Notes

- Please read Pentland and Williams, "Good vibrations", SIGGRAPH'89
- 1st bug in assignment 4
  - Thanks Igor and Richard
  - v=calloc(3*nv,sizeof(float)) fixed online

Cloth Collisions

- From last time
  - Use the same basic algorithm as for particle systems
  - But ignore elastic collisions - elastic/inelastic comes from internal dynamics (damping), not collisions
  - Also update post-collision velocity with an implicit step (smoothing filter from damping)
- Looking at collision resolution pipeline now

Repulsions

- Pipeline starts with repulsion forces
  - Simplest thing for resting contact
  - Can even model the thickness of the cloth, its compressibility, etc.
  - Avoids chainmail friction
    - Without a soft buffer, sharp folds lock together like alligator teeth

Pointwise collisions

- Next stage in pipeline: need to check for fast moving simple collisions
- Do point vs. triangle and edge vs. edge (as before, talking about rigid bodies)
  - Solve cubic for collision time, check if close enough to collision, etc.
- Will want to iterate a few times

Rigid impact zones

- Last stage in pipeline
  - Need a last resort to clean up complicated collisions
- Provot's approximation:
  - Glue together elements that are still colliding
  - Too stable (too much friction, too inelastic) but that's OK
- Algorithm:
  - Start off with each triangle as a separate "impact zone"
  - Check if any two zones collide - if so, merge, rigidify (preserve linear and angular momentum)

Why it works

- If the points in a mesh move rigidly, mesh cannot self-intersect
  - Thus no self-intersections inside an impact zone
- We keep merging zones: very quickly end up with no more collisions
- Technical detail: if we use an Euler step to advance positions with rigid bodies, we actually induce a dilation
  - But non-intersection property still holds for general affine maps!
  - In fact, can generalize momentum… related to modal analysis

When this fails

- Constraints on cloth are incompatible with rigid impact zone
  - Could cut time step to solve,
or relax constraints,
or solve cloth self-intersection problem
- Collision adjustments strain the cloth too much (and strain limiting etc. goes crazy)
  - See this around sharp points
  - Ultimately, may want to forget black-box approach: directly incorporate contact forces into implicit time stepping, e.g. Baraff & Witkin, "Large steps…", SIGGRAPH'98 (but they don't get robustness)

Wind force on cloth

- Use simple approximation: ignore effect of cloth on wind
  - Take given wind velocity field v_{wind}
  - Look at a piece of cloth with velocity v and normal n
  - Apply traction $t=D(v_{wind}-v)\cdot n\ n$
  - Implement triangle by triangle: multiply traction by area of triangle to get force
  - Distribute force on triangle equally to vertices
  - $D>0$ is some parameter to tune
- This misses some effects, but is often adequate

Cloth speed-ups

- Curvature-accelerated interference detection
  - Form a hierarchy, find normal cone for each
  - Self-intersection is unlikely if normals all lie in same half-space
  - See Volino and Magnenat-Thalmann EG'94
- Take liberties with implicit integration
  - More approximations for speed
- Use selective collisions
  - Only run collision code on parts of clothing that are likely to come into contact with other things
- …

3D Elasticity

- We basically covered the theory earlier
  - Tensors are now 3x3 instead of 2x2 or 3x2
- What about numerics?
  - [work out FVM/FEM on tetrahedra]
- Tetrahedra can lock - may prefer hexahedra
  - Meshing is more difficult - may prefer embedding geometry in elements

Inverted elements

- If too much force is applied to an element, it will be crushed and even inverted
  - The constitutive model should not let this happen
  - [simple example with springs]
- In cloth (2d material in 3d world) this is not an issue
  - But 2d in 2d has exactly the same problem!
- Usually not a problem for stiff materials, but often with soft objects undergoing large deformation (e.g. **point-based collisions**!)
- Can partially alleviate with design of mesh

Volume springs

- Volume of tetrahedron $x_1, \dots, x_4$ is

$$V = \tfrac{1}{6}\left(x_2 - x_1\right) \times \left(x_3 - x_1\right) \bullet \left(x_4 - x_1\right)$$

  - Actually signed (to indicate right- or left-handed labeling)
- Crushed element: V=0
- Inverted element: V<0
- We generally want to control excess volume changes (most materials are incompressible in bulk), so natural to add soft constraint (potential energy) $K/2(V-V_0)^2$

