Animation

Animation

Shape Specification as a Function of Time.

Animation Representation

- many ways to represent changes with time
- intent
  - artistic motion
  - physically-plausible motion
- efficiency
  - typically not a major problem
- control
  - most algorithms deals with this

Animation Editing

- different techniques for different processes
- key-framing
  - describe key poses, interpolate the rest
  - man-made process: laborious but artistic
  - good for characters
- procedural animation
  - motion expressed algorithmically
  - good for small secondary motion or special effects
    - e.g. clock animation

Animation Editing

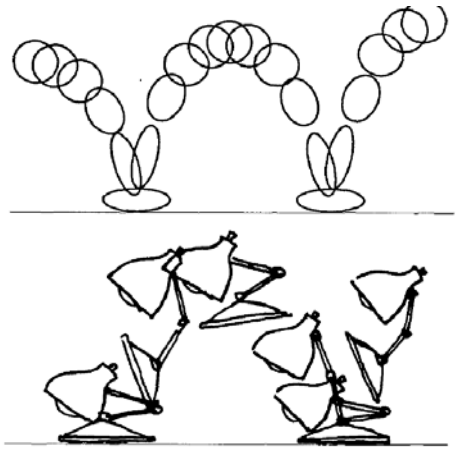
- different techniques for different processes
- motion capture:
  - reproducing performances
  - good for character, but requires lots of hand-tuning
- physically-based simulation
  - assign physical properties
  - simulate physics
  - realistic, but difficult to set up and control style

Principles of Animation

- describe artistic choices in hand-drawn anim.
  - now used to describe computer animation
  - non-technical description
    - can't build algorithms on
    - but useful to think about what the goals are

Principles of Animation

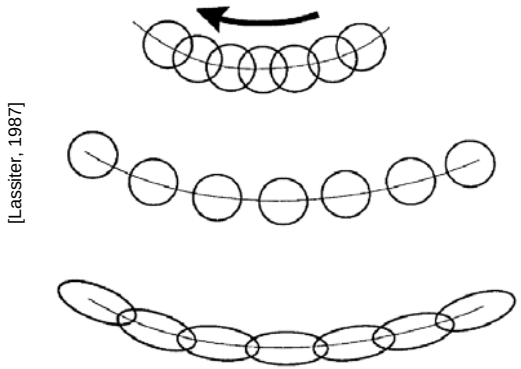
- squash-and-stretch



[Lassiter, 1987]

Principles of Animation

- squash-and-stretch



[Lassiter, 1987]

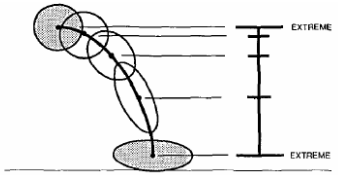
slow motion

fast motion

fast motion w s.s.

Principles of Animation

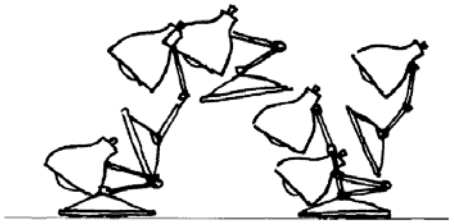
- timing



[Lassiter, 1987]

Principles of Animation

- anticipation



[Lassiter, 1987]

Movie Time

How Animation Works?

- flip very fast a set of fixed images
- perceived as motion by our visual system

- how many images per second?
  - should be above flicker fusion: > 60 Hz
  - NTSC TV signal: 60 half-frames per second
  - movies: 24 fps repeated 3 times

Motion Blur

- avoid aliasing over time
  - equiv. to color “averages” to remove “jaggies”

[Cook et al., 1984]

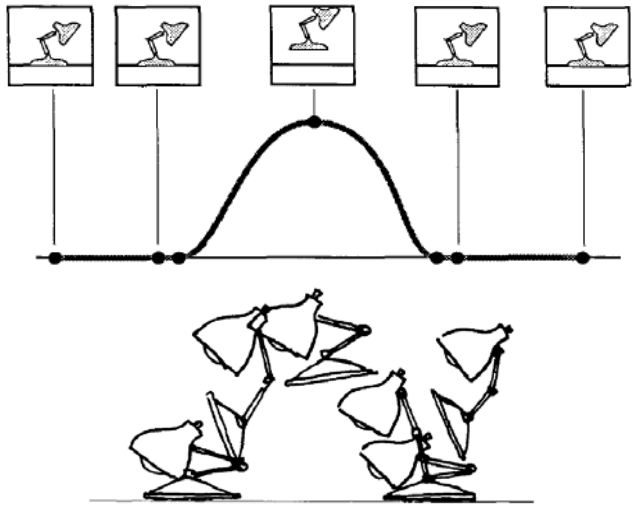
Representing Changes

- one frame-at-a-time
  - inefficient and cumbersome
- key-pose animation
  - define key poses
  - interpolate in the middle

Key-frame Animation

- used in 2d hand-drawn animation
  - head animators define key poses
  - inbetweeners define intermediate poses
- same conceptual framework
  - animator defines key poses
  - computer interpolates intermediate poses

