Estudio y Desarrollo de técnicas interactivas de Iluminación Global

Tesis doctoral presentada por:
Rubén J. García Hernández
para la obtención del grado de Doctor en Informática.

Director
Carlos Ureña Almagro
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Granada

19 de mayo de 2009

Editor: Editorial de la Universidad de Granada
Autor: Rubén J. García Hernández
D.L.: En trámite
ISBN: En trámite

Agradecimientos

Durante el desarrollo de mi tesis, me he apoyado en mucha gente. Quiero agradecer en primer lugar a mi director de tesis, Carlos Ureña, que ha seguido siempre mi investigación intentando llevarme por el buen camino. Además, durante estos años he trabajado estrechamente con Miguel Lastra, Rosana Montes y Jorge Revelles, desarrollando software colaborativamente e intentando publicar los resultados de nuestra investigación. También quiero saludar aquí al resto de miembros del grupo de gráficos, a los que he preguntado dudas en muchas ocasiones. Las versiones preliminares de mi tesis han sido leídas por compañeros del departamento, y me han dado muchos consejos útiles. En la parte de mi trabajo relacionada con el estudio teórico de algoritmos, me he apoyado en compañeros del departamento de matemática aplicada, a los que agradezco su interés.

Por otra parte, durante estos años mucha gente me ha preguntado cuándo leería la tesis, tanto compañeros del departamento, como amigos y familia. A todos ellos dedico este documento.

Este trabajo ha sido parcialmente financiado por los proyectos del Ministerio de Educación y Ciencia TIC2001-2392-C03-03, TIN2004-07672-C03-02 y TIN2007-68066-C04-01, y por la beca de Formación de Profesorado Universitario AP2001-3238.

Índice general

Índice de cuadros

Índice de figuras

Índice de algoritmos

Resumen

El ordenador se ha usado para generar imágenes prácticamente desde su invención. En un primer momento eran imágenes sencillas de conceptos matemáticos (gráficas de funciones, por ejemplo), juegos muy simples (el archiconocido pong), fractales (el copo de nieve de Koch), etc. Conforme las capacidades de los ordenadores crecían, se empezaron a generar imágenes más complejas (el fractal de Mandelbrot).

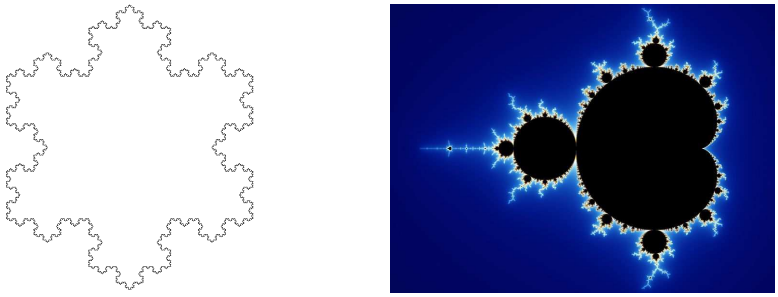


Figura 1: Copo de nieve de Koch (izquierda) y fractal de Mandelbrot (derecha).

Paralelamente, también se comenzó a usar el ordenador para diseño, modelando mundos virtuales y representándolos en la pantalla. Al principio se usaba una representación de alambre, que permitía intuir la forma de los objetos y se podía realizar con poco cómputo. Al pasar el tiempo, las imágenes han aumentado el realismo, pasando por alambres con eliminación de partes ocultas, relleno de color, modelos de iluminación simples (locales), raytracing (para escenas con muchas superficies transparentes o especulares) y finalmente modelos de Iluminación Globales (figura 2).

Los métodos que generan imágenes más realistas requieren mayor cómputo y por tanto más tiempo en una misma máquina. Los procedimientos (o algoritmos) de generación de imágenes se dividieron en interactivos (iluminación sencilla, tiempo de cómputo pequeño) y no interactivos (iluminación relativamente más compleja, tiempo de cómputo del orden de minutos u horas). Actualmente se puede hacer Iluminación Global con escenas pequeñas en tiempo real. Sin embargo, las escenas grandes requieren aún tiempo de cómputo del orden de

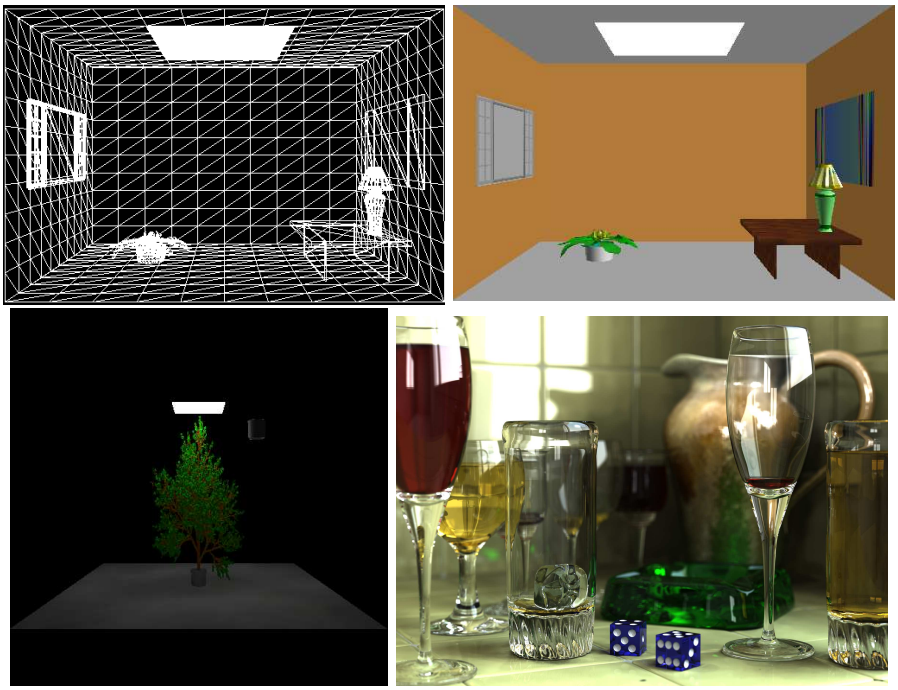


Figura 2: Alambre, Iluminación Local, Radiosidad, Raytracing

minutos (y no es extraño encontrar imágenes que requieren horas).

