EG 2004

Tutorial 5: Programming Graphics Hardware

Advanced Rendering Techniques

Matthias M Wloka

nVIDIA®

Nalu

Acknowledgements

- Hubert Nguyen
- William Donnelly
- NVIDIA Demo Team

Long Blonde Hair

- Long
  - Requires dynamic animation
    - Thus cannot bake lighting
  - Requires lots of hair
    - Thus shading has to be fast

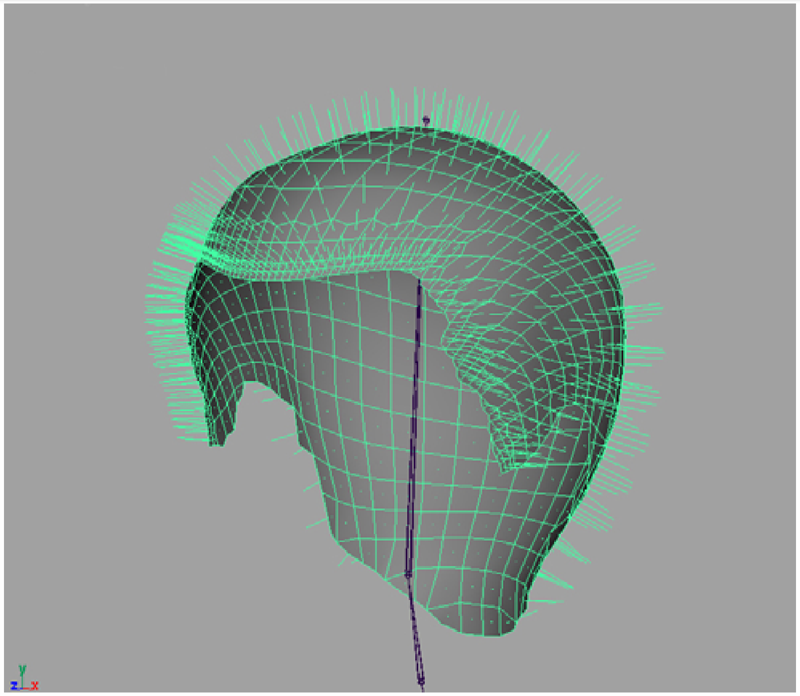
- Blonde
  - Three visible highlights, black only has one
  - Shadows much more visible

Hair Rendering: Overview

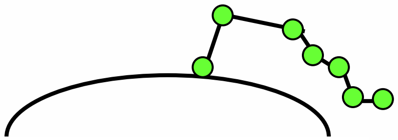
- Geometry and dynamics
- Shading
- Shadowing

Hair Geometry, Part 1

- "Skull cap" specifies
  - Where control hairs grow
  - Which direction to grow
  - Growth is non-linear

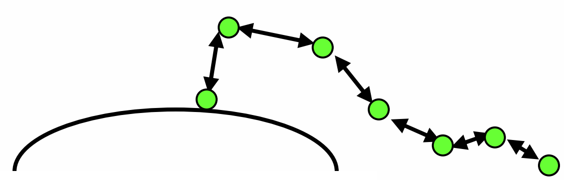


- 762 control hairs
  - Each is 7 vertices long



Hair Dynamics

- Treat control hairs as particle system
- For all (7 * 762) vertices in control hairs do
  - Physics simulation
  - Collision detection and reaction
  - Vertices of each control hair
    - Linked
    - Distance-constrained

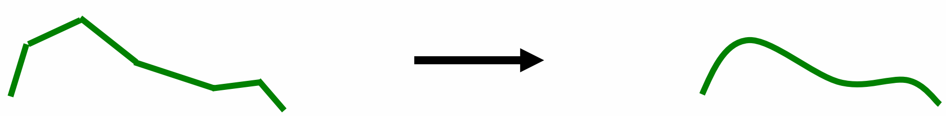


Physics Simulation

- Uses Verlet integration
  - Previous frame's position computes velocity
  - Less sensitive to frame rate
- Apply forces, then apply constraints
  - Iteratively
  - Particles converge
  - Thus take head-motion into account

