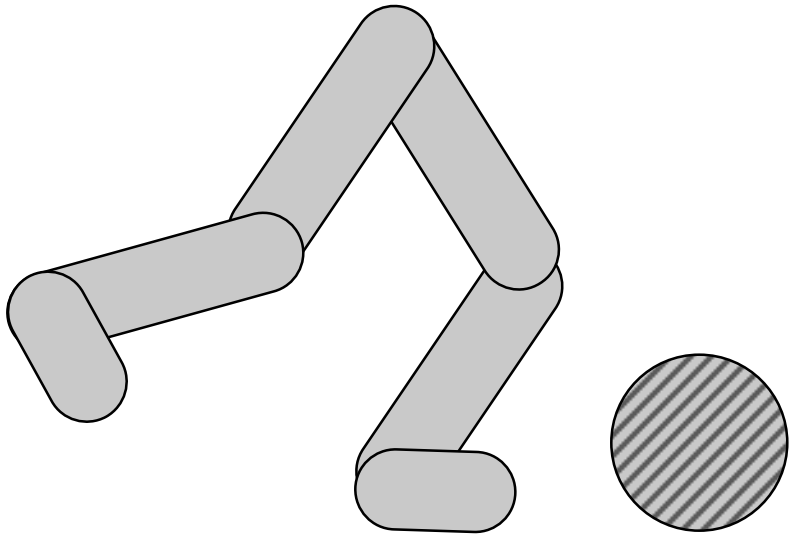
COMPUTER ANIMATION

15-497/15-861

02/26/02

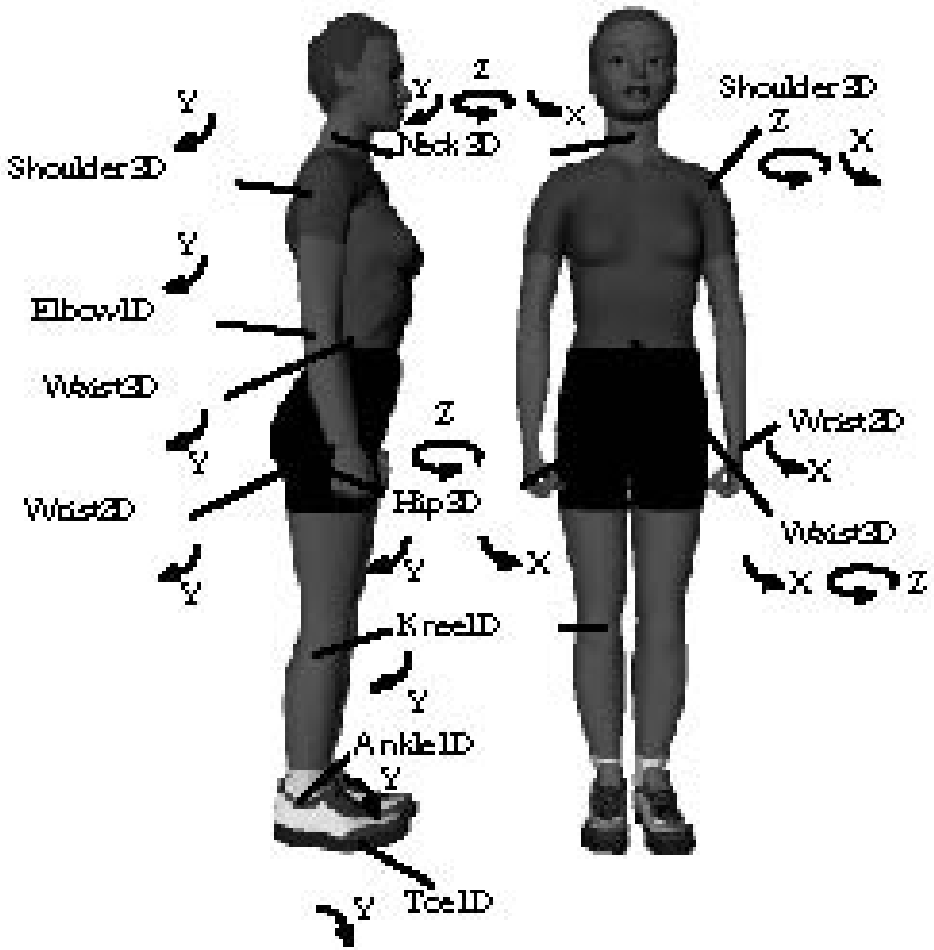
Kinematics

- Last two classes: how to interpolate positions/translations and orientations
- But we also need to set the keyframes—make the foot come into contact with the ball, for example



