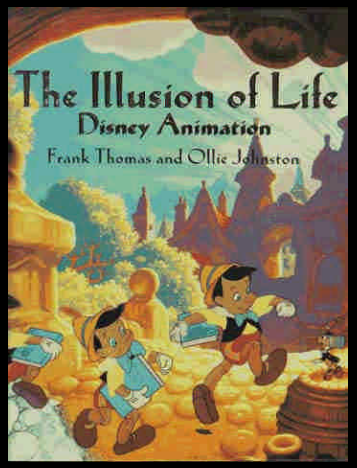
Computer Animation III



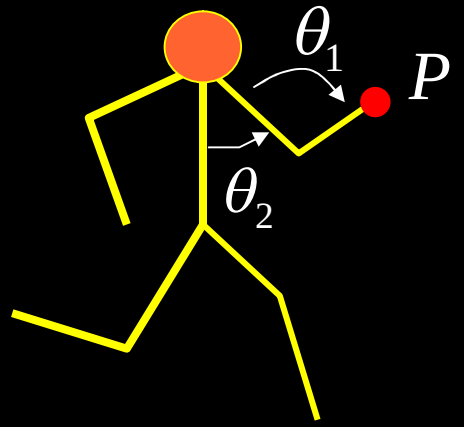
Wen-Chieh (Steve) Lin

National Chiao-Tung University

Shirley, Fundamentals of Computer Graphics, Chap 16

Joint Space vs. Cartesian Space

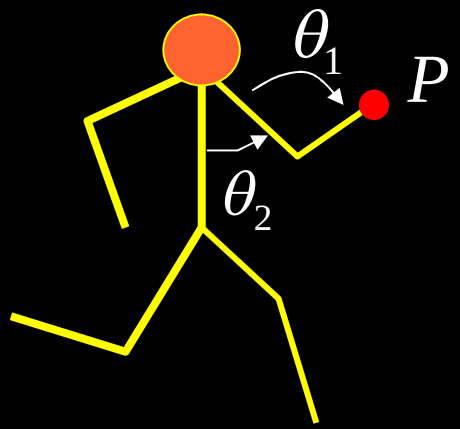
- Joint space
  - space formed by joint angles
  - position all joints—fine level control
- Cartesian space
  - 3D space
  - specify environment interactions



$$P = f(\theta_1, \theta_2)$$

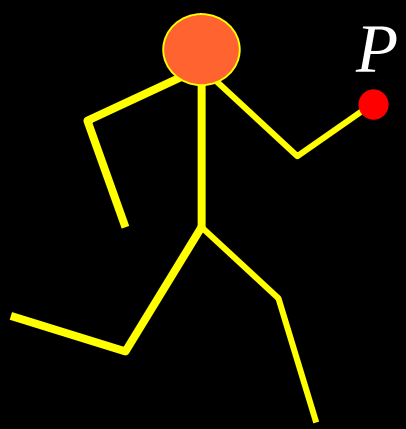
Forward and Inverse Kinematics

- Forward kinematics
  - mapping from joint space to cartesian space
- Inverse kinematics
  - mapping from cartesian space to joint space



