Hair Cloth and soft bodies

Lesson 11

Juraj Onderik | onderik@sccg.sk

Lesson 08 Outline

- Problem definition and motivations
- Modeling deformable solids with mass-spring model
- Position based dynamics
- Modeling cloths with mass-spring model
- Modeling hair with mass-spring model
- Demos / tools / libs

Simulation of Deformable Solids

- Lagrangian Mesh Based Methods
  - Continuum Mechanics Based Methods
  - Mass-Spring Systems
- Lagrangian Mesh Free Methods
  - Loosely Coupled Particle Systems
  - Smoothed Particle Hydrodynamics (SPH)
  - Mesh Free Methods for the solution of PDEs
- Reduced Deformation Models and Modal Analysis
- Eulerian and Semi-Lagrangian Methods
  - Fluids and Gases
  - Melting Objects

Mass-spring

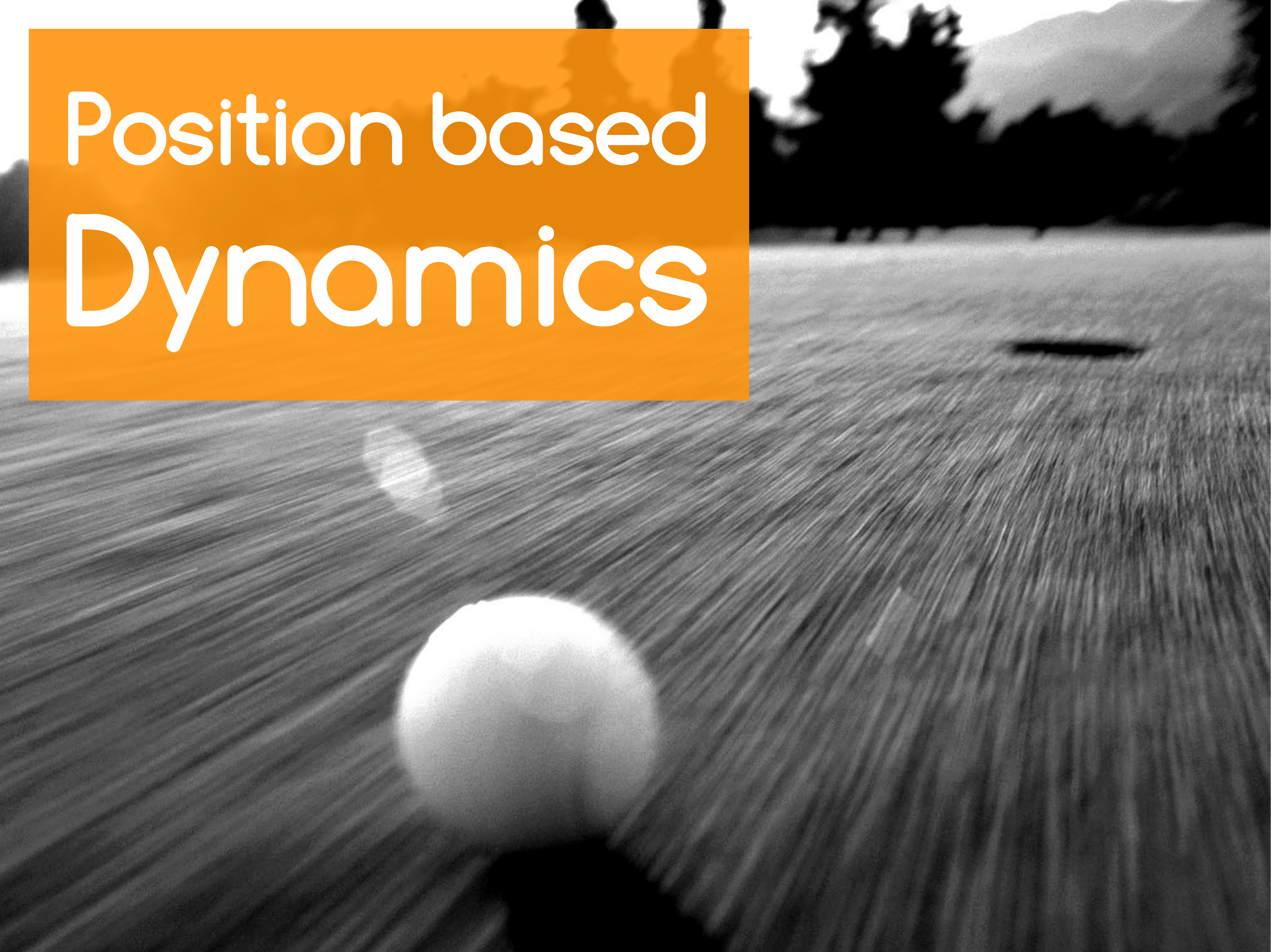
Model

Mass-spring Model

- Each deformable solid is modeled as a graph (mesh) of particles (with mass) connected with mass-less springs
- Particle Model
  - Each particle is defined at least by its Mass (mi), Position (pi), Velocity (vi)
  - Additionally there can be force, acceleration, momentum …
  - Usually particles can be incident to any number of springs
- Spring Model
  - Springs usually connects 2 particles and exerts force on them
  - Usually sprigs have non-zero rest length and some constant material properties

Hook's Spring Model

- Hook's Law: Strain is directly proportional to stress
- Formally: $f = - k_s\, x$
  - x is the displacement of the end of the spring from its equilibrium position
  - f is the restoring force exerted by the material
  - k_s is a material constant called spring stiffness
- Using rest length and velocity damping
- $f = - [k_s(|l| - l_0) + k_d(v_a - v_b)/|l|]\ (l/|l|)$
  - k_d is damping factor


Position based
Dynamics

Position based Dynamics

- Traditional force based dynamics must solve ODE using some integration scheme. Using simple and fast explicit methods can lead simulation to inaccuracy and instability
- This can be prevented by solving large systems of equations (using implicit methods) or
- Using more geometric approach by directly modify positions into stable and more accurate states.
- Such approach (position based dynamics) gives more control over animation and easily models constraints.

Position based Dynamics

- Object Representation
  - We represent dynamic object with a set N vertices
  - Each vertex has: Mass (m_i), Position (p_i), velocity (v_i)
- Constraint Representation