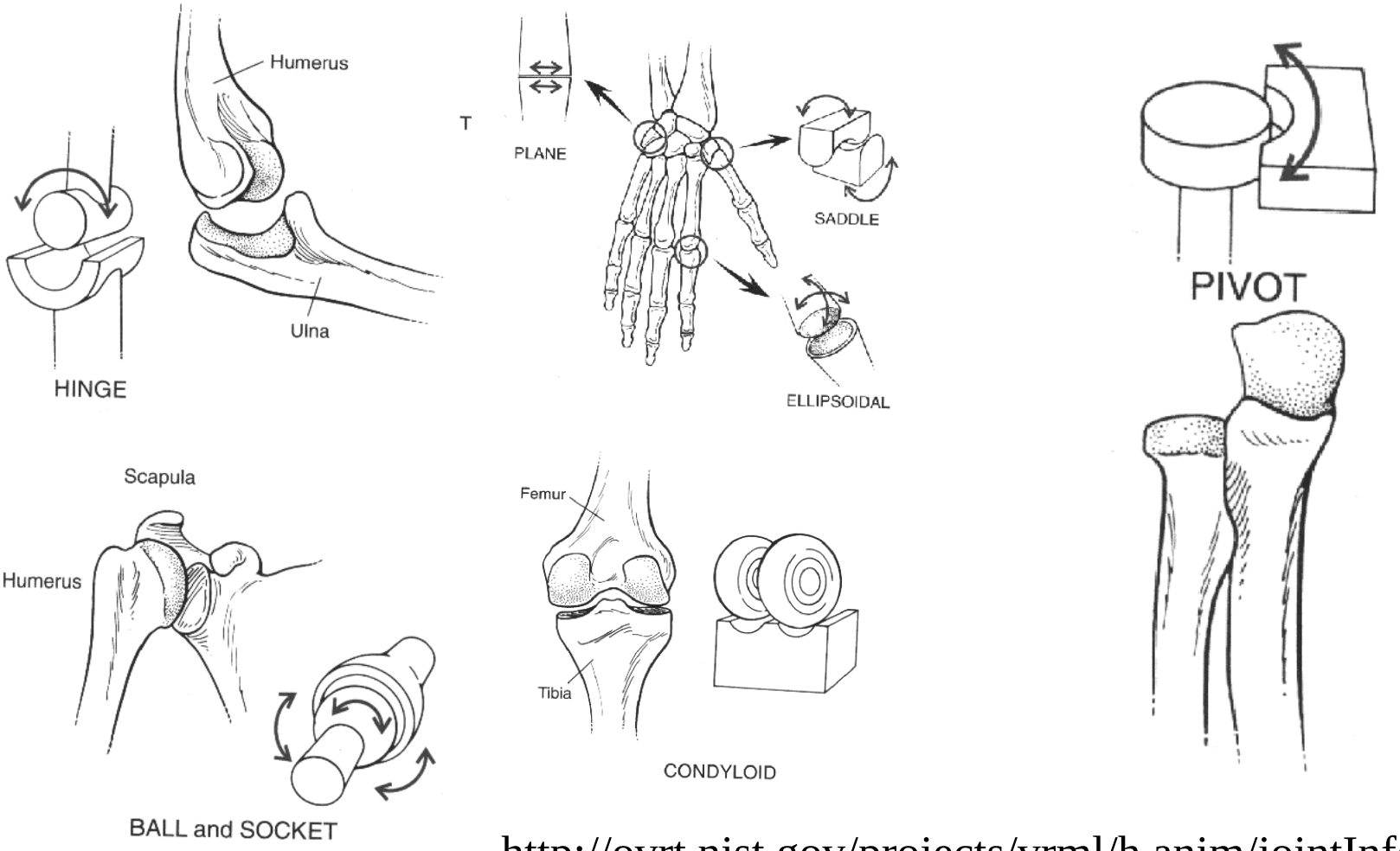
Hierarchical Models

- Want structure for body to be maintained
  - Pivot joint
  - Prismatic joint
  - Ball and socket joint