Forward Kinematics

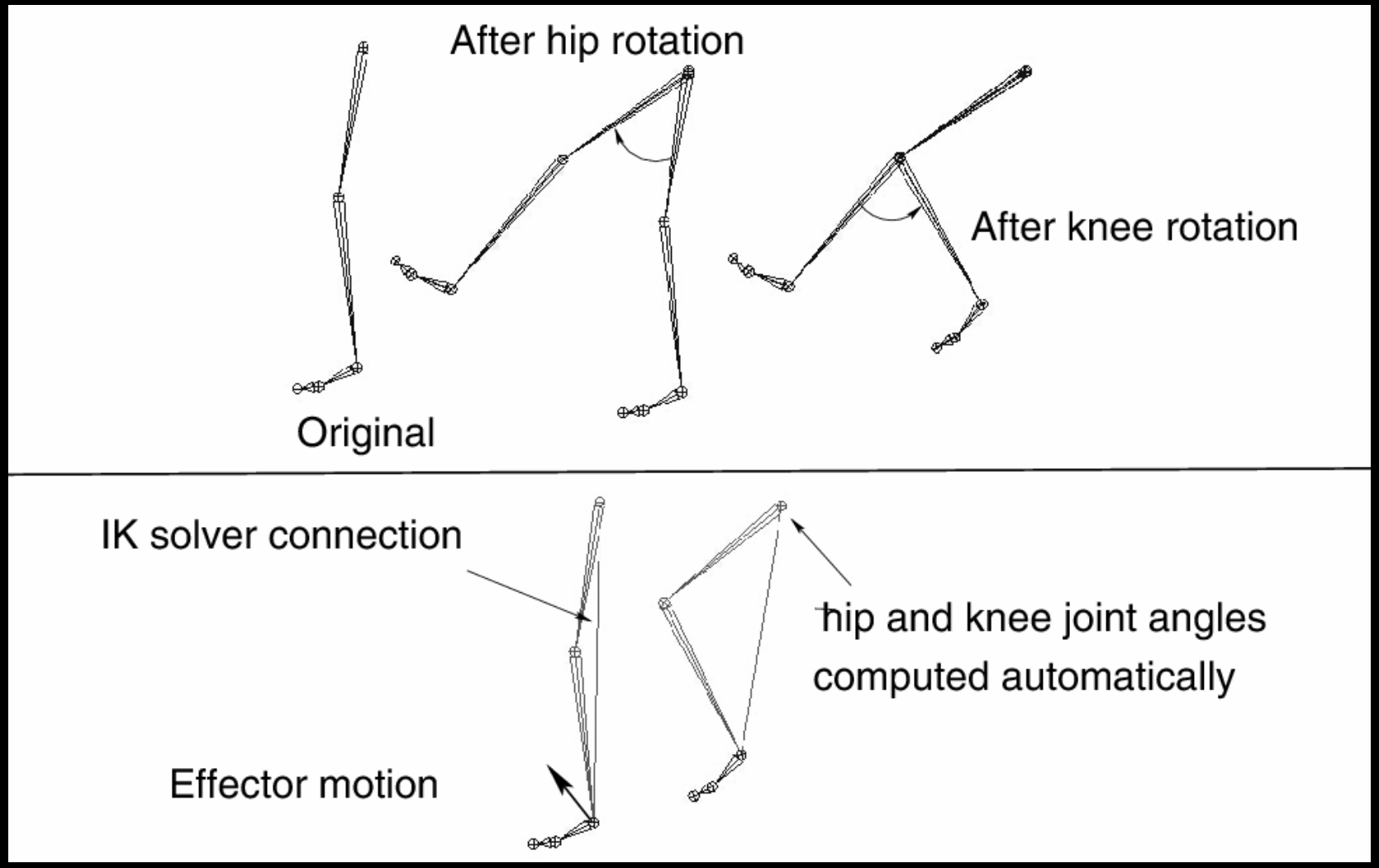
$$P = f(\theta_1, \theta_2)$$



Inverse Kinematics

$$\theta_1, \theta_2 = f^{-1}(P)$$

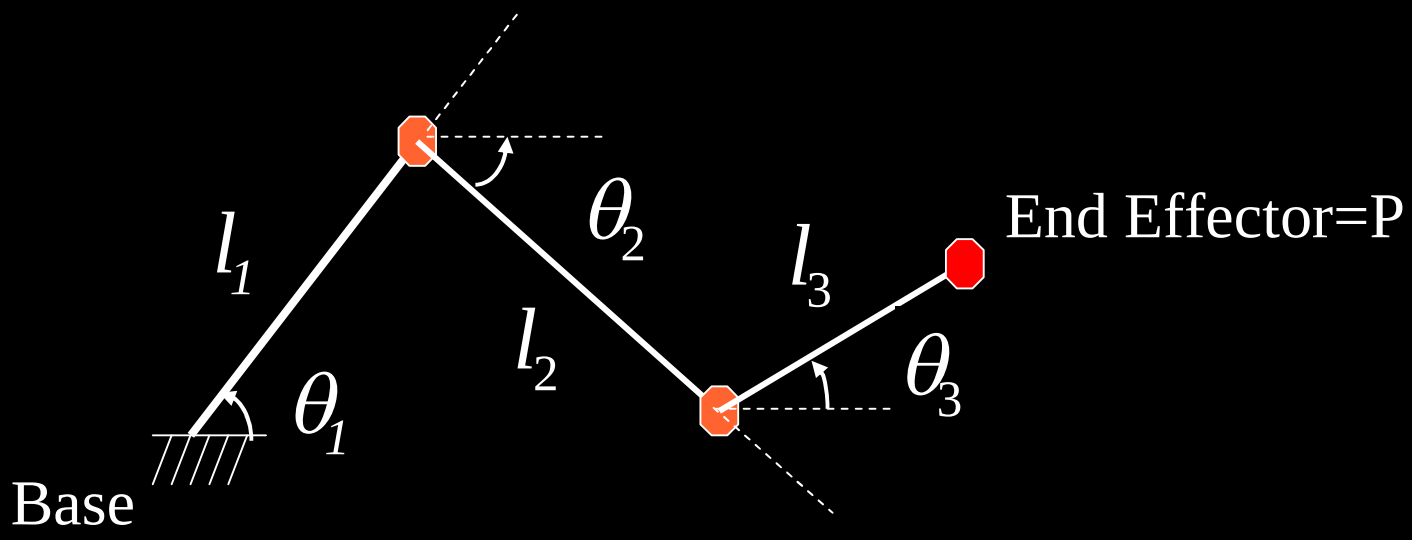
Forward & Inverse Kinematics (cont.)



Forward & Inverse Kinematics (cont.)

- Forward kinematics
  - good for rendering
- Inverse kinematics
  - good for specifying environment interaction
  - good for controlling a character—fewer parameters

Example: Forward Kinematics

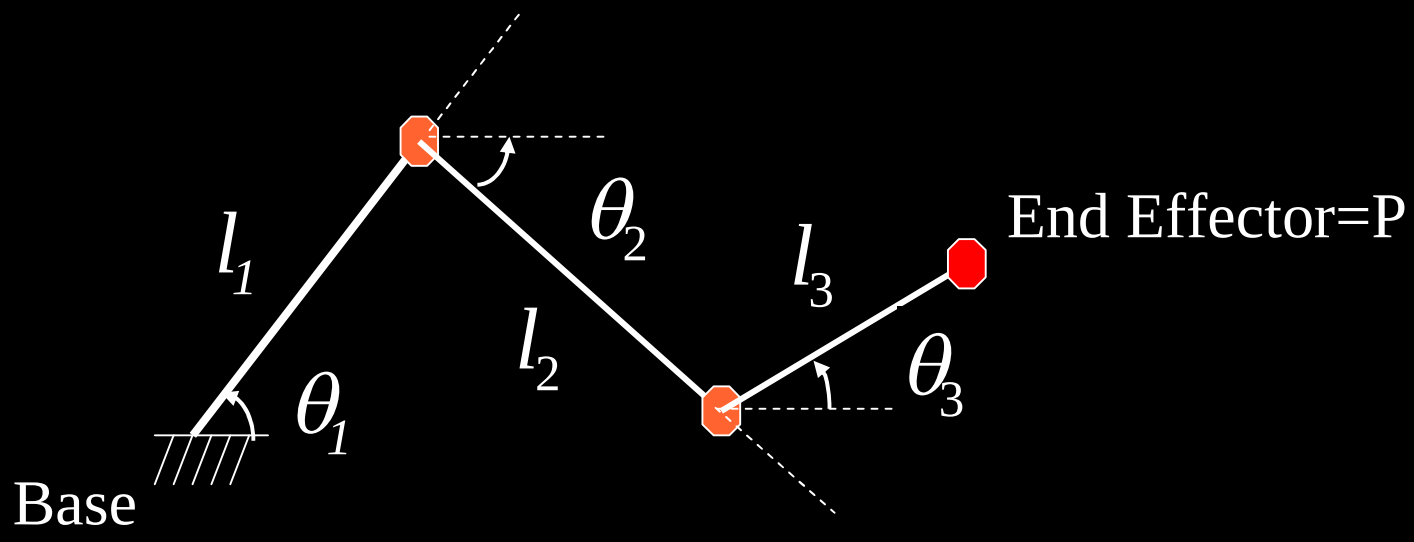


$$P = \mathrm{f}(\theta_1, \theta_2, \theta_3)$$

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

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

Example: Inverse Kinematics



$$\theta_1, \theta_2, \theta_3 = \mathrm{f}^{-1}(P)$$

Redundancy in IK

- Our example
  - 2 equations (constraints)
  - 3 unknowns

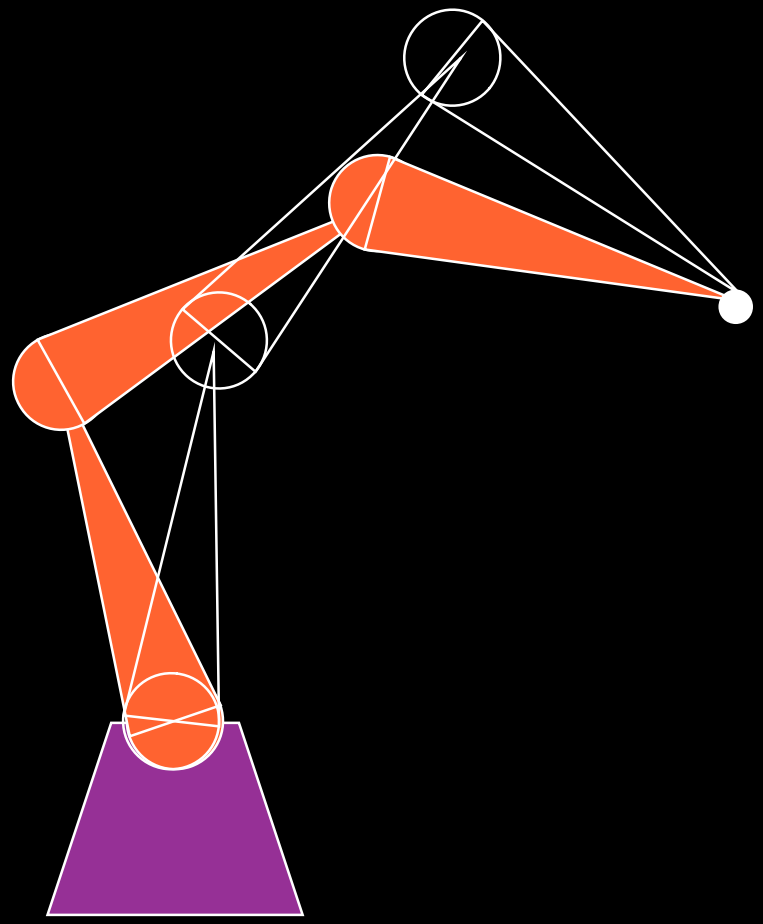
$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3)$$

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

  - Multiple solutions exist!
- This is not uncommon!
  - see how you can move your elbow while keeping your finger touching your nose

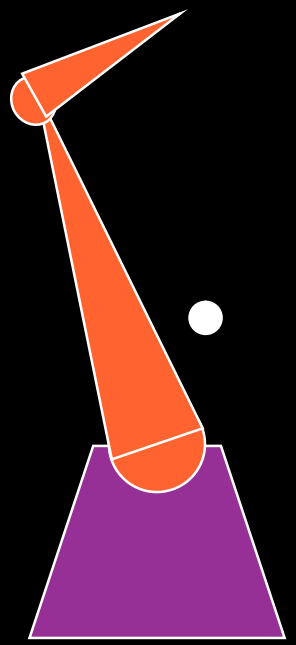
Other problems in IK

- Infinite solutions



Other problems in IK

- No solutions



Why is IK hard?

