Outline

- Fur Generation Tool
- Normal Mapper Tool
- Shadow Techniques Used In 9700 Animusic Demo
Real Time Fur
Shells & Fins

Shells
Original geometry grown in the direction of normal for each edge.

Fins
Additional Geometry extruded along direction of normal for each edge.
ATI’s Fur Generation Tool

Developed by John Isidoro, ATI

ATI Mojo Day North America – September 19, 2002
Fin Textures Generated by FurGen

- Fins – 2 RGBA textures as output
  - RGB(DirOfAniso) & Alpha(Opacity)
  - R(uOffset), G(0), B(Thinning), & Alpha(LengthCulling)
Sample Fur Output

Straight Fur

Curly Fur
Fur Dilation Filter (Hole Filling)

- A 25-tap filter (5x5) is used to fill in the holes in the fin and shell textures. This is necessary for generating useful mip maps and for bilinear texture fetches.

For every black pixel, a weighted average of the neighboring 24 pixels are used based on:

$$\frac{1.0}{\text{distanceToPixel}}$$
Shell Textures Generated by FurGen

• Shells – 2 RGBA textures output for each shell
  – RGB(DirOfAniso) & Alpha(Opacity)
  – RG(uvOffset), B(Thinning), & Alpha(LengthCulling)
ATI Mojo Day North America – September 19, 2002

Fur Generation Tool Demo
Normal Mapper Tool

Developed by Dave Gosselin, ATI

ATI Mojo Day North America – September 19, 2002
Normal Maps

- Commonly created from a height map:

- Using low and high resolution models is becoming widely adopted:
ATI’s Normal Mapper Tool

- Command line interface that generates a bump map from a high and low resolution model (and a height map for fine details)
- High resolution model requires vertices and normals
- Low resolution model requires vertices, normals, and one set of texture coordinates
- The output normal map is applied to the low resolution model at runtime
Ray Casting

- For each texel mapped onto the low resolution model, a ray is cast out from the low resolution model and intersected with the high resolution model (Multiple rays if super sampling)
Ray Casting (Continued)

• If multiple polygons on the high resolution model are intersected, the one closest to the origin of the ray is chosen.
• Super Sampling: RADEON 9700 Car Paint Demo was generated with 36 rays per texel.
• Negative intersections are valid:
Beam Optimization

- As a preprocess, to minimize the number of intersection tests for each ray, a list of polygons on the high resolution model is generated for each polygon on the low resolution model.
- A 3-sided beam consisting of 6 planes is used. Each side of the beam is represented by 2 planes. If the plane test passes for either of the 2 planes, the test for that side of the beam passes.
Combining With A Height Map

• A height map can be combined with the ray cast normal map to provide micro detail to the normal map
• Scars on skin, imperfections in rock, bumps in bricks, etc.

<table>
<thead>
<tr>
<th>Normal Map</th>
<th>Height Map</th>
<th>Final Map</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Normal Map" /></td>
<td><img src="image2" alt="Height Map" /></td>
<td><img src="image3" alt="Final Map" /></td>
</tr>
</tbody>
</table>
Authoring Normals For Low Resolution Model

- Artists must make sure that rays cast based on the normals from the low resolution model, can accurately represent the high resolution model.
Spacing Texture Coordinates

• Space must be left between surfaces so there is room for a dilation filter (similar to fur)
• Bilinear fetches and mip map generation requires this
Normal Map Dilation Filter For Bilinear Fetches

- A 4-tap filter is used to grow the normals to neighboring pixels. This is necessary for generating mip maps and bilinear fetches. ATI’s tool runs this filter four times.
Tangent Space

- Tangent space in the normal map computations must match the tangent space of the run-time model.

- If normal map is generated in world space, tangent space isn’t needed, unless model is skinned.
Shadow Techniques Used In Animusic Demo
Static Shadows

- Shadows cast by non-moving light sources
- Scene geometry that doesn’t move
  - Terrain, rocks, buildings, etc.
- Great opportunity to optimize out the brute-force nature of dynamic shadow volumes!
Static Shadows in ATI’s Demo Engine

- Precomputed shadows are cut directly into the artist-generated geometry.
Advantages

• Looks like global stencil shadow volumes without the fill overhead!
• We can draw the polygons that are in shadow with a simpler vertex and pixel shader since fewer lights are affecting those pixels.
• The light color that casts the shadows are still animated! Dimming and color change is possible.
Beam Basics

- A beam is a closed volume created by 4 planes.
- The 4 planes are constructed from a single triangle and a light’s position.
- 3 of the 4 planes are defined by an edge of the polygon and the light’s position.
- The 4\textsuperscript{th} plane is simply the plane of the triangle.
- Near vs. Far beam: flip the normal of the 4\textsuperscript{th} plane.
Shadow Cutting Algorithm

• For each polygon in scene (polygon A)
  – For each polygon that falls into polygon A’s “near beam” (polygon B)
  • “Far beam” polygon B into polygon A and mark fragments of A that fall inside of beam B as in shadow.

![Diagram of shadow cutting algorithm](image)
Difficult Cases

• Due to the simplicity of the algorithm, there is no recursion. Only original artist-created polygons form beams.

• This automatically solves for cyclically overlapping polygons without special code:
Result of Shadow Cutting
Dynamic Shadows

- Dynamic shadows are used on animated geometry using stencil shadow volumes.
- These shadows are combined with the precomputed shadows by drawing a quad over the screen utilizing destination alpha.
Shadow Volumes

Sphere

Sphere’s Shadow Volume
Shadow Volume Extrusion Setup

We insert 2 degenerate polygons between the original polygons which share the appropriate face normal encoded in the vertex.

Original bordering polygons.
Stencil Buffer

- Dynamic shadow volumes are drawn into the stencil buffer.
- Pixels are essentially tagged as in or out of shadow based on their stencil value.

- How do we get the shadows into the color buffer?...
Destination Alpha

• As the scene is drawn initially, we also write a value to destination alpha.
• The dest alpha value solves the following equation per-pixel:
  \[ \text{destColor} \times \text{destAlpha} = \text{ambient} \]
  \[ \text{destAlpha} = \frac{\text{ambient}}{\text{destColor}} \]
• All pixel shaders write something useful to destination alpha as the scene is being drawn.
Contents of Destination Alpha
Shadow Quad

- A full-screen quad is drawn with the following blending enabled:
  - SrcBlend = Zero
  - DestBlend = DestAlpha

- This provides:
  - $\text{SrcColor} \cdot \text{Zero} + \text{DestColor} \cdot \text{DestAlpha}$
Questions?

Email devrel@ati.com to be notified about availability of our tools

Slides will be available on ATI’s Developer Relations web site:

ati.com/developer

Alex Vlachos – AVlachos@ati.com