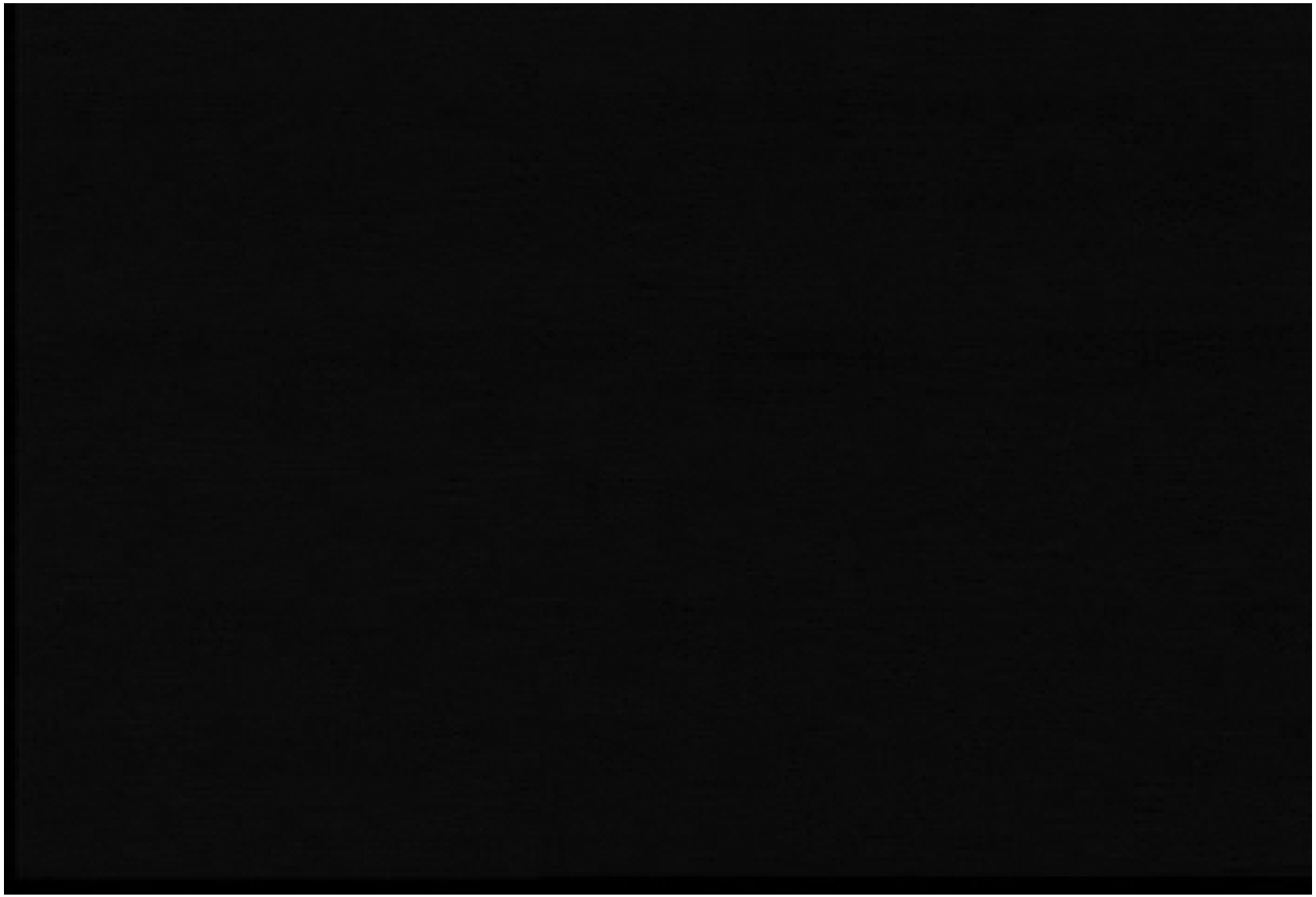
Human Models

Articulated figures are a horrible approximation



http://ovrt.nist.gov/projects/vrml/h-anim/jointInfo.html

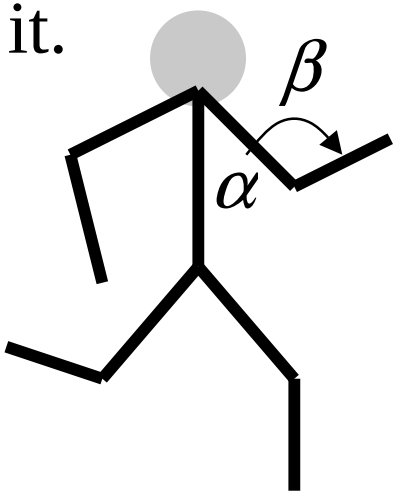
Hierarchical Models in Simulation

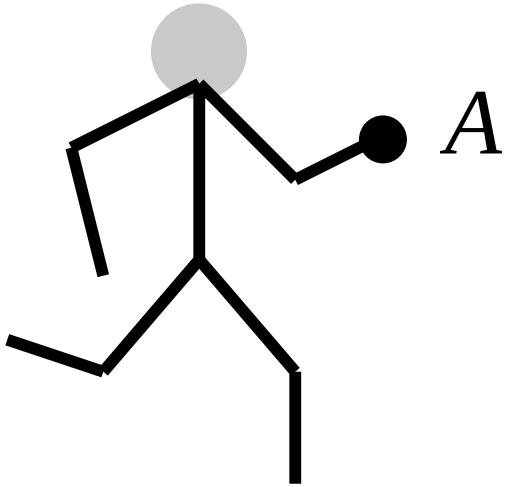


Wayne Wooten, Gatech

Kinematics

- The study of motion without regard to the forces that cause it.





