I am Adam Martinez, a senior shader writer at Sony Pictures Imageworks (SPI). Today I am going to talk about the work we did for *Oz The Great and Powerful* and how *Open Shading Language* (OSL) – along with some key rendering improvements at SPI – contributed to that work.

OSL

First a brief review about what OSL is, how it works, and where to go for more information.

OSL

OSL is for artists…

Adam Martinez
Sony Pictures Imageworks

OSL is a specification as well as an open-source codebase that provides artists with a simple yet robust language for describing materials for use in advanced rendering systems. OSL provides a secure and performance-focused environment that allows shader writers to focus on feature deployment and not on code optimization. Shaders are essentially decoupled from the low-level rendering API. Therefore shaders are protected from API changes, and the renderer is highly protected from crashing or incorrect behavior.

OSL

…and for developers

Adam Martinez
Sony Pictures Imageworks

For developers, OSL provides a rich framework to deliver a much-needed feature to users, without incurring a penalty for API compatibility. Renderer developers are free to pursue the evolution of their engine without too much concern for breaking existing shader libraries. Additionally, support issues for inappropriate use of API functions are greatly reduced. At SPI we've done it both ways and we will never go back given the stability and advantages.

OSL

How does it work?

OSL shaders return symbolic *closures*, not colors

Closures can be evaluated and sampled at any time

Separation of material description from ray tracing algorithms

Sampling is in the hands of the renderer

```
surface my_shader (
    float Ks          = 0.5,
    color ColorA      = 1,
    float Roughness   = 0.1,
    float eta         = 1.3
)
{
    if(Ks)
    {
        Ci = microfacet_beckmann(N, Roughness, eta) * ColorA * Ks;
    }
}
```

Adam Martinez
Sony Pictures Imageworks

The first thing to understand is that shaders do not compute final sample colors. Instead they return a list of weighted closures which correspond to renderer API functions. The renderer can analyze, evaluate and sample the closures or store them for later evaluation. Having the recipe for the description of the surface means that the renderer can now determine the best strategy for integration, and can do so very differently – if necessary – depending on context.

OSL

Improvements

- Stabilized Windows support
- Significant runtime optimization improvements

New Features

- Gabor and "simplex" noise
- `pointcloud_write()`
- `isconnected()`

Adoption

- OSL fully integrated in the *Cycles* renderer in Blender www.blender.org
- Chaos Group's V-Ray will ship with OSL support www.chaosgroup.com
- OSL is incorporated into Autodesk's Beast middle-ware solution

Adam Martinez
Sony Pictures Imageworks

I'd like to quickly point out some updates to the OSL project over the past year. The codebase has come quite a long way in terms of Windows support, and has incorporated many optimizations that came out of development both inside and outside of SPI. A few new features have been added, such as robust implementations of Gabor noise and simplex noise, plus new functions allowing shaders to write point clouds and retrieve information about the network topology. OSL has also been officially and fully integrated into the Cycles renderer, which is part of the Blender project. Furthermore, Chaos Group is nearing is first OSL enabled release of their popular V-Ray renderer.