- Redundancy
- Natural motion
  - joint limits
  - minimum jerk
  - style?
- Singularities
  - ill-conditioned matrix
  - shown later

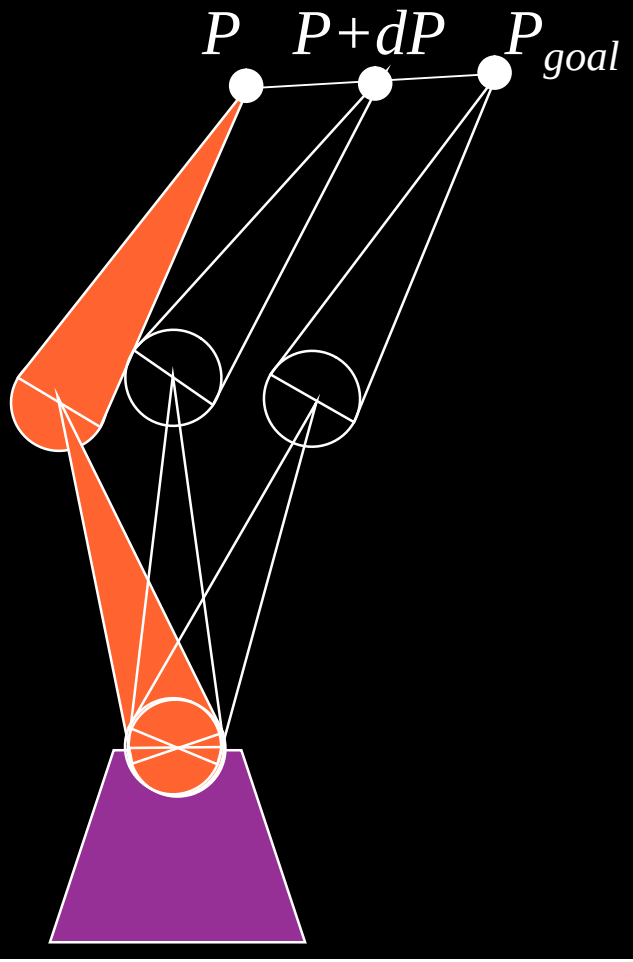
Solving IK using Inverse-Jacobian

- Iteratively change the joint angles to approach the goal position and orientation

$$f(\theta) = P$$

$$J(\theta)d\theta = dP \qquad J_{ij} = \frac{\partial f_i}{\partial \theta_j}$$

$$\theta_{k+1} = \theta_k + \Delta t J^{-1}(\theta_k)V$$



Jacobian

$$f(\theta) = P \qquad P \in R^n \ \ (n = 6 \text{ usually})$$

$$\theta \in R^m (m = \text{DOFs})$$

- Jacobian is the *n* by *m* matrix relating differential changes of θ to differential changes of *P* (*dP*)

$$J(\theta)d\theta = dP \qquad J_{ij} = \frac{\partial f_i}{\partial \theta_j}$$

- Jacobian maps velocities in joint space to velocities in cartesian space $J(\theta)\dot{\theta} = V$