Históricamente, se comenzaron usando métodos de elementos finitos para las primeras imágenes de Iluminación Global. Estos métodos discretizan las ecuaciones integro-diferenciales que caracterizan la función de radiancia, y lo hacen mediante proyección en conjuntos de funciones base. A partir de esto, se puede calcular la Iluminación Global en toda la escena resolviendo las ecuaciones. Estas técnicas suelen utilizarse en entornos totalmente difusos, ya que aumentar la dimensión del problema para poder usar materiales que reflejen la luz de forma arbitraria (BRDFs generales) requiere un costo en tiempo y memoria enorme. Además presentan elevados tiempos de cálculo incluso en el caso de superficies difusas, aunque existen distintas variaciones sobre esta técnica que disminuyen los tiempos de cálculo.

Posteriormente, se aplicaron técnicas de Monte Carlo a Iluminación Global. Los métodos de Monte Carlo usan procesos estocásticos (cadenas de Markov) para obtener estimadores de los valores de la función de radiancia o funcionales de la misma. Estas técnicas no tienen la restricción de superficies difusas, pero presentan ruido. Además, el costo de generar las cadenas de Markov es bastante alto. Una optimización de estas técnicas es la estimación de densidades, que permite obtener imágenes con menos ruido calculando la densidad de energía en las cercanías de los puntos donde deseamos conocer la iluminación. La técnica

más famosa dentro de este enfoque es Photon Maps [Jensen 95, Jensen 96]. Sin embargo, Photon maps sigue teniendo los problemas de elevado tiempo de cálculo, errores grandes en superficies pequeñas, y dificultad para su uso en escenarios interactivos.

Es necesario por tanto diseñar nuevos algoritmos que permitan obtener Iluminación Global con unos tiempos de síntesis de imágenes suficientemente cortos como para permitir interactividad aún para escenas complejas. En esta memoria de tesis se describe el estudio, diseño e implementación de algoritmos para Iluminación Global interactiva en escenas de complejidad media-alta. Actualmente se considera que los basados en métodos estocásticos tienen mejor rendimiento que los de elementos finitos [Szirmay-Kalos 99, Veach 98]. Dentro de los métodos de Monte Carlo, una parte de ellos se basan en la emulación del historial de un conjunto amplio de fotones (idealizados como partículas puntuales), seguido de una estimación de densidades de impactos de estos en las superficies. La estimación de densidades puede hacerse de varias formas, las más elemental (y la primera descrita en Informática Gráfica) es la Cuenta de Impactos [Arvo 86]; otros, más avanzados, son la Estimación de Densidades [Walter 97], el anteriormente mencionado Photon Maps [Jensen 95, Jensen 96] y Estimación de Densidades en el Plano Tangente (DETP) [Lastra 02b]. El consenso es que los basados en derivados de Photon Maps son los que más rendimiento obtienen. Estos métodos están pensados para escenas estáticas y tienen unos tiempos de cálculo excesivamente altos al aplicarlos a interactividad.

Hemos observado que la mayoría de las escenas de complejidad alta, consisten en un escenario estático y un conjunto de objetos relativamente pequeños y dinámicos. Esto hace que la mayoría de la iluminación sea muy similar a la de fotogramas anteriores. Reutilizar la iluminación que sigue siendo válida y recalcular los fotones que hayan sido afectados por los objetos dinámicos es una posibilidad. En la bibliografía se encuentran algunos trabajos en esta línea, por ejemplo, Selective Photon Tracing de Dmitriev [Dmitriev 02], que se basa en Photon Maps, guarda la edad de los fotones y elimina los fotones más antiguos conforme tira nuevos fotones para actualizar la iluminación. Sin embargo, al no controlar si los fotones antiguos siguen siendo válidos, elimina información todavía válida.

Pretendemos obtener un algoritmo de cálculo interactivo de la iluminación resolviendo las limitaciones de las técnicas anteriores. Se estudiaron los métodos de Cuenta de Impactos, Photon Maps y DETP y se observó que el método de DETP era el que menor tiempo requería para un valor de error dado. Por ello se utilizó este algoritmo como base de nuestro trabajo. Utilizando el hecho de que la mayoría de la iluminación puede reutilizarse, se realizó un algoritmo que sigue el espíritu de Dmitriev, pero que sigue usando los fotones mientras sean válidos, por muy antiguos que sean. Además, al utilizar métodos de estimación de densidades más avanzados, mejoramos aun más la eficiencia.

Entre otras posibilidades, para incrementar el rendimiento se puede hacer uso de hardware avanzado. En esta linea hemos llevado a cabo implementaciones usando juegos de instrucciones SIMD y hardware gráfico de altas prestaciones. Aparte de esto, hemos estudiado de forma teórica la eficiencia del algoritmo de

DETP para encontrar sus limitaciones y diseñar nuevos algoritmos que obtienen tiempos aún menores, no solo para determinadas escenas, sino con mejora de eficiencia probada matemáticamente. Parte de este trabajo se ha publicado previamente en [García 03, García 04, García 05, García 06, García 07]. También se ha enviado un resumen de esta memoria a la revista Computer Graphics Forum, que se encuentra en proceso de revisión.

