Rendering Fur in "Life of Pi"

Ivan Neulander
Google Inc.

Toshi Kato, Kevin Beason
Rhythm & Hues Studios

We present several technical advancements developed at Rhythm & Hues for the efficient rendering of photorealistic fur in Ang Lee's Oscar-winning feature film *Life of Pi*. We drew heavily on this work to stereoscopically render the tiger Richard Parker and several other animals, all with sufficient realism and aesthetic control to capture the director's ambitious vision.

We summarize existing work on which our implementation builds, and describe in detail some recent improvements in the areas of hair shading, performance optimizations, and post-rendering tools for motion blur and stereo image synthesis.

Hair Shading

Life of Pi was our first feature film to rely almost exclusively on area lights. We used multiple importance sampling (MIS) to combine light-based and BSDF-based sample rays, deploying our adaptive importance sampler [Neulander 2011] to weed out occluded or otherwise unimportant light paths during heavy ray gathering. We used a combination of an HDRI-mapped infinite sphere to capture distant lighting, with smaller rectangular and finite-sphere lights to model nearby source of illumination. Image-based importance sampling, based on precomputed two-dimensional CDF tables, was critical to rendering low-noise, high-contrast lighting from the HDRIs, which were generally sampled at full resolution in order to preserve crisp shadow detail.

The BSDF used for our hair strands was based on the *cone-shell* model described in [Neulander 2010], which strikes a good balance of forward and backward scattering, and a relatively simple derivation. For *Life of Pi* we implemented two enhancements: 1) a secondary specular lobe that could be shifted along a hair strand and whose shape and color could vary independently from the primary specular lobe (this mimics the TRT component of Marschner's model). 2) We opted for a more aggressive BSDF-based importance sampler, using a precomputed CDF table to approximate the Wigner semicircle distribution (the optimal choice for this BSDF). When used with MIS for specular reflections, this produced less noise than the flatter piecewise-linear importance PDF originally proposed.

Renderer Optimizations

We made extensive use of the *hair reflection cache* described in [Neulander 2010], which sparsely stores primary hair shading samples along each strand in a temporally coherent yet refinable fashion. We extended this cache to store individual contributions from various light paths, and optimized its memory layout using clustered allocations to reduce fragmentation. We also implemented a read-copy-update mechanism to minimize thread locking while accessing the cache. The use of this cache cut our render times in half, simultaneously reducing noise relative to uncached hair shading.

A large proportion of render time was spent testing occlusion along hair-based gather rays, with layers of semitransparent fur being the most common and expensive occluder. We employed two techniques to accelerate this: The first was a modified form of the volumetric occlusion approximation described in [Neulander 2010], which reliably identified rays that were likely to be blocked by nearby skin, alowing the renderer to avoid tracing them. The remaining non-skin-bound rays were attenuated using an accurate raytraced estimation of their occlusion. The second optimization leveraged our renderer's dual representation of fur as both scanline triangles and raytraced hair primitives. This allowed using screen door transparency to accelerate the ray tracing: For the ray-occluding fur, semitransparent tube primitives were made opaque and their thicknesses were correspondingly reduced so as to preserve coverage. These operations were applied at the control vertices of each strand, allowing for precise lengthwise variation in opacity and thickness. By making these strands thin and opaque, we eliminated the need to trace multiple levels of transparency rays through them, and we also reduced the number of ray intersections by shrinking the ray targets. This produced a dramatic, artifact-free speed increase for secondary gather rays, while preserving the desired look of true transparency for the primary rays.

Postprocessing

Due to the immense geometric complexity of our fur, computing motion blur in the renderer by explicitly sampling visibility over time would have greatly increased our render times. So we made extensive use of our *pixmotor* tool [Neulander 2007] to synthesize motion blur by postprocessing static images using motion vectors (efficiently output by our renderer). For *Life of Pi*, we redesigned *pixmotor*'s high-resolution work buffer to reduce its memory footprint. This was entailed by the high number of color layers stored in each rendered image (containing individual contributions of various lights and various scatter events per light). Our solution was to store normalized device coordinates in the work buffer, rather than explicit colors as before. While this added a level of indirection in accessing pixel colors, it ultimately sped up *pixmotor* by significantly reducing memory bandwidth, especially with multithreading.

To improve efficiency on stereoscopic projects such as *Life of Pi*, we extended *pixmotor* to synthesize right-eye images from rendered left-eye images, a postprocess we labeled *pixstereo*. Primarily horizontal motion vectors were accurately computed from the parallax between the left and right cameras, and these were used in a specialized single-iteration mode of *pixmotor* to generate a pixel-shifted image as seen from the right-eye camera. We applied more aggressive hole-filling algorithms here than with *pixmotor*, and used an even higher-resolution work buffer (often 6x) to preserve quality. The above memory optimization was crucial, since storing all color channels at 6x resolution would have exceeded our RAM budget. For some rendered elements, the *pixstereo* images were of sufficient quality for final shots, but mostly they were suitable only for preview. However, with runtimes under a minute, the right-eye images effectively came for free, allowing many iterations of animation and lighting to be viewed stereoscopically without the added cost of right-eye rendering.

References

NEULANDER, I. 2007. Pixmotor: a pixel motion integrator. In *ACM SIGGRAPH 2007 sketches*, SIGGRAPH '07.

NEULANDER, I. 2010. Fast furry ray gathering. In *ACM SIGGRAPH 2010 Talks*, SIGGRAPH '10.

NEULANDER, I. 2011. Adaptive importance sampling for multi-ray gathering. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11.