  - Let $p = (p_1, \ldots, p_N)$ be the generalized position
  - The constraint is a functions $C_j(p) = C_j(p_1, \ldots, p_N) : R^{3N} \rightarrow R$
  - Cardinality m_j is the number of “used” parameters
  - Stiffness parameter $k_j \in \{0\ldots1\}$ is a material property
  - We define equality (bilateral) constraint as: $C_j(p) = 0$
  - We define inequality (unilateral) constraint as: $C_j(p) \geq 0$

PBD: Algorithm

- 1: **forall** vertices i: **initialize** $p_i = p_i^0$; $v_i = v_i^0$; $w_i = 1/m_i$
- 2: **loop**
  - 3: **forall** vertices i **do** { $v_i \leftarrow v_i + \Delta t\, w_i\, f_{ext}(x_i)$ }
  - 4: DampVelocities($v_1, \ldots, v_N$)
  - 5: **forall** vertices i **do** { $q_i \leftarrow p_i + \Delta t\, v_i$ }
  - 6: **forall** vertices i **do** { CreateCollisionConstraints($x_i \rightarrow p_i$) }
  - 7: **loop** n_s **times** { ProjectConstraints($C1, \ldots, C_{M+Q}, q_1, \ldots, q_N$) }
  - 8: **forall** vertices I **do** { $v_i \leftarrow (q_i - p_i)/\Delta t$; $p_i \leftarrow q_i$; }
  - 9: VelocityUpdate($v_1, \ldots, v_N$)
- 10: **endloop**

PBD: Algorithm

- First all masses, positions and velocities are initialized to rest state
- With each simulation frame we do
  - First we modify velocities due to external forces (3:)
  - Next we add artificial damping to the system (4:)
  - Then we predict new positions (q_i) with simple Euler step (5:)
  - Next we detect and construct all collision constraints (6:)
  - We apply “projection” several times on all constraints (7:)
  - We find correct velocities and set projected positions (8:)
  - We apply friction and restitution impulses on velocities (9:)

PBD: Constraint Projection

- Assuming constraint is violated ie. $C(\rho) \mathrel{!}= 0$ (<0) we must find correction $\Delta\rho$ such that $C(\rho + \Delta\rho) = 0$ (≥ 0)
- By linearization we get: $C(\rho+\Delta\rho) \approx C(\rho) + \nabla_\rho C(\rho) \cdot \Delta\rho$
  - To conserve both momentums correction must be along direction of constraint function gradient $\nabla_\rho C(\rho)$ ie:
  - $\Delta\rho = \lambda \nabla_\rho C(\rho)$; λ (Lagrange multiplier) is a scalar
  - $\lambda = - C(\rho) / |\nabla_\rho C(\rho)|^2 f$
  - For i-th particle with mass m_i ($w_i = 1/m_i$)
  - $\Delta\rho_i = \lambda w_i \nabla_\rho C(\rho_1, \ldots, \rho_N)$
  - $\lambda = - w_i C(\rho_1, \ldots, \rho_N) / \sum_j w_j |\nabla_{\rho j} C(\rho_1, \ldots, \rho_N)|^2$