Example: Jacobian Matrix

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

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

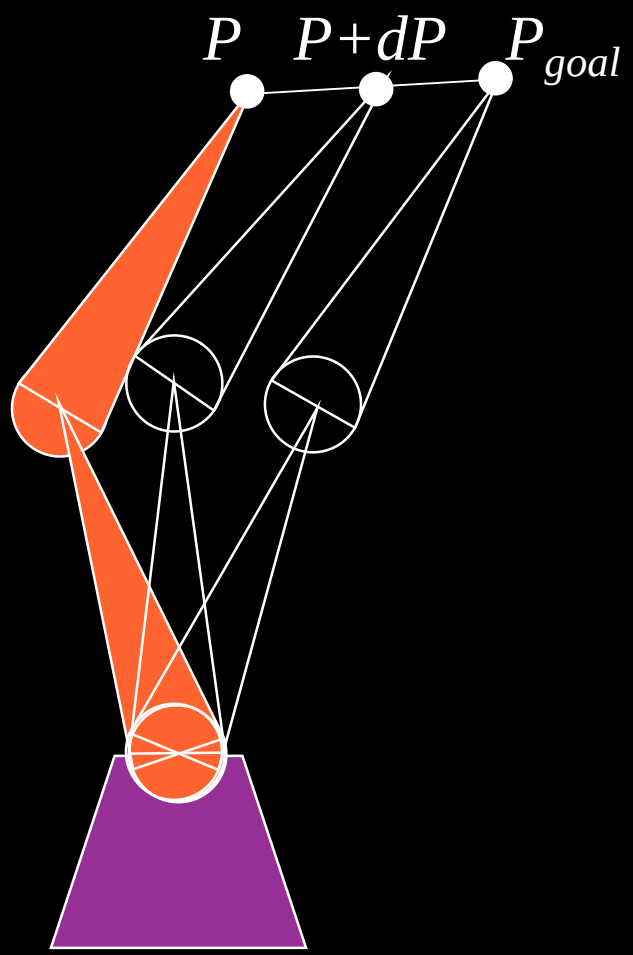
Iteratively Solving

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

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

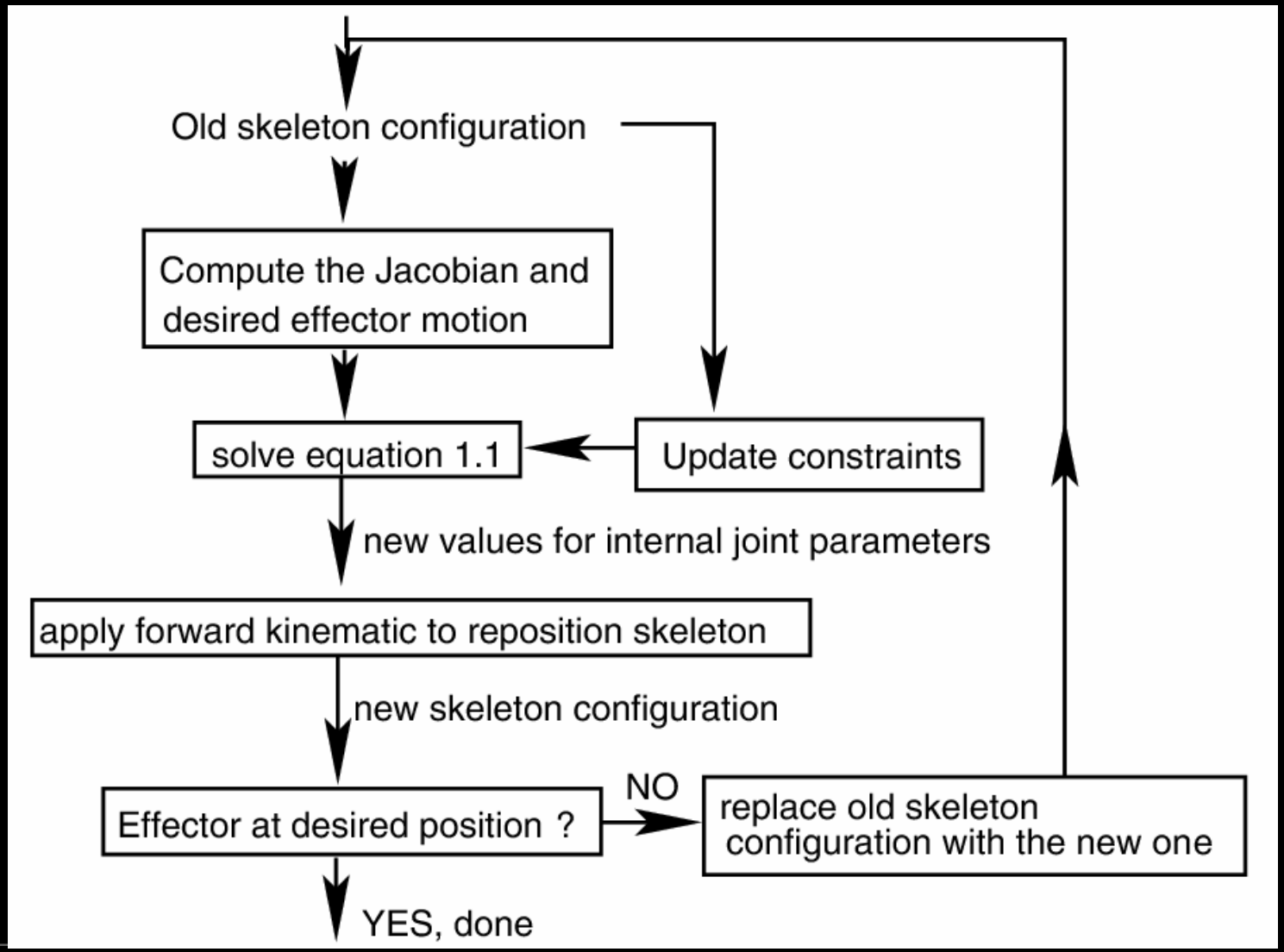
$$\dot{\theta} = J^{-1}(\theta)V$$

$$\theta_{k+1} = \theta_k + \Delta t J^{-1}(\theta_k)V$$



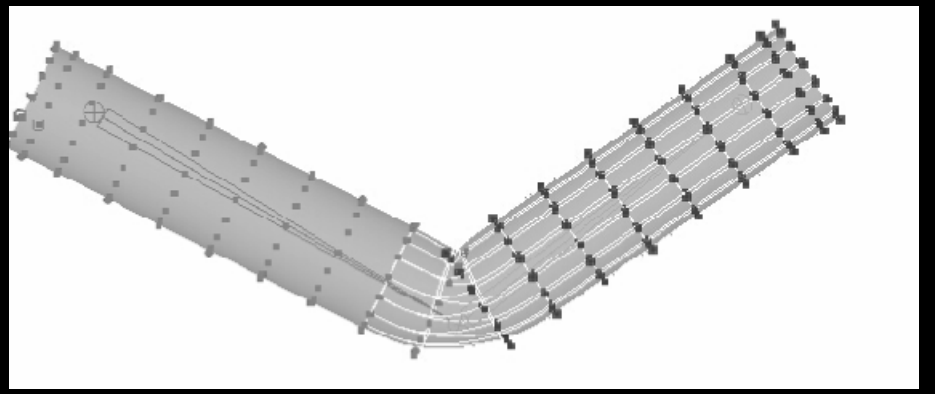
- Linearize about θ_k locally
- Small increments

Iteratively Solving (cont.)



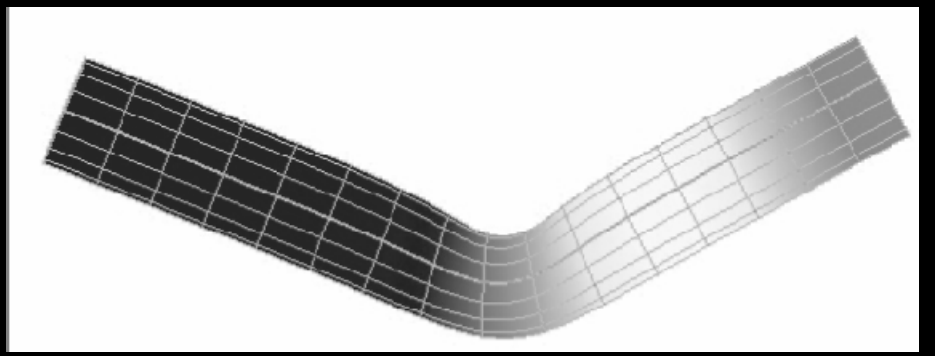
Skinning

- Skin deformation driven by skeleton
- Rigid skinning
  - Each skin vertex moves with one joint



- Smooth skinning
  - Each skin vertex moves with multiple joints

$$\mathbf{d} = \sum w_i \mathbf{d}_i \quad \sum w_i = 1$$



Motion Capture

- Record the animation from live action
  - simplest method - rotoscope (trace) over video of real motions
- Real time input devices
  - electronic puppeteering
- Motion capture
  - track motion of reference points
    - body or face or hands
  - magnetic
  - optical
  - exoskeletons
  - convert to joint angles (not always straightforward)
  - use these angles to drive an articulated 3-D model
  - These motion paths can be *warped*

Types of Motion Capture System

optical

magnetic

mechanical

Application: Special Effects

Optical Mocap System used in “I, Robot”

Application: Video Games

- Optical MOCAP system

Applications: Computer Puppetry

- Performance-based animation using a magnetic MOCAP system

Shin et al., “Computer puppetry”

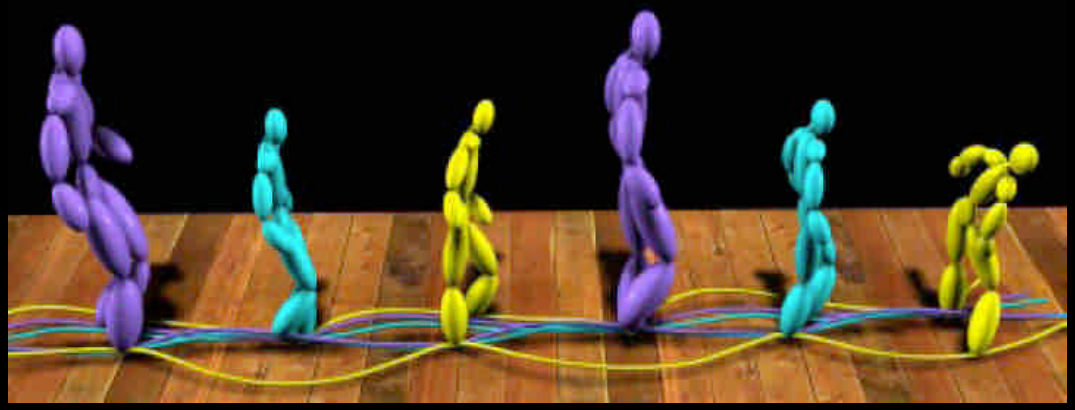
Application: Human Computer Interface

- Mechanical Motion Capture
- Data glove
  - Bend sensor + optical tracking
  - 6 DOF
- video