Forward: $A = f(\alpha, \beta)$

Draw graphics

Backward: $\alpha, \beta = f^{-1}(A)$

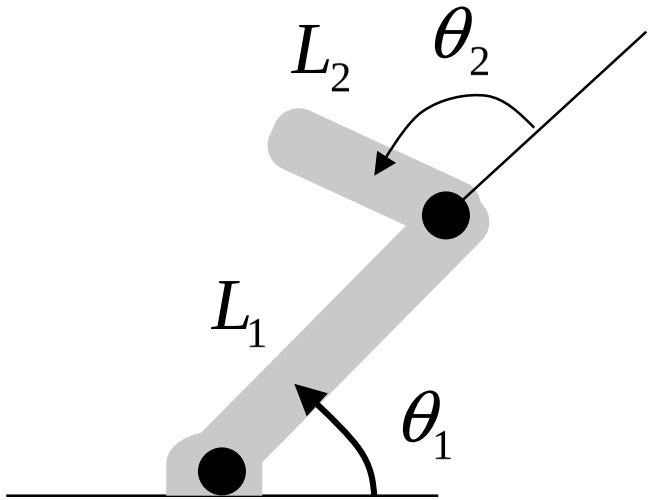
Specify fewer degrees of freedom
More intuitive control of dof
Maintain contact with the environment
Calculate desired joint angles for control

Forward Kinematics

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} & & & \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$



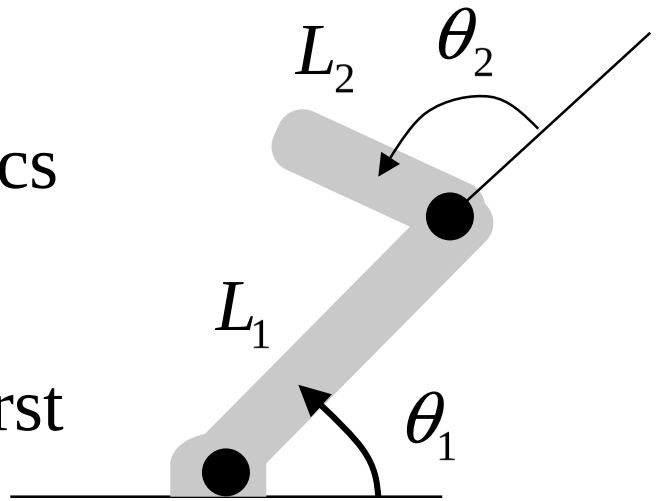
$$\begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} = [transL_2][rot\theta_2][transL_1][rot\theta_1]$$

Why Inverse Kinematics?

- Pick up an object or place feet on the ground

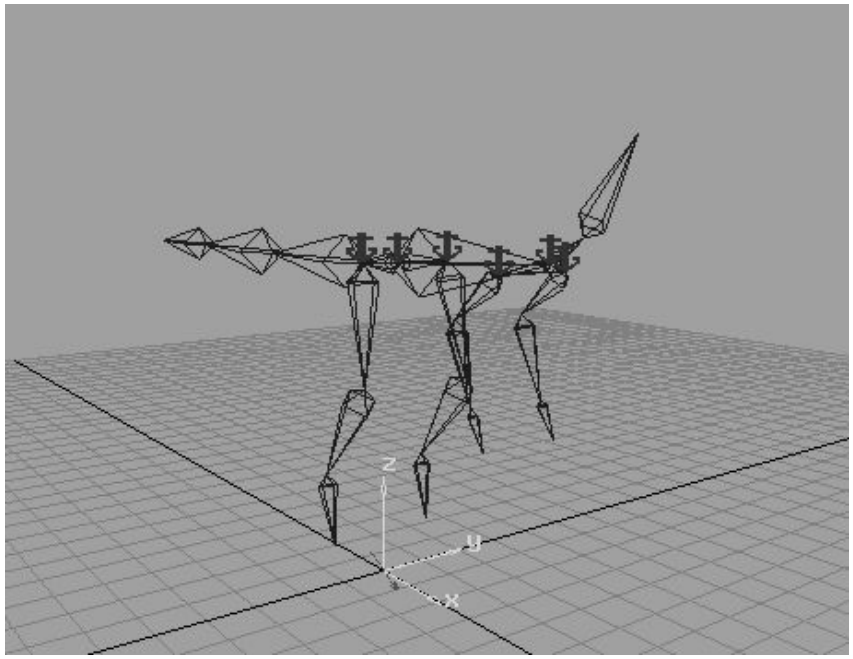
  Hard to do with forward kinematics

- Allow animator to set fewer parameters or at least get a good first approximation