Como resultados concretos, podemos destacar lo siguiente:

- Se ha realizado un estudio teórico de la convergencia, el sesgo, la varianza y la complejidad de los algoritmos de Cuenta de Impactos, Photon Maps, DETP y Ray Maps.
- Hemos estudiado empíricamente el rendimiento de Cuenta de Impactos, Photon Maps y DETP. DETP tiene tiempos de cálculo muy inferiores al resto de técnicas para un error dado.
- También se ha estudiado teóricamente el rendimiento de DETP con gran detalle. Este estudio nos ha permitido calcular automáticamente los parámetros óptimos del algoritmo y su eficiencia. Los resultados del estudio teórico son la demostración de los siguientes puntos:
  - DETP básica y la Caché de Esferas son lineales con respecto al número de rayos y al número de puntos
  - La Caché de Esferas con ordenación de puntos, aunque igualmente lineal, tiene una constante oculta proporcional al cociente de los cuadrados del radio del disco y el radio de la esfera envolvente de la escena (la fracción de rayos en una esfera cuyo radio es el de los discos), que hace que el algoritmo sea bastante rápido en la práctica.
  - Para escenas pequeñas, este último algoritmo puede probarse que es proporcional al producto del número de rayos y la raíz cúbica del número de puntos.
- Por otra parte, se han diseñado dos técnicas para mejorar los tiempos:
  - La primera se basa en DETP y mejora los tiempos de cómputo, especialmente en el caso de escenas con un número de rayos no muy grande. Esta técnica (Indexación de Discos) indexa los discos asociados a los puntos de irradiancia y suma las contribuciones de cada rayo con todos los discos con los que interseca. Hemos obtenido resultados de incrementos de eficiencia respecto de otros métodos propuestos en la literatura de hasta el 50 %.
  - La segunda es una técnica de cálculo incremental de la iluminación que aprovecha el hecho de que la mayoría de escenas de complejidad alta tienen una gran parte estática durante la mayor parte de la animación. La técnica utiliza la Caché de Esferas para la parte móvil e Indexación de Discos para la escena estática. La mejora de los tiempos es de un orden de magnitud para móviles de un tamaño del 10 % de la escena.

Usando estas técnicas, podemos obtener interactividad para escenas de complejidad media.

- Hemos estudiado teóricamente la eficiencia en tiempo de las dos técnicas que hemos desarrollado.
  - Indexación de Discos es lineal en el número de rayos y con tiempo en el orden $O(n_P^{1/3} \log n_P)$ para árboles no balanceados y $O(n_P^{1/3})$para balanceados, siendo n_P el número de puntos.
  - En cuanto al cálculo incremental, obtenemos una aceleración proporcional al cuadrado del cociente de radios de la esfera englobante del móvil y la escena estática. También se ha demostrado que es óptimo usar la Caché de Esferas y la Indexación de Discos para el móvil y la parte estática respectivamente.

Esta memoria de tesis doctoral está estructurada siguiendo los siguientes capítulos:

- El capítulo uno comienza repasando los métodos de indexación espacial, base de los algoritmos gráficos eficientes. Tras ello se realiza una introducción a técnicas para iluminación global, tanto *offline* como interactiva, mostrando sus ventajas y limitaciones. Se comenta con detalle la Estimación de Densidades en el Plano Tangente, punto de origen de esta investigación. Finalmente se introduce el trabajo relacionado con el estudio teórico de los algoritmos junto con las bases matemáticas necesarias.
- El capítulo dos explica la mejora de DETP para aumentar la eficiencia (Indexación de Discos), que mejora sustancialmente los tiempos de cálculo, y la compara con los algoritmos anteriores.
- El capítulo tres comienza estudiando de forma teórica el sesgo, la varianza y la eficiencia de distintas técnicas de estimación de densidades para iluminación global. Tras ello se estudia la eficiencia de DETP y sus mejoras, incluyendo la explicada en el capítulo dos, y obtiene el orden de eficiencia de cada técnica y los valores óptimos de los parámetros.
- El capítulo cuatro explica el algoritmo de cálculo incremental de la iluminación, que obtiene grandes mejoras de tiempo para escenas quasi-estáticas.
- El capítulo cinco describe el software desarrollado.
- Finalmente el capítulo seis describe los resultados obtenidos en la tesis y el trabajo futuro.

Capítulo 1

Introducción y trabajo previo

1.1. Objetivos

En esta memoria de tesis se describe el estudio, diseño e implementación de algoritmos interactivos para Iluminación Global para escenas complejas. Las técnicas de Monte Carlo (y en particular las técnicas de estimación de densidades basadas en Photon Maps) son las que mejor rendimiento obtienen para el cálculo de Iluminación Global. Hemos realizado un estudio de la aplicabilidad de estas técnicas a interactividad. Para ello, se comenzó estudiando qué cambios son necesarios en los algoritmos para proporcionar un recálculo incremental de la iluminación, y se compararon los resultados de Estimación de Densidades en el Plano Tangente (en lo sucesivo DETP) con los algoritmos de Cuenta de Impactos y Photon Maps. Posteriormente, se diseñaron nuevas optimizaciones de DETP para disminuir los tiempos de cálculo.

Los algoritmos basados en DETP han sido estudiados de forma teórica para poder elegir en cada caso el algoritmo más eficiente. El cálculo incremental de iluminación también ha sido estudiado teóricamente.

En este capítulo se explica primero qué es la Iluminación Global y las distintas formas de calcularla: métodos de elementos finitos y métodos de Monte Carlo. Posteriormente, se describe el estado del arte en algoritmos de cálculo de Iluminación Global basados en Monte Carlo: Cuenta de Impactos [Arvo 86], Estimación de Densidades [Walter 97], Photon Maps [Jensen 95], Estimación de Densidades en el Plano Tangente [Lastra 02b] y Ray Maps [Havran 04]. Tras ello se comentan algunos algoritmos interactivos para Iluminación Global: Selective Photon Tracing [Dmitriev 02] y los algoritmos paralelos optimizados de Wald [Wald 02b]. La sección termina comentando el algoritmo de reuso de caminos de Sbert para animación de la fuente de luz [Sbert 04].

A continuación se describen técnicas de indexación espacial, que son fundamentales en la implementación eficiente de algoritmos que requieran una gran cantidad de intersecciones rayo-objeto, como es el caso de la Iluminación Global usando Monte Carlo.

La sección 1.7 describe técnicas de estudio teórico de algoritmos para indexación espacial, Ray Tracing e Iluminación Global. Finalmente se describen técnicas matemáticas y de teoría de la información necesarias para el desarrollo de los algoritmos presentados: teoría de probabilidad, fundamentos de Geometría Integral y el teorema de muestreo de Nyquist-Shannon.