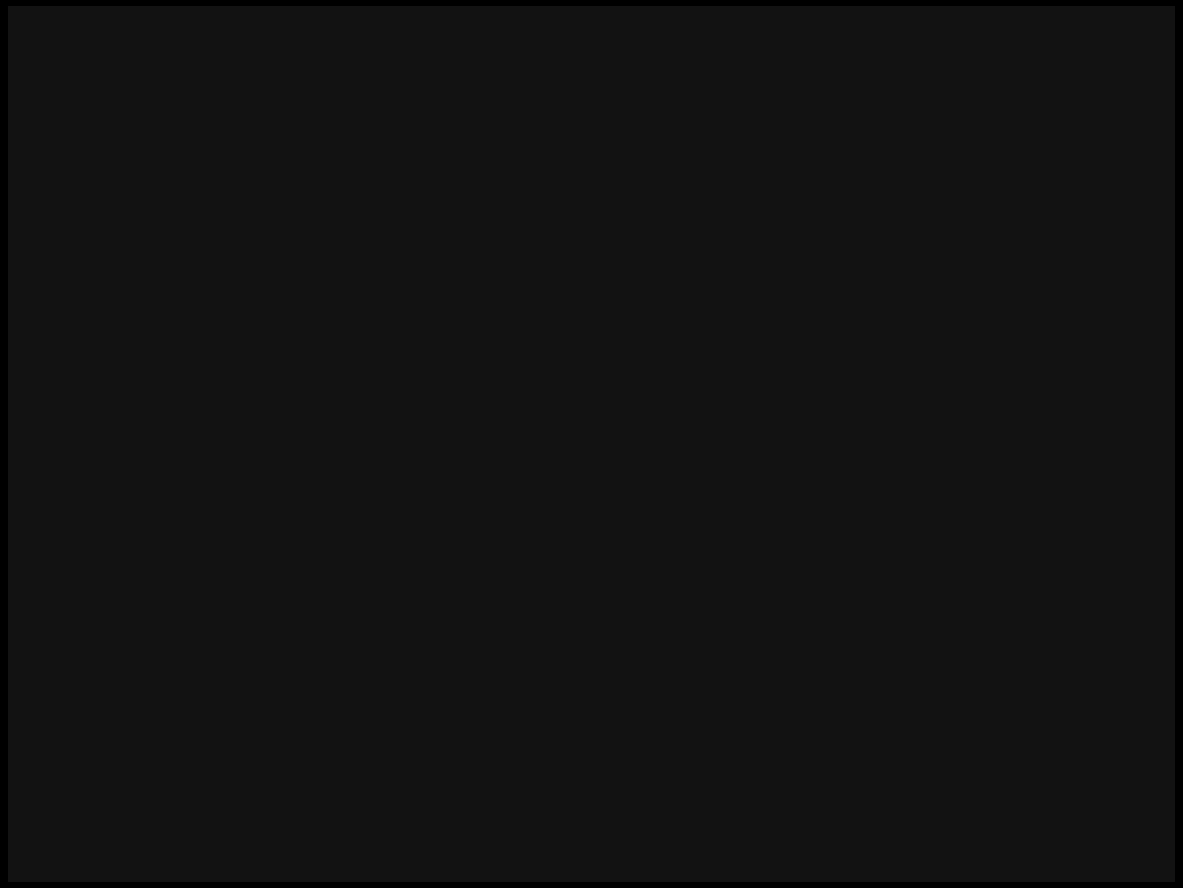
http://www.vrealities.com/glove.html

Motion Editing

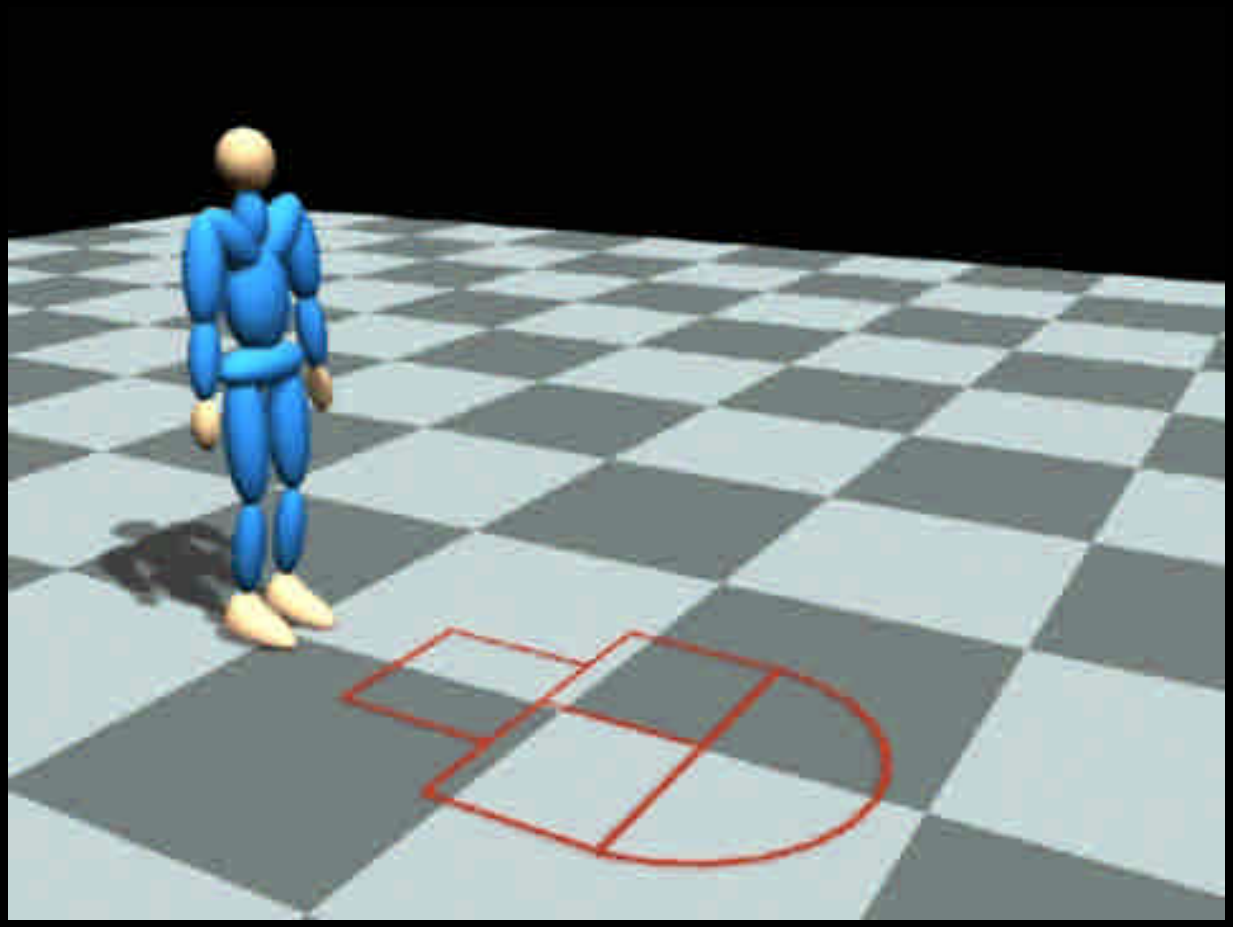
- Get a specific motion
  - from capture, keyframes
  - specific character, action, style
- Want something else while preserving original
  - which part to preserve is case dependent
  - cannot characterize/distinguish motions well enough



Motion Retargeting Video



Optimization-based Motion Synthesis



Liu and Popovic, SIGGRAPH'02

Procedural Animation

- Define the motion using formulas
  - Hand-crafted (rule-based)
  - Physically based
- The animator must be a programmer
- Keyframing starts to become procedural as expressions are added
- At some level of complexity it becomes easier/more efficient than keyframing.

Behavioral Animation

- Animating by describing an actor's behavior
- An actor's behavior defines how the actor interacts with other actors and the environment

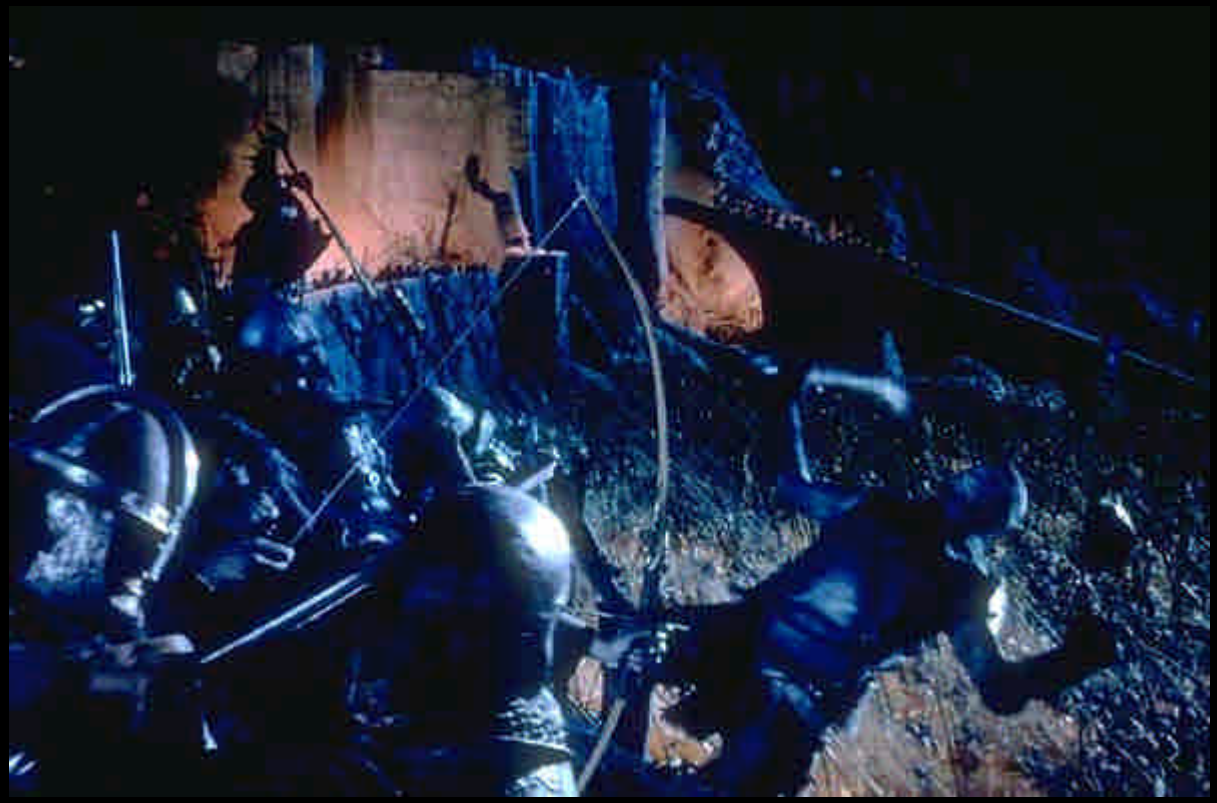
```
TRex()
if(player is close)
        eatPlayer()
else if(can see player)
        chasePlayer()
else
        wander()
```