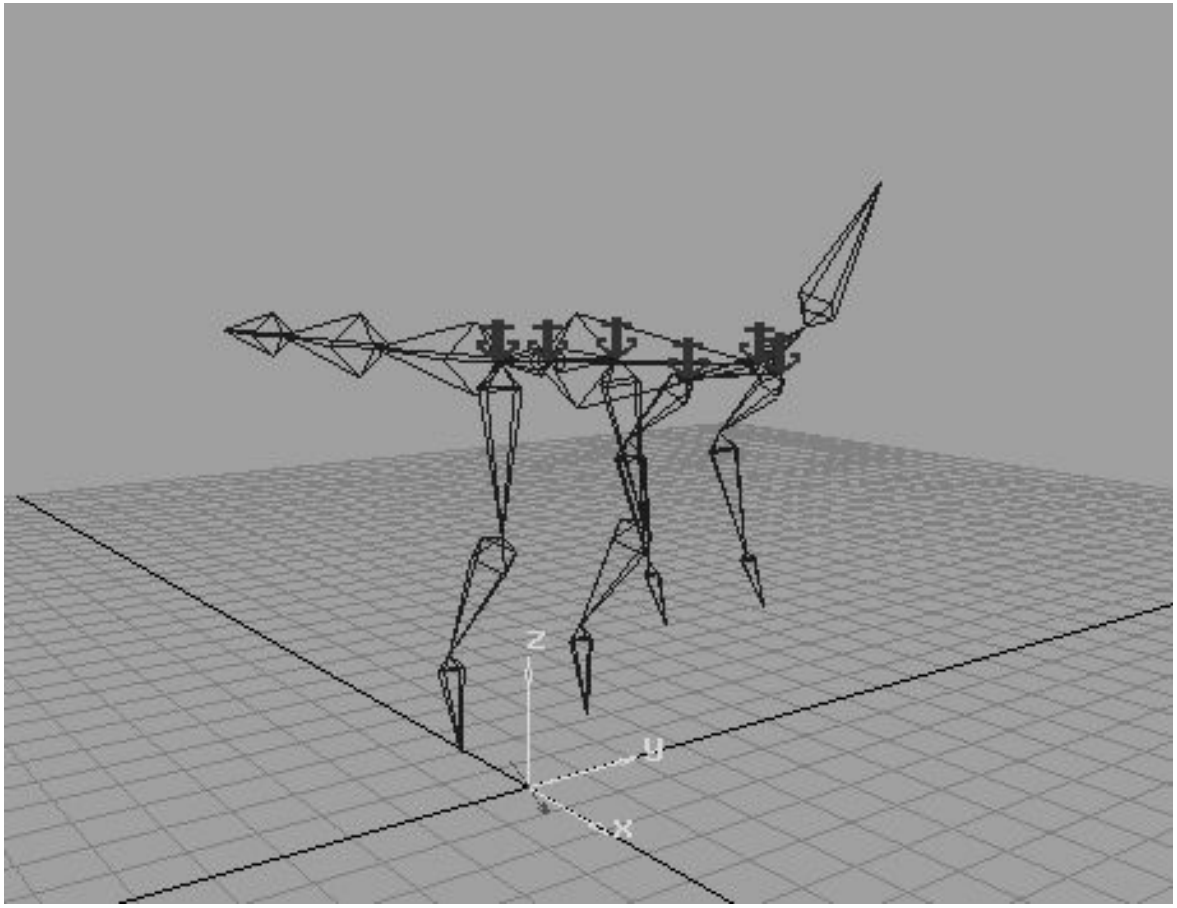
User Control via Inverse Kinematics

- Joint space
  - Position all joints—fine level of control
- Cartesian space
  - Specify environmental interactions easily
  - Most degrees of freedom computed automatically