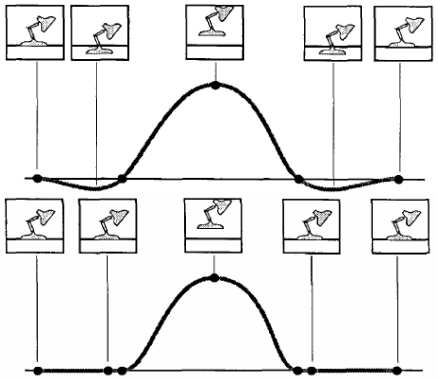
Key-frame Animation



[Lassiter, 1987]

Key-frame Animation

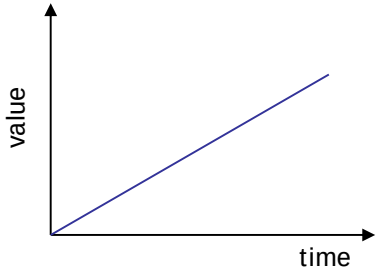
- how to define interpolating function
  - choose smooth curve formulation: splines
  - interpolation is not rock-solid

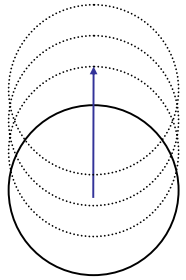


[Lassiter, 1987]

Key-frame Animation

- feedback comes in various forms
  - animation playback
  - parameter curve
  - ghosting





Key-frame Animation

- what to interpolate?
  - shape are defined by control points
    - too many controls for animation purposes
  - express *deformation* with meaningful parameters
    - deformation: changes in shape
- degrees of freedom
  - modeling: number of control points
  - animation: parameters of deformations
  - ui: parameters of manipulators
  - use smallest number of degrees of freedom

Deformation Examples

- rigid body transformation
  - translation/rotation
  - shape is unchanged

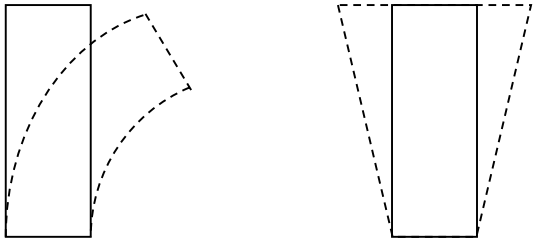
$$\mathbf{P}' = M \times \mathbf{P}$$

- modeling representation
  - move all the control points: *n* x 3 DOFs
- deformation/animation transformation
  - translation + rotation vector: 6 DOFs

Deformation Examples

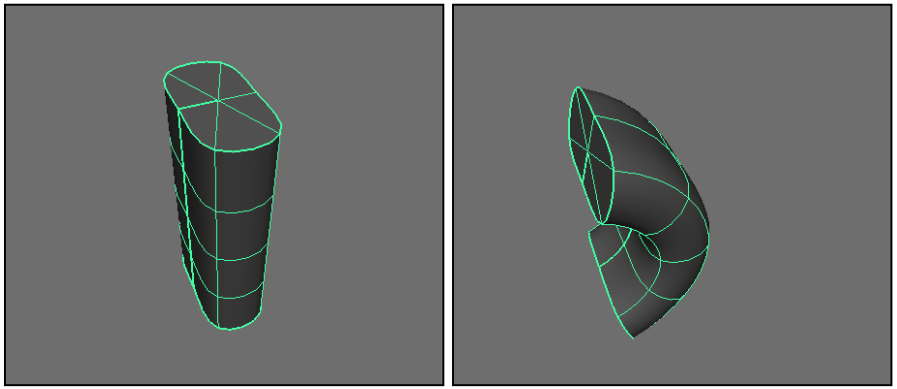
- deformations change shape
  - introduce different functions
  - limits on the type of deformation