Group Behavior

- Useful for crowd animations

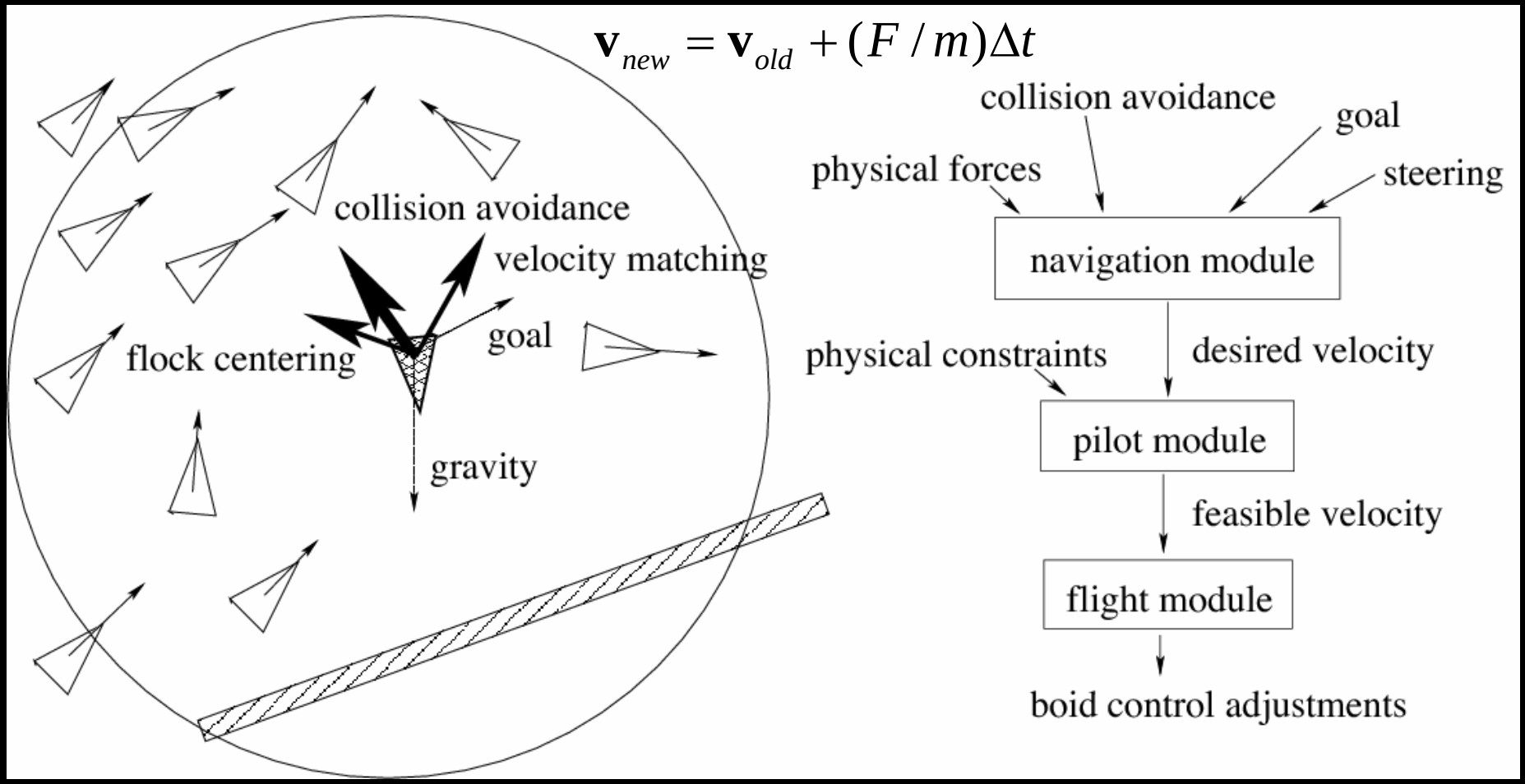
Group Behavior

- Marching Uruk-Hai
- Birds



Battle of Helm's Deep, the Lord of the Rings

Flocking Behavior: Boids, Craig Reynolds



http://www.red3d.com/cwr/boids/

PSCrowd Demo: real-time crowd engine for PS3

Dynamics and Simulation

- Generate motion based on physical laws (e.g., Newton's laws, Fluid dynamics)
- Simulated physical phenomena
  - gravity
  - momentum
  - collision
  - friction
  - fluid flow (liquid, gas, turbulence)
  - flexibility, elasticity, fracture

Dynamics – Particle Systems

Particle Systems [Reeves83]

Represent “fuzzy” objects (such as fire, smoke) as a collection of particles

Particles contain local state

- Position
- Velocity
- Age
- Lifespan
- Rendering properties

Dynamics – Simulated Flames

Duc Quang Nguyen, Ronald Fedkiw and Henrik Wann Jensen, SIGGRAPH 2002

Dynamics – Fluids

Fluid Simulation

- Navier Stokes plus topology changes

Nick Foster and Ronald Fedkiw, SIGGRAPH 2001

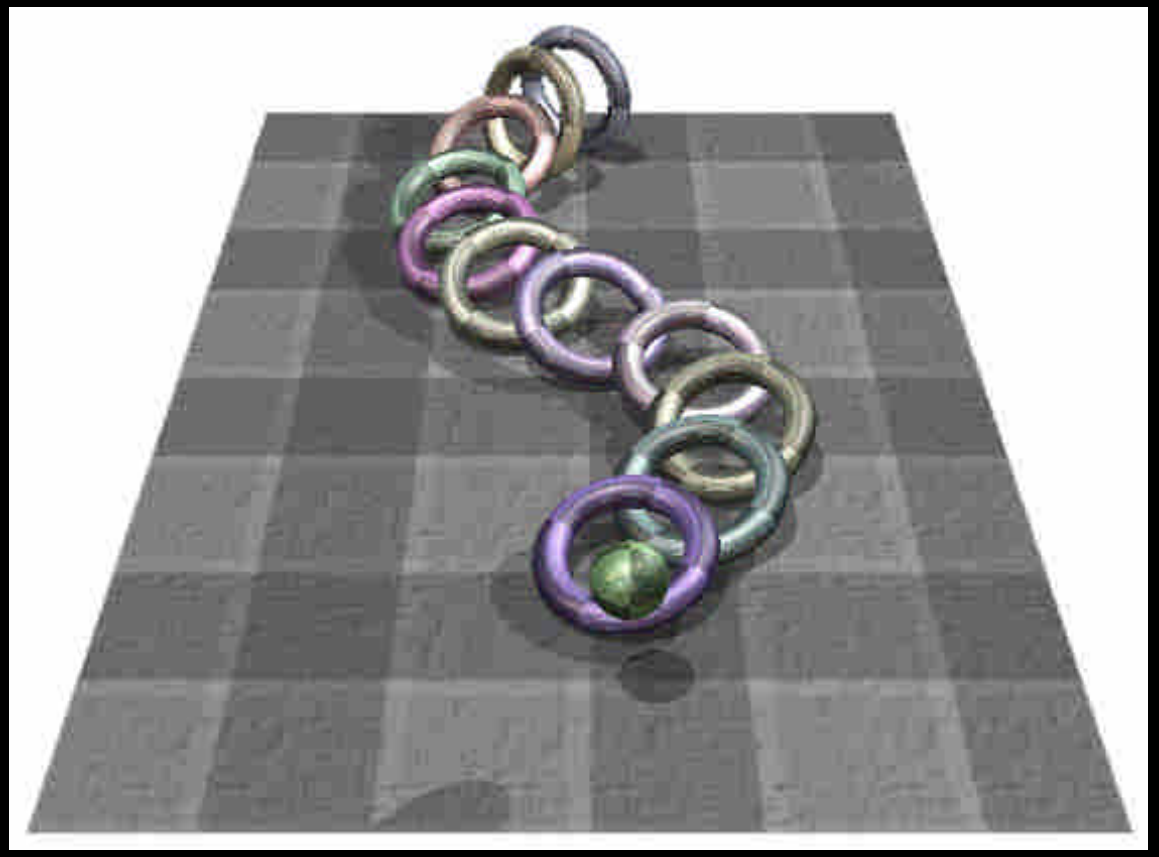
Simulated Water in Films

- Antz water simulation, similar techniques were used in Shrek (Nick Foster & Ronald Fedkiw)

