15-462 Computer Graphics I

Lecture 12

Physically-Based Modeling

Overview
Newtonian Particles
Spring Forces
Solving Particle Systems
Constraints

[Angel, Ch 11.1-11.5]

March 12, 2002
Frank Pfenning
Carnegie Mellon University

http://www.cs.cmu.edu/~fp/courses/graphics/

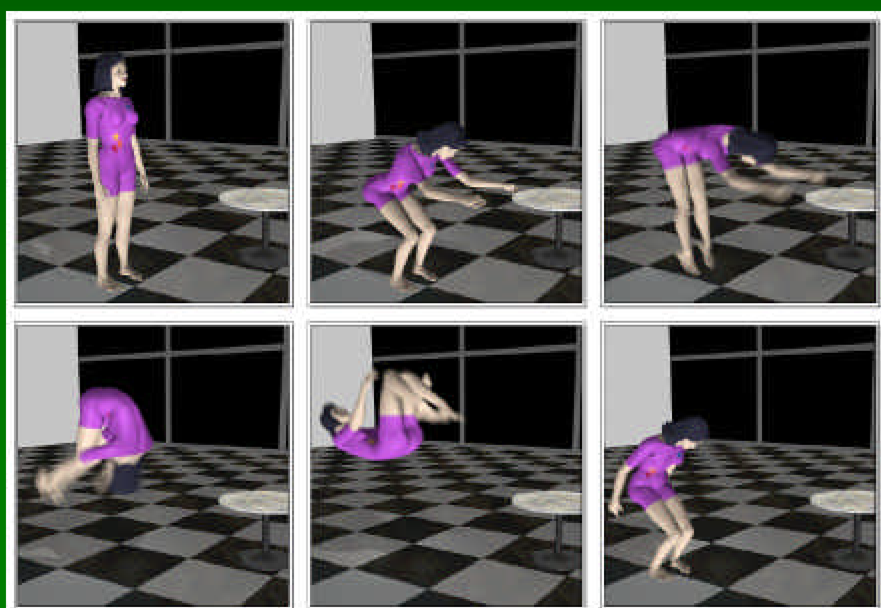
Motivation

- Animation
  - Frame-by-frame or keyframing (artist’s intuition)
  - Performance-based (motion capture)
  - Physically-based (dynamic simulation)
  - Trade-offs and combinations
- Interaction
  - Model deformation and acquisition (haptics)

Dynamic Simulation

- Realistic motion
- High-level control
- Passive vs active systems
- Control and constraints

Antz PDI/Dreamworks



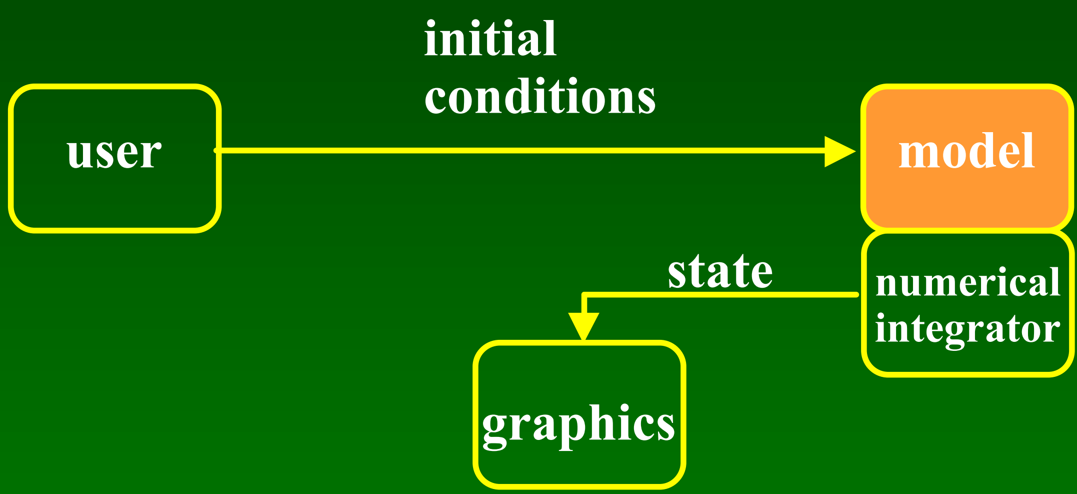
J. Hodgins

Scientific Computation and Graphics

- Scientific computation
  - Specified error tolerance
  - Must be numerically accurate to tolerance
  - Often difficult to compute and unstable
- Computer graphics
  - Viewer's tolerance dominates
  - Need not be numerically accurate
  - Can be faster to compute
- Cross-fertilization
  - Scientific visualization
  - Physically-based modeling