$$\mathbf{P}' = f(\mathbf{P}, \{\alpha_i\})$$



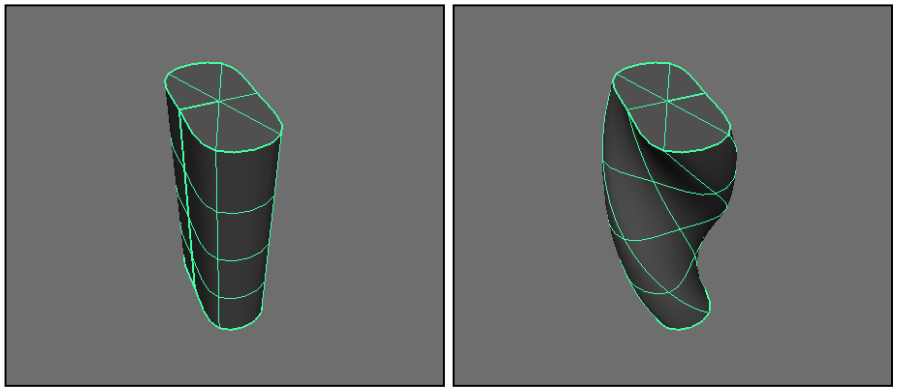
Deformation Examples

bend



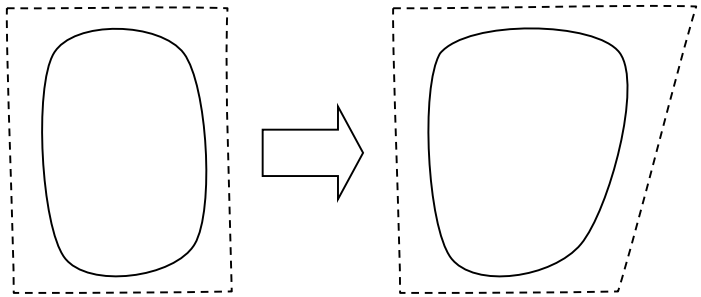
Deformation Examples

twist



Deformation Examples

- using a lattice of control points
  - key idea: size of lattice is smaller than model



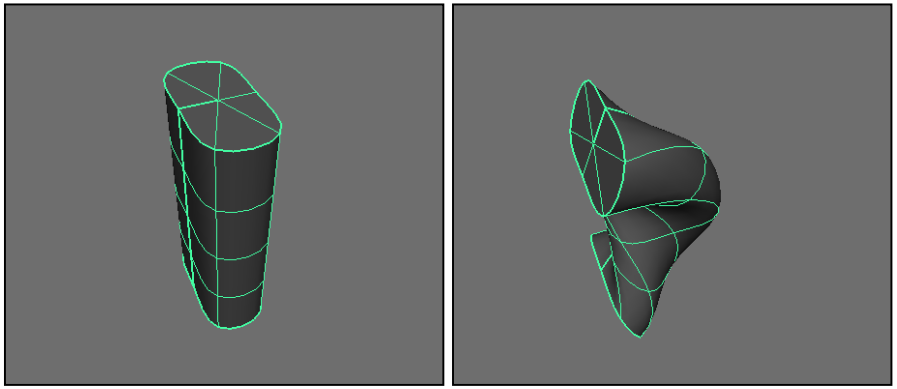
Complex Deformations

- complex deformations by function composition
  - no unified description
  - so apply one after the other

$$\mathbf{P}' = f_1(f_2(\mathbf{P}))$$

Complex Deformations

bend + twist



Deformation and Control Points

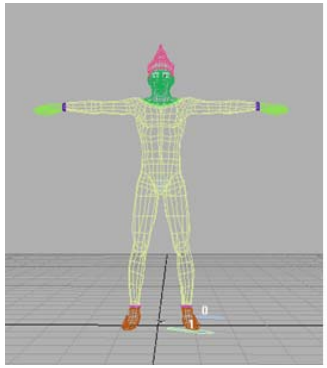
- should we deform control points or the surface?
- in general, deforming control points is wrong
  - cannot prove that the surface is equivalent
- in practice, deforming control points is ok
  - control mesh is tessellated enough
  - many useful transform are well-behaved

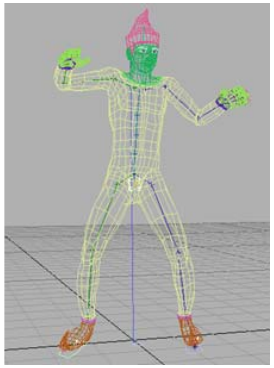
Deformations for Characters

- often combination of lots of deformations
- specialized deformations
  - mesh skinning: body deformation
  - blend shapes: face deformation

Mesh Skinning

- deform surface around a skeleton

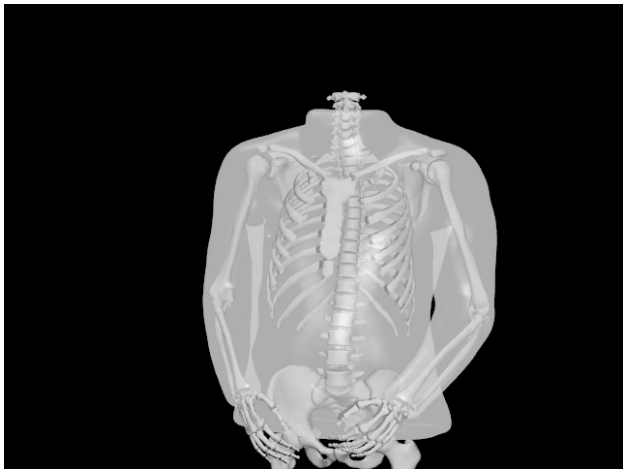




[Domine/NVIDIA]

Mesh Skinning

- concepts based on skin/bone interactions



[Fedkiw et al.]

