Introduction to Computer Graphics
Marie-Paule Cani & Estelle Duveau

04/02 Introduction & projective rendering
11/02 Procedural modeling, Interactive modeling with parametric surfaces

25/02 Introduction to OpenGL + lab: first steps & modeling
04/03 Implicit surfaces 1 + lecture/lab: transformations & hierarchies
11/03 Implicit surfaces 2 + Lights & materials in OpenGL
18/03 Textures, aliasing + Lab: Lights & materials in OpenGL
25/03 Textures in OpenGL: lecture + lab
01/04 Procedural & kinematic animation + lab: procedural anim
08/04 Physically-based animation: particle systems + lab: physics 1
22/04 **Physically-based animation: collisions, control** + lab: physics 2
29/04 Animating complex objects + Realistic rendering

06/05 Talks: results of cases studies
Physically-based models

Interactions (collisions) between objects

Processing them: an advantage of physically-based models!

• Continuous solutions
  – Intersections of trajectories
  – Back to the contact time!

• Discrete time solutions
  1. Detect penetrations
  2. Model contact
  3. Respond to collisions
Physically-based models
Interactions between objects

1. Detect interpenetrations
   • Broad phase
     – Event-based processing
     – Use a space grid
     – Use bounding volumes
   • Narrow phase
     – Intersection of geometry
Physically-based models

Interactions between objects

1. Detection: broad phase

- **Event-based detection**
  - For rigid solids with bounded acceleration
    Guarantee that a pair cannot collide before …
  - Use a temporal queue to store the next tests

- **Space grid**
  - Each cell: list of objects intersecting it
  - Tests: pairs of object in the same cell
Physically-based models

Interactions between objects

1. Detection: broad phase
   • Use bounding volumes
     − Spheres
       distance > R1 + R2 ?
     − Axes parallel bounding boxes (ABB)
       X1-max > X2-min ?
     − Oriented bounding boxes (OBB)
Physically-based models

Interactions between objects

1. Detection: broad phase

- Hierarchies of bounding volumes
  - Divide & conquer approach
    - refine if the parents intersect
  - Constant time, approximate detection
    - Stop when needed
Physically-based models

Interactions between objects

1. Detection: narrow phase
   • For each pair of object
     Use the geometric description
     - Polygonal models: intersection between pairs of faces (O(n^2))
     - Tests point/field function if one of the objects is implicit (O(n))

   (restrict to points of faces in bounding volume)
Physically-based models

Interactions between objects

1. Detection: narrow phase
   • For each pair of object
     Use the geometric description …

Notes
- Many recent methods are based on the graphics hardware (GPU)
- Difficult case: thin, deformable objects can cross between time steps
Physically-based models

Interactions between objects

2. Contact modeling
   - Rigid objects
     - Back to a « valid configuration »?
       Inequalities expressing non-penetration
       Global system to be solved
     - Virtual reality: fast solution for a single collision
       Display non-penetrating copies
   - Deformable models
     - Deform objects without moving them?
Physically-based models

Interactions between objects

Problem for thin, deformable objects: Untangling cloth

Image from Bridson et al.
3. Response to collisions

- Rigid bodies: 2 possible solutions
  - Impulses
    \[ V = V_t + V_n \]
    Modified speed: \[ V := V_t - k V_n \]
    (mirror with energy decay in normal direction)
  - Contact forces

- Soft objects
  - Contact forces
3. Response to collisions

- “Penalty method” for response forces
  - Normal force fct of penetration
  - + Friction forces (viscous, dry…)

Overshooting problem

- Go back in time?
- Project the object to the closest point?
- Control energy after rebounce?
- Use adaptive time-steps?
Motion Control
What the art director would like

« Help to realism »

• Master the scénario
  – Give approximate trajectories
  – Control some Dof, synchronize

• Use simulation
  – Realistic motion of floating parts
  – Collision detection and response
  – Improve realism of trajectories
Physically-based models

*Motion control?*

- Hard for inanimate objects
  - Unpredictable effect of collisions!
  - Instable

- Impossible for a character?
  - Animation governed by muscle forces over time
  - Ex: a dinoausaur descending stairs
    - More than 150° degrees of freedom to synchronize
    - Keep equilibrium!
Physically-based models

Motion control

Technics for combining realism and control?

1. Imposing the motion of some DoF
2. Improving a trajectory given by key-frames
3. Using/generating motion controllers

(and combinations of the above!)
Physically-based models
Motion control

1. Imposing the motion of some Dof

• **Goals**
  1. Imposed motion for some degrees of freedom
  2. The simulation computes the rest

• **Examples**
  – Swim : impose rotations of the arms
  – Swing : impose rotations of the legs
Physically-based models

Motion control

1. Imposing the motion of some Dof
   - Some resolution methods
     - Inverse dynamics
     - Constraint forces (optimization)
     - Displacement constraints
       - Animate each part as independant
       - Iterate displacements until each constraint is reached
Physically-based models

Motion control

1. Imposing the motion of some DoF

Results

Objects move as puppets (some parts pull the others)

For controlled DoF:

- No help to realism
- No deviation from ideal motion due to collision
2. Improving trajectories given by key-frames

*Simple method: Following a target*

Physically-based model
- Attracted by a geometric target
- Computes speed, collisions…

- **Results**
  - Object are pulled as puppets
  - Fake realism!
Physically-based models

Motion control

2. Improving trajectories given by key-frames

Space-time constraints [Witkin, Kass 88]

1. The user specifies constraints (position/orientation at $t_i$)

2. The trajectory is improved through optimisation
   - Temporal discretization: unknowns $X_i, F_i$
   - Mecanics laws are used as constraints
   - A criteria is minimized (amount of internal energy used)
Physically-based models

Motion control

2. Improving trajectories given by key-frames

Space-time constraints [Witkin, Kass 88]

Results

– Attractive idea: “physically-based interpolation”
– Collisions cannot be handled automatically
Physically-based models
Motion control

3. Use/generate motion controllers

- *Method inspired from robotics*
  - Use a real simulation (ex walking: maintain equilibrium)
  - Muscular forces computed by a “controller”

Current state: $x, v$

Captors

Controller

Simulation (collisions)

Motors
Physically-based models

Motion control

3. Use/generate motion controllers

- Controllers can act by pulling towards a succession of poses
  - Blind control (mecanic toys)
  - Reactive control: take contacts into account (captors)
Physically-based models

Motion control

3. Use/generate motion controllers

- Synthesis of controllers
  - Manual: example of athletic motion \([Hod95]\)
  - Optimisation: random search, selection, improvement \([VdP93-95]\)
  - Genetic algorithms: population, crossings \([NM93]\)

\textit{Find how a given creature can use its muscles!}
Physically-based models
Motion control

3. Manual tuning of motion controllers

[Miller 89]  [Hodgins 95]
Physically-based models
Motion control

3. Automatic generation of motion controllers

[Van de Panne 93-2000]
3. Use/generate motion controllers

Complex motion

- Transition graphs between controllers
  Ex: walk + equilibrium, fall, get up
- Each controller is itself a graph of desired postures

*The “captor” data play an essential part!*