Dynamics – Rigid Bodies

Rigid Bodies

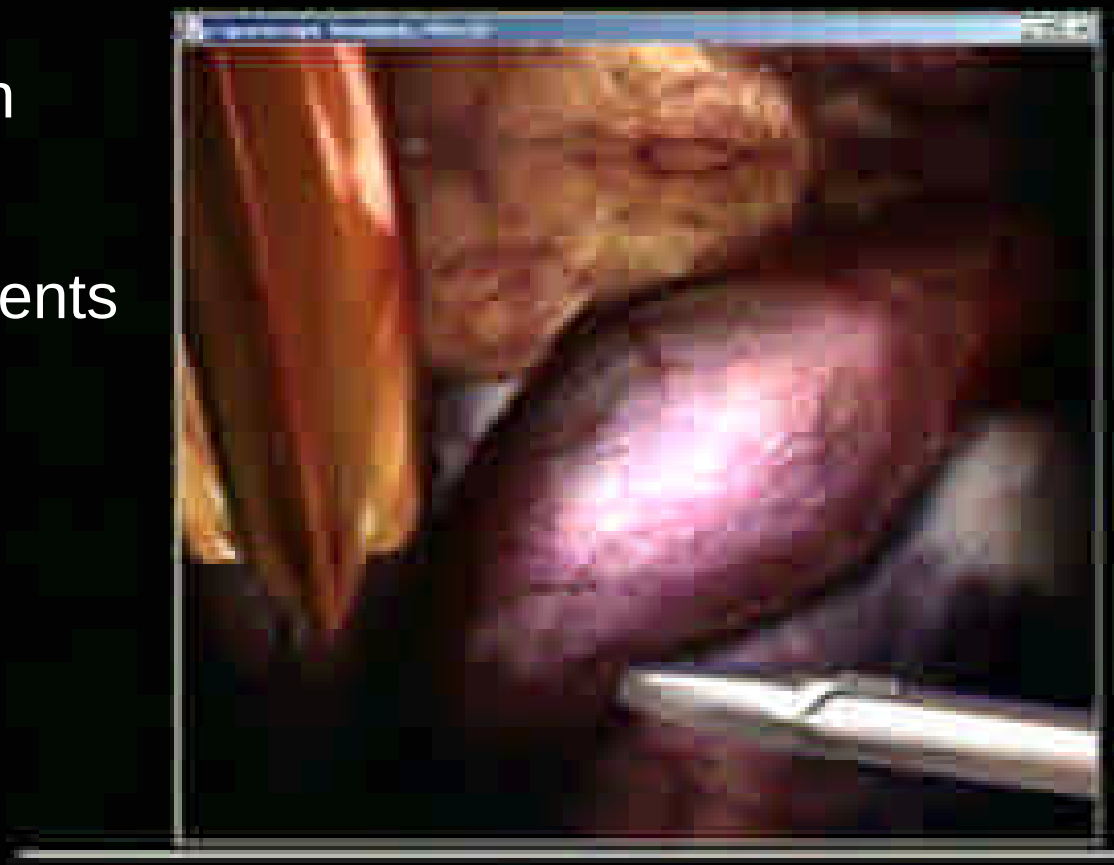
- Integration
- Collisions
- Constraints



Dynamics – Deformable Objects

Deformable Objects

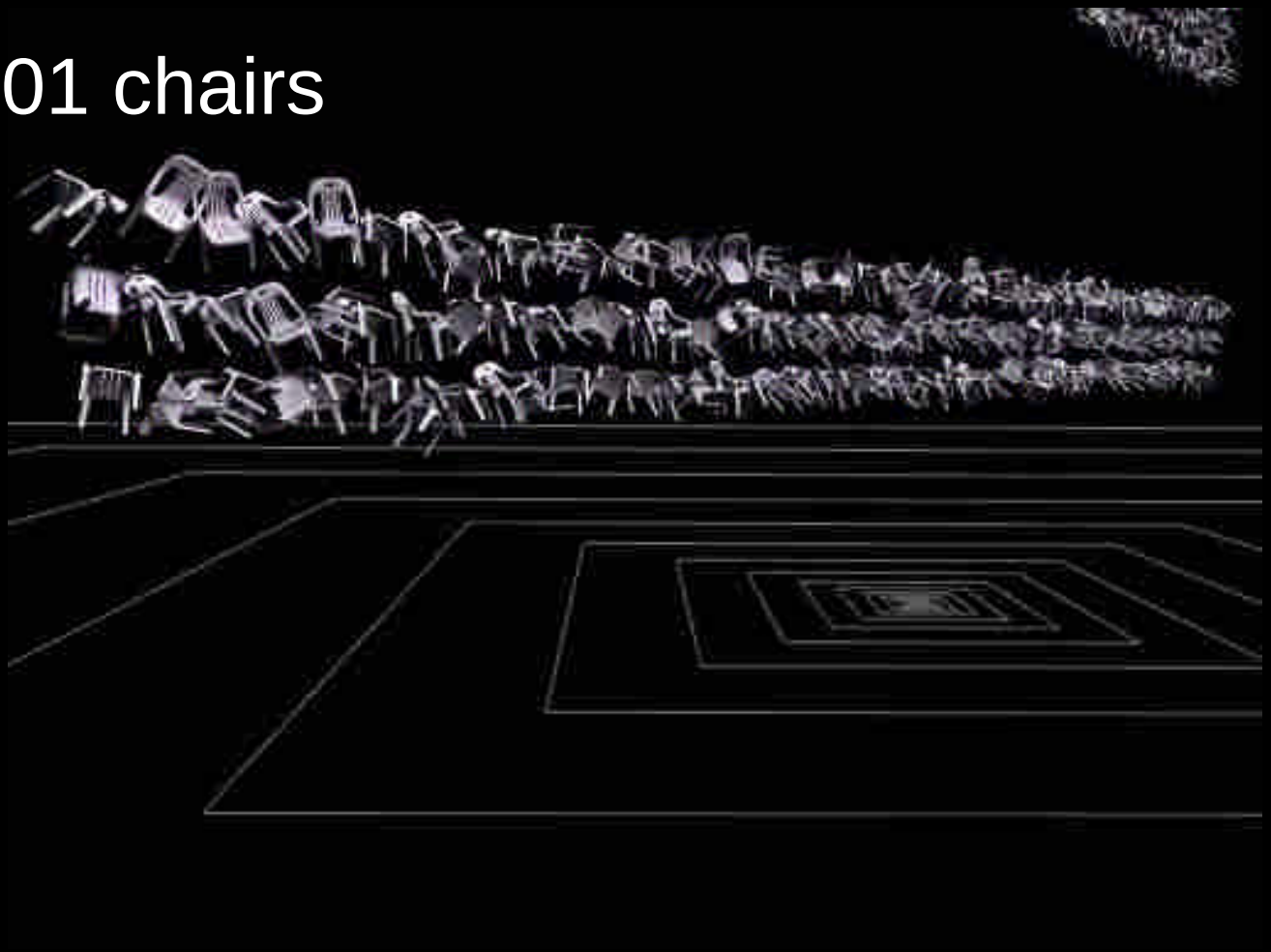
- Free-form deformation
- Elastics
- Finite Elements