1.2. Iluminación Global

Supongamos una escena, definida mediante una geometría. Si tenemos para cada punto de la geometría una función que describa cómo interacciona la luz con ese material, podemos simular el transporte de la luz para obtener cálculos realistas de la iluminación en esa escena.

La luz se genera en las fuentes de luz, y atraviesa la escena. Parte de la luz se refleja en los objetos e ilumina de forma indirecta otros objetos. Tras una serie de rebotes, los fotones se absorben por algún objeto de la escena o salen de la escena.

Los algoritmos que simulan el transporte de la luz de forma físicamente realista se llaman algoritmos de Iluminación Global, y formalmente consisten en calcular numéricamente valores aproximados de la función de radiancia L, que es solución de la ecuación funcional

$$L = L_e + \mathcal{T}L \tag{1.1}$$

donde L_e es la radiancia emitida y $\mathcal{T}$ es el operador integral que modela el transporte de la luz [Arvo 95a, Szirmay-Kalos 99].

Existen dos técnicas principales para resolver esta ecuación, los métodos de elementos finitos y los métodos de Monte Carlo, explicados en las siguientes secciones.

1.3. Métodos de elementos finitos

Los métodos de elementos finitos discretizan la ecuación 1.1 mediante proyección sobre un conjunto de funciones base. Una forma bastante utilizada es descomponer la escena en zonas llamadas parches, y calcular una aproximación a la ecuación de transporte de la luz suponiendo que la densidad de energía radiante incidente y reflejada es constante en cada parche.

Formalmente usan un método de resolución de ecuaciones iterativo, como Gauss-Seidel o Southwell [Cohen 93, Ureña 97], para calcular la irradiancia. Hay una enorme variedad de enfoques y variaciones sobre estos métodos. El cuadro 1.1 contiene una selección.

1.3.1. Métodos interactivos

Dentro de los métodos de elementos finitos existen algunos algoritmos que permiten actualizar la iluminación de forma eficiente cuando hay cambios en la

Año	Método	Referencia
1986	Radiosity Method for Non-diffuse Environments	[Immel 86]
1987	Synthesis of Raytracing and Radiosity	[Wallace 87]
1988	Procedural Refinement	[Shao 88]
1988	Shading Method for Computer Generated Images	[Malley 88]
1989	Illumination Networks	[Buckalew 89]
1989	Two-pass method for Specular and Diffuse Reflection	[Sillion 89]
1991	A global illumination solution for general BRDFs	[Sillion 91]
1991	Hierarchical Radiosity	[Hanrahan 91]
1992	Importance-driven Radiosity	[Smits 92]
1993	Importance and Discrete Three Point Transport	[Aupperle 93]
1993	Integral Geometry for Fast Form Factors	[Sbert 93]
1994	Progressive Refinement	[Gortler 94]
1994	Haar Wavelet	[Pattanaik 94]
1994	Wavelet Radiance	[Christensen 94]
1994	Wavelet Methods for Radiance Computations	[Schröder 94]
1995	Multiresolution B-spline Radiosity	[Yu 95]
1995	Hemicube	[Cohen 85]
1997	Update of Illumination Using a Line-space Hierarchy	[Drettakis 97]
1999	Group Accelerated Shooting Methods	[Rouselle 99]
2001	Incremental Rapid Glossy Global Illumination	[Granier 01]

Cuadro 1.1: Iluminación global usando elementos finitos.

geometría. Esto permite diseñar algoritmos interactivos, en los que el usuario puede modificar la escena y obtener una iluminación recalculada. Conforme el algoritmo de resolución del sistema de ecuaciones de la radiancia converge a la solución correcta, la imagen se aproxima a la imagen final correcta.

Drettakis y Sillion [Drettakis 97] se basaron en un método de radiosidad jerárquica para poder recalcular la iluminación de forma interactiva al mover la geometría. Usan enlaces entre parches con el conjunto de todas las líneas que intersecan ambos parches y utilizan el espacio de las líneas para intersecar los objetos móviles con estas líneas de forma eficiente y averiguar qué parches deben actualizarse. Posteriormente Granier y otros [Granier 01] añadieron un método híbrido basado en técnicas de Monte Carlo para las superficies no perfectamente difusas.

Rouselle et al [Rouselle 99] usan el hecho de que las superficies cercanas tienen factores de forma relativamente grandes. Esto permite el uso de métodos de agrupación iterativa que aumentan la velocidad de convergencia. Al usar además ideas de Refinamiento Progresivo, obtienen un algoritmo interactivo.

1.4. Técnicas de Monte Carlo

Las técnicas de Monte Carlo son métodos de integración de funciones que se basan en métodos estocásticos diseñando una variable aleatoria que sigue una

distribución de probabilidad cuya media es el resultado de la integral. Después, se muestrea la variable aleatoria un número determinado de veces para obtener una aproximación de la integral con determinado margen de confianza del error porcentual cometido. El libro de Rubinstein [Rubinstein 81] explica cómo usar estas técnicas en general.

Año	**Técnica**	**Referencia**
1984	Distributed Ray Tracing	[Cook 84]
1986	Path Tracing	[Kajiya 86]
1986	Cuenta de Impactos	[Arvo 86]
1988	Irradiance Cache	[Ward 88]
1990	Adaptative Radiosity	[Heckbert 90]
1991	Hierarchical Radiosity	[Hanrahan 91]
1992	Light Path Tracing	[Pattanaik 92]
1993	Potential Equation for Global Illumination	[Pattanaik 93]
1994	Bidirectional Estimators for Light Transport	[Veach 94]
1994	Bidirectional Path Tracing	[Lafortune 94]
1994	Importance Driven Monte Carlo	[Dutré 94]
1995	Bidirectional Path Tracing	[Veach 95]
1995	Hierarchical Radiosity with Clustering	[Sillion 95]
1995	Particle Tracing	[Shirley 95]
1996	Photon Maps	[Jensen 96]
1997	Metropolis Light Transport	[Veach 97]
1997	Light Maps	[Tobler 97]
1997	Estimación de Densidades	[Walter 97]
1998	Irradiance Volume	[Greger 98]
2001	Rapid Glossy Global Illumination	[Granier 01]
2002	Estimación de Densidades en el Plano Tangente	[Lastra 02b]
2002	Selective Photon Tracing	[Dmitriev 02]
2002	Advanced Radiance Estimation for Photon Maps	[Hey 02]
2002	Iluminación Global Interactiva	[Wald 02b]
2004	Ray Maps	[Havran 04]

Cuadro 1.2: Iluminación global usando Monte Carlo.