PBD: Distance Constraint

- Let $C(p) = C(p_1, p_2) = |p_1 - p_2| - d = 0$
  - $\nabla_{p1} C(p_1, p_2) = (p_1 - p_2)/|p_1 - p_2|$
  - $\nabla_{p2} C(p_1, p_2) = (p_1 - p_2)/|p_1 - p_2|$
  - $\lambda = (|p_1 - p_2| - d) / w_1 + w_2$ where $w_1 = 1/m_1$ and $w_2 = 1/m_2$
  - $\Delta p_1 = (w_1 / (w_1 + w_2))(|p_1 - p_2| - d)(p_1 - p_2)/|p_1 - p_2|$
  - $\Delta p_2 = (w_2 / (w_1 + w_2))(|p_1 - p_2| - d)(p_1 - p_2)/|p_1 - p_2|$
- For equality constraints we always do projection
- For Inequality we project only when $C(p) < 0$
- Finally we multiply Δp with stiffness k (Δpk)
  - Due to iterations use $k' = 1 - (1 - k)^{1/ns}$. Stiffness is applied linearly after n_s iterations

PBD: Collisions

- Given old position pi and predicted position qi we detect if a ray (p_i, q_i) enters some object. If yes we compute entry point q_c and collision normal n_c
- Next add collision constraint with stiffness k = 1 $C(p) = (p - q_c) \cdot n_c \geq 0$ (ensures non-penetration)
  - When scene contains more dynamic bodies we must provide constraint from all bodies into one “scene” solver
  - For triangle meshes with face (p_1, p_2, p_3): $n_c = (p_2 - p_1) \times (p_3 - p_1)$
  - Collision constraint generation is done outside the solver loop, to speed up simulation. Artifacts are negligible

PBD: Damping

- Velocities are damped
- **forall** vertices i
  - $\Delta v_i = v_{cm} + \omega \times r_i - v_i$
  - $v_i \leftarrow v_i + k_d \Delta v_i$
- **endfor**
- Δv_i only damps local deviations
  - Here $v_{cm} + \omega \times r_i$ is the velocity due to global body motion

- Global "body" variables
  - $p_{cm} = (\sum_i p_i m_i)/(\sum_i m_i)$
  - $v_{cm} = (\sum_i v_i m_i)/(\sum_i m_i)$
  - $L = \sum_i r_i \times (m_i v_i)$
  - $J = \sum_i (r^x_i)(r^x_i)^T m_i$
  - $\omega = J^{-1} L$
  - $r_i = p_{cm} - p_i$
  - r^x_i is cross product matrix

Position based Dynamics - Summary

- Control over explicit integration with no typical instability problems
- Positions of vertices and objects parts can directly be manipulated during the simulation
- Simple handling of general constraints in the position based setting
- The explicit position based solver is easy to understand and implement.

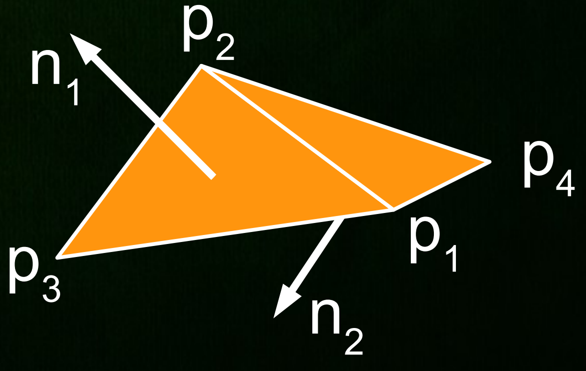

Modeling
Cloth

Cloth: Representation

- Cloth is represented with arbitrary manifold triangular mesh (no need for regular lattice)
- Each mesh vertex become a simulation particle
- Given cloth density and thickness we calculate mass of each triangle.
- Mass of each particle is sum of 1/3 of the mass of each adjacent triangle.
- Constraints are defined along edges and faces
- Cloth tearing is performed on vertices with large deformations