Volume springs

- Take derivative w.r.t. x_i to get force
  - Example: $F_4 = -\frac{1}{6}K(V - V_0)(x_2 - x_1) \times (x_3 - x_1)$
  - To do the other forces easily, rewrite volume formula so that x_i appears only in the dot-product
- Note cross-product is proportional to area-weighted normal of opposite triangle
  - So these are "altitude springs" (with the right scaling automatically)
- To fully enforce no inversion, can change force to go to infinity as V approaches 0
  - But may have stiffness problems…

Small deformation

- We can use linear elasticity
  - Except if there's rigid body rotation
  - Can fix this: rotate to approximate rest configuration, do linear elasticity, rotate back
    - Rigid body + small deformation
    - May need to keep track of Coriolis and centripetal pseudo-forces
  - Terzopoulos & Witkin GI'88, others
- Use of linear elasticity opens up new doors
  - Modal Analysis
  - BEM
- Also: appropriate for acoustics

Modal Analysis

- Discretization of linear elasticity boils down to

$$M\ddot{x} = -Kx - D\dot{x} + F_{ext}$$

- M, K, and D are constant matrices
  - M is the mass matrix (often diagonal)
  - K is the stiffness matrix
  - D is the damping matrix: assume a multiple of K
- This a large system of coupled ODE's now
- We can solve eigen problem to diagonalize and decouple into scalar ODE's
  - M and K are symmetric, so no problems here - complete orthogonal basis of real eigenvectors

Eigenstuff

- Say $U=(u_1 \mid u_2 \mid \dots \mid u_{3n})$ is a matrix with the columns the eigenvectors of $M^{-1}K$ (and also $M^{-1}D$)
  - $M^{-1}Ku_i=\lambda_i u_i$ and $M^{-1}Du_i=\mu_i u_i$
  - Assume λ_i are increasing
  - We know $\lambda_1=\dots=\lambda_6=0$ and $\mu_1=\dots=\mu_6=0$ (with $u_1, \dots, u_6$ the rigid body modes)
  - The rest are the deformation modes: the larger that λ_i is, the smaller scale the mode is
- Use this with $\ddot{x} = -M^{-1}Kx - M^{-1}D\dot{x} + M^{-1}F_{ext}$

Decoupling into modes

- Take $y=U^T x$ (so $x=Uy$) - decompose positions (and velocities, accelerations) into a sum of modes $U\ddot{y} = -M^{-1}KUy - M^{-1}DU\dot{y} + M^{-1}F_{ext}$
- Multiply by U^T to decompose equations into modal components:

$$U^T U\ddot{y} = -U^T M^{-1}KUy - U^T M^{-1}DU\dot{y} + U^T M^{-1}F_{ext}$$
$$\ddot{y} = -diag(\lambda_i)y - diag(\mu_i)\dot{y} + U^T M^{-1}F_{ext}$$

- So now we have 3n independent ODE's
  - If F_{ext} is constant over the time step, can even write down exact formula for each

Examining modes

- Mode i: $\ddot{y}_i = -\lambda_i y_i - \mu_i \dot{y}_i + u_i \cdot M^{-1}F_{ext}$
- Rigid body modes have zero eigenvalues, so just depend on force
  - Roughly speaking, rigid translations will take average of force, rigid rotations will take cross-product of force with positions (torque)
  - Better to handle these as rigid body…
- The large eigenvalues (large i) have small length scale, oscillate (or damp) very fast
  - Important acoustically, generally not visually
- Left with small eigenvalues being important

Throw out high frequencies

- Only track a few low-frequency modes (e.g. 5-10)
- Time integration is blazingly fast!
- Essentially reduced the degrees of freedom from thousands or millions down to 10 or so
  - But keeping full geometry, just like embedded element approach
- Collision impulses need to be decomposed into modes just like external forces
  - But may have a harder time resolving collisions, like rigid bodies - not much freedom left

Simplifying eigenproblem

- Low frequency modes not affected much by high frequency geometry
  - And visually, difficult for observers to quantify if a mode is actually accurate
- So we can use a very coarse mesh to get the modes, or even analytic solutions for a block of comparable mass distribution
- Or use a Rayleigh-Ritz approximation to the eigensystem (eigen-version of Galerkin FEM)
  - E.g. assume low frequency modes are made up of affine and quadratic deformations
  - [Do FEM, get eigenvectors to combine them]

More savings

- External forces (other than gravity, which is in the rigid body modes) rarely applied to interior, and we rarely see the interior deformation
- So just compute and store the boundary particles
  - E.g. see James and Pai, “DyRT…”, SIGGRAPH’02 -- did this in graphics hardware!