Passive Systems

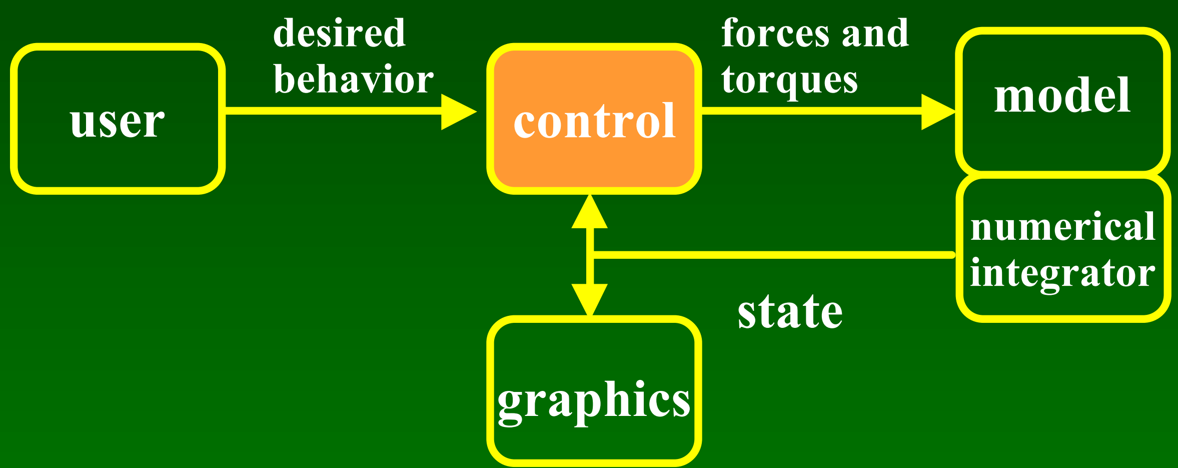
- No muscles or motors (no internal sources of energy)



- Examples
  – Particle systems
  – Leaves
  – Water spray
  – Clothing

Active Systems

- Internal source of energy or control



- Examples
  - Running human
  - Swimming fish
  - Driven car
  - Airplane

Dynamics

- Generating motion by applying physical laws
  - Typical: Newton’s laws, Hook’s law
  - Particles, soft objects, rigid bodies
- Simulates physical phenomena
  - Gravity
  - Momentum (inertia)
  - Collisions
  - Friction
  - Fluid flow (drag, turbulence, surface tension, ...)
  - Solidity, flexibility, elasticity
  - Fracture

Outline

- Overview
- Newtonian Particles
- Spring Forces
- Solving Particle Systems
- Constraints

Newtonian Particles

- Basis for many forms of simulation
- m**a** = **f**
  - Scalar m: mass
  - Vector **a**: accelaration
  - Vector **f**: force
- Particle determined by position and velocity

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \\ \frac{dz}{dt} \end{bmatrix}$$

Newton's Law

- Six coupled first-order differential equations

$$\begin{aligned}\dot{\mathbf{p}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \frac{1}{m}\mathbf{f}(t)\end{aligned}$$

- Rendering of each particle is orthogonal issue
  - Just a dot
  - Person or animal
- Can simulate many independent particles

Simple Pseudocode Template

- Assume n particles

```
Float time, delta;
Float state[6n], force[3n];
State = get_initial_state();
For (time = init; time < final; time += delta)
{
  force = force_function(state, time);
  state = ode(force, state, time, delta);
  render(state, time);
}
```

Independent Particles

- Example: gravity for particle *i* :

$$\mathbf{f}_i(\mathbf{p}_i, \mathbf{v}_i) = \mathbf{f}_i = \mathbf{g} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix}$$