Mesh Skinning

- every bone has a transform M_j
- every bone-vertex has a weight w_{ij}
  - user provided, often zeros for most pairs
  - typically not zero only for close enough "bones"
- deformation is weighted average of positions transformed by every bone
  - weighted by the vertex weight

$$\mathbf{p}_i' = \sum_j w_{ij} M_j \mathbf{p}_i$$

Mesh Skinning

- deformation often defined for a rest pose
  - position are defined in model space
  - reference matrices for model-to-bone transform

$$\mathbf{p}_i' = \sum_j w_{ij} M_j M_{ref\ j}^{-1} \mathbf{p}_i$$

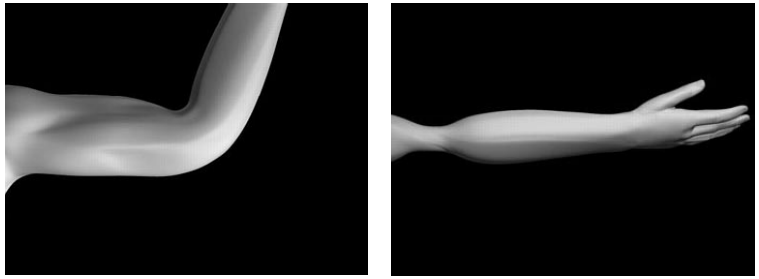
- normals use inverse-transpose formulation

Mesh Skinning

- solution for body deformation
- efficient to compute
  - hardware acceleration available
- good control
  - but hard to set up proper weights
- often used in games as-is
- used in movies as part of more complex setups

Mesh Skinning Issues

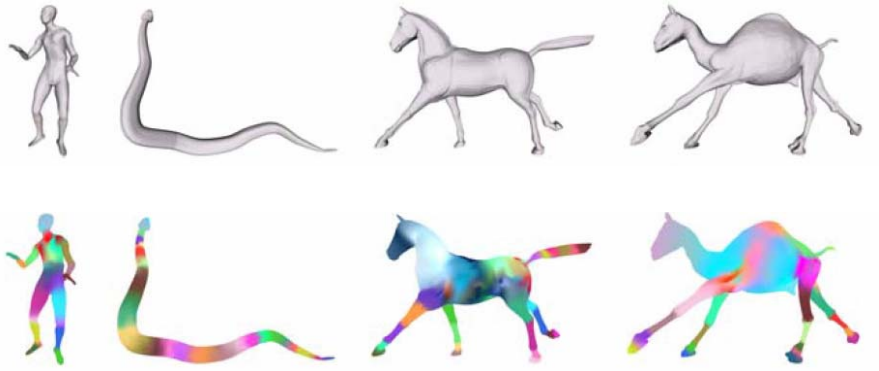
- surface collapse
  - around joints or for strong twists
- hard to fix
  - cannot be fixed by tweaking weights



[Lewis et al., 2000]

Mesh Skinning - Defining Weights

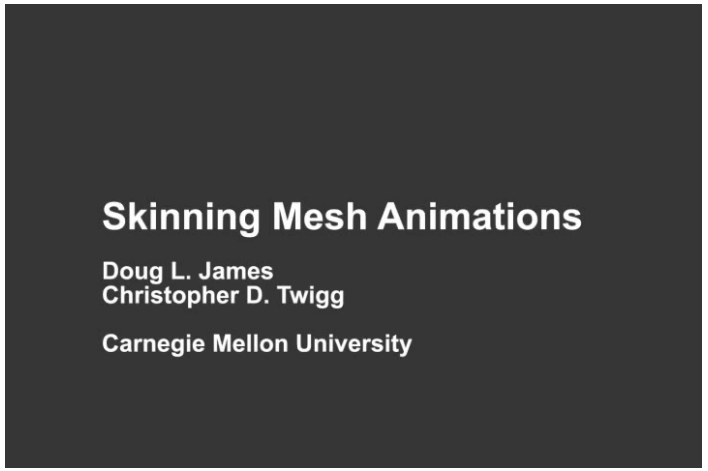
- often very time consuming: active research



[James and Twigg, 2005]

Mesh Skinning – Efficiency

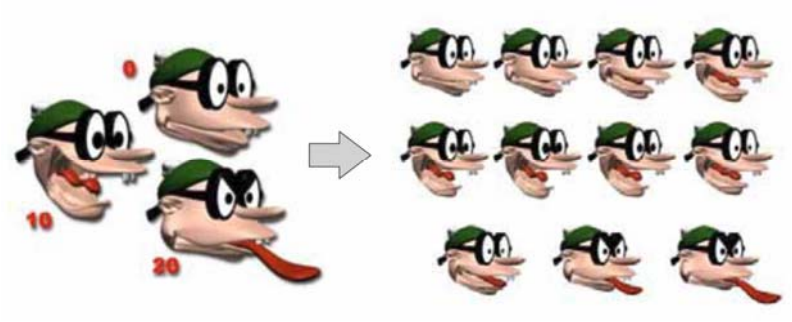
- hard to use, but really fast implementations



[James and Twigg, 2005]

Blend Shapes

- interpolate set of meshes



[3DMax docs/Discreet]

Blend Shapes

- user provides a set of meshes with same topology
- final mesh is weighted sum of base meshes
  - weights have to sum to 1

$$\mathbf{p}_i' = \sum_j w_j \mathbf{p}_{ji} \qquad \sum_j w_j = 1$$

Blend Shapes

- solution for face deformation
- efficient to compute
  - albeit increase memory usage
- great control
  - but only works for small deformation
  - cannot produce “novel” shapes
- often used in games