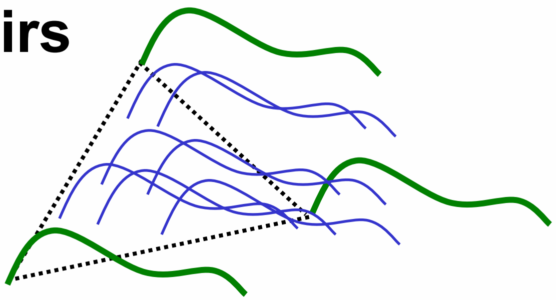
Now Have 762 7-Vertex Control Hairs

- Turn each control hair into 6 basic Bezier curves
  - 1 control hair has 6 segments
  - 1 basic Bezier requires 2 points and 2 tangents
- Concatenate and tessellate each set of 6 basic Bezier curves
  - Creates smooth control hair



Interpolate Control Hairs

- Interpolate 3 smooth control hairs at a time
  - Generates total of 4095 individual hairs



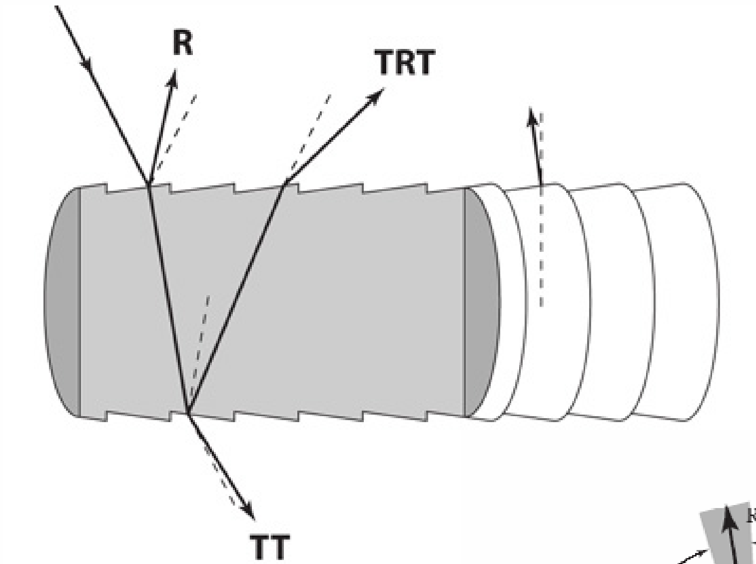
- Interpolation is post-tessellation
  - Performance reasons:
  - Tessellation is expensive
- Generates ~123k total vertices for hair alone

Wire-Frame Demo

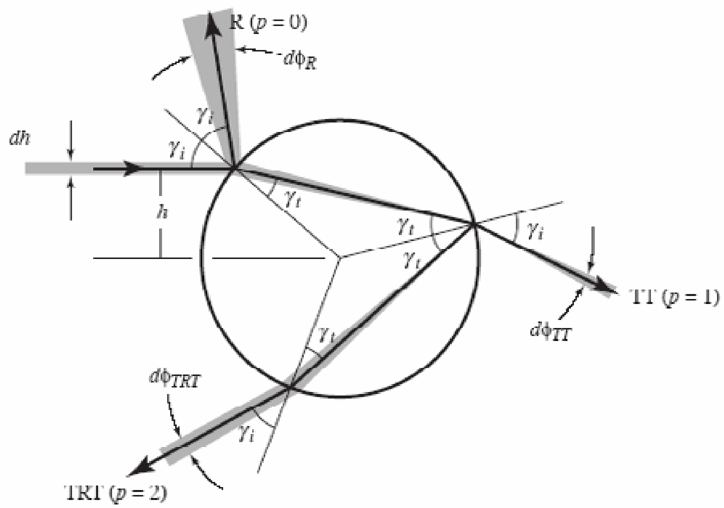
Hair Shading Based On

- “Light Scattering from Human Hair Fibers”
- By Steve Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan
- SIGGRAPH 2003

