Computer Animation

Robert Bridson (rbridson@cs.ubc.ca)

(preview of CPSC 426)

Computer Animation

- Offline: generate a film, play it back later
- We long ago reached the point of being able to render anything an artist could model
- The problem is: how to model?
  - Tools/UI for directly specifying model+motion (the traditional technique)
  - Procedural modeling (e.g. particle systems)
  - Data-driven modeling (e.g. motion capture)
  - Physics-based modeling (e.g. fluid simulation)

Real-Time Animation

- For example, games
- Rendering limited, modeling even more limited
- “Traditional” technique - replay scripted motions
  - But scalability/realism are becoming a problem
  - Need to generate more new motion on the fly

Traditional CG animation

- Grew out of traditional animation
- [Pixar]
- Every detail of every model is parameterized
  - E.g. position and orientation of base of lamp, joint angles, lengths, light intensity, control points for spline curve of power cord, …
- Associate a “motion curve” with each parameter - how it changes in time
- Animating == designing motion curves

Motion Curves

- Keyframe approach:
  - Artist sets extreme values at important frames
  - Computer fills in the rest with splines
  - Artist adjusts spline controls, slopes, adds more points, adjusts, readjusts, re-readjusts, …
- Straight-ahead approach:
  - Artist simply sets parameters in each successive frame
- Layering approach:
  - Design the basic motion curves first, layer detail on afterwards

Motion Curve Tools

- Retiming: keep the shape of the trajectory, but change how fast we go along it
  - Add a new abstract motion curve controlling distance traveled along trajectory
- Inverse Kinematics (IK):
  - Given a skeleton (specified by joint angles)
  - Artist directly controls where parts of the skeleton go, computer solves for the angles that achieve that

Procedural Modeling

- Write programs to automatically generate models and motion
- For example, "flocking behaviour"
- Build a flock of birds by specifying simple rules of motion:
  - Accelerate to avoid collisions
  - Accelerate to fly at preferred distance to nearby birds
  - Accelerate to fly at same velocity as nearby birds
  - Accelerate to follow "migratory" impulse
- Let it go, hope the results look good

Data-Driven Modeling

- Measure the real world, use that data to synthesize models
  - Laser scanners
  - Camera systems for measuring reflectance properties
  - Image-Based Rendering - e.g. Spiderman
  - …

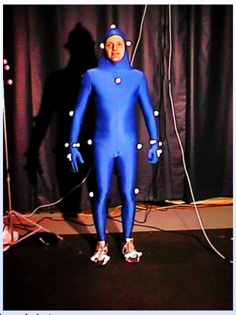
Data-Driven Motion

- Record real motion (motion capture = mocap)
- Then play it back
- But life is never that simple
  - Real motion is hard to measure
  - Measurements are noisy
  - Won't quite fit what you needed
  - Not obviously adaptable to new environments, interactive control, etc.

Marker-based mocap

- Stick performer in a tight black suit, stick markers on body, limbs, …
- Film motion with an infrared strobe light and multiple calibrated cameras
- Reconstruct 3D trajectories of markers, filling in gaps and eliminating noise
- Infer motion of abstract skeleton
- Clean up data
- Drive CG skeleton with recorded motion curves

What it looks like…



(from Zoran Popovic's website)

Footskate and Clean Up

- Most common problem: footskate
  - Feet that in reality were stuck to floor hover and slip around
- Fix using IK: determine target footplants, automatically adjust joint angles to keep feet planted
  - Often OK to even adjust limb lengths…

Motion Control

- How do you adapt mocap data to new purposes?
  - Motion graphs (remixing)
  - Motion parameterization (adjust mocap data)
  - Motion texturing (add mocap details to traditional animation)

Motion Graphs

- Chop up recorded data into tiny clips
  - Aim to cut at common poses
- Build graph on clips: connect two clips if the end pose of one is similar to the start pose of another
- Then walk the graph
  - Figure out smooth transitions from clip to clip
  - Navigate a small finite graph instead of infinite space of all possible motions

Physics-based modeling

- Like procedural modeling, only based on laws of physics
- If you want realistic motion, simulate reality
- Human motion:
  - Specify muscle forces (joint torques), simulate actual motion
  - Has to conserve momentum etc.
  - Can handle the unexpected (e.g. a tackle)
  - But need to write motion controllers
- Passive motion:
  - Figure out physical laws behind natural phenomena
  - Simulate (close cousin of scientific computing)

Example: particle dynamics

- Newton's law: F=ma
  - Rewrite as $\frac{dv}{dt} = a = \frac{F}{m}, \quad \frac{dx}{dt} = v$
  - Simplest good solver: symplectic Euler

$$v^{new} = v + \Delta t \frac{F(x,v)}{m}$$

$$x^{new} = x + \Delta t v^{new}$$

Particle forces

- For next animation, damped spring between nearby particles
  - Elastic force pushes/pulls between two nearby particles to make them preferred distance apart
  - Damping force slows down relative motion
  - Cut off to zero for particles too far away from each other
- Also collision forces
  - If particles penetrate container, push them out

More advanced physics

- Rigid bodies
  - Include orientation and rotation
- Constrained dynamics
  - E.g. jointed skeletons
- Solid mechanics
  - Generalize springs to multiple dimensions
- Fluid mechanics
  - Pressure and viscosity forces