Interpolating Deformations

- just interpolate deformation parameters
  - translation: position
  - rotation: quaternions
  - bending/twisting: angle
  - skinning: skeleton translation/rotation
  - blending: blend weights

Interpolating Translations

- linearly blend the translation keys

$$\mathbf{v}(t) = (1 - f(t))\mathbf{v}_0 + f(t)\mathbf{v}_1$$

  - for linear *f*, v(0.5) is the midpoint between the keys

- define a spline passing by the translation values
  - additional controls define speed and acceleration

Interpolating Rotations

- how many degrees of freedom do rotations have
  - 3, 1 for angle and 2 for direction

- what is a good representation for rotation?
  - matrices
  - Euler angles
  - quaternions

Interpolating Rotations

- using matrices is problematic

$$M(t) = (1 - f(t))M_0 + f(t)M_1$$

  - for linear *f*, M(0.5) may not be a rotation
    - e.g. M_0 is identity, M_1 is 90° rotation around x
    - M(t) is not a rotation, since MM^T is not I

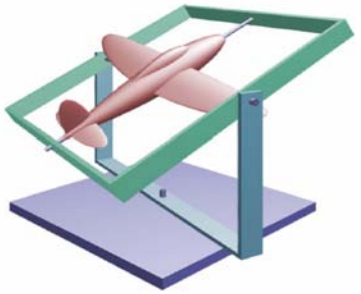
$$M(0.5) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & -0.5 & 0.5 \end{bmatrix}$$

[adapted from MIT course]

Interpolating Rotations

- Euler angles
  - rotation around 3 different axis
  - can represent any rotation

- interpolation is unnatural
  - 90°around Z then Y = 120° around (1,1,1)
  - 30°around Z then Y = 40° around (1,1,1)

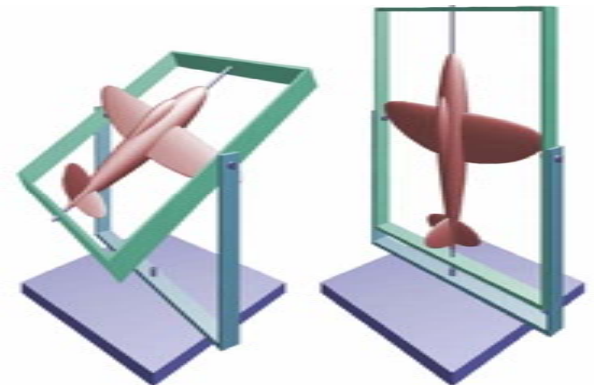


[Hoffman, http://www.fho-emden.de~Ehoffmann/]

[adapted from MIT course]

Interpolating Rotations

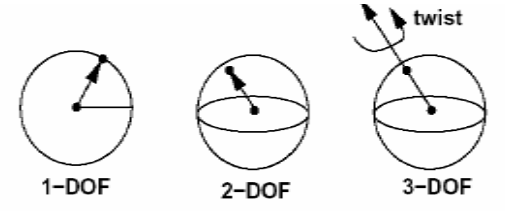
- gimbal lock
  - may look degrees of freedom when interpolating



[Hoffman, http://www.fho-emden.de~Ehoffmann/]

Interpolating Rotations

- matrices: incorrect rotation
- Euler angles: unnatural and gimbal lock
- quaternions: nice mathematical framework
  - won't cover in depth
  - intuition: interpolate rotation as point on sphere



[adapted from MIT course]

Providing Deformation Parameters

- kinematics
  - provide transformation parameters directly
  - hand-editing
    - forward kinematics
    - inverse kinematics
  - motion capture
- dynamics
  - solve physics equations of motion

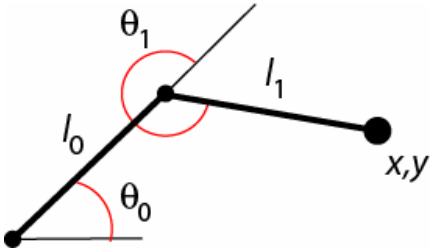
Forward Kinematics

- artists defines transformation parameters directly
- hierarchical transformations
  - used for bone structures in character animation
    - e.g. skeletons or robots
  - hard to define what happens at end of chains
    - e.g. which angles should the leg be to have the foot touch the floor?
    - done by trial and error

Forward Kinematics

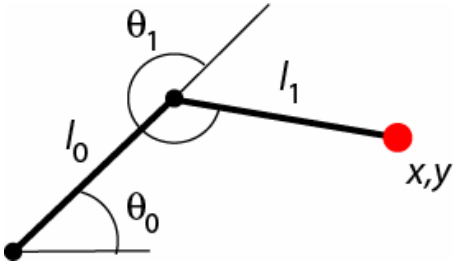
- position at end of the chain

$$p_x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \qquad p_y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$



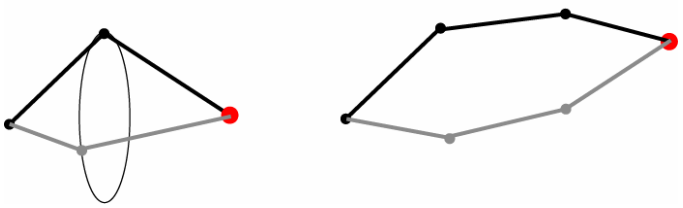
Inverse Kinematics

- specify directly the position at the end of chain
  - easier to control motion, less trial and error
  - joints angles solutions by inverting previous eqs.