Cloth: Constraints

- Stretching Constraints
  - Along each mesh edge we define fixed stretching constraint as simple equality distance constraint (spring)
  - $C_s(p_1, p_2) = |p_1 - p_2| - l_0$ where l_0 is rest length
  - Stiffness k_s is usually higher to overcome springiness
- Bending Constraints
  - For each pair of adjacent triangles (p_1, p_3, p_2) and (p_1, p_2, p_4) we define a bending constraint
  - $C_b(p_1, p_2, p_3, p_4) = \mathrm{acos}(n_1, n_2) - \varphi_0$ where
    - $n_1 = ((p2 - p1) \times (p3 - p1)) / |(p2 - p1) \times (p3 - p1)|$
    - $n_2 = ((p2 - p1) \times (p4 - p1)) / |(p2 - p1) \times (p4 - p1)|$



Cloth: Collisions and Tearing

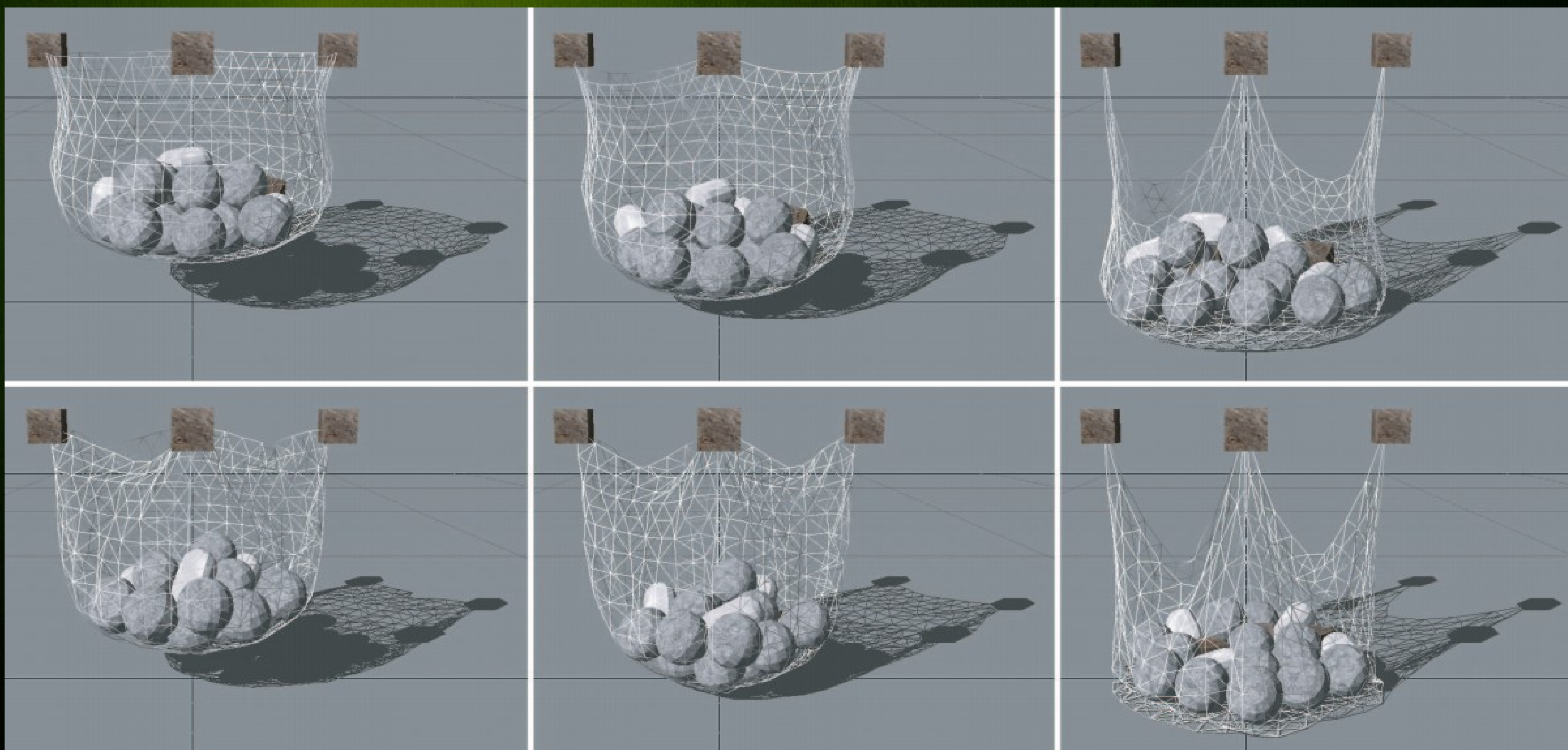
- Inequality collision constraints is defined as
- $C_b(q, p_1, p_2, p_3) = (q - p1) \cdot n - h$
  - q is collided point with face (p_1, p_2, p_3)
  - n is face normal
  - h – distance to the face.
- Collision with rigid body exerts impulse $m_i p_i / \Delta t$ at p_i
- More involved self-collision detection must be done cloth becomes to be tangled