En el caso de Iluminación Global, utilizamos estas técnicas para calcular una aproximación a la integral que define la solución a la ecuación de transporte de la luz (ecuación 1.1 de la sección 1.2). Existen dos formas principales de resolver la integral: simulando el transporte físico de la luz, ya sea desde las fuentes de luz o desde el observador (líneas locales) y usando el concepto de líneas globales. El cuadro 1.2 contiene una selección de técnicas de líneas locales.

Cuando se usan algoritmos de líneas globales, se generan aleatoriamente líneas siguiendo una distribución uniforme en la esfera envolvente de la escena. Estas líneas se intersecan con la escena y se usan para transferir energía de forma bidireccional entre los distintos parches de la escena. Cada línea equivaler a varias líneas locales si la oclusión de la escena es grande. Ejemplos de estos

métodos fueron propuestos por Buckalew et al [Buckalew 89] y Neumann et al [Neumann 95].

Tal y como propusieron Sbert y otros [Sbert 96b], si se realiza una primera pasada con líneas locales para realizar una distribución inicial de la energía se evita desperdiciar trabajo al intersecar las líneas globales con parches que no hayan recibido energía aún. [Sbert 96a, Martínez 09] usan esta idea.

Las siguientes cinco secciones explican diferentes métodos avanzados de resolver la ecuación de transporte de la luz usando líneas locales de forma eficiente en tiempo de cálculo y con un error lo más pequeño posible.

1.4.1. Cuenta de Impactos

El método más básico de estimación de densidades es la Cuenta de Impactos, desarrollada por Arvo [Arvo 86], y extendida por Heckbert [Heckbert 90] y Pattanaik [Pattanaik 92]. Se basa en el modelo de partículas de la luz. Los fotones se emiten desde las fuentes de luz e interactúan con las superficies hasta que son absorbidos o dejan la escena. Los puntos de interacción (o reflexión) entre los fotones y las superficies, así como la dirección de incidencia se guardan para poder reconstruir la radiosidad en las superficies. Para obtener la radiosidad en una superficie, se suma la energía de los impactos en las caras. Luego, para cada vértice, se calcula un valor promedio teniendo en cuenta las caras a las que pertenece y se interpola trilinealmente en el triángulo.

1.4.2. Estimación de Densidades

Un método útil para estimar la integral de la radiancia es el método de Estimación de Densidades, popularizado por [Walter 97]. Tiene tres fases. La primera fase simula los fotones desde las fuentes de luz. Cuando un fotón interacciona con una superficie, se almacena la posición y la energía del fotón. Tras el proceso de todos los fotones, el resultado de esta fase es una estructura de datos con la posición y energía de todos los fotones.

La segunda (estimación de densidades propiamente) estima la radiancia. El método suma la energía de los fotones cercanos ponderados con una función que depende de la distancia, de modo que cuanto más lejos esté el fotón menor sea el peso (esta función es el núcleo de la estimación de densidades). La distancia a la que se buscan fotones se llama el ancho de banda del núcleo. El problema es que como sólo hay impactos en las superficies, cerca de los bordes se subestima la radiancia. Esto se soluciona disminuyendo el ancho de banda cerca de los bordes, resolviendo un problema de regresión.

La tercera fase simplifica la geometría tras el cálculo de iluminación. Esta última fase a menudo se obvia porque se considera fuera del ámbito de estimación de densidades.

1.4.3. Photon Maps

Jensen [Jensen 95, Jensen 96] diseñó el conocido método de Photon Maps. Consiste en encontrar los k fotones más cercanos al punto donde se estima la irradiancia, sumar su energía y dividir por el área del círculo máximo de la esfera que contiene los k fotones (k está predefinido).

Existen dos modos de usar este algoritmo. El primero se denomina visualización directa del Photon Map. Consiste en realizar un raytracing desde el observador, y para cada punto obtenido, hacer una consulta al Photon Map. El segundo modo se denomina *final gathering* y consiste en reflejar siguiendo la BRDF el rayo en cada punto, obteniéndose un número de rayos secundarios salientes (normalmente 50) en cada punto. La consulta al Photon Map se realiza en las intersecciones de los rayos salientes con la escena, y la radiancia se calcula a partir de la contribución de cada rayo [Jensen 01]. Se obtiene un gran incremento en la calidad de la imagen generada, aunque el coste es mucho mayor.

La limitación más conocida de Photon Maps es que cuando se calcula la irradiancia en un punto, los fotones cercanos deberían estar en el mismo plano y en un área circular alrededor del punto. Hey y Purgathofer [Hey 02] presentan un algoritmo que resuelve esta limitación usando información geométrica en las cercanías del punto.

Otra limitación menos conocida de Photon Maps y el algoritmo propuesto por Hey y Purgathofer mencionado en el párrafo anterior es que si hay superficies en la escena relativamente muy pequeñas, estas zonas tienen comparativamente una alta varianza, y tienden a aparecer muy brillantes o muy oscuras, si el número de fotones no es lo suficientemente alto. Véase la figura 1.1 (izquierda).

1.4.4. Estimación de Densidades en el Plano Tangente

Un método que evita la alta varianza de Photon Maps mencionada en la sección anterior, consiste en guardar los rayos y usar un disco de tamaño fijo centrado en el punto donde se calcula la irradiancia y que está contenido en el plano tangente a la superficie. Los rayos que intersequen un disco dado se usan para calcular la irradiancia en el punto central [Lastra 02c]. El algoritmo

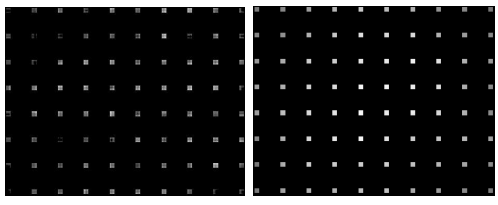


Figura 1.1: Photon Maps (izquierda) y Estimación de Densidades en el Plano Tangente (derecha).

se llama Estimación de Densidades en el Plano Tangente (DETP). Nótese que este algoritmo guarda la trayectoria de los fotones (origen, dirección y distancia al impacto), mientras que Photon Maps guarda sólo el punto de impacto y la dirección. Véase la figura 1.2. Como se pueden aprovechar mejor los rayos, es factible hacer visualización directa y tener una calidad aceptable.