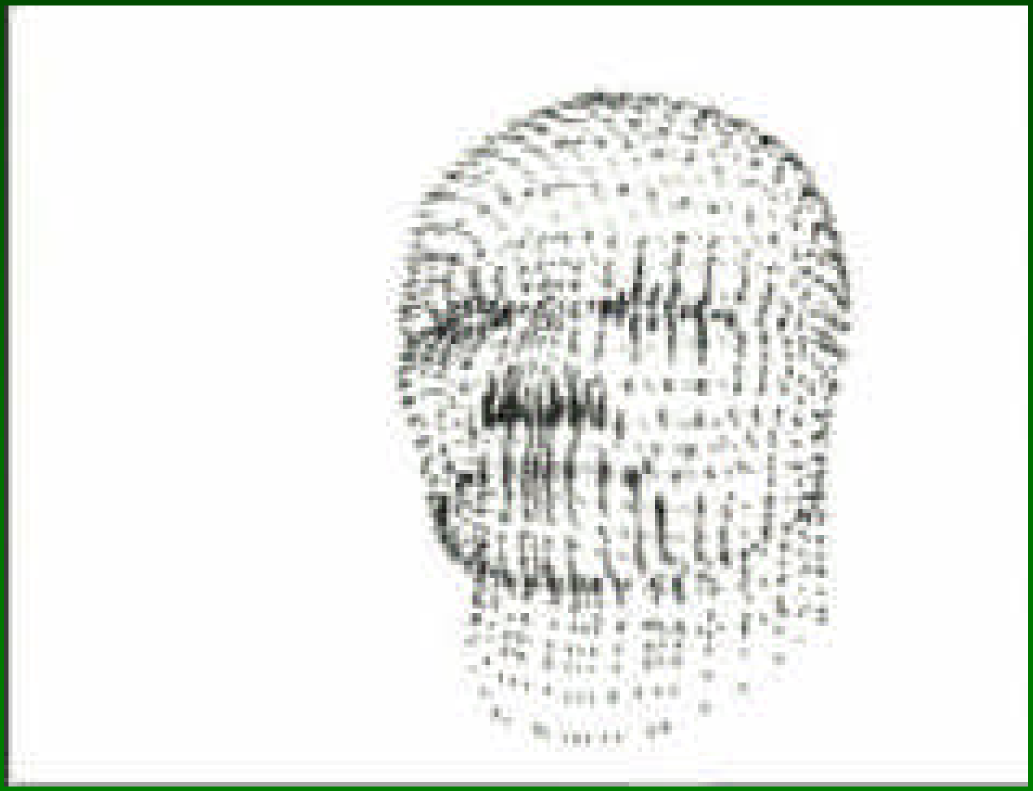
- Simple approximate computation

$$\begin{aligned} \mathbf{v}' &= \mathbf{v} + \mathbf{g}\Delta t \\ \mathbf{p}' &= \mathbf{p} + \frac{\mathbf{v}+\mathbf{v}'}{2}\Delta t \end{aligned}$$

- May be enough, depending on application
- Numerical integration is more general solution

Particle Systems, Example I

- Clip from Karl Sims, *Particle Dreams*, 1988



Particle Systems, Example II

- Clip from *Star Trek II: The Wrath of Khan*

Particle Systems: Other examples

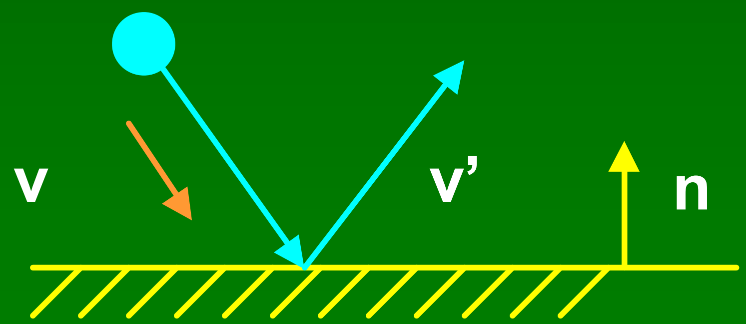
- Clouds
- Smoke
- Fire
- Waterfalls
- Fireworks
- *Batman Returns*, using Reynolds' flocking algs.
- *The Lion King*, Wildebeest stampede

Particle Systems

- Creation
  - Number
  - Position and velocity
  - Shape, size, material properties
  - Lifetime
- Update of position and velocity
- Deletion
- Rendering style
  - Motion blur
  - Compositing

Other Forces

- Unary
  - Gravity: **f** = m **g**
  - Viscous drag: **f** = -k **v**
- N-ary
  - Spring: **f** = -k **d**
- Spatial interaction force
  - Collisions
  - **v' = v** – 2(**v** · **n**)**n**

