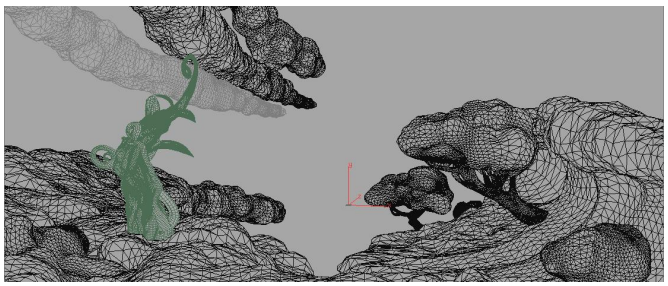
Cloud Modeling And Rendering for "Puss In Boots"

Brett Miller Ken Museth Devon Penney Nafees Bin Zafar

DreamWorks Animation, SKG



The story of DreamWorks Animation's "Puss In Boots" called for Puss and his crew to climb up to a world of clouds. Alongside natural looking clouds, there would be trees and buildings made of clouds. The art direction called for a lot of detail in the density, and "magic hour" lighting tones with strong reds and blues. To achieve this effect we created a system that took a rough digital sky set, created high resolution volumetric cloud representations, and allowed artistic lighting controls with multiple scattering effects.

High-Resolution Modeling Of A Digital Sky

FX required a workflow that would introduce an accurate representation of the clouds as early as possible in our pipeline. Conversely, layout and character animation preferred to work with proxy representations of the clouds. To this end, we modeled modular clouds as polygonal meshes that layout could dress the set with, and that animation could immediately animate characters on. When they arrive in the FX department, the polygonal clouds were then scan-converted into narrow-band level sets, represented in a compact data structure. This data structure, dubbed VDB[Museth 2012], forms the basic volume representation for our clouds system. VDB models a virtually infinite 3D index space that allows for fast and cache-coherent data access into sparse volumes of extreme resolution. It imposes no topology restrictions on the sparsity of the volumetric data, and it supports fast random insertion, retrieval and deletion of data. Additionally, the underlying hierarchical structure of VDB facilitates bounding-volume acceleration during rendering. The level sets representations of the polygon models were next displaced using procedural noise, after which additional procedural noise was added to create a variety of textures, from packed, clumpy cumulonimbus to feathery nimbus clouds. Final details were applied using a suite of animated volume primitives such as wispy curves and spheres. These density volumes were stored both as camera aligned frustum buffers, in order to maximize the capture of fine detail, and as coarse rectangular buffers, for lighting calculations and extra-frustum deep shadow generation. Typical VDB resolutions were $15,000 \times 900 \times 500$.

Rendering

Our sparse approach to cloud modeling posed certain challenges and benefits for rendering. Typically density for shadow maps was zero outside the bounds of the frustum VDBs which resulted in light leaks duirng the beauty pass. We addressed this issue by generating low resolution rectangular grids for density queries outside the frustum bounds during shadow map generation.

We found that FX artists didn't need complex and general-purpose lights while rendering expensive direct illumination passes. Thus, we optimized the light shaders and underlying API by selectively removing infrequently used parameters and features, but seamlessly making them available for special cases.

Many shots required rendering numerous large overlapping grids simultaneously. Rays were segmented according to the overlapping bounding boxes to efficiently reduce empty space traversal [Wrenninge et al. 2010]. Additionally, motion blur calculations were optimized by simultaneously marching temporal rays rather than treating them sequentially.

Lighting Clouds

The primary challenge to lighting clouds was to create a multiple scattering effect. Though brute force approaches have been explored for this problem, we favored a set of visual approximations. In our model a cloud has two zones dominated by different lighting effects: a first order scattering zone, and a multiple scattering zone. The former is the part of the cloud facing the light, and the latter is the "underside" of the cloud.

We posed the multiple scattering effect as a wavefront propagation problem, specifically as the Eikonal equation. The Henyey-Greenstein phase function is formulated as the speed function for this partial differential equation. A few voxels closest to the light in the density grid are initialized as the boundary. The Fast Iterative Method, an expanding wavefront scheme, is used to solve the Eikonal equation efficiently [Jeong and Whitaker 2008]. The solution represents a time of arrival for the light, which is mapped to radiance values. Clearly this result is only valid for light which propagates from a single location. For the sake of efficiency we make the simplifying assumption that far away from the boundary, on the "underside" of the cloud, the result faithfully approximates the mean free path through the medium. We cannot prove the correctness of this assumption, but we observe that it looks plausible.

Finally we add an ambient occlusion pass to the cloud lighting model. The ambient occlusion is calculated efficiently using spherical harmonics. A few thousand sampling locations are generated in the cloud volume. The visibility in all directions is sampled at these locations by ray marching through the volume, and visibility function coefficients are computed by projecting the results to the spherical harmonics basis [Sloan et al. 2002]. During lighting the local ambient occlusion term is generated by sparse spatial interpolation of the coefficients of the visibility function.

JEONG, W.-K., AND WHITAKER, R. T. 2008. A fast iterative method for eikonal equations. *SIAM J. Sci. Comput. 30* (July), 2512–2534.

MUSETH, K. 2012. VDB: High-resolution sparse volumes with dynamic topology. *Accepted for publication in ACM Transactions On Graphics*.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21* (July), 527–536.

WRENNINGE, M., BIN ZAFAR, N., CLIFFORD, J., GRAHAM, G., PENNEY, D., KONTKANEN, J., TESSENDORF, J., AND CLINTON, A. 2010. Volumetric methods in visual effects. In *ACM SIGGRAPH 2010 Courses*, ACM, New York, NY, USA, SIGGRAPH '10, ACM.