Ten Unsolved Problems in Rendering

Paul S. Heckbert

PIXAR
P.O. Box 13719
San Rafael, CA 94913
USA

It is helpful to step back and review developments in image synthesis as a whole. First we discuss progress in rendering over the past ten years, and then we list ten unsolved problems.

RECENT PROGRESS

In 1977 Newell and Blinn reviewed progress in raster graphics and wrote a prophetic list of six outstanding problems in realistic image synthesis [1], which we paraphrase here (our comments in italics):

1. complexity: There is a huge disparity between the complexity of scenes in the real world and synthetic scenes.

Since 1977 we have made considerable progress in complexity due to faster computers, larger memories, improvements in algorithm efficiency, and increased use of procedural modeling, but we cannot yet regard the complexity problem as solved. We have found that handling complexity is as much of a modeling and logistics problem as it is a rendering problem.

2. surface properties: Many objects in nature are not well described as smooth surfaces with small perturbations. Hair and clouds are out of reach of current techniques.

Fractals have become popular in recent years for modeling rough surfaces. They perform well for highly regular structures such as snowflakes or highly random structures such as moon craters, but are less applicable to complex organic and physical objects like trees and clouds. Good animation and rendering of hair and clouds is closer, but still out of reach.

3. transparency: Refraction has not been satisfactorily simulated.

Refraction can now be considered solved due to Whitted's recursive ray tracing algorithm.

4. lighting models and shadows: All existing models use point light sources.

In recent years Nishita has simulated polygonal light sources and their penumbras, producing some of the most realistic pictures to date. Cook's probabilistic (''distributed'') ray tracing allows penumbra calculation for more general shapes.

The radiosity method introduced by Goral simulates indirect reflection from diffuse surfaces, simulating diffuse reflection globally for the first time. Recent work by Immel generalizes radiosity to specular reflection. Present radiosity methods are only practical, however, for simple, static, diffuse, polygonal models.

5. antialiasing: Aliasing is a well understood phenomenon, and antialiasing of two dimensional graphics has been accomplished, but there are still problems antialiasing small details in hidden surface algorithms, and temporal aliasing is poorly understood.

Since 1977 our understanding of antialiasing has improved. A number of efficient algorithms for antialiased hidden surface elimination and motion blur have been developed using both numerical and analytic techniques. Cook's stochastic sampling and probabilistic ray tracing techniques simultaneously provide new algorithms for spatial antialiasing, motion blur, penumbras, depth of field, gloss, translucency, and dispersion. Though very general, stochastic sampling is inefficient when the probability distribution is wide. Kajiya's ''importance sampling'' helps optimize the integration of wide distributions such as diffuse reflection, but his ray tracing technique for simulating radiosity cannot yet be considered practical. Analytic integration methods are often superior under specialized situations such as polygon antialiasing and linear motion blur.

6. systems integration: The various individual techniques must be combined in a consistent way.

Ray tracing has provided a flexible framework within which a variety of modeling and shading techniques can be tested, but most other existing rendering systems are much more limited, often being restricted to polygonal objects and local shading models.

A theory development that was unanticipated by Newell and Blinn is Kajiya's ''rendering equation'', which lends conceptual unity to several formerly disjoint techniques: the "Utah approximation", radiosity, and ray tracing. The rendering equation suggests an elegant taxonomy of past and future shading models.

UNSOLVED PROBLEMS

Below are listed ten unsolved problems in rendering which, when solved, will define the future state of the art. Most of the problems consist of implementation, optimization, or systems integration of the new techniques mentioned above.

1. conversion of implicit models to parametric form

Scan conversion rendering algorithms expect surface (B-Rep) models as opposed to implicit functions or CSG models. Only ray tracers are currently capable of rendering arbitrary implicit models. Can algorithms be found to parameterize implicit models, for example to convert a blobby model to polygons, or better yet, to patches?

2. high quality texture filtering

Methods exist for high quality texture filtering, and for efficient texture filtering, but we have no method for general, random access, efficient, high quality texture filtering. Current high quality texture filters have unbounded cost per pixel, so most animation production is done using cheaper methods that are restricted to box filtering of rectangular areas, resulting in poorer quality. Can an efficient, high quality texture filter be found?

3. antialiasing

Our understanding of antialiasing is improving, but not complete. The theory for uniform resampling of signals is adequate, but non-uniform resampling problems such as perspective are poorly understood. Can antialiasing of highlights and procedural textures be mastered? And is there any prospect for general analytic antialiasing? The plethora of ad-hoc antialiasing techniques need to be systematized and interrelated.

4. shadows outside ray tracing

Most pictures containing shadows are rendered using ray tracing. A number of shadow algorithms have been proposed (shadow volume, preprocess, z-buffer) but they are not in regular use because they are inefficient, often doubling rendering time. Most shadow algorithms are limited to polygonal objects and point light sources. Can shadow algorithms be generalized to penumbras, motion blur, and non-polyhedral objects without resorting to ray tracing?

5. practical ray tracing

Ray tracing is currently the rendering algorithm of choice among image realists, but it is the bane of animators because of its slow speed. Can it be made practical? Is there a fundamental new algorithm to be discovered that will make rendering efficient? As Turner Whitted once asked, "can we find the counterpart of the FFT algorithm for computer graphics"?

6. practical radiosity

Many of us enjoy the radiosity pictures Cornell brings to SIGGRAPH each year, but is anyone out there really using it? How can radiosity methods be optimized to make them practical for high complexity animated scenes and how can they be applied to scenes containing arbitrary curved surface primitives?

7. frame-to-frame coherence

There's been much talk of frame-to-frame coherence over the years, but little progress. We have few methods for automatically detecting and exploiting frame-to-frame coherence. Experienced computer animators have a large body of tricks for optimizing rendering and recording of animation: exploiting static backgrounds, cyclic motion, ignoring off-screen objects, and compositing 2-D elements. Can we automate these tricks to help reduce the logistical drudgery of animation production?

8. automating model culling

Multiple levels of detail and hierarchical bounding boxes are well known methods for optimizing the rendering of detailed or off-screen objects, but they are seldom used because of the tedium of setting up the necessary databases. Can the creation of levels of detail and hierarchical bounding boxes be automated?

9. smooth model transitions

In computer graphics we often find that it's more important to be continuous than to be correct. When switching between levels of detail of a polygonal model, for example, how can we make the transition smoothly? Perlin has suggested a micro-macro continuum of models: at a macroscopic scale, a bumpy surface should be modeled using geometry, at an intermediate scale it is best represented by a bump map, and at a microscopic scale the bumpiness becomes the statistical distribution of a shading function. Can these transitions between levels of detail be made without visible artifacts?

10. affordable real-time rendering hardware

We'd all like a personal computer capable of generating realistic renderings in real-time, but such a price/performance combination seems years off. What algorithms and machine architectures are best suited for real time? Will brute force prevail over algorithmic optimizations?

REFERENCES

[1] Martin E. Newell, James F. Blinn, ''The Progression of Realism in Computer Generated Images'', *ACM 77 Proceedings*, Oct. 1977, pp. 444-448.