OSL

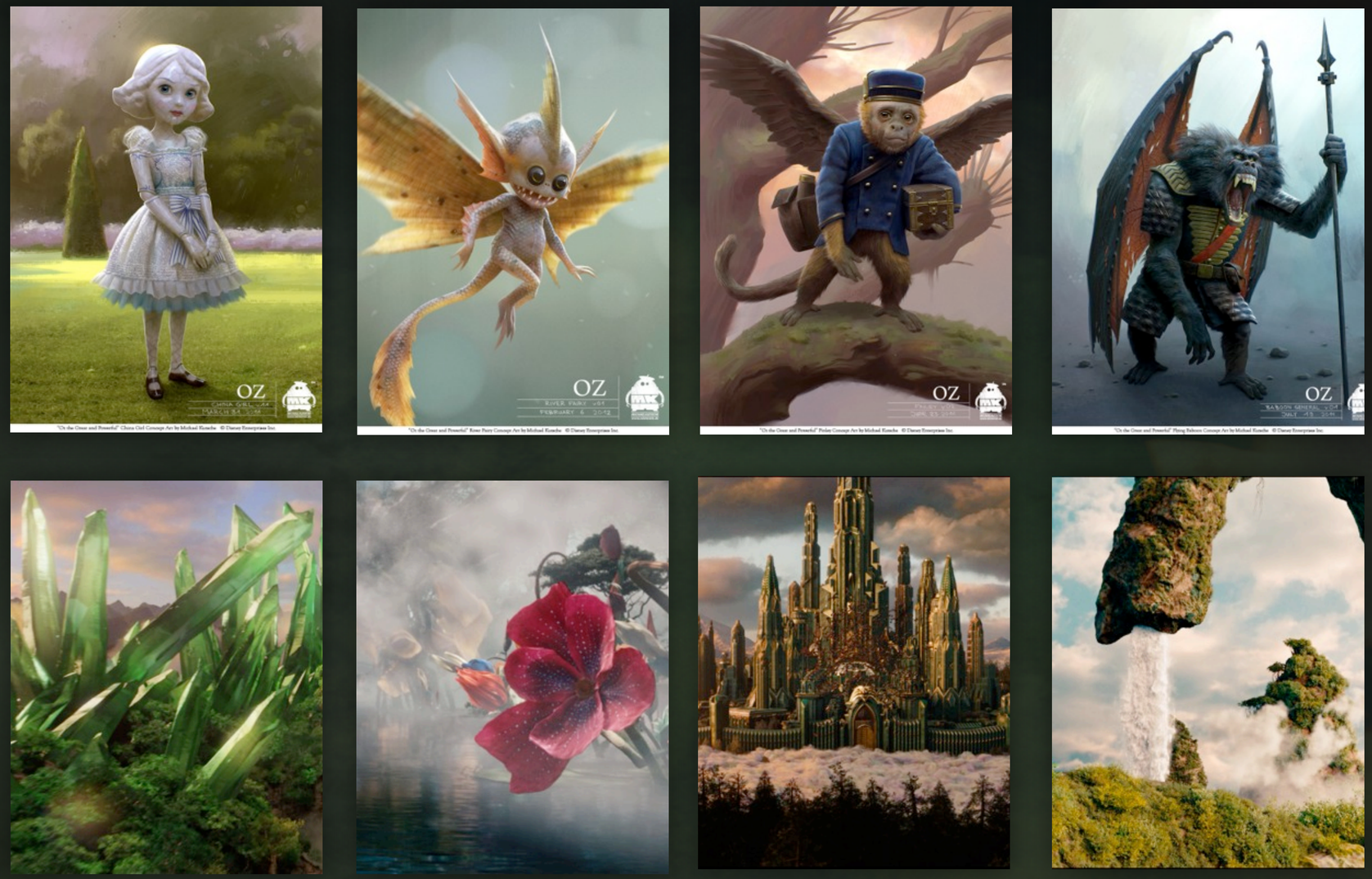
More Information

OSL is hosted on GitHub at https://github.com/imageworks/OpenShadingLanguage

Siggraph 2012 course http://blog.selfshadow.com/publications/s2012-shading-course/

Adam Martinez
Sony Pictures Imageworks

You can find out more about OSL and start trying it for yourself by acquiring the source from GitHub which includes a test render application. You can also learn more about the inner workings and history of OSL at SPI by grabbing the notes from last year's Physically Based Shading course.