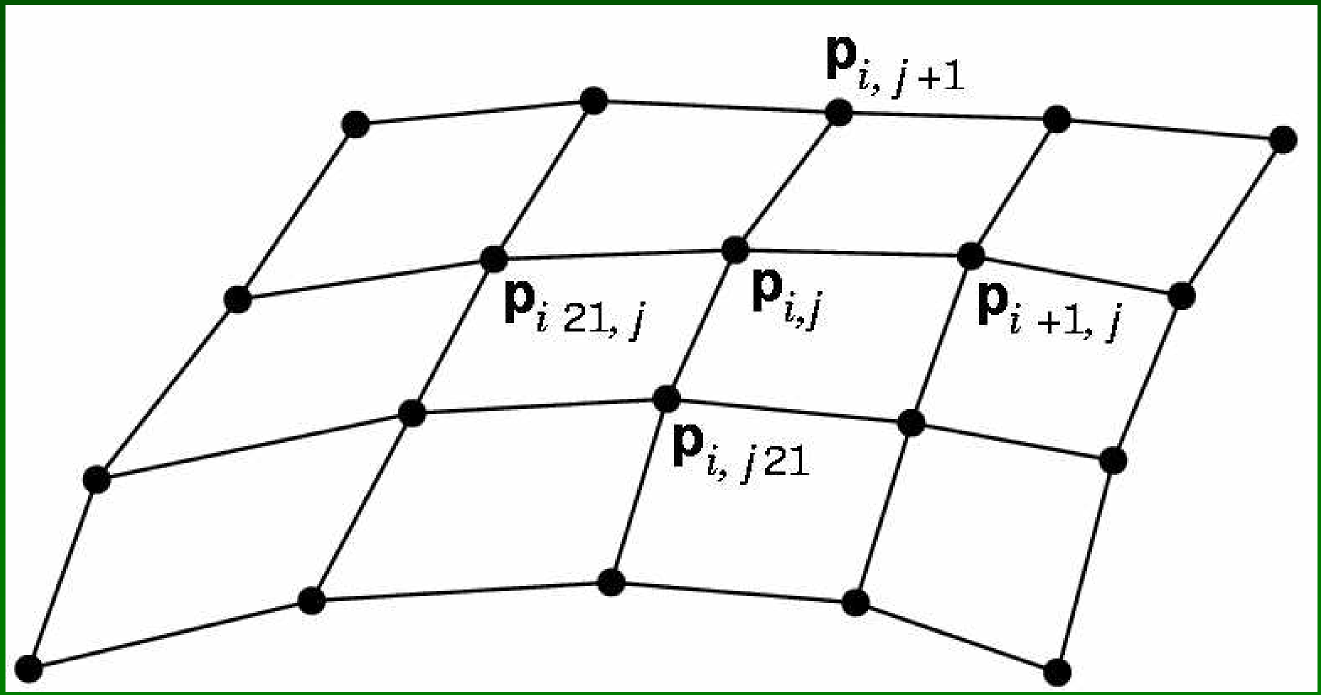


Outline

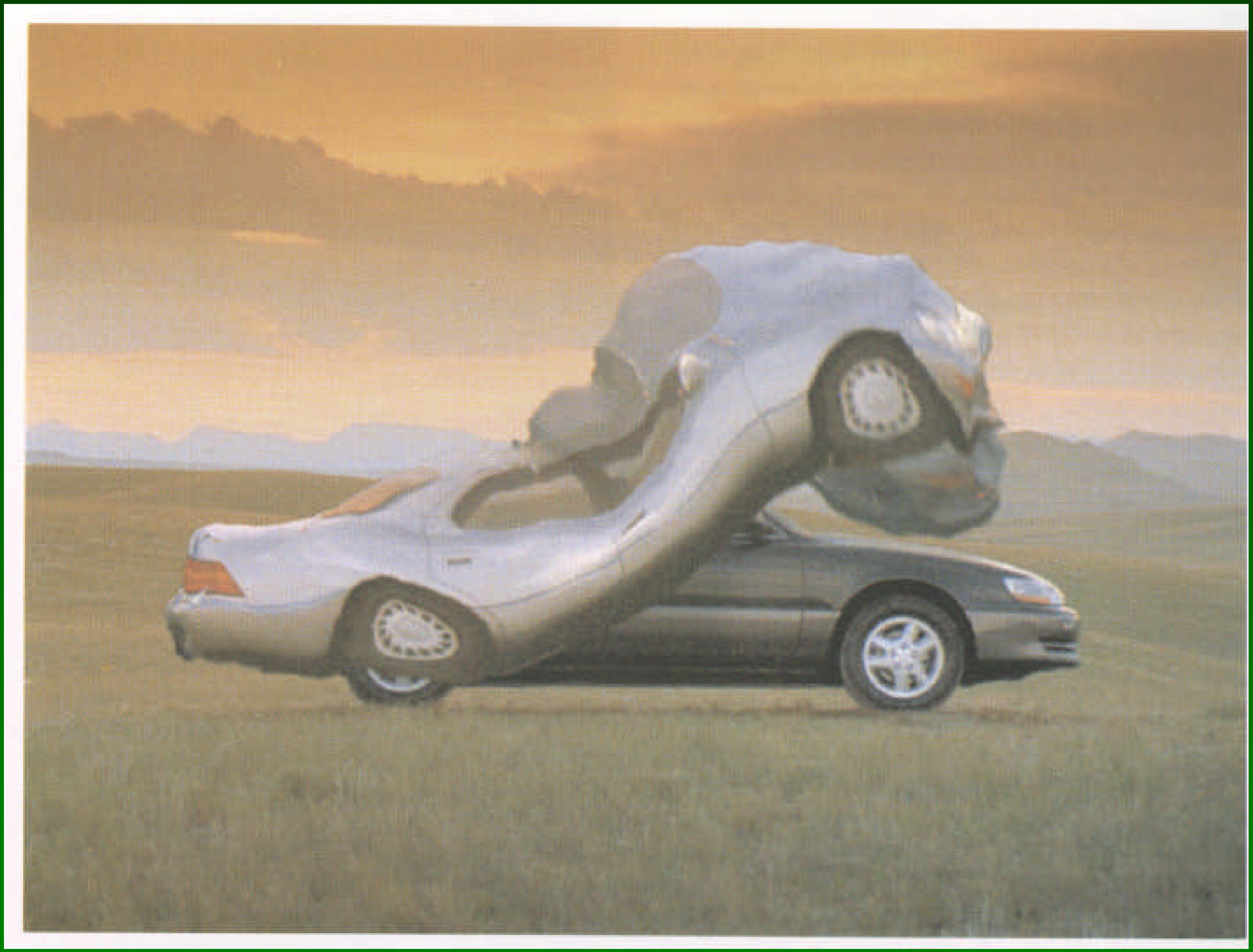
- Overview
- Newtonian Particles
- Spring Forces
- Solving Particle Systems
- Constraints

Spring-Mass Systems

- Examples: cloth in 2D, jello in 3D
- In general, O(n²), sometimes can limit to O(n)

