USING CACHING AND BREADTH-FIRST SEARCH TO SPEED UP RAY-TRACING

(extended abstract)

Pat Hanrahan

Abstract

Ray-tracing is an expensive image synthesis technique because many more ray-surface intersection calculations are done than are necessary to shade the visible areas of the image. This paper extends the concept of beam-tracing so that it can be coupled with caching to reduce the number of intersection tests. Two major improvements are made over existing techniques. First, the cache is organized so that cache misses are only generated when another surface is intersected, and second, the search takes place in breadth-first order so that coherent regions are completely computed before moving onto the next region.

Introduction

Ray-tracing has attracted considerable attention recently because of the super-realistic images that can be produced. Lighting and shading effects that require information about the global environment, such as shadows, reflections and refractions, can be calculated by recursively tracing rays from the surfaces they intersect [Whitted, 1980]. Distributed or stochastic ray-tracing can be used to simulate other optical effects, such as motion blur, finite-sized light sources, prismatic effects, etc., and to remove many of the artifacts due to point sampling the image [Cook, Porter and Carpenter, 1984]. Ray-casting can also be used to generate line drawings and sectioned views, and to perform the volume integrals needed for the calculation of mass properties [Roth, 1982]. Another advantage of ray-tracing is that it is conceptually elegant and easy to implement. The models that comprise the scene can be rendered if a procedure to intersect a ray with their surfaces is provided. Because of the object-oriented architecture, a ray-tracing system is easy to maintain and extend. The number of geometric primitives that can be ray-traced is quite large and continues to grow.

The major disadvantage of the standard ray-tracing algorithm is that the time needed to generate an image is equal to the number of geometric primitives *times* the size of the output image. This is because when an individual ray is being traced all the objects in the scene