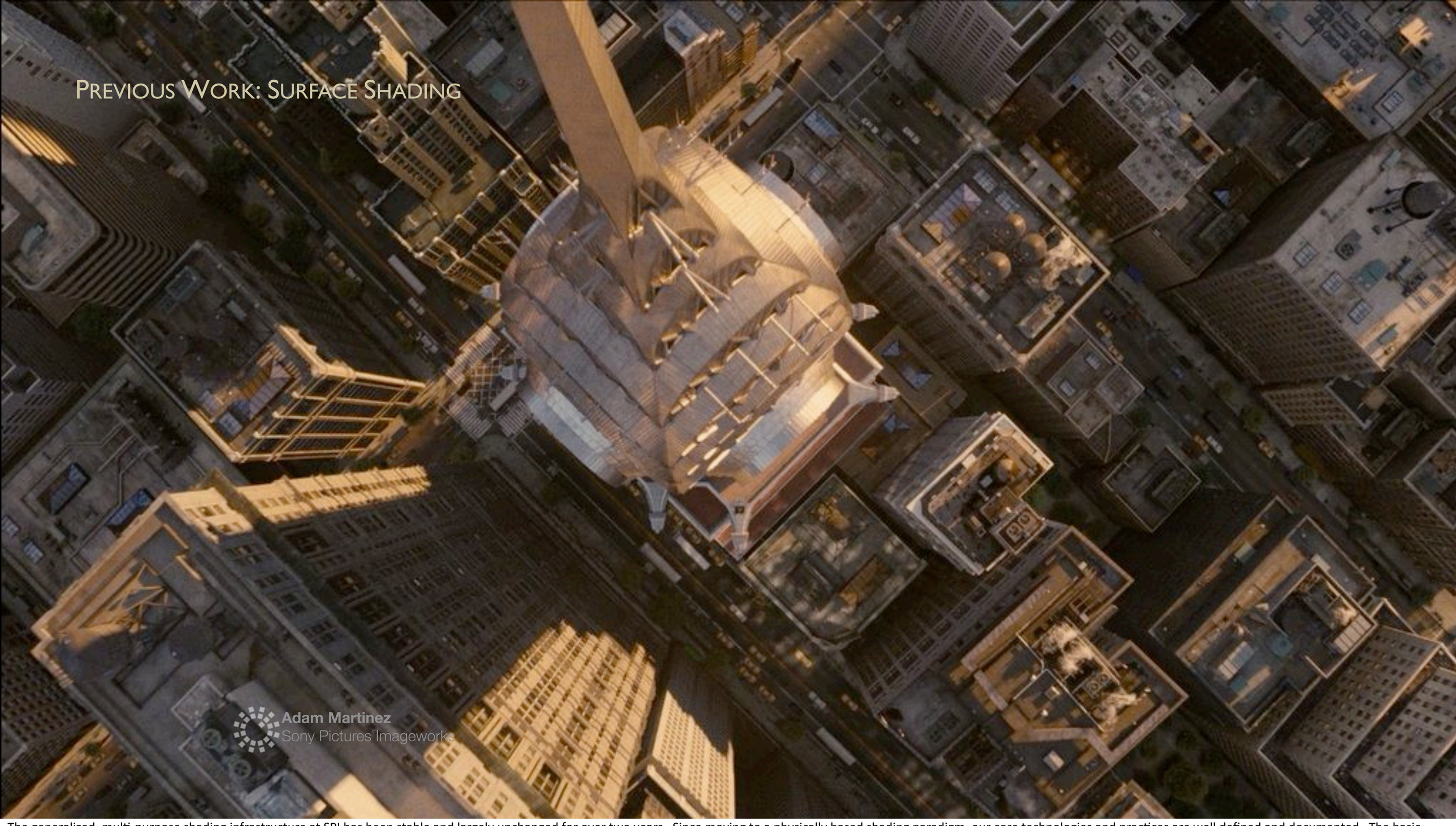


Adam Martinez
Sony Pictures Imageworks

©2013 Disney Enterprises, Inc.

The land of Oz: vicious flying baboons, cute but deadly river fairies, a talking monkey, a marionette with a mean streak...
Oz is a land of multitudes.
The locations for the film were richly detailed and treated very much like characters in themselves. From expansive mountain ranges to crystal gorges, dramatic forests to fantastic swamps, cloudscapes, cities and villages, each had their own flavor and the concept art went to great lengths to demonstrate the richness, variety and complexity that needed to exist on screen. The top row of images here are concept illustrations for a few of the characters for the film. The bottom row are final frames of a few of the environments.
Before look development and lighting began in earnest, we took stock of the tools we had available with a critical eye towards what we would need to complete *Oz*.

Previous Work: Surface Shading



The generalized, multi-purpose shading infrastructure at SPI has been stable and largely unchanged for over two years. Since moving to a physically based shading paradigm, our core technologies and practices are well defined and documented. The basic materials provide a broad palette of features that can be combined in an energy conserving manner – or not, if required – and additional visual complexity can be added using arbitrary material layering techniques. Even so, our materials are designed to be heavily texture driven. Almost every parameter is controllable via a painted or procedural texture.

This all-CG shot of New York City from *Men In Black 3* uses the same basic material in layered and non-layered configurations.

Previous Work: Skin Shading

Previously, skin shading at SPI had largely been an exercise in reinvention for every new show. The flexibility and feature set of our general purpose shading network, combined with – at times aggressive – material layering allowed artists to approximate skin reflectance in a very flexible way. These materials could be complicated enough to simulate multi-layer transmission, or highly simplified and stylized.

Unfortunately, these materials were highly specific to their application, and hardly ever survived beyond the show for which they were designed. They also had differing texture mapping requirements, which in some cases were far from straightforward. Additionally, these solutions tended to become complicated and hard to dial very quickly, plus material manipulation had to be very tightly controlled.

Previous Work: Hair Shading

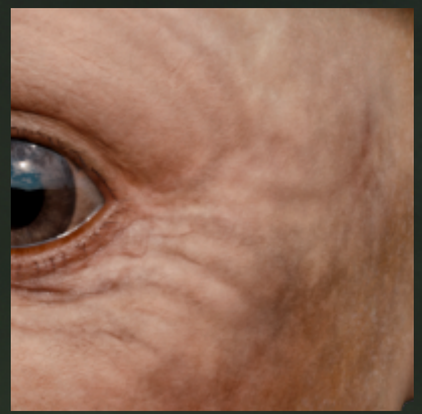
Until recently, hair shading at SPI had been fairly standardized since production on *Stuart Little*, even though the core technologies continued to evolve. Hair specular and diffuse have largely relied on common Kajiya-Kay models applied to curve primitives, with much of the development focused on texturing techniques. Our specific implementation merged a single diffuse component with multiple specular lobes to form a versatile foundation for many kinds of hair effects.