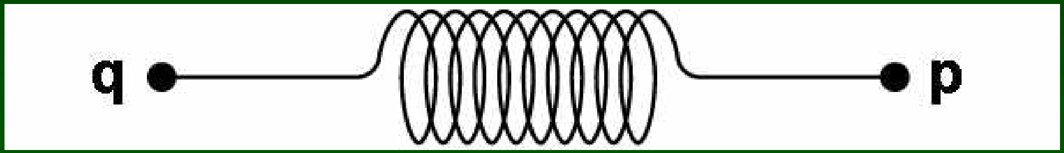


Spring-Mass Systems: Example



Hook's Law

- Simple linear spring force approximation



- Hooke's law

$$\mathbf{f} = -k_s(|\mathbf{d}| - s)\frac{\mathbf{d}}{|\mathbf{d}|}$$

f = force
k_s = spring constant
d = **p** – **q** (distance)
s = resting length

Damping Term

- Damping or drag depends on velocity
- Acts in direction of spring force
- Proportional to projection of velocity vector onto distance vector

$$\mathbf{f} = -\left(k_s(|\mathbf{d}| - s) + k_d \frac{\dot{\mathbf{d}} \cdot \mathbf{d}}{|\mathbf{d}|}\right) \frac{\mathbf{d}}{|\mathbf{d}|}$$

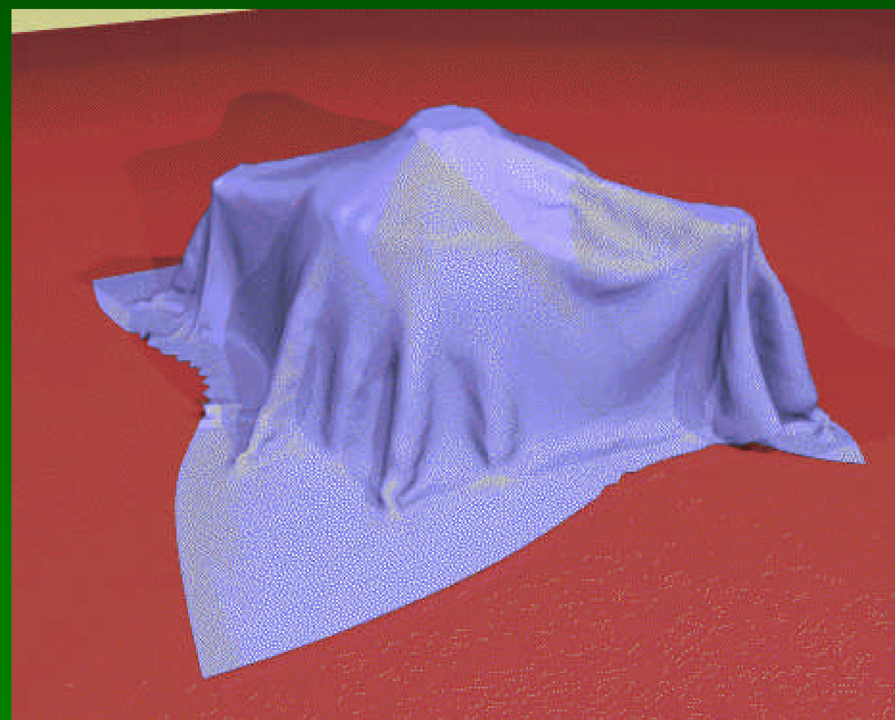
$$\dot{\mathbf{d}} = \dot{\mathbf{p}} - \dot{\mathbf{q}}$$

k_d = damping constant

Cloth

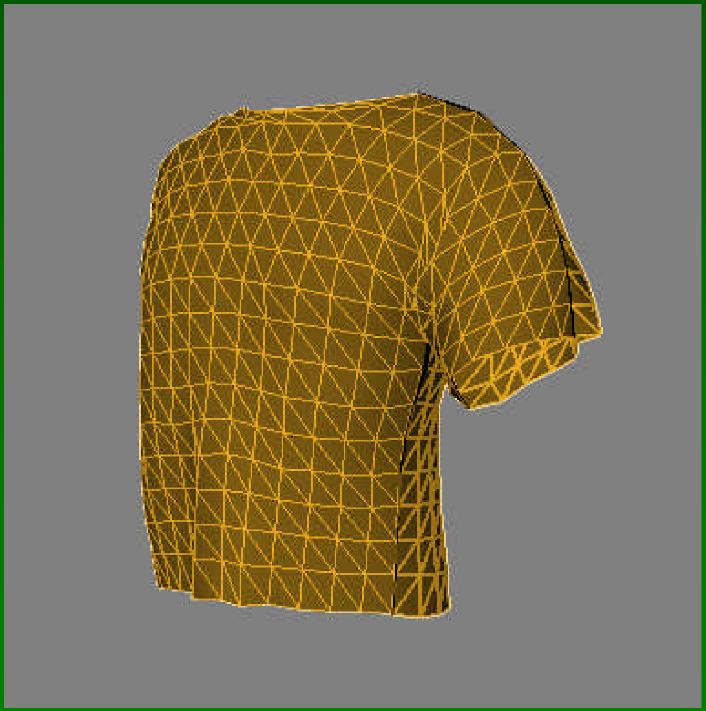
- Example of spring-mass system
- Many different varieties