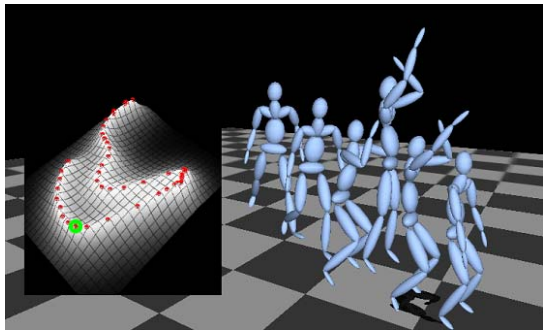
Inverse Kinematics

- more bones results in under-constrained system
  - infinite number of solutions
  - which solution to pick?
  - impose constraints: minimize energy function based on plausible motion



Inverse Kinematics

- or try to capture "styles"
  - by learning from data sets



[Grochow et al., 2004]

Motion Capture

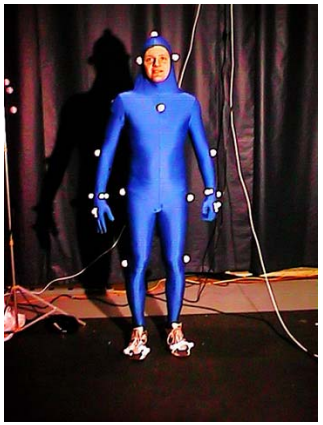
- record motion and play it back
  - how to record: motion capture systems
  - how to apply motion to digital characters
    - motion editing
    - motion retargeting

Motion Capture Usage

- heavily in games, a bit in movies
  - not very expressive, but more high expectation

Motion Capture Systems

mechanical optical



[Popovic]

Motion Capture Editing

- motion capture generates too much raw data
  - how to edit it? try to fit with lower DOFs models
- motion retargeting
  - capture from actor A, but apply to actor B
  - how to do this in a believable manner?
- clean up motion
  - noise present in data / too little DOFs
  - how to clean it up?
- often just starting point for manual animation

Kinematics vs. Dynamics

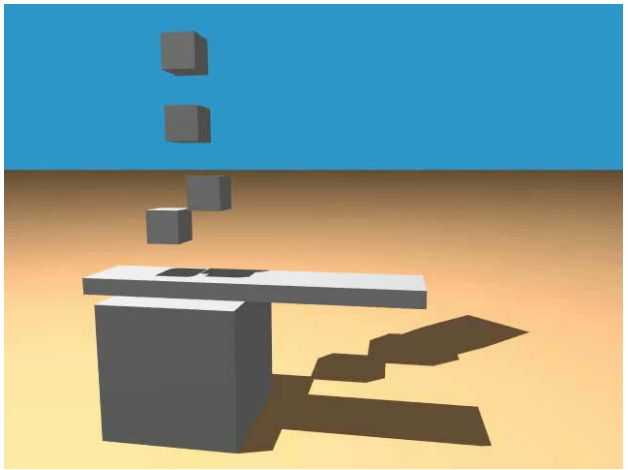
- kinematics: specify parameters directly
- dynamics: solve the equations of motion
  - physically based animation
- rigid body dynamics
  - solve rigid body equations
  - collision detection
  - doable in many cases
- more complex cases almost impossible
  - cannot model physics accurately enough
  - simply for good-enough solutions

Dynamics

- animation from dynamics is accurate
  - since we are simulating physics
- at the price of less artistic freedom
  - cartoon physics anyone?
- control-vs-correctness triage often hard
  - interests in mixing dynamics with kinematics
  - open research issue

Dynamics

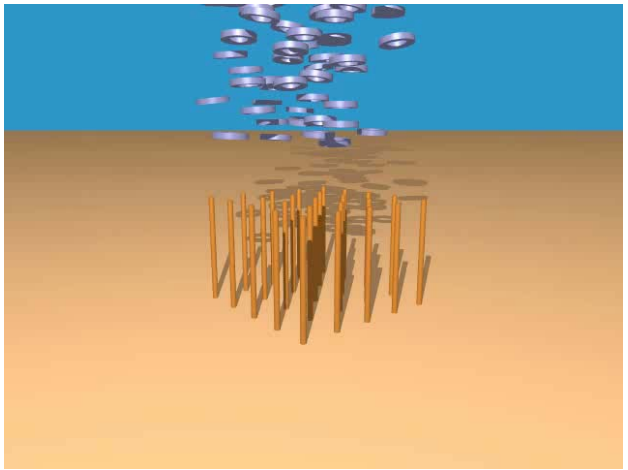
- simulating simple objects



[Fedkiw et al.]