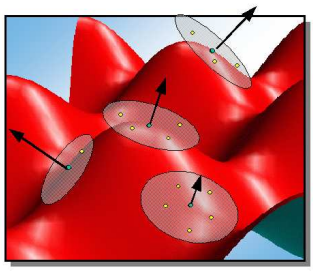


Figura 1.2: Estimación de Densidades en el Plano Tangente.

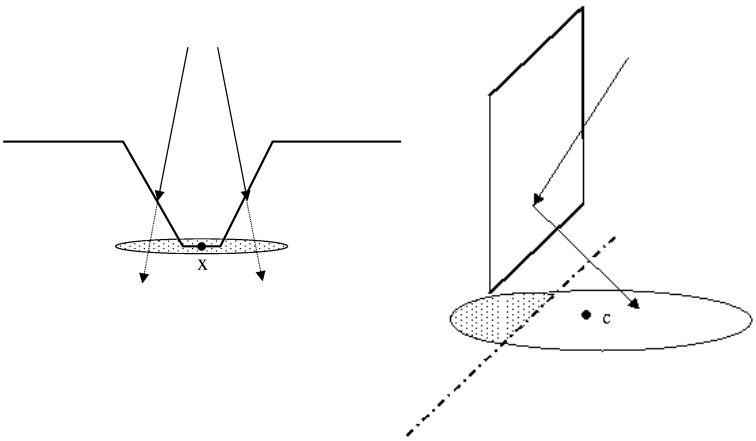


Figura 1.3: Artefactos de DETP.

Para evitar sombras propias en superficies cóncavas, existe una modificación denominada *artifact control*. Esta modificación disminuye el sesgo a costa de aumentar un poco el tiempo de cálculo, y evita tener en cuenta las zonas del disco a las que los rayos no pueden acceder (figura 1.3). Hay que tener en cuenta dos cosas:

- Un rayo contribuye a un disco si la intersección está estrictamente antes de la segunda intersección con la escena (figura 1.3, izquierda).

- Si una porción del disco no es visible desde el origen del rayo, la contribución del rayo al disco debe dividirse por el área de la zona visible en lugar del área completa (figura 1.3, derecha). Esto es debido a que la parte no visible del disco no contribuye a la medida del ángulo sólido desde c, y esto provoca un sesgo inherente a cualquiera de las técnicas de estimación de densidades en su formulación habitual.

Este método usa discos de radio fijo [Lastra 02c]. Sin embargo, el algoritmo compensa de forma óptima entre exactitud y varianza cuando el radio del disco es del orden de magnitud de la mitad de la distancia entre muestras de irradiancia. Si el radio del disco fuera más pequeño, los rayos que intersecan el plano tangente cerca del punto medio entre dos muestras de irradiancia se ignorarían. Si fuera más grande, las intersecciones se usarían para varios cálculos, creando un suavizado artificial.

Este método presenta un sesgo debido a una limitación inherente derivada del hecho de que se aproxima la radiancia en un entorno del punto de interés. Este problema se llama en la literatura *light leaks* (fugas de luz: zonas que deberían ser oscuras que aparecen iluminadas debido a la presencia de rayos cercanos) y aparece cuando el tamaño del disco es mayor que la anchura de los objetos que arrojan sombra. Tobler et al [Tobler 06] propone usar *geometry feelers* (tirar rayos desde el punto donde se calcula la irradiancia en varias direcciones buscando geometría que tape la luz, u *octoboxes* (octogonos irregulares sobre el disco que se ajustan a la geometría de la escena). Otra solución es disminuir el tamaño del disco, y aumentar el número de rayos en la escena para mantener el error constante. Para las fugas de luz que se producen en los bordes de sombras de objetos lejanos (como bajo una mesa), los *geometry feelers* y *octoboxes* no son suficientes y hay que usar el cambio del tamaño del disco.

Caché de Esferas

El método de DETP en su forma básica tiene el problema de que el cálculo de las intersecciones rayo-disco es más complejo en tiempo que la búsqueda de fotones puntuales de Photon Maps, ya que las trayectorias son objetos extensos.

Se puede disminuir el número de las intersecciones rayo-disco necesarias descartando los rayos que estén lejos del punto de interés, con lo que el tiempo de cómputo disminuye. La Caché de Esferas [Lastra 02c] usa esta idea. El método consiste en crear una jerarquía de esferas de radio decreciente y guardar los rayos que intersecan cada esfera para disminuir el número de tests de intersección rayo-disco.

Primero, se crea una esfera tangente a la caja englobante de la esfera. Esta esfera interseca todos los rayos. Después, como se ve en la figura 1.4, se crean esferas de radio decreciente unas dentro de otras (la razón entre el radio de dos esferas consecutivas es un parámetro llamado Q), hasta que el radio está justo por encima del radio del disco o hasta que la esfera tiene menos de un número prefijado de rayos (el límite de subdivisión).

Cada esfera tiene una estructura de datos asociada que contiene los rayos

que interseca dicha esfera. Estos rayos se calculan intersecando la esfera con los rayos de la esfera inmediatamente superior.

El primer punto en el que se ha de calcular la irradiancia se usa como el centro de las esferas. Por tanto, el primer disco está en la esfera interna. La irradiancia se puede calcular comprobando qué rayos de la esfera interna intersecan el disco, y sumando su energía. El número de intersecciones rayo-disco se reduce claramente.

