Today

- Introduction to Simulation
  - Basic particle systems
  - Time integration (simple version)
Physically Based Animation

- Generate motion of objects using numerical simulation methods

Physically Based Animation

Monday, November 24, 2008
Particle Systems

- Single particles are very simple
- Large groups can produce interesting effects
- Supplement basic ballistic rules
  - Collisions
  - Interactions
  - Force fields
  - Springs
  - Others...

Karl Sims, SIGGRAPH 1990
Particle Systems

- Single particles are very simple
- Large groups can produce interesting effects
- Supplement basic ballistic rules
  - Collisions
  - Interactions
  - Force fields
  - Springs
  - Others...

Basic Particles

- Basic governing equation \( \ddot{x} = \frac{1}{m} f \)
- \( f \) is a sum of a number of things
  - Gravity: constant downward force proportional to mass
  - Simple drag: force proportional to negative velocity
  - Particle interactions: particles mutually attract and/or repel
    - Beware \( O(n^2) \) complexity!
  - Force fields
  - Wind forces
  - User interaction
Basic Particles

- Properties other than position
  - Color
  - Temp
  - Age
- Differential equations also needed to govern these properties
- Collisions and other constrains directly modify position and/or velocity

Integration

- Euler's Method
  - Simple
  - Commonly used
  - Very inaccurate
  - Most often goes unstable

\[
x^{t+\Delta t} = x^t + \Delta t
\]

\[
\dot{x}^{t+\Delta t} = \dot{x}^t + \Delta t
\]
Integration

- For now let's pretend $f = mv$
  - *Velocity* (rather than acceleration) is a function of force

\[ \dot{x} = f(x, t) \]

Witkin and Baraff

Note: Second order ODEs can be turned into first order ODEs using extra variables.
Integration

- With numerical integration, errors accumulate
- Euler integration is particularly bad

\[ \dot{x} := x + \Delta t \cdot f(x, t) \]

Integration

- Stability issues can also arise
  - Occurs when errors lead to larger errors
  - Often more serious than error issues

\[ x = \begin{bmatrix} -\sin(\omega t) \\ -\cos(\omega t) \end{bmatrix} \]
Integration

- Modified Euler
  \[ x^{t+\Delta t} = x^t + \frac{\Delta t}{2} (\dot{x}^t + \dot{x}^{t+\Delta t}) \]
  \[ \dot{x}^{t+\Delta t} = \dot{x}^t + \Delta t \ddot{x}^t \]
  \[ x^{t+\Delta t} = x^t + \Delta t \dot{x}^t + \frac{(\Delta t)^2}{2} \ddot{x}^t \]

- Midpoint method
  a. Compute half Euler step
  b. Eval. derivative at halfway
  c. Retake step

- Other methods
  - Verlet
  - Runge-Kutta
  - And many others...
Integration

- Implicit methods
  - Informally (incorrectly) called backward methods
  - Use derivatives in the future for the current step
    \[
    x^{t+\Delta t} = x^t + \Delta t \dot{x}^{t+\Delta t} \\
    \dot{x}^{t+\Delta t} = \dot{x}^t + \Delta t \ddot{x}^{t+\Delta t}
    \]
    \[
    x^{t+\Delta t} = V(x^{t+\Delta t}, \dot{x}^{t+\Delta t}, t + \Delta t) \\
    \dot{x}^{t+\Delta t} = A(x^{t+\Delta t}, \dot{x}^{t+\Delta t}, t + \Delta t)
    \]

- Solve nonlinear problem for \( x^{t+\Delta t} \) and \( \dot{x}^{t+\Delta t} \)
- This is fully implicit backward Euler
- Many other implicit methods exist...
- Modified Euler is partially implicit as is Verlet
Integration

- Semi-Implicit
  - Approximate with linearized equations
    \[ V(x^{t+\Delta t}, x^{t+\Delta t}) \approx V(x^t, x^t) + A \cdot (\Delta x) + B \cdot (\Delta x) \]
    \[ A(x^{t+\Delta t}, x^{t+\Delta t}) \approx A(x^t, x^t) + C \cdot (\Delta x) + D \cdot (\Delta x) \]

\[
\begin{bmatrix}
  x^{t+\Delta t} \\
  \dot{x}^{t+\Delta t}
\end{bmatrix} =
\begin{bmatrix}
  x^t \\
  \dot{x}^t
\end{bmatrix} + \Delta t \left( \begin{bmatrix}
  \ddot{x}^t \\
  \dddot{x}^t
\end{bmatrix} + \begin{bmatrix}
  A & B \\
  C & D
\end{bmatrix} \begin{bmatrix}
  \Delta x \\
  \Delta \dot{x}
\end{bmatrix} \right)
\]
Integration

- Explicit methods can be conditionally stable
  - Depends on time-step and stiffness of system
- Fully implicit can be unconditionally stable
  - May still have large errors
- Semi-implicit can be conditionally stable
  - Nonlinearities can cause instability
  - Generally more stable than explicit
  - Comparable errors as explicit
    - Often show up as excessive damping

Integration

- Integrators can be analyzed in modal domain
- System have different component behaviors
- Integrators impact components differently
Suggested Reading

- Physically Based Modeling: Principles and Practice
  - Andy Witkin and David Baraff
- Numerical Recipes in C++
  - Chapter 16
- Any good text on integrating ODE's