Paper Models 3 Distinct Highlights



- Uses path notation
  - R is reflection
  - T is transmission



Figures from "Light Scattering from Human Hair Fibers" (see previous slide)

R and TRT Highlights

- R – white primary highlight
- TRT – colored secondary highlight

Picture from "Light Scattering from Human Hair Fibers" (see previous slides)

TT Highlight

- TT – strong forward scattering component
  - Important for underwater hair

Hair Model Is 4-Dimensional Function

- Factor into lower dimensional terms
  - M_R (thetaH) * N_R (thetaD, phiD)
    + M_TT (thetaH) * N_TT (thetaD, phiD)
    + M_TRT(thetaH) * N_TRT(thetaD, phiD)

- Use 2D textures to encode as look-up tables
  - cos(thetaL), cos(thetaE)
    → M_R, M_TT, M_TRT, cos(thetaD)
  - cos(thetaH), cos(phiD)
    → N_R, N_TT, N_TRT

Make Most Aspects Tweakable

- Highlights:
  - Separation
  - Strength
  - Width
- Hair albedo
- Extinction coefficient
- Index of refraction

Hair Shading Demo

Shadowing

- "Opacity Shadow Maps"
- By Tae-Yong Kim and Ulrich Neumann
- SIGGRAPH 2001

Why Opacity Shadow Maps

- Opacity shadow maps ask:
  - What percentage of light is blocked from here?
  - Vs. Is the light blocked from here?
- Thus supports AA edges and volumetric rendering
- Regular shadow maps alias around edges
- Hair is 100% edges

Pictures From Tae-Yong Kim’s Website

No Shadows

15 slices

255 slices

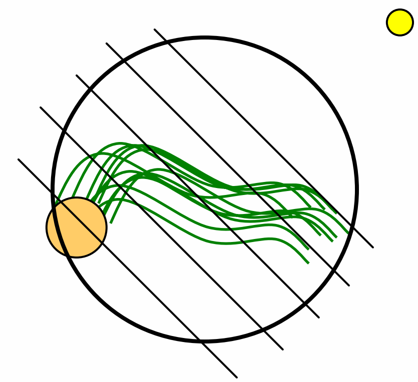
For Each Point In Map Compute:

$$\tau(z) = \exp\left(-\int_0^z \kappa(z')\,dz'\right)$$

- T(z): amount of light penetrating to depth z
- For hair:
  - Integral is sum over all strands between light and point being shadowed
- Compute sum via additive blending
  - "Extinction coefficient" K controls darkness of shadows

Creating the Opacity Maps

- Choose 16 slicing planes in hair
  - Uniform distribution
  - In hair bounding sphere



- For each hair-pixel and for each plane
  - Is hair-pixel closer to light than plane?
  - Yes: add hair to contribution (plane)
  - No: do nothing

Opacity Map Creation Implementation

- Render all hairs to 4 render targets
  - Each ARGB stores 4 planes
  - 4 passes
- Render all hairs to 4 MRTs
  - 1 pass
  - MRT shader is simple: 4 SLT and 4 MUL instructions

Using the Opacity Maps

- Hair-pixel position determines
  - Which opacity maps to look in
  - Where in opacity map to look in
- Hair-pixel positions generated by lines
  - Linearly interpolated vertex values are equivalent

Using Opacity Maps Implementation

- Vertex-shader computes
  - Texture coordinates for all 16 maps
  - Blend-weights to use
- Pixel-shader combines 16 look-ups
  - Via 5 dot4 instructions
- Add z-bias due counter limited z-resolution
  - Just like regular shadow maps

Shadowing Demo

Before

After

Questions

- http://developer.nvidia.com
  The Source for GPU Programming
- Matthias Wloka (mwloka@nvidia.com)