Dynamics

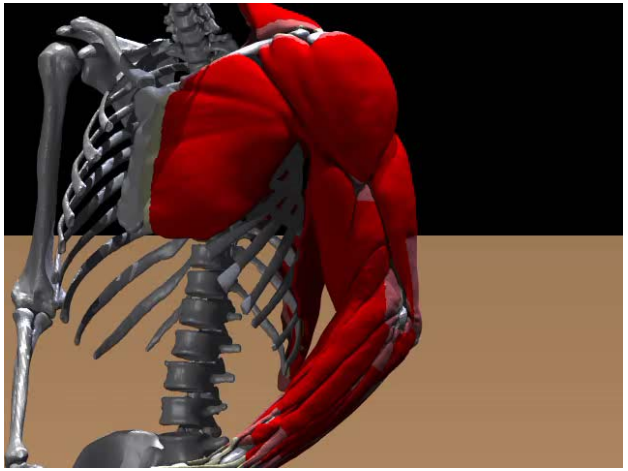
- simulating complex situations



[Fedkiw et al.]

Dynamics

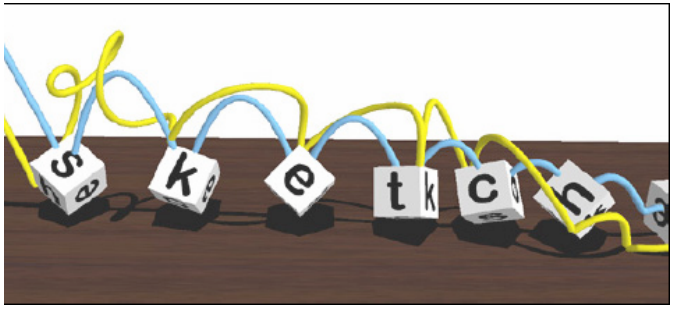
- simulating complex objects



[Fedkiw et al.]

Controlling Dynamics

- basic principle: cheat where you can



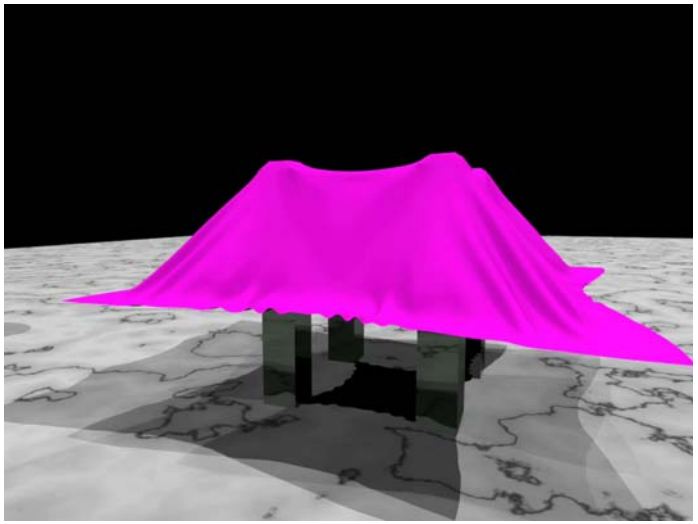
[Popovic et al., 2003]

Movie Time

Natural Phenomena

- often done by physical simulation
  - looks like computational physics
- simulation domain
  - choose based on phenomena to define
  - e.g. smoke uses volumetric adaptive grids
  - e.g. cloth uses points/springs systems
- simulation algorithms
  - very different ones depending on simulation domain
    - lots of open research

Natural Phenomena

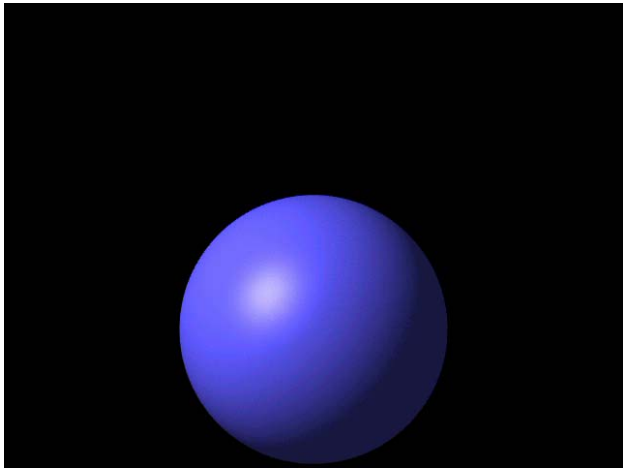


[Fedkiw et al.]

