Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Add links

Cone tracing

From Wikipedia, the free encyclopedia

Cone tracing[1] and beam tracing are a derivative of the ray tracing algorithm that replaces rays, which have no thickness, with thick rays.

Contents [hide]

Principles [edit]

This is done for two reasons:

From a physics of light transport point of view [edit]

The energy reaching the pixel comes from the whole solid angle by which the eyes see the pixel in the scene, not from its central sample. This yields the key notion of **pixel footprint** on surfaces or in the texture space, which is the back projection of the pixel on the scene.

The description above corresponds to the pinhole camera simplified optics classically used in computer graphics. Note that this approach can also represent a lens-based camera and thus depth of field effects, using a cone whose cross-section decreases from the lens size to zero at the focal plane, and then increases.

Moreover, a real optical system does not focus on exact points because of diffraction and imperfections. This can be modeled as a point spread function (PSF) weighted within a solid angle larger than the pixel.

From a signal processing point of view [edit]

Ray-tracing images suffer strong aliasing because the "projected geometric signal" has very high frequencies exceeding the Nyquist-Shannon maximal frequency that can be represented using the pixel sampling rate, so that the input signal has to be low-pass filtered - i.e., integrated over a solid angle around the pixel center.

Note that contrary to intuition, the filter should not be the pixel footprint since a box filter has poor spectral properties. Conversely, the ideal sinc function is not practical, having infinite support and possibly negative values. A Gaussian or a Lanczos filter are considered good compromises.

Computer graphics models [edit]

Cone and Beam early papers rely on different simplifications: the first considers a circular section and treats the intersection with various possible shapes. The second treats an accurate pyramidal beam through the pixel and along a complex path, but it only works for polyedrical shapes.

Cone tracing solves certain problems related to sampling and aliasing, which can plague conventional ray tracing. However, cone tracing creates a host of problems of its own. For example, just intersecting a cone with scene geometry leads to an enormous variety of possible results. For this reason, cone tracing has remained mostly unpopular. In recent years, increases in computer speed have made Monte Carlo algorithms like distributed ray tracing - i.e. stochastic explicit integration of the pixel - much more used than cone tracing because the results are exact provided enough samples are used. But the convergence is so slow that even in the context of off-line rendering a huge amount of time is required to avoid noise.

Differential cone-tracing, considering a differential angular neighborhood around a ray, avoids the complexity of exact geometry intersection but requires a LOD representation of the geometry and appearance of the objects. MIPmapping is an approximation of it limited to the integration of the surface texture within a cone footprint. Differential ray-tracing [2] extends it to textured surfaces viewed through complex paths of cones reflected or refracted by curved surfaces. The full differential cone-tracing - including geometry and appearance filtering - is mostly applicable as volumetric cone-tracing,[3] relying on 3D MIPmapping.

A recent SVO implementation by Crassin *et al.*[4] has generalized this approach to global illumination and adapted it to GPU, showing remarkable quality images at 25–70 frames per second.[5]

References [edit]

1. ^ Amanatides. "Ray Tracing with Cones" Siggraph'84 http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.129.582
2. ^ Homan Igehy. "Tracing Ray Differentials". http://www.graphics.stanford.edu/papers/trd/
3. ^ Fabrice Neyret. "Modeling Animating and Rendering Complex Scenes using Volumetric Textures". http://hal.inria.fr/inria-00537523
4. ^ Cyril Crassin, Fabrice Neyret, Miguel Sainz, Simon Green, Elmar Eisemann. "Interactive Indirect Illumination Using Voxel Cone Tracing". http://www.icare3d.org/research-cat/publications/interactive-indirect-illumination-using-voxel-cone-tracing.html
5. ^ https://www.youtube.com/watch?v=fAsg_xNzhcQ

Categories: Global illumination algorithms | Computer graphics