Cloth: Overpressure and Tear

- Overpressure inside the closed mesh is modeled as
  - $C(\rho_1, \ldots, \rho_N) = \sum_j (\rho_{j1} \times \rho_{j2}) \cdot \rho_{j3} - k_\rho V_0$
  - $\nabla_{\rho 1} C = \sum_j (\rho_{j2} \times \rho_{j3}) + \sum_j (\rho_{j3} \times \rho_{j1}) + \sum_j (\rho_{j1} \times \rho_{j2})$
- Cloth Tearing Process
  - Whenever the stretching of an edge exceeds a specified threshold value, we select one of the edge's adjacent vertices
  - We then put a split plane through that vertex perpendicular to the edge direction and split the vertex into 2 new vertices
  - All triangles above the split plane are assigned to the original vertex while all triangles below are assigned to the duplicate

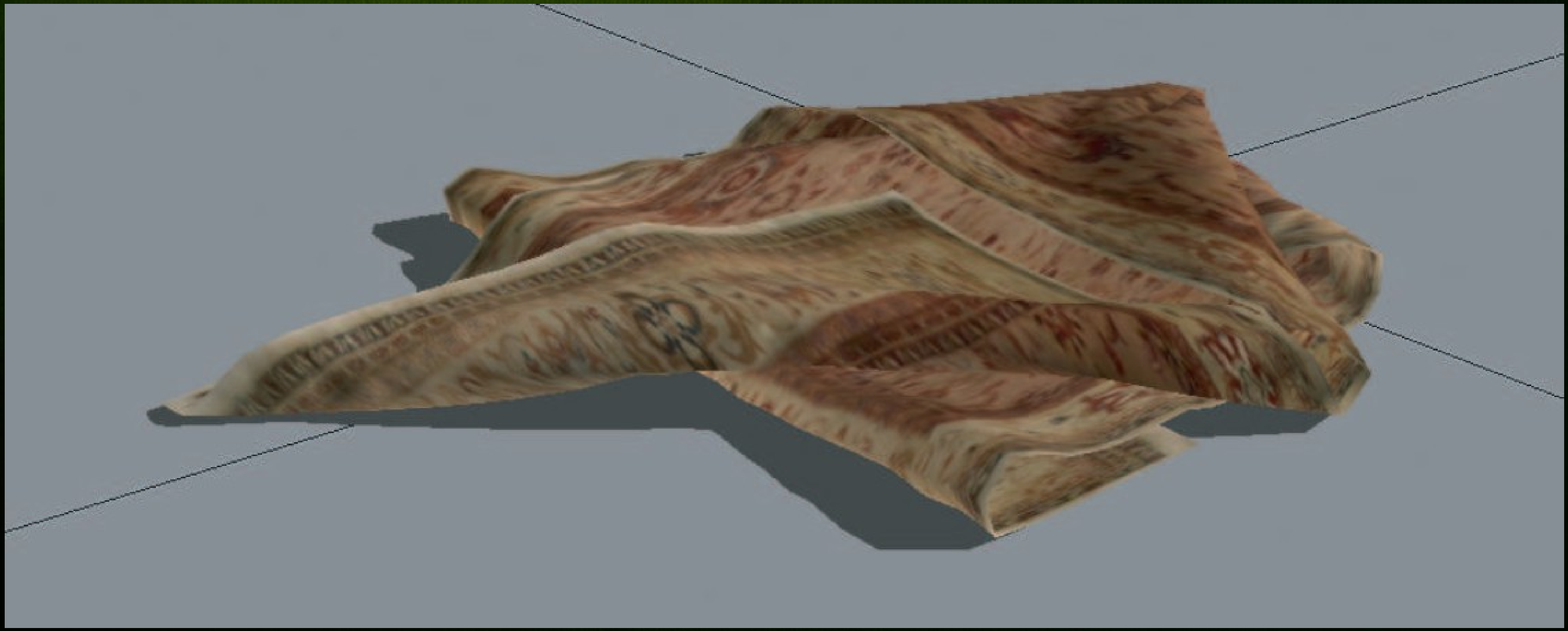
Cloth: Stiffness and Bending

$(k_s; k_b) = (1; 1)$ $(k_s; k_b) = (0.5; 1)$ $(k_s; k_b) = (0.01; 1)$

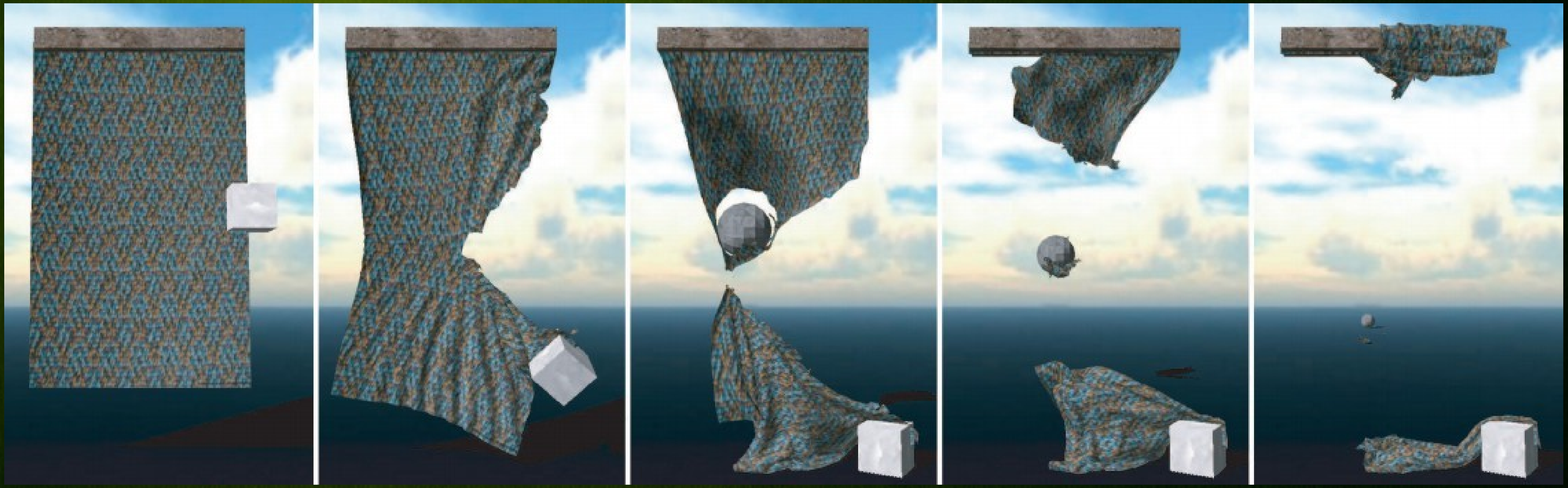


$(k_s; k_b) = (1; 0)$ $(k_s; k_b) = (0.5; 0)$ $(k_s; k_b) = (0.01; 0)$

Cloth: Self Collisions and Balloons



Cloth: Examples




Modeling
Hair

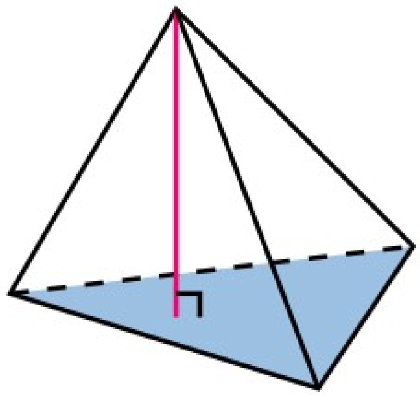