Para el resto de puntos, si el disco centrado en el punto está dentro de la esfera interna, el disco se interseca con los rayos de esta esfera. En caso contrario tenemos un fallo de caché: se descarta la esfera y se comprueba el resto de esferas hasta que una contenga el disco. En ese momento se recalcula la jerarquía de esferas, usando el punto como centro (véase la figura 1.4 derecha). Finalmente,

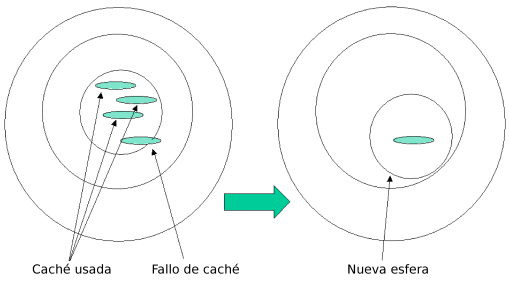


Figura 1.4: Caché de Esferas.

el disco se interseca con los rayos de la esfera interior, del mismo modo que si no se necesita recalcular ninguna esfera.

Ordenación de puntos

El rendimiento de la Caché de Esferas depende de la coherencia espacial de los discos, ya que esta coherencia permite reusar las esferas. [Lastra 02c] contiene un algoritmo que reordena los puntos para incrementar la coherencia espacial. Este algoritmo se basa en la curva de Lebesgue, descrita en [Sagan 94].

El algoritmo transforma las coordenadas de los puntos en tres enteros sin signo de 16 bits, de modo que la caja englobante de la escena va de 0 al máximo valor representable en cada coordenada. Los puntos se ordenan con la siguiente función de comparación: Para dos puntos A y B, se comprueba el bit más significativo (MSB) de la coordenada X de A con la de B, después el MSB en Y y el Z. Después se comprueban los siguientes bits, en orden. En cuanto uno de los bits sea menor, el punto se considera menor. Un ejemplo de la ordenación puede verse en la figura 1.5.

La eficiencia de la ordenación de puntos depende de la coherencia espacial de la curva que rellena el espacio. Para evitar los saltos bruscos que aparecen en los cambios de bits de la ordenación de Lebesgue, otra posibilidad es usar una

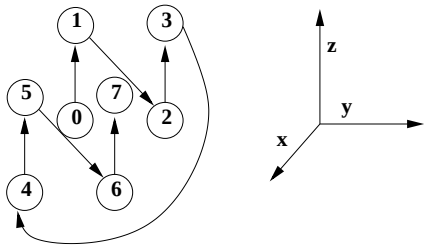


Figura 1.5: Ordenación de puntos de Lebesgue.

ordenación basada en la curva de Hilbert [Alber 03]. La ordenación puede verse en la figura 1.6, y proporciona una ligera mejora del rendimiento, como se ve en el cuadro 1.3.

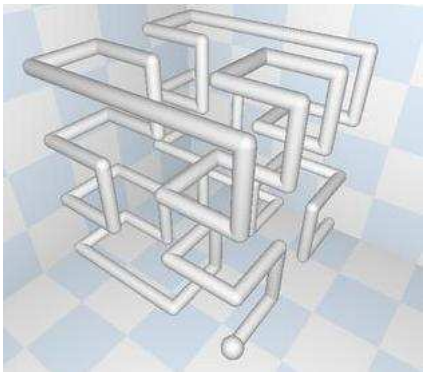


Figura 1.6: Ordenación de puntos de Hilbert.

Radio/ N° de rayos	**1 000 000**	**100 000**	**10 000**	**1 000**
0,05 %	13,3 %	33,7 %	69,8 %	78,3 %
0,5 %	10,1 %	28,8 %	66,6 %	78,0 %
5,0 %	0,5 %	2,4 %	27,7 %	64,7 %

Cuadro 1.3: Aceleración del cálculo usando la curva de Hilbert.

Caché de Esferas Multilista

Cuando no es factible usar la ordenación de puntos (por ejemplo en algunos raytracers interactivos como [Tawara 04] o [Dmitriev 02], en los que sólo se actualizan unas cuantas muestras por fotograma debido a limitaciones de tiempo), la caché de esferas tiene un rendimiento bajo.

En esta situación, se puede permitir a cada esfera tener más de un descendiente (conservando las esferas creadas tras un fallo de cache). Esto hace que la coherencia sea menos importante, ya que las esferas pueden usarse para calcular eficientemente la estimación de radiancia de puntos contenidos en ellas aunque éstos no se procesen secuencialmente.

El resultado es una estructura de datos en árbol que denominamos Caché de Esferas Multilista. Para controlar el tamaño de esta estructura, podemos o bien controlar el número máximo de hijos de cada esfera, o bien controlar el número máximo de esferas existentes, y eliminar las esferas antiguas con una cola FIFO.

1.4.5. Ray Maps

Havran [Havran 05a] describe otra forma de utilizar todo el recorrido de los rayos para disminuir la varianza. Organiza los rayos en un kd-tree construido bajo demanda que permite hacer búsquedas de proximidad de rayos según diferentes criterios:

- Intersección con un dominio: disco, hemisferio, esfera o caja alineada a los ejes.
- Búsqueda del vecino más cercano: distancia entre la intersección del rayo con el plano tangente, distancia al rayo, distancia a la línea soporte del rayo

El recorrido del kd-tree está optimizado para el caso de búsquedas coherentes, y se controla el uso de memoria usando una estrategia LRU para colapsar y expandir el árbol. Los rayos se organizan en función de su dirección para optimizar las búsquedas.

1.5. Iluminación Global Interactiva

En esta sección comentaremos algunos trabajos enfocados a obtener Iluminación Global con tiempos de cómputo pequeños que permitan interactividad. Los trabajos de Dmitriev buscan reusar información de la iluminación para disminuir los tiempos de cálculo, mientras que los de Wald usan clusters de computadores para disminuir los tiempos de cálculo usando paralelismo.

1.5.1. Selective Photon Tracing

Dmitriev, en [Dmitriev 02] propone un método que permite recalcular la iluminación y la radiosidad a lo largo de un número de fotogramas, y también encontrar rápido los fotones que deben ser recalculados. El método se basa en la secuencia de Halton multidimensional, que debido a su quasi-periodicidad, permite encontrar fácilmente los fotones que tienen un camino similar al que se está recalculando. La mayoría de estos fotones también tendrán que ser recalculados. Usa *raytracing* desde las fuentes de luz, así que su solución es independiente de la vista.