The traditional hair shading methods are computationally fast and very easy to implement, however they do not fit very well in a physically based shading ecosystem. One problem is that the lobes cannot sample the environment in a manner consistent with the BRDF. The reflectance of the specular model is also inconsistent with the micro-facet models we use on surfaces, which can cause some visual disparity. Finally, multiple scattering effects are out of reach.

Previous Work: Volume Shading

Rendering volume effects is an ongoing area of development across the industry. At SPI, our core volume engine was external to the 'primary' renderer and was extremely fast and efficient. It was, however, *entirely* external and had to be set up with an exclusive scene and light description; there was no direct translation of a scene into the volume renderer. Geometric integration was largely handled by depth and holdout passes that were pre-generated from the primary renderer. Volume elements usually required considerable compositing effort to integrate into a shot, and much of this was served by additional passes so that the lighting could be reassembled in 2D. Finally, these elements didn't contain global illumination, and it was a significant effort to get reflections of volumes onto surfaces. So, any efficiencies gained from the volume engine were usually eclipsed by the labor and synchronization of data between multiple renderers.

OSL



Adam Martinez
Sony Pictures Imageworks

It was in these last three key areas – skin, hair and volumes – that significant development was needed.

Skin

Efficient Rendering of Human Skin (D'Eon, Luebke, Enderton EUROGRAPHICS 2007)

Diffusion profile expressed as a sum of Gaussians

6 BSSRDF calls, 2 specular lobes (`microfacet_beckmann()`)

Rapidly prototyped, developed and deployed

Texture driven

Limited to skin only

Adam Martinez
Sony Pictures Imageworks

As the result of a facility-wide rendering summit in 2011, the shading department at SPI recognized the need for a standardized approach to rendering skin. The primary concerns were realism and flexibility, with performance being a tertiary concern. It had to look good and allow for a broad range of modification for different purposes. We selected the 2007 Eurographics paper by Eugene D'Eon, David Luebke, and Eric Enderton, entitled *Efficient Rendering of Human Skin*. In this paper, the authors present an approximation to multiple scattering layers using a sum-of-Gaussians approach. This method is highly dependent on texture data for final color and detail – a very desirable property for us – and also does a very good job of approximating the measured data upon which much of the research is based.

Skin

Variance (mm^2)	RGB Weights Red	Green	Blue
0.0064	0.233	0.455	0.649
0.0484	0.100	0.336	0.344
0.187	0.118	0.198	0
0.567	0.113	0.007	0.007
1.99	0.358	0.004	0
7.41	0.078	0	0

```
// Skin diffusion profile
closure color scattering = 0;
scattering  =  bssrdf_gaussian(0.0064) *  color( 0.233, 0.455, 0.649);
scattering  += bssrdf_gaussian(0.0484) *  color( 0.1,   0.336, 0.344);
scattering  += bssrdf_gaussian(0.187)  *  color( 0.118, 0.198, 0.0);
scattering  += bssrdf_gaussian(0.567)  *  color( 0.113, 0.007, 0.007);
scattering  += bssrdf_gaussian(1.99)   *  color( 0.358, 0.004, 0.0);
scattering  += bssrdf_gaussian(7.41)   *  color( 0.078, 0.0,   0.0);
```

Adam Martinez
Sony Pictures Imageworks

Our implementation of this formula loosely follows the approach described in GPU Gems 3, chapter 14, *Advanced Techniques for Realistic Real-Time Skin Rendering.* Here we have the diffusion profile from the paper, and the corresponding OSL code of the diffusion profile. OSL allowed us to express and manipulate a very complex series of operations in a very simple and intuitive manner. Energy conservation in the scattering component is implied by the normalized weighting of the profile. Diffuse and specular component weighting is managed by shader parameters.

Skin

From left to right, this image shows a snap shot of each component that makes up the final look for Finley's skin. The six subsurface lobes are followed by the diffuse-only component and the additional specular components. These are all summed together to produce the final look of the skin. Texture convolution is handled automatically by the renderer. We allow a portion of the texture contribution into the pre-convoluted illumination information. Specifically the diffuse lighting is weighted by the square root of the albedo texture. The remaining texture information is introduced after the Gaussian sums are computed. Keep in mind that this solution is entirely path traced in a single render session. There is no need pre-process textures or illumination data.

