PRMan as a CG Tool Bench

Apurva Shah
CVO, Prana Studios, ashah@pranastudios.com
PhotoRealistic RenderMan is a rendering system that generates high-quality 2D images from 3D scene-description information. Scene descriptions are created through the RenderMan Interface and are typically comprised of:

- a description of the geometry present in the scene,
- references to functions that describe how the geometry should be shaded,
- specifications of light sources (type, position, and orientation), and
- a specification of the virtual camera through which the resultant picture is to be imaged.
Is this a beautiful still from a movie...
...or several rendering techniques balanced together?

- Subdivision Surfaces
- RiCurve Primitives
- Subsurface Scatter
- Anisotropic highlights on fur
- Multi-bounce Raytraced Reflections & Refraction
- Depth map shadows
- Glow on practical light sources
- Depth of field
- Soft reflections optimized with brickmap caches
- Occlusion

Thursday, January 23, 14
In the context of visual filmmaking, the renderer is a tool that serves a singular purpose -- creating compelling imagery that assist in telling the story.
Renderer Architecture

- Optimized for complex, photo-real scenes.
- Hybrid renderer: scanline for hider, RT for light transport.
- Run hider first -- complexity bound by screen resolution not geometry.
- Shade grids not points (Reyes) -- computing partials becomes a simple differencing operation.
Geometry & Transforms

- All geometry ultimately maps to a RiPrimitive: RiSphere, RiPoints, etc. (http://renderman.pixar.com/resources/current/rps/geometricPrimitives.html)

- Transforms can be specified with RiTransform (matrix), RiTranslate, etc. (http://renderman.pixar.com/resources/current/rps/graphicsState.html)

- Scope with RiTransformBegin/RiTransformEnd

- Geometry is modeled in tools like Maya or digitized.

- RiProcedural generate RiPrimitives in memory, on the fly.
Geometry & Transforms

Things to try:

- Create simple skeleton with spheres & cylinders connected with transforms.
- Write a C program/python script to generate fractal with cubes into a rib.
- Use an open source physics engine (ex: ODE) to simulate a particle system and export RiPoints to rib per frame. Batch render to create an animated sequence.
- Convert one of the high resolution Stanford scans to a rib and render image.

(http://graphics.stanford.edu/data/3Dscanrep/)
Surface Shaders

- Shaders provide visual variety.
- Written in RSL and compiled with nshader.
- Surface shaders have three distinct parts: displacement, texture and illumination.
- Photos and painted textures can be used, may require a parametric space (uv, ptex) to map onto geometry.
- Texture use mip-mapping for efficient filtering.
- Shaders are a generalized concept and used elsewhere in the renderer: lights, imagers, etc.
Surface Shaders

Shader can access key variables: P, N, etc.

Shader sets output variables Ci and Oi.

By default variables in camera space. Need to convert to a more suitable space with convenience functions.

Anti-aliasing, spatial & temporal, tricky aspects of writing robust shader. Renderer provides derivatives to assist.

Attributes can be used to modulate shader behavior per object or prim.
constant.sl
noise.sl
texture.sl
Surface Shaders

Things to try:

- Write a shader that uses the RSL cellnoise function.
- Create a layered shader for aged plaster. Plaster should have cracks that expose brick underneath and should be covered with patches of grime and dirt.
- Write a displacement shader to give the appearance of cobblestone.
Micropolygons (MPs) atomic unit of shading.

In MP surface shader, run each light shader and add its contribution to surface illumination -- this is the light loop.

Surface and lights can support different illumination types (diffuse, specular, etc.) corresponding to lobes of a BRDF.
plastic.sl
pointlight.sl
spotlight.sl
Things to try:

- Implement a Cook-Torrance shader using the following Pixar note. (http://renderman.pixar.com/view/cook-torrance-shader)

- Implement anisotropic hair shader from Marschner paper. (http://www.cs.cornell.edu/~srn/publications/SG03-hair-abstract.html)

- Implement polynomial texture maps and use it to create a cloth shader. (http://www.hpl.hp.com/research/ptm/papers/ptm.pdf)
Camera Projection

- In rib, transforms and fov before RiWorldBegin establish the projection matrix.
- MPs once shaded are projected into an A-buffer.
- Energy of MP is spread out based on motion blur and DOF. Important cinematic cues.
- Render in energy linear space, then tone map for specific looks or display characteristics.
- Output of one render pass often fed back as input to another pass: shadows, env maps, etc.
Additional References

- A Brief Introduction to Renderman.  (http://renderman.pixar.com/view/brief-introduction-to-renderman)
- Advanced Renderman by Tony Apodaca.  (http://www.amazon.in/Advanced-RenderMan-Creating-Pictures-Kaufmann/dp/1558606181)