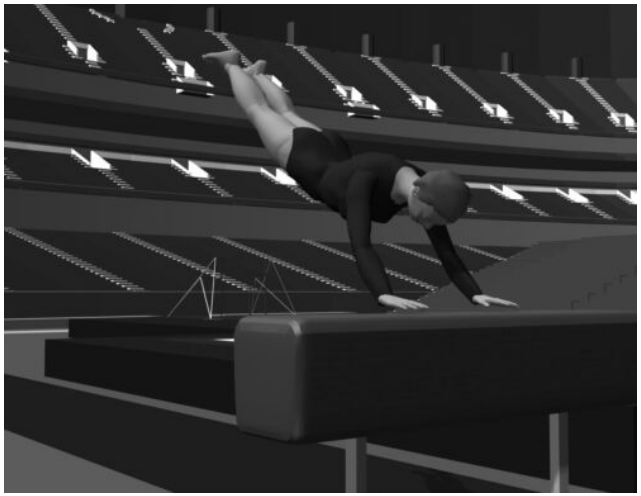
What do we need?

- Natural looking path or just a goal position? (time coherency)
- Local or global solution?



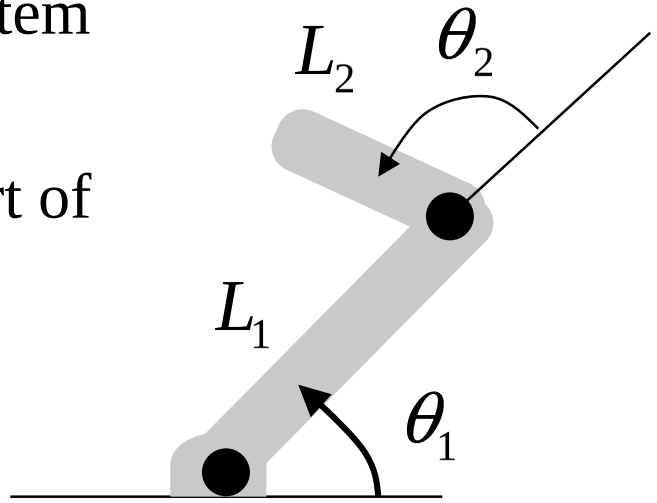
Why Inverse Kinematics—Control

- Balance---keep center of mass over support polygon
- Control---position vaulter's hands on line between shoulder and vault
- Control---compute knee angles that will give the runner the right leg length



What do we need?

- Skeleton with 1,2,3 degree of freedom joints
- Solve from root position to end effector position
- Arbitrary position constraints
- Direction/orientation constraints
- Joint limits
- Techniques for handling unconstrained system
  - Adding constraints
  - Heuristics to push solution into right part of space
  - Optimization based on some criterion



Inverse Kinematics—Closed Form Solution

$$\cos(\theta_T) = \frac{x}{\sqrt{x^2 + y^2}}$$

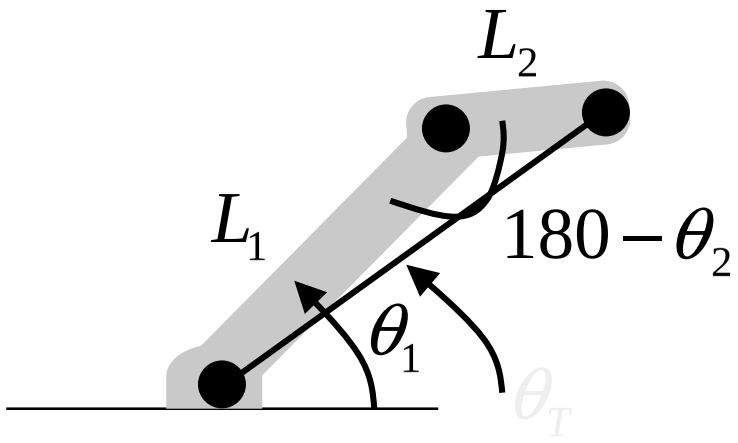
$$\theta_T = a\cos\left(\frac{x}{\sqrt{x^2 + y^2}}\right)$$

$$\cos(\theta_1 - \theta_T) = \frac{L_1^2 + x^2 + y^2 - L_2^2}{2 - L_1\sqrt{x^2 + y^2}} \quad \text{cosine rule}$$

$$\theta_1 = a\cos\left(\frac{L_1^2 + x^2 + y^2 - L_2^2}{2 - L_1\sqrt{x^2 + y^2}}\right) + \theta_T$$

$$\cos(180 - \theta_2) = \frac{(-x^2 - y^2 + L_1^2 + L_2^2)}{2L_1L_2} \quad \text{cosine rule}$$

$$\theta_2 = a\cos\left(\frac{(-x^2 - y^2 + L_1^2 + L_2^2)}{2L_1L_2}\right)$$



Methods

- Closed form—only for fairly simple mechanisms
- Iterative solutions
- Solutions
  - No solution (outside workspace, too few dof)
  - Multiple solutions (redundancy)
  - Single solution