Breen, 1995



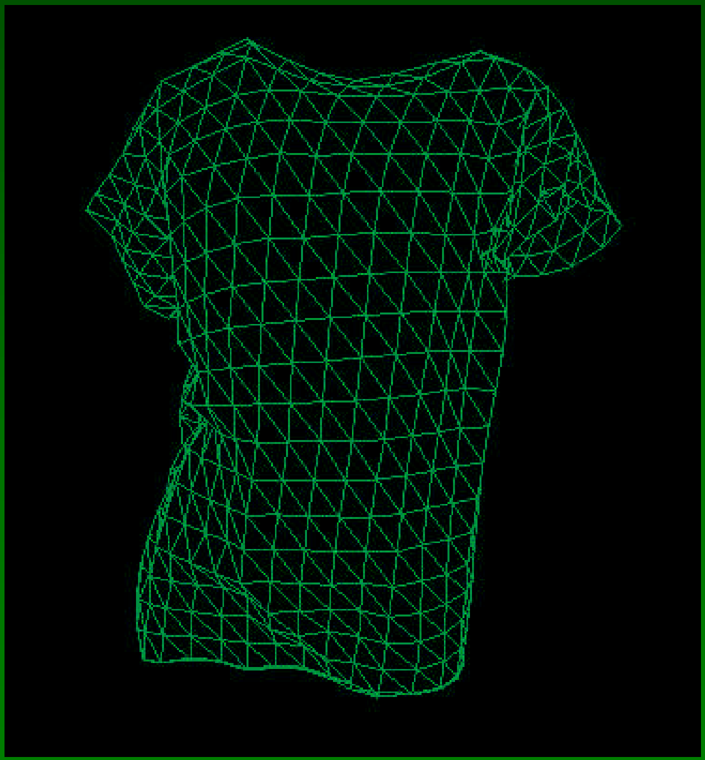
Mesh Size vs Computation Time

- Higher resolution is slower to compute
- Resolution in space vs resolution in time
- Example: modeling for clothing



Collisions for Clothing

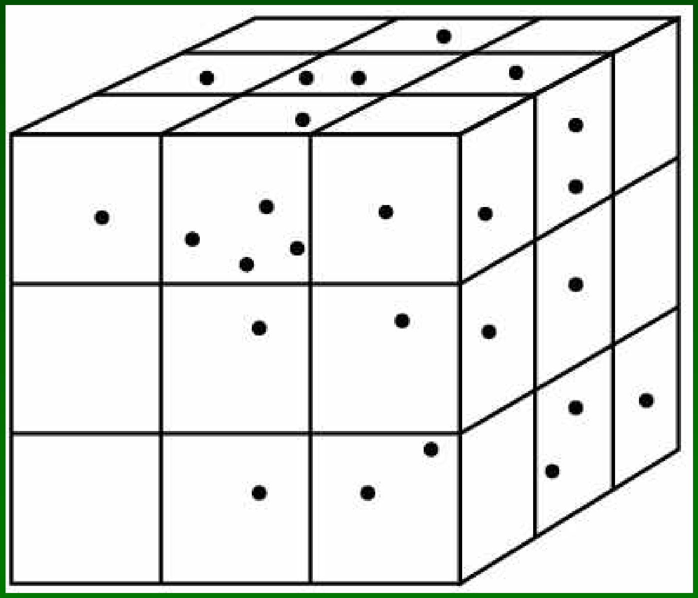
- Computationally very expensive
- Using bounding box hierarchy to reduce tests



Attractive and Repulsive Forces

- Avoid collisions (e.g., stampede)
- Example: force is inversely proportional to square of distance

$$\mathbf{f} = -k_r \frac{\mathbf{d}}{|\mathbf{d}|^3}$$



- Divide space into cells to avoid $O(n^2)$ calcn.

Outline

- Overview
- Newtonian Particles
- Spring Forces
- Solving Particle Systems
- Constraints