Los requisitos hardware para su método son un ordenador de gama media. Explica el procedimiento de actualización de la iluminación, en el que sólo usa los rayos recalculados. Se usa hardware gráfico para calcular las sombras, y tras ello sólo se recalcula la iluminación indirecta, que se añade a la imagen resultante.

La principal ventaja de este método es que permite obtener iluminación global interactiva con un gasto de memoria muy bajo, ya que se puede evitar almacenar los fotones usando la secuencia de Halton para recrearlos en caso necesario. Esto permite que el método sea útil para escenas mucho más complejas.

1.5.2. Implementaciones optimizadas paralelas de Photon Maps

Wald et al, en [Wald 02b, Wald 02a], explican un método que aplica técnicas de programación distribuida a Ray Tracing. Usa un cluster de PCs para obtener Iluminación Global en tiempo real.

Las mejoras incluyen ensamblador generado a mano, optimización del layout y alineamiento de las estructuras de datos, prefetching, manejo de la cache e instrucciones SIMD para seguir varios rayos (lo que resulta en una gran mejora del rendimiento de la cache) y usar el buffer de discontinuidad y el muestreo entrelazado [Keller 01] para disminuir el ruido.

Una extensión a Photon Maps para cáusticas, consistente en meter en una tabla hash los impactos de los fotones, se usa para poder usar superficies transparentes. Las reflexiones especulares se calculan en un paso de *raytracing*. La integración de Quasi Monte Carlo se usa en su sistema para acelerar la convergencia.

Benthin [Benthin 03] describe los cambios en el sistema para obtener una escalabilidad y rendimiento mejorados. [Günther 04] y [Wald 04b] describen algunas optimizaciones más.

El sistema permite obtener iluminación realista en tiempo real para escenas complejas, ya que aprovecha la gran capacidad de cómputo y el paralelismo del cluster.

Limitaciones de los algoritmos

El algoritmo que proponen Wald et al [Wald 02b] se basa en un sistema de raytracing en tiempo real, pero tiene algunas limitaciones:

- No puede mapearse a una arquitectura de un PC estándar aislado, que es la plataforma más común.
- Aunque Wald obtiene tasas de refresco interactivas mientras los objetos se mueven, la iluminación correcta tarda dos o tres segundos en computarse una vez que los objetos dejan de moverse.

El problema de este esquema es que las sombras se calculan con la información de los fotogramas anteriores, y la posición de los objetos ha cambiado. Esto

hace que la posición de las sombras vaya detrás de su posición correcta. Un par de segundos después de que el objeto se haya detenido, las sombras llegan a su posición correcta. El problema es análogo al que se presenta en los métodos interactivos de elementos finitos mencionados en la sección 1.3.1. La divergencia viene del hecho de que Wald usa su algoritmo para la visualización previa de Iluminación Global en las escenas, así que esperar unos segundos para que la iluminación se estabilice no es problema. Por otra parte, nuestro principal objetivo es obtener una aproximación realista a la Iluminación Global en todos los fotogramas, lo que es necesario por ejemplo en los juegos 3D.

Su nuevo sistema oculta el problema para escenas de complejidad media porque aumenta en gran medida la escalabilidad y usa un gran cluster. Sin embargo pensamos que el uso del sistema de tiempo real de Wald et al mejoraría todos los métodos basados en trazado de fotones que presenten coherencia en los rayos, como es el método presentado en el capítulo 4.

1.5.3. Reuso de caminos

El algoritmo de reuso de caminos fue propuesto por Sbert et al. en Real-time Light Animation [Sbert 04]. Dada una escena estática y una fuente de luz dinámica, se pueden reutilizar los caminos de las reflexiones de los fotones en distintos fotogramas uniéndolos con la posición nueva de la fuente de luz, y aplicando pesos para evitar un algoritmo sesgado. Es necesario lanzar por lo menos un camino en cada fotograma para que el algoritmo sea no sesgado, y la combinación de los caminos puede realizarse eficientemente en la GPU. También es necesario eliminar la contribución de la fuente de luz al camino si ésta está oculta por algún objeto (el peso en este caso sería cero).

Los pesos de los caminos se recalculan usando Multiple Importance Sampling [Veach 98]. La aceleración es como máximo el número de fotogramas de la animación. Este método también puede aplicarse en el caso interactivo utilizando sólo los caminos creados en el pasado, y la velocidad del algoritmo es buena incluso en este caso.

1.6. Métodos de Indexación espacial

En informática gráfica, los métodos de indexación espacial son estructuras de datos que dividen de forma jerárquica el espacio en zonas, y almacenan las primitivas geométricas en función de su posición, permitiendo calcular fácil y rápidamente intersección entre una primitiva geométrica (típicamente un polígono, segmento o semirrecta) y una geometría compleja, detección de colisiones, etc. Si se posee una geometría compleja, estos métodos son indispensables para obtener algoritmos eficientes. Estos métodos constituyen una parte muy importante de los algoritmos eficientes de iluminación global mencionados en la sección anterior.

Cuando se crea una indexación espacial, es necesario tener en cuenta varias cosas:

- Método de construcción y coste asociado.
- Calidad de la indexación generada (si está balanceada, si las primitivas se han repartido por los nodos uniformemente, etc).
- Método de recorrido y coste asociado.

En el campo de Iluminación Global con algoritmos estocásticos, nos interesa especialmente que la intersección rayo-objeto sea lo más eficiente posible, ya que es la parte más costosa. El cuadro 1.4 contiene una lista con los algoritmos más conocidos. Las siguientes secciones contienen una descripción de los algoritmos que se han implementado durante el desarrollo de esta tesis, o que por sus características se integrarían bien con los algoritmos desarrollados (la sección 6.2 tiene más detalles). Usaremos métodos avanzados de indexación espacial para mejorar la eficiencia del algoritmo de Estimación de Densidades en el Plano Tangente