What makes IK hard?--Redundancies

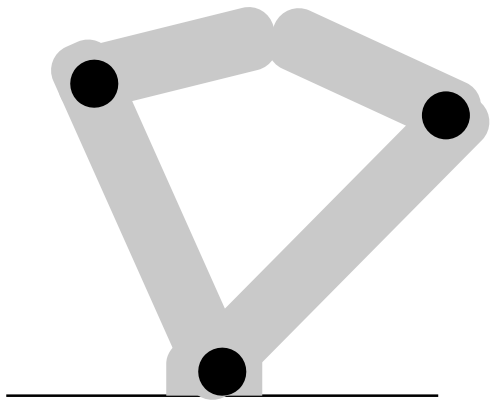
a subspace $\{\theta_x\}$ defined by

$\theta(\theta_1, \theta_2 ..., \theta_n) \in \theta_x$ if $f(\theta) = \mathrm{X}$

Add constraints to reduce redundancies

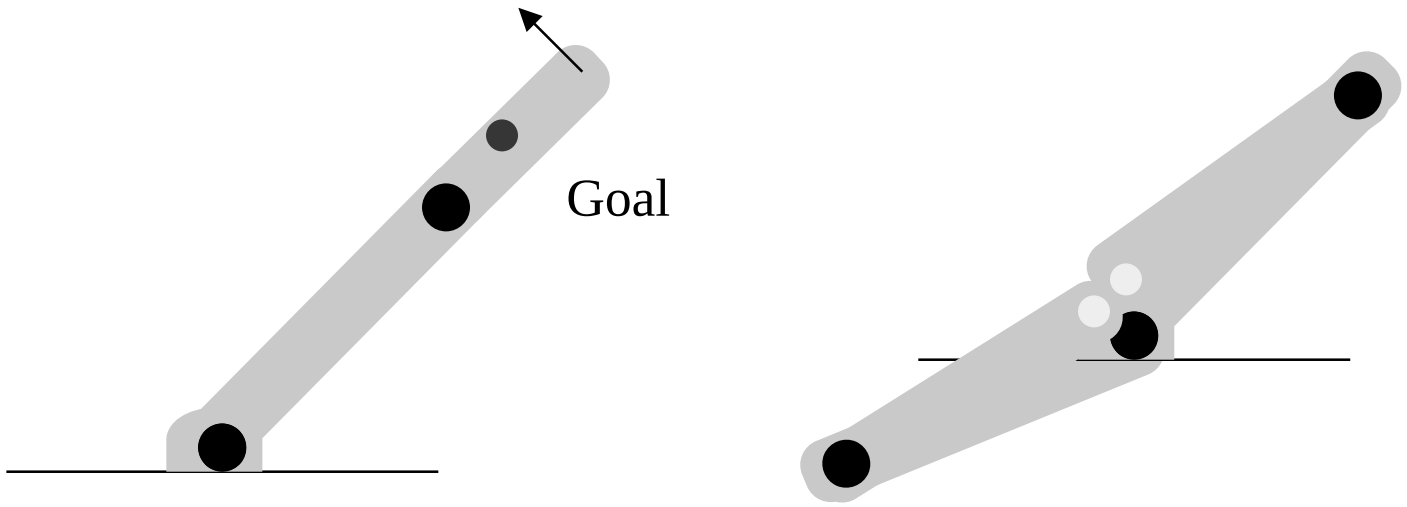
Choose solution that is

- "closest" to current configuration
- Move outermost links the most
- Energy minimization
- Minimum time
- Natural looking motion???



What makes IK hard?--singularities

- Ill-conditioned near singularities
- High state space velocities for low cartesian velocities



Reachable Workspace

$$L_1 - L_2 \leq \sqrt{x^2 + y^2} \leq L_1 + L_2$$

Again, not so simple for more complex mechanisms

Iterative Method

Used in Girard and Maciejewski 1985
Described in Parent

Uses inverse of Jacobian to iteratively step all the joint angles towards the goal

The Jacobian

$f(\theta) = x$ x is of dimension n (generally 6)

θ is of dimension m (# of dof)

Jacobian is the *nxm* matrix relating differential changes of θ ($\partial\theta$) to differential changes x (∂x)

$J(\theta)\partial\theta = \partial x$ where the *ij*th element of J is

$$J_{ij} = \frac{\partial f_i}{\partial x_j}$$

Jacobian maps velocities in state space to velocities in cartesian space. Jacobian is dependent on state (and must be recomputed frequently).