Doug James & Dinesh Pai, SIGGRAPH 2002

Simulated Deformable Objects with Collisions

- 12201 chairs

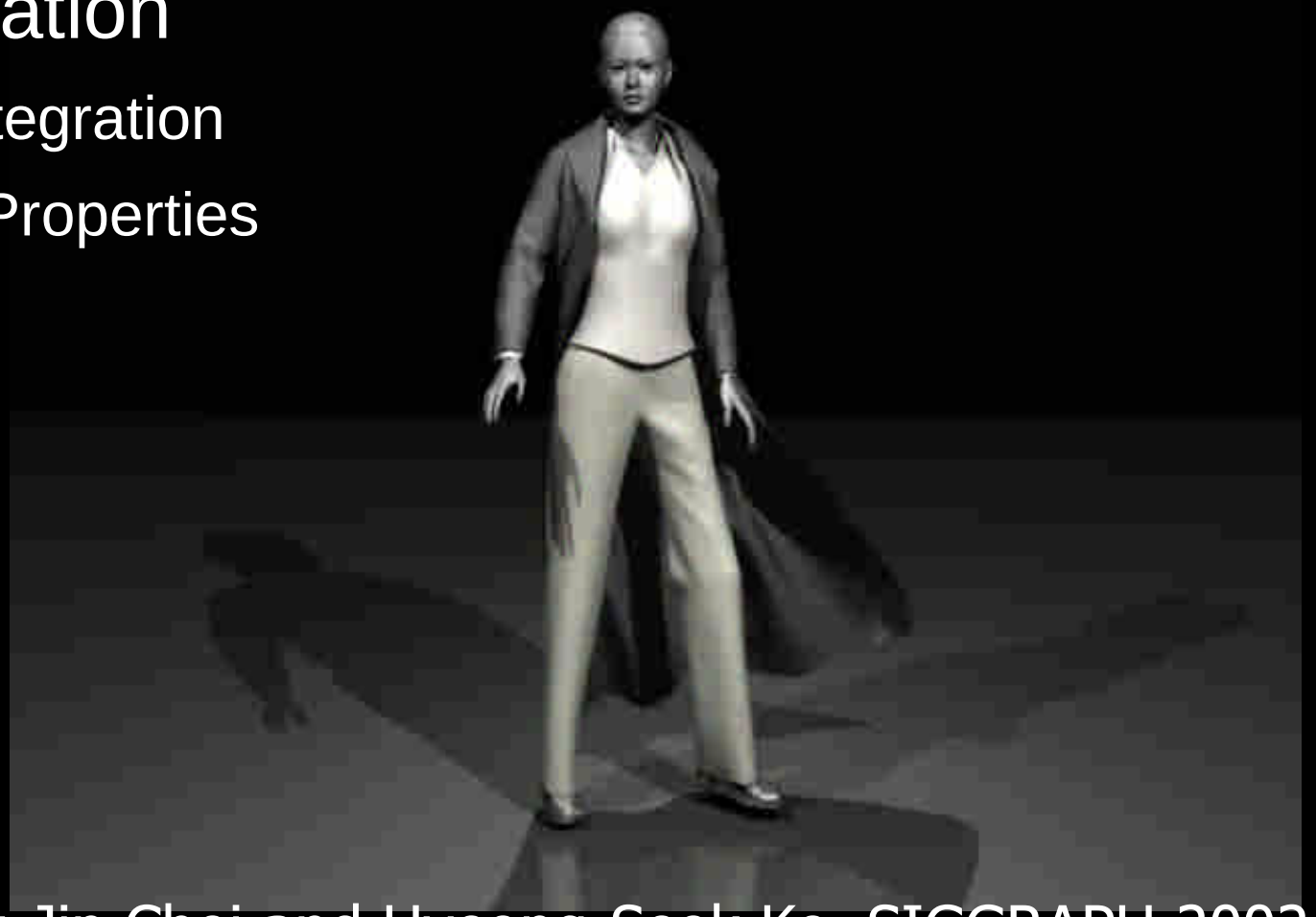


Doug James et al., http://graphics.cs.cmu.edu/projects/bdtree/

Dynamics – Cloth

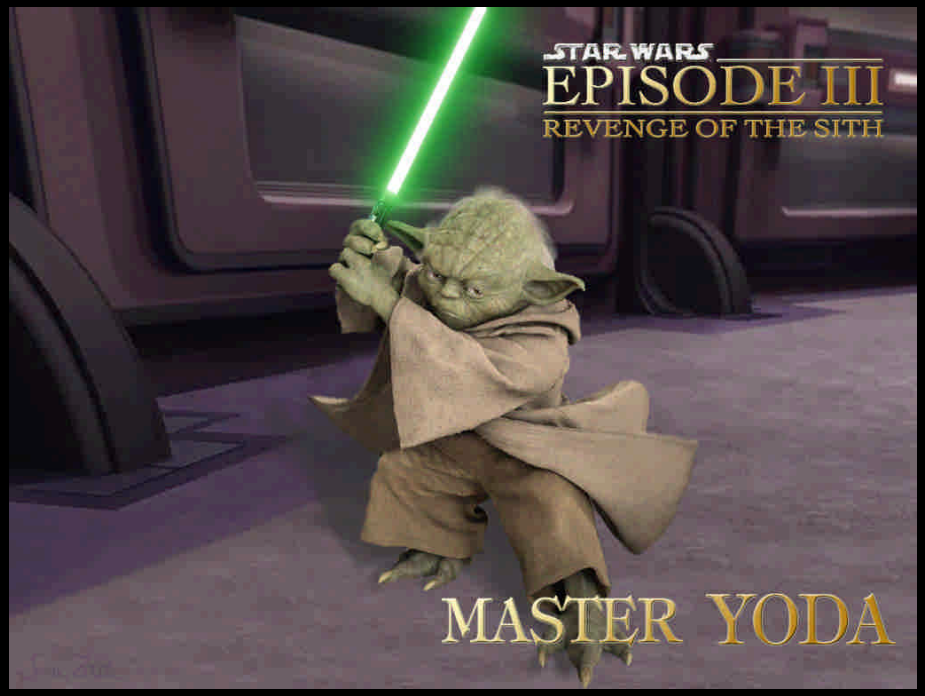
Cloth Simulation

- Stable Integration
- Material Properties

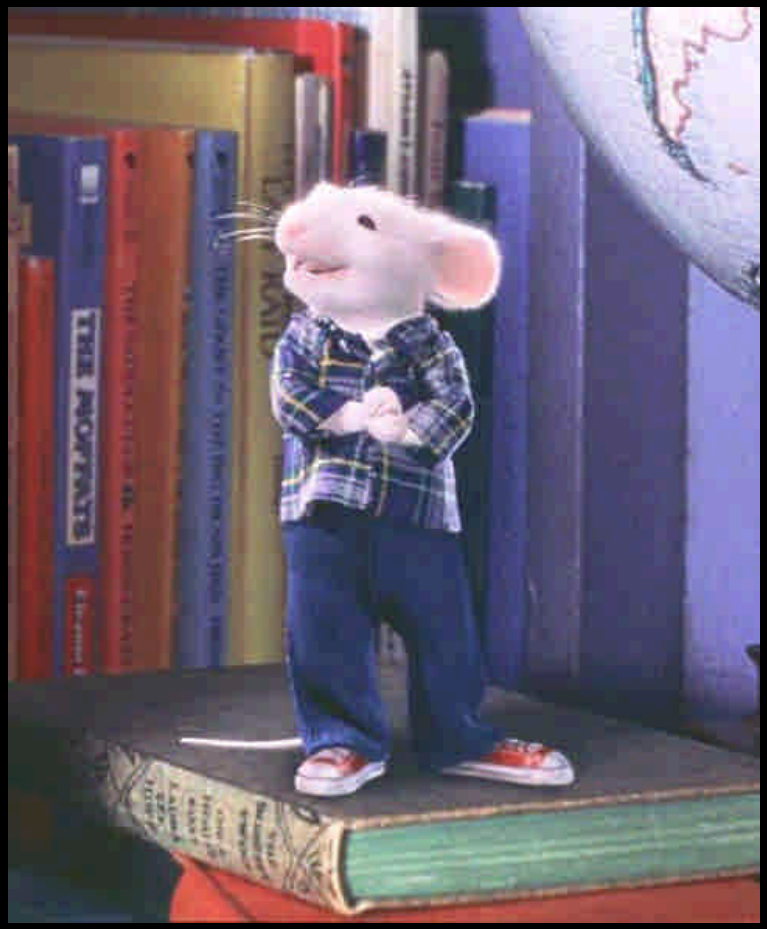


Kwang-Jin Choi and Hyeong-Seok Ko, SIGGRAPH 2002

Simulated Cloth in Films



Star Wars



Stuart Little

Summary

- Keyframing: interpolating between keyframes
- Skeletal animation: human and animals
  - Kinematics: representing and posing a character
  - Motion capture
  - Motion editing and synthesis
- Procedural animation
  - Behavior animation: group and crowd
  - Dynamics and simulation: passive objects