Ordinary Differential Equations

- Applicable when forces are simple

$$\dot{\mathbf{u}} = \mathbf{g}(\mathbf{u}, t)$$

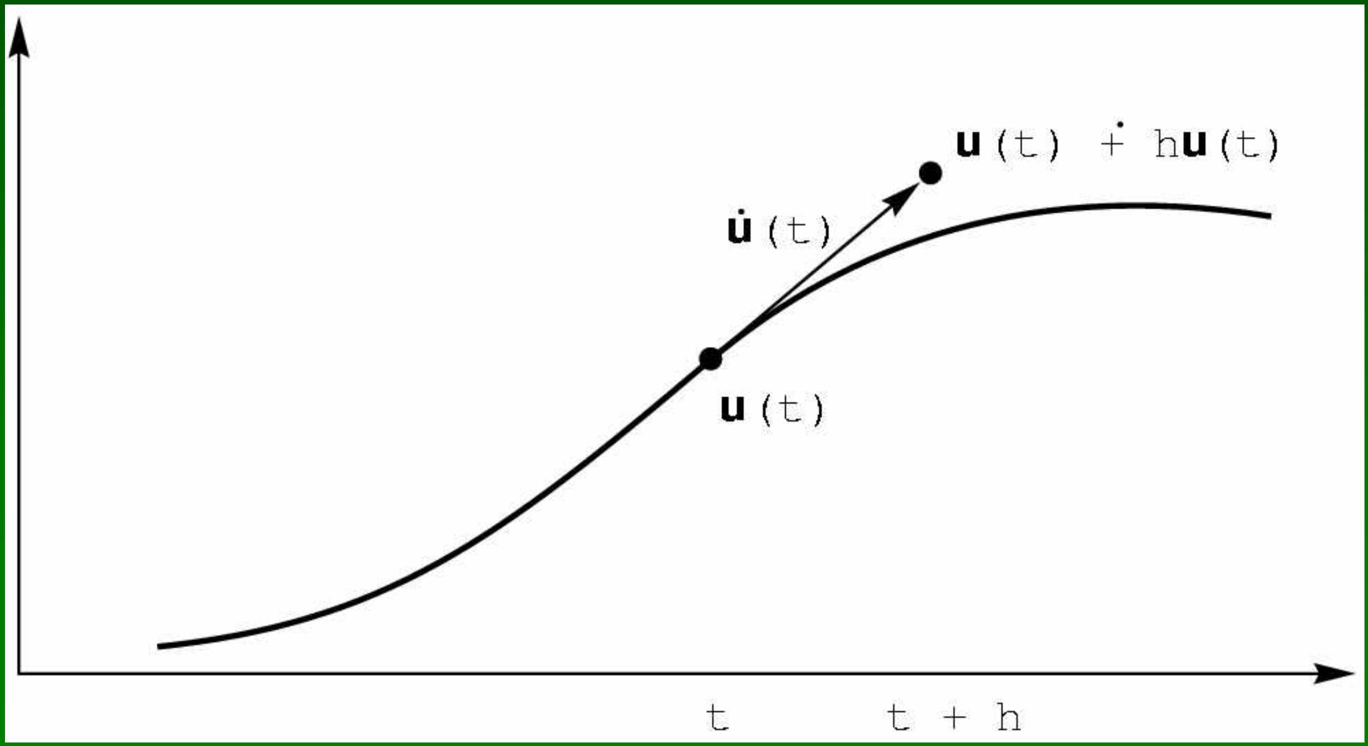
- **u** represents both position and velocity vectors
- **g** includes external forces
- Use numerical ODE solvers
- Approximate **u** over short time *h*

$$\int_t^{t+h} \dot{\mathbf{u}} d\tau = \mathbf{u}(t+h) - \mathbf{u}(t) = \int_t^{t+h} \mathbf{g}(\mathbf{u}, \tau) d\tau$$

Euler's Method

- If *h* is small, approximate **g** over [t, t+h] by **g**(t)

$$\mathbf{u}(t+h) \approx \mathbf{u}(t) + h\mathbf{g}(\mathbf{u}(t), t)$$



Assessing Euler's Method

- Particularly easy to implement

$$\mathbf{u}(t+h) = \mathbf{u}(t) + h\mathbf{g}(\mathbf{u}(t), t) + O(h^2)$$

- Accuracy depends on step size
- Accumulating errors (numerical instability)
- Specialized method for certain applications
  - Cloth, clothing, soft object

Runge-Kutta Method

- Straightforward improvement

$$\mathbf{u}(t+h) = \mathbf{u}(t) + \int_t^{t+h} \mathbf{g}(\mathbf{u},\tau)d\tau$$

- Use average of **g** over [t, t+h] to approx integral

$$\int_t^{t+h} \mathbf{g}(\mathbf{u},\tau)d\tau \approx \frac{h}{2}(\mathbf{g}(\mathbf{u}(t),t)+\mathbf{g}(\mathbf{u}(t+h),t+h))$$

- Need to evaluate **g** twice
- Error only $O(h^3)$, other orders similar

Outline

- Overview
- Newtonian Particles
- Spring Forces
- Solving Particle Systems
- Constraints

Constraints

- Hard constraints
  - Collisions
  - Contact force
  - Joints
- Soft constraints
  - Preservation of energy
  - Other examples?