Hair: Representation

- Each hair strand is modeled as a set of vertices connected by edges into series of line segments
- Each vertex is used as simulation particle
- Given material density and strand thickness we can calculate volume/mass of each segment. Particle mass is average of incident edge masses
- Strand constraints are applied along edges, additional (virtual) edges and newly created particles
- Hair tearing is performed on vertices with large deformations

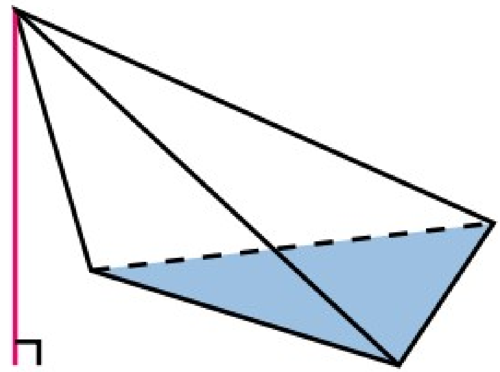
Hair: Constraints

- We model Curly Hair and Straight Hair
- Stretching Constraints (springs)
  - Linear springs between every consecutive particle
- Bending Constraints (springs)
  - Linear springs between every other particle
  - The edge springs and bending springs together form triangles that implicitly represent the orientation of the hair
- Torsion Constraints (springs)
  - Twist is modeled by attaching torsion springs that connect each particle to a particle three particles away from it
- Altitude Constraints (springs)
  - See figure