While this shader was designed specifically for human skin, it was applied to any creature that had a 'skin' aspect to it. The river fairy was an extreme example of manipulating the diffusion profile to create something not quite as human as Finley.
For more information about the specifics of the sampling technique, please check out Chris Kulla's talk *BSSRDF Importance Sampling* at 3:45 in Ballroom E

Hair

New physically based shading model

Accurate importance sampling

Marschner decomposition of components (Stephen R. Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan, 2003)

Supports multiple scattering

```
closure color R = hair_R( tangent, roughness, eta );
closure color TRT = hair_TRT( tangent, rotation, roughness2, eta );
closure color TT = hair_TT( tangent, roughness_theta, roughness_phi, eta );

Ci = ( R + TRT + TT );
```

Adam Martinez
Sony Pictures Imageworks

Hair shading at Imageworks has evolved considerably. *Oz* is the first show which utilized our new hair shading model developed by Alejandro Conty. It is a physically based model that provides accurate importance sampling and multiple scattering capabilities. It follows the Marschner decomposition of components as described in the SIGGRAPH 2003 paper *Light Scattering from Human Hair Fibers*.

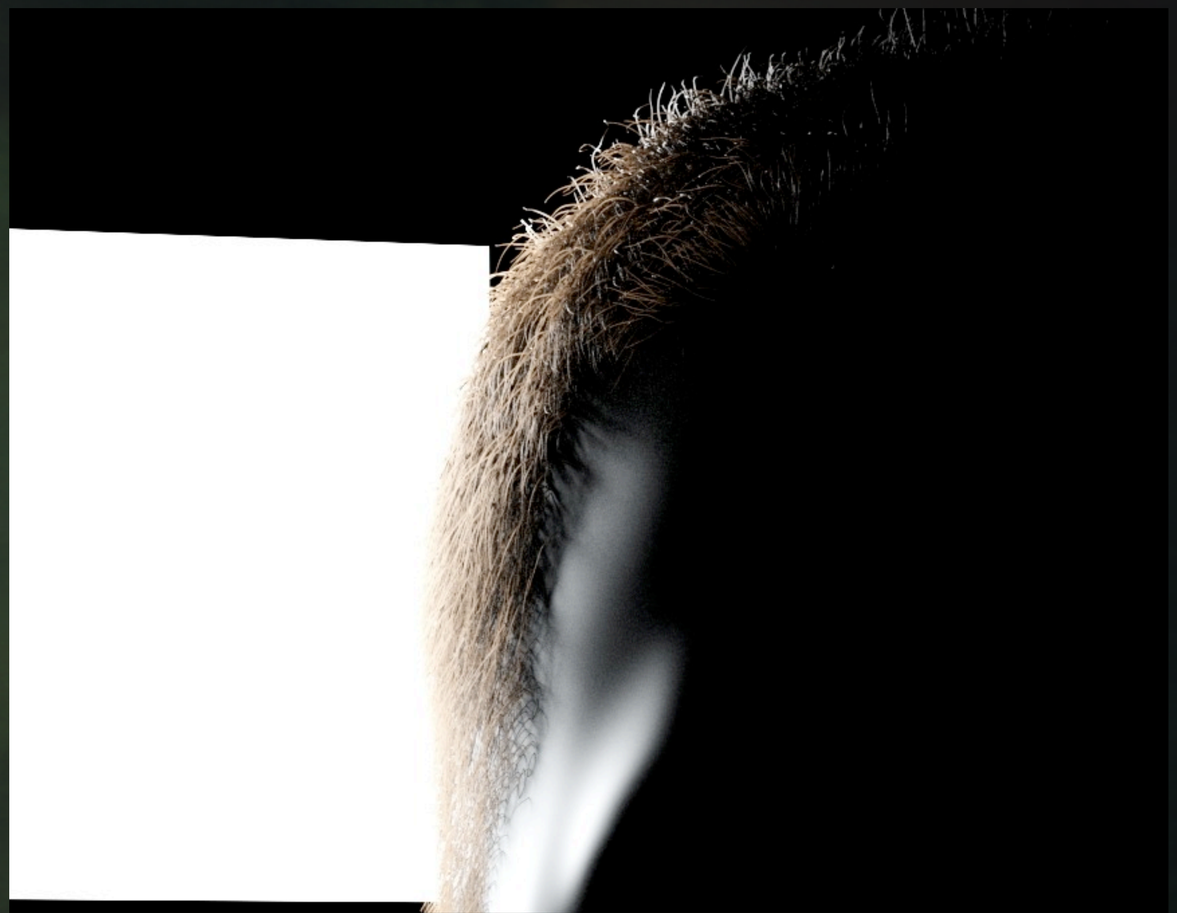
The model is implemented as three separate closures that are weighted and added together.
hair_R is the primary specular component, hair_TRT is the secondary specular component and hair_TT is the transmission component.
In this partial OSL code you can see all three closures take the same eta parameter. In this way each closure is modulated such that the combination of all three is energy conserving. This approach is very similar to the Kelemen/Szirmay-Kalos glossy + diffuse model presented in *A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling* (EuroGraphics 2001).

Hair

Adam Martinez
Sony Pictures Imageworks

The scan of actor Thomas Hayden-Church from *Spiderman 3* served as the primary test data for these new shading methods. In each of these pairs, the image on the right is the photographic reference, and the image on the left is the hair rendered with our new model.