The Jacobian—using Parent's notation

$$V = \left[v_x, v_y, v_z, \omega_x, \omega_y, \omega_z \right]^T$$

$$\dot{\theta} = \left[\dot{\theta}_1, \dot{\theta}_2 \ldots \dot{\theta}_n \right]^T$$

$$J_{ij} = \frac{\partial V}{\partial \theta}$$

$$J = \begin{bmatrix} \frac{\partial v_x}{\partial \theta_1} & \mathrm{K} & \frac{\partial v_x}{\partial \theta_n} \\ & \mathrm{O} & \mathrm{M} \\ & & \\ & & \frac{\partial \omega_z}{\partial \theta_n} \end{bmatrix}$$

IK and the Jacobian

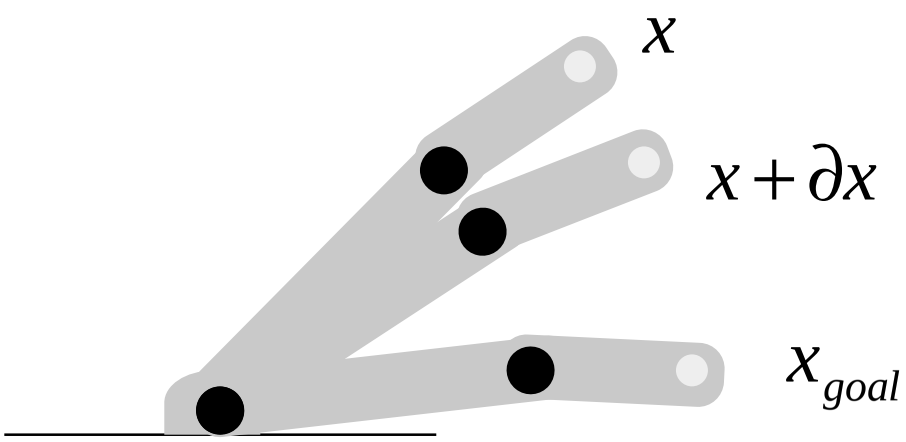
$f(\theta) = x$ x is of dimension n (generally 6)

θ is of dimension m (# of dof)

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

$\partial x = J \partial \theta$

$\partial \theta = J^{-1} \partial x$



$\theta_{k+1} = \theta_k + \Delta t J^{-1} \partial x$

linearize about θ_k

An iterative solution

Inverting the Jacobian

J is nxm not square in general

compute pseudo-inverse J^+

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

$$J^T V = J^T J\,\dot{\theta}$$

$$\left(J^T J\right)^{-1} J^T V = \left(J^T J\right)^{-1} J^T J\,\dot{\theta}$$

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

where $J^+ = \left(J^T J\right)^{-1} J^T = J^T \left(J J^T\right)^{-1}$

Using the Jacobian for IK

- Singularities cause the rank of the Jacobian to change

- Jacobian is valid only for the configuration for which it was computed

- Pseudo inverse minimizes joint angle rates (locally)

- Could minimize other quantities

Non-Linear Optimization

Zhao and Badler, TOG 1994 (pointer from syllabus)

Non-linear programming—numerical method for finding the (local) minimum of a non-linear function

Objective function
Constraints
Non-linear optimization routine

Objective Function

•Position of end effector $P(x) = (p - x)^2$

$$\nabla_x P(x) = 2(x - p)$$

•Or position and orientation

$$P(x_{e,} y_e) = (x_g - x_e)^2 + (y_g - y_e)^2$$

where x_g, y_g is the orientation goal as a pair of orthonormal vectors

$$\nabla_{x_e} P(x_e, y_e) = 2(x_g - x_e)$$

$$\nabla_{y_e} P(x_e, y_e) = 2(y_g - y_e)$$

•Or aiming at

•For example, you might want the hand to slide along the tabletop or to keep the glass upright.

Formulation

$$\text{minimize } \mathrm{G}(\theta) \text{ subject to } a_i^T \theta = b_i$$

$$a_i^T \theta < b_i$$

joint limits are inequalities $-\theta_i \leq -l_i$

$$\theta_i \leq u_i$$

Solution

$$G(\theta) \text{ and } \nabla_\theta G(\theta) = \left(\frac{\partial e}{\partial \theta} \right)^T \nabla_x P(e)$$

use a standard numerical technique

Summary of Kinematics

- Forward is straightforward
- Inverse usually requires a numerical solution
- May not always get the “right” answer
  - Frame-to-frame coherence (fix long segments)
  - Natural looking motion (how defined?)
  - If you don’t like the solution add more constraints—hardly an elegant solution…

The study of motion without regard to the forces that cause it---so, what about those forces?

Simulation---Technique #3