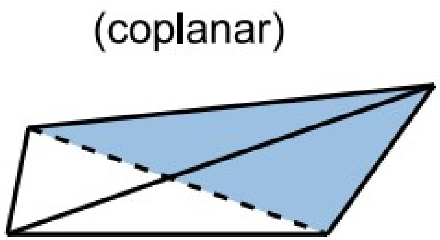
Point/Face Altitude Springs



(a) Spring has all non-negative barycentric weights

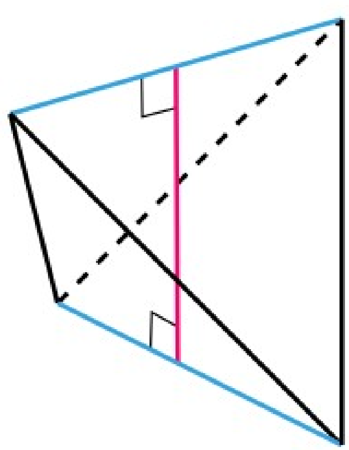


(b) Spring has negative barycentric weights

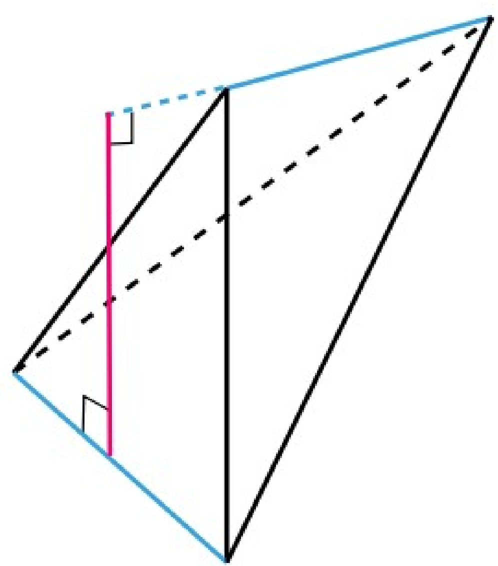


(c) Degenerate: all point/face springs have negative barycentric weights

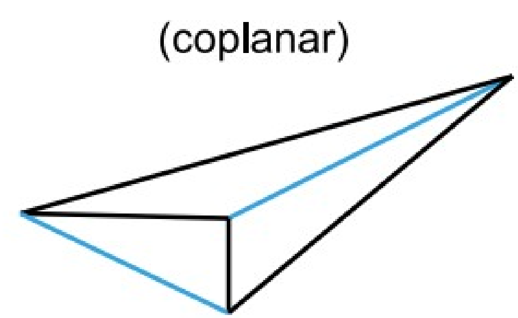
Edge/Edge Altitude Springs



(d) Spring has all non-negative barycentric weights



(e) Spring has negative barycentric weights



(f) Degenerate: all edge/edge springs have negative barycentric weights

Hair: Altitude Springs

- Point/Face Altitude Springs
  - Perpendicular to the face starting from the given point
  - Length is $l = 6V/|u \times v|$ where u and v are the vectors of the base triangle and V is the signed volume of the tetrahedron
- Edge/Edge Altitude Springs
  - Perpendicular to common spring and bending spring
  - Length is $l = 6V/|u \times v|$ where u and v are the stretch and bend spring and V is the signed volume of the tetrahedron
  - For any tetrahedron, the edge/edge or point/face spring with minimal length is guaranteed to have all non-negative barycentric weights, preventing unbounded forces