Hair



Adam Martinez
Sony Pictures Imageworks

This is a before and after comparison of the global illumination response of SPI's hair shading models. The white square is emissive geometry, not an explicit light source. On the left, the Kajiya-Kay specular model fails to sample the reflection of the emissive surface in the hair, while on the right the new model is sampling two specular lobes accurately and with very little noise.
Currently this shading model is unpublished and is based on Conty's own research into importance sampling, and a lot of trial-and-error work with production data.

Hair

Adam Martinez
Sony Pictures Imageworks

©2013 Disney Enterprises, Inc.

We modified the T-TRT-TT hair model to add a modified Kajiya-Kay diffuse component to capture a dry look for our primate. This hair diffuse incorporates a cosine-distributed importance sampling function for indirect illumination, along with the appropriate albedo modulation to conserve energy when combined with the specular closures. From left to right we have diffuse, transmission and multiple hair_R lobes in keeping with SPI's traditional hair shading approach. The far right is the final summed result.

The baboon characters made heavy use of the capabilities of the new hair shading model.

The baboon’s hair is almost entirely specular using this new method. Here you can see the baboons hair is largely defined by it’s primary specular response.

Finley combined the new skin and hair models to the greatest extent in the film. The combination of physically plausible skin shading coupled with accurately sampled hair meant that each element could be lit consistently.

Volumes

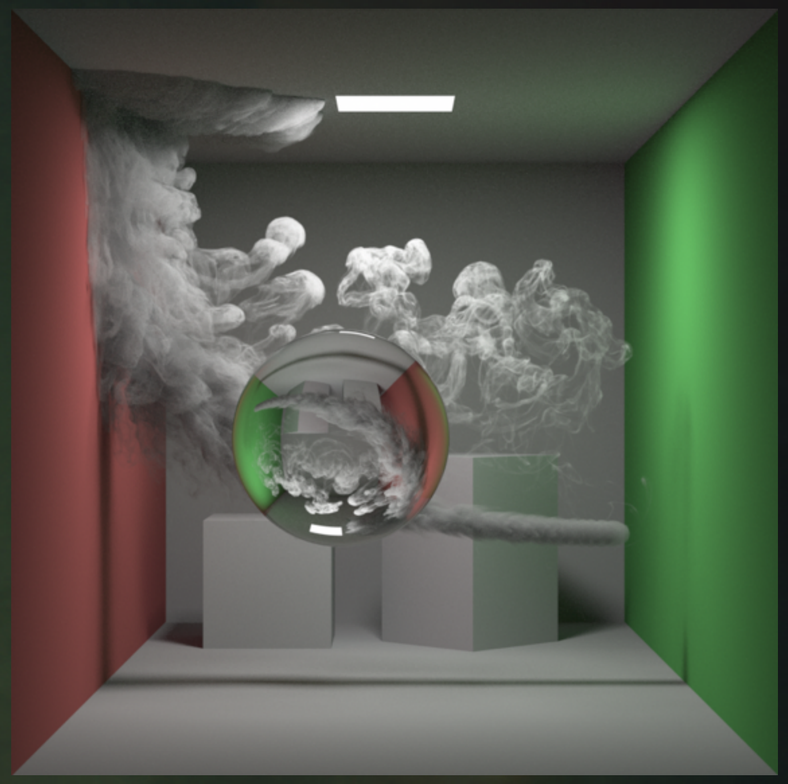
Describes volume properties at integration

Can be expressed as density and albedo

Standardized OSL volume material library

Applied to volume primitives and polygon containers

```
closure color S = Scattering_Color * henyey_greenstein(eccentricity);
closure color A = Absorption_Color * absorption();
closure color E = Emission_Color * emission();
Ci = S + A + E;
```