Natural Phenomena

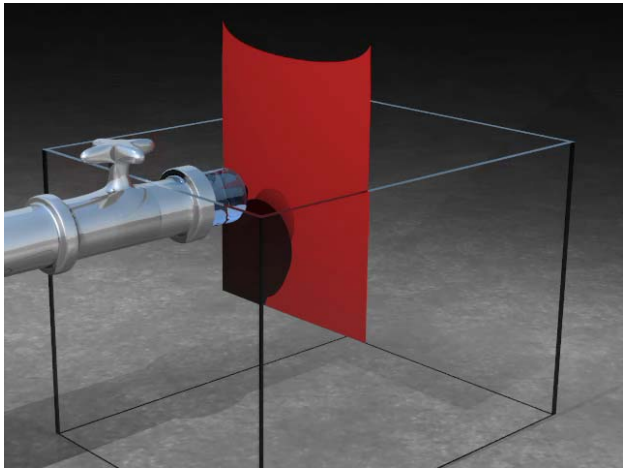
[Fedkiw et al.]

Natural Phenomena



[Fedkiw et al.]

Natural Phenomena



[Fedkiw et al.]

Particle Systems

- collection of particles
  - simple, since it is just simulating point dynamics
  - used heavily in special effects
    - complex phenomena represented as point/force collections
  - simplest dynamics formulation
- point properties
  - dynamics: position/velocity/acceleration
  - varying properties: color/temperature/lifespan
  - constant properties: mass/lifetime

Particle Systems

- for each frame
  - create new random particles
    - where to create? along point/line/surface
    - artistic control
  - delete expired particles
    - random/lifespan/collision
  - update particles based on dynamics
  - render particles

Particle Dynamics

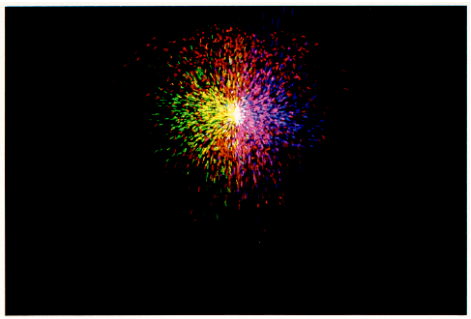
- Newton equation

$$\mathbf{v} = \frac{d\mathbf{p}}{dt} \qquad \mathbf{a} = \frac{d^2\mathbf{p}}{dt^2} \qquad \mathbf{F}(\mathbf{p}, \mathbf{v}, t) = m\mathbf{a}$$

- find position at time t
  - given position, velocity and acceleration at time 0
  - initial value problem: use Euler method
    - more efficient methods exist

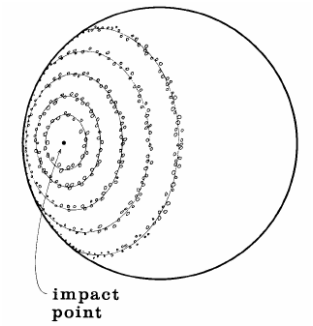
$$\begin{aligned} \mathbf{p}(0) &= \mathbf{p}_0 \\ \mathbf{v}(0) &= \mathbf{v}_0 \\ \mathbf{a}(0) &= \mathbf{a}_0 \end{aligned} \longrightarrow \begin{aligned} \mathbf{v}(t+\Delta t) &= v(t) + \mathbf{a}(t)\Delta t \\ \mathbf{p}(t+\Delta t) &= \mathbf{p}(t) + \mathbf{v}(t)\Delta t + \mathbf{a}(t)\Delta t^2 / 2 \end{aligned}$$

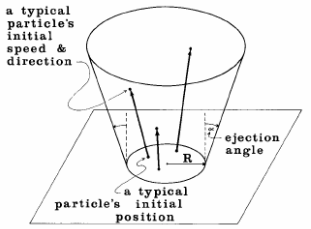
Particle Systems Example



[Reeves, 1983]

Particle Systems Example



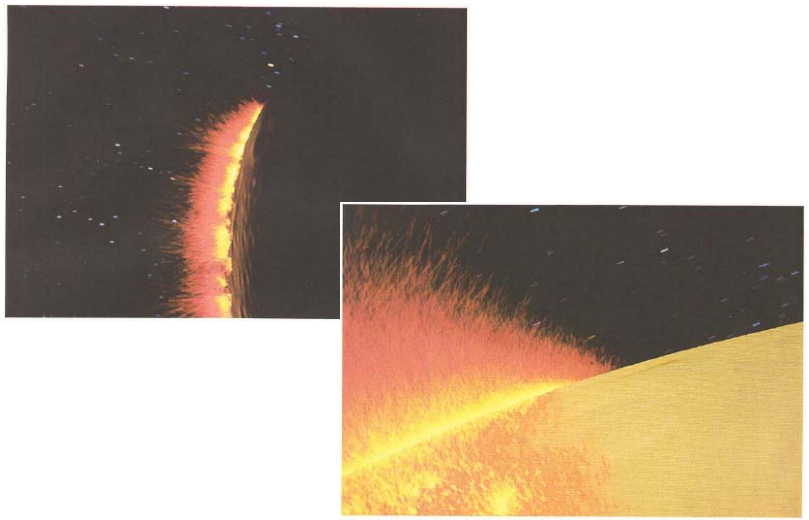


[Reeves, 1983]

Particle Systems Example

[Reeves, 1983]

Particle Systems Example



[Reeves, 1983]