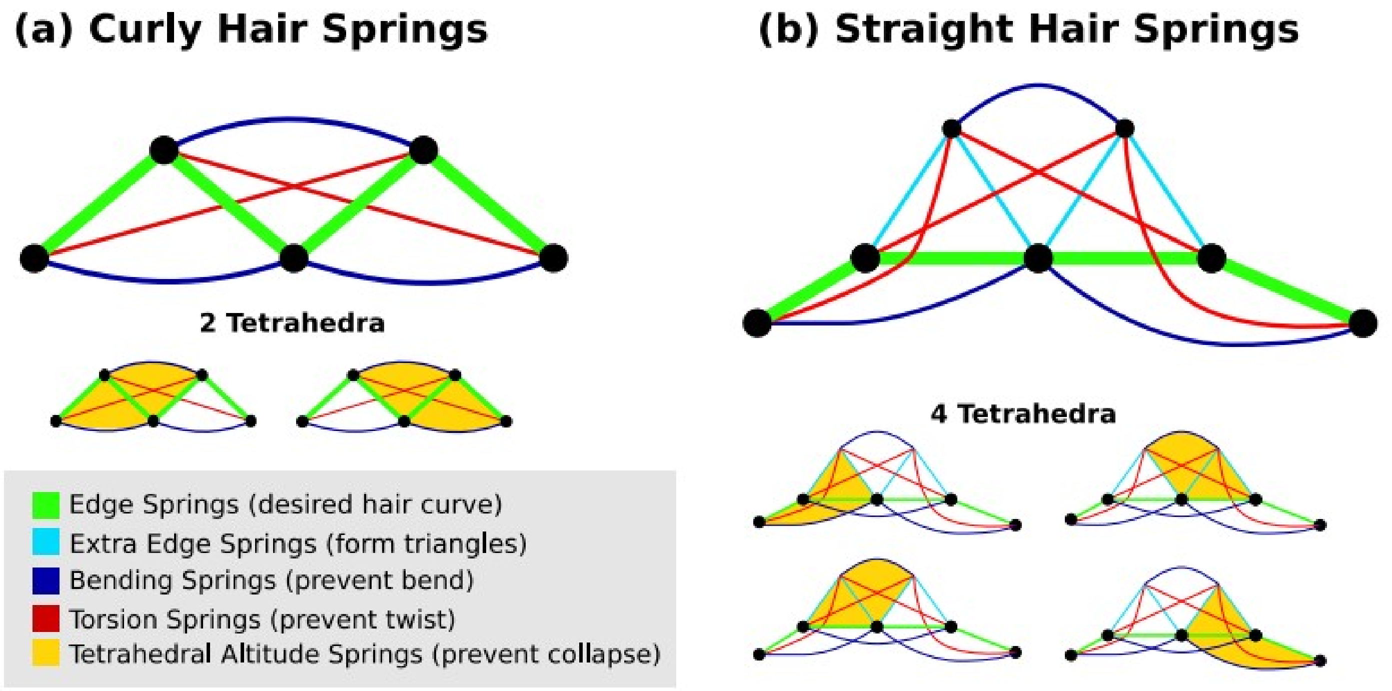


Figure 7: *Straight and curly hair models using edge, bending, torsion, and altitude springs preserving the implied tetrahedra.*

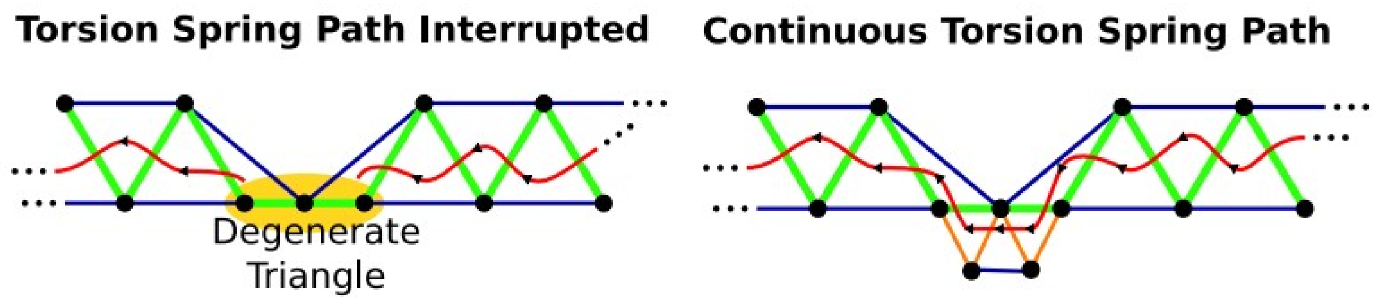


Figure 8: *Triangles define orientations for penalizing twist, and torsion springs "trace" a continuous path through the non-degenerate triangles — but they are blocked at straight hair segments (left). The subdivision and perturbation of our method removes degeneracies so the path becomes continuous (right).*

Hair: Linear Strands

Figure 9: *A simulation of 10,000 long straight hairs with 50 segments each (1,000,000 total particles) on a character shaking his head from side to side.*

Hair: Curly Strands

Figure 14: *A simulation of 5,000 long curly hairs with 50 segments each (250,000 total particles) on a character spinning around from back to front.*


The End