Adam Martinez
Sony Pictures Imageworks

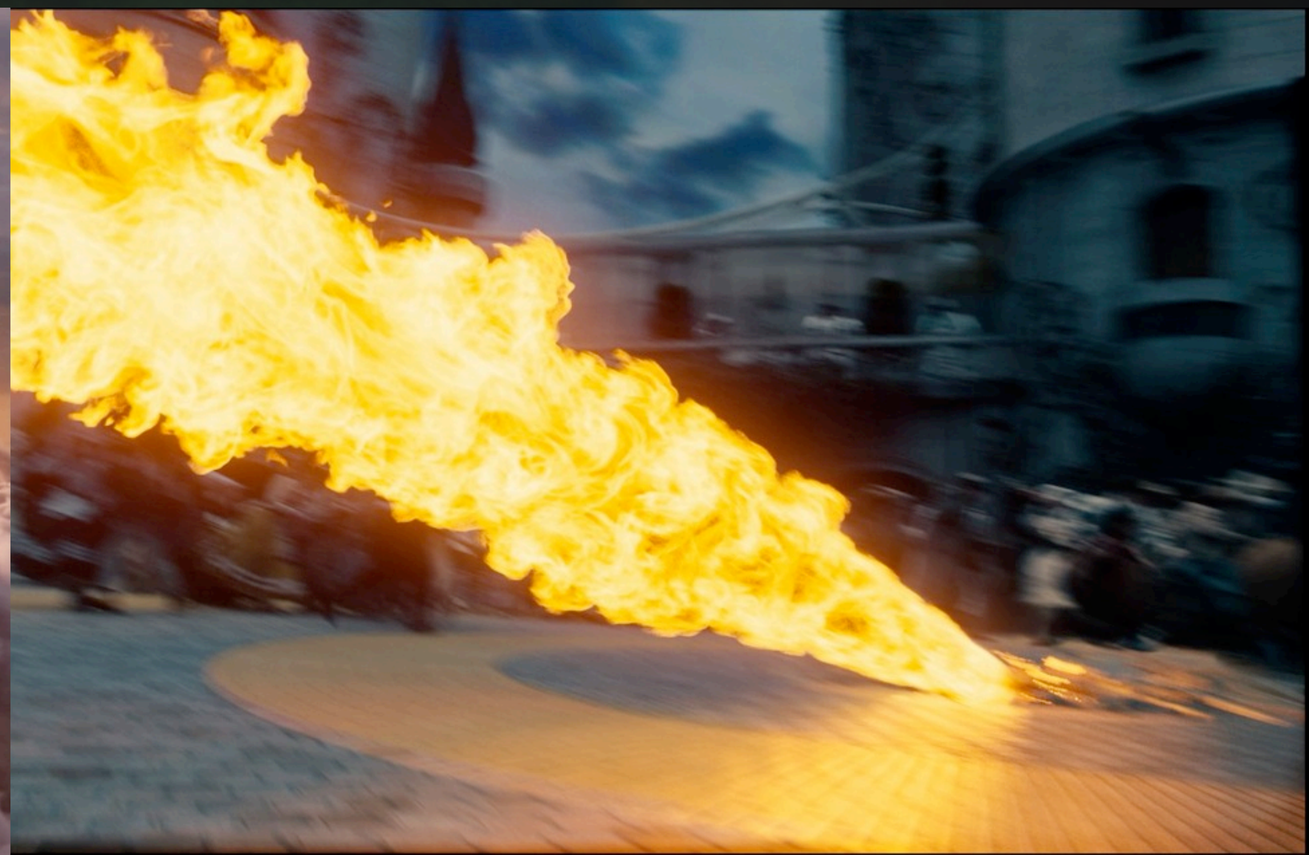
Since *Men In Black 3* SPI has been rendering all volume elements inside the primary path tracing engine. Volumes are importance-sampled and they receive and contribute to global illumination. Volumes can be defined by volume primitives (field 3d assets) or polygon mesh containers. At SPI we have built volume-specific materials around these building blocks, which provide general and special-case capabilities to both fx and lighting technical directors.

In OSL, volume materials are described as a combination of three closures: scattering, absorption and emission. In this code example the scattering component employs the Henyey-Greenstein scattering function which is the SPI facility standard. The weighting of each closure is the primary function of the OSL volume shader. The shader knows nothing about step sizes or other integration details and largely functions much like any other OSL shader.

©2013 Disney Enterprises, Inc.

These new methods – together with refined practices – finally made it possible to achieve complicated volume effects that interacted with the environment realistically.