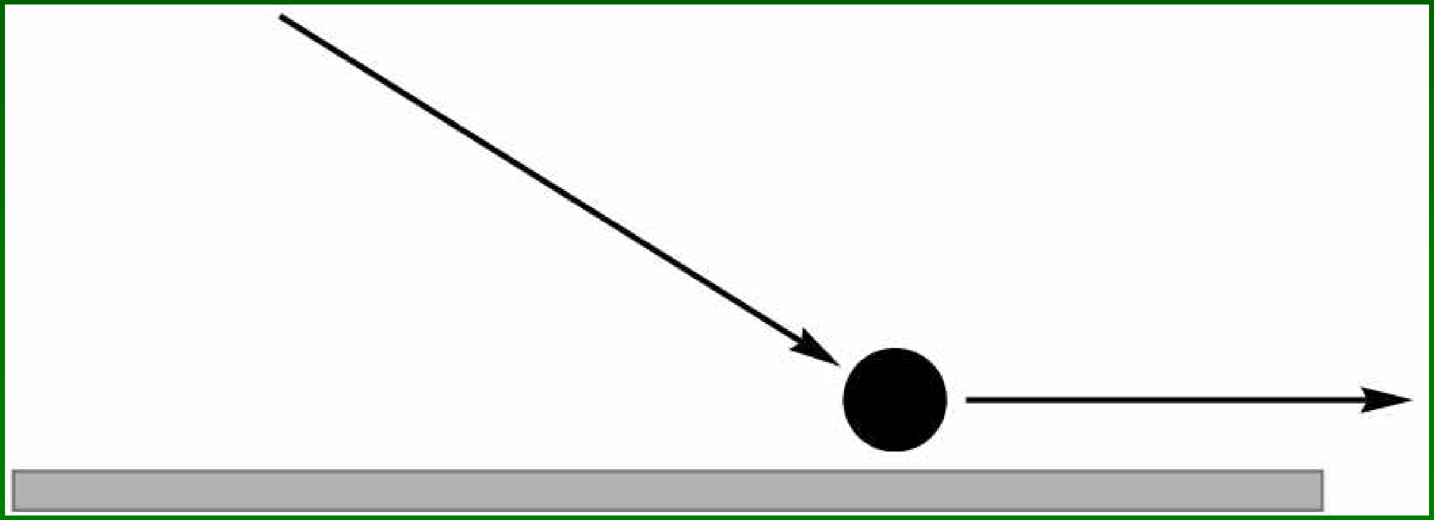
Collisions

- Often expensive to compute (bounding boxes)
- Elastic collision
  - Particle retains all energy
  - Reflected like a light beam
  - Need surface normal
- Inelastic collision
  - Coefficient of restitution: fraction of normal velocity
  - Example: simple bouncing ball
- Example: particles inside a sphere

[Angel, Ch 11.5.2]

Contact Forces

- Useful for programming assignment 5
- Decompose force vector into normal and tangential component
- Apply frictional term in tangential component

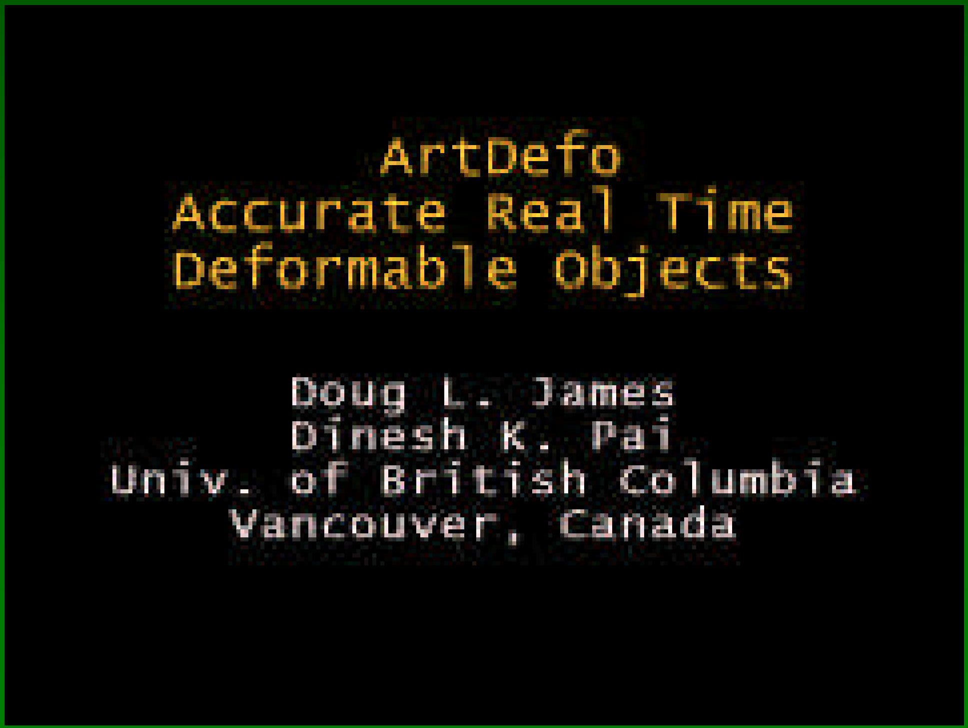


Summary

- Overview
- Newtonian Particles
- Spring Forces
- Solving Particle Systems
- Constraints

Physically-Based Modeling

- Only scratched the surface (so to speak)
- Active and exciting research area
- Example: Doug James, CMU faculty candidate



Preview

- Thursday: Texture Mapping [Shayan Sarkar]
- Next week: the “end” of the graphics pipeline
- Assignment 5 due next week
- Pick up midterm now