Volumes



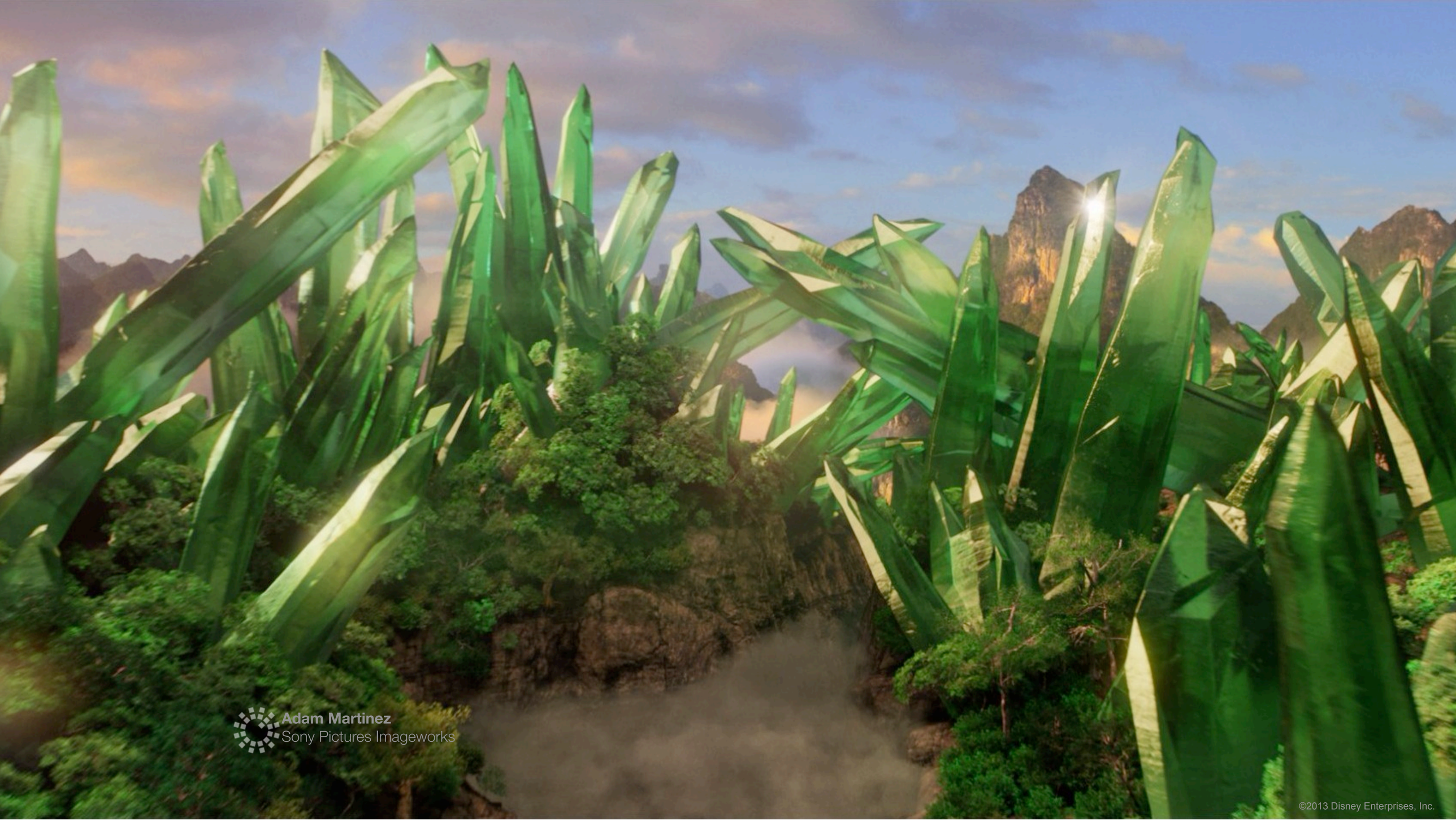
Adam Martinez
Sony Pictures Imageworks

©2013 Disney Enterprises, Inc.

Our basic volume materials are focused on delivering volume looks for clouds, mist and other phenomena. On the left, the clouds are rendered with a combination of volume data and procedural noise.

Our pyroclastic shaders make use of fluid simulation data to render fire and smoke effects. On the right, the fire is primarily rendered via the emission closure weighted by a black body radiation function.

Simulating multiple scattering events inside the volumes is a function of the renderer, not of the OSL shader. This means that we are free to explore alternative methods for approximation without having to redesign the shaders or modify user settings. This is similar to the way we are able to implement new integration methods for surface materials without fear of breaking compatibility with existing shaders.



Having volumes and surfaces integrated in the same renderer, and described in the same shading language, was a huge win for much of the work we did on *Oz The Great and Powerful.* It reduced complexity in lighting setups and allowed for greater continuity among elements of the same shot.

In some cases this integration was critical, as in the case of hair interacting with smoke effects. At 3:45 today in Ballroom E, the talk *Oz: The Great and Volumetric* will go into more detail about the GI rendering pipeline for volumes including integration of Field3D, multiple scattering approximations and the manipulation of volumes using Katana.

To summarize, much of the rendering work executed on *Oz The Great and Powerful* made use of significant advancements in both the in-house renderer as well as OSL. Using the same language to describe phenomena such as hair and clouds as we do to describe surfaces, reduces inconsistencies in both the interface and the approach. This means that there are far fewer exceptions in the rendering pipeline from data management to creative workflow. Using physically based techniques and applying them where ever possible similarly affords greater visual consistency.

OSL

THANK YOU

amartinez@imageworks.com

https://github.com/imageworks/OpenShadingLanguage

Adam Martinez
Sony Pictures Imageworks

I would like to thank you for your attention and if you have any questions about this information please feel free to e-mail me at this address, or join the OSL community by following the links on the GitHub repository.
These slides will be made available on-line after the conference. Due to copyright restrictions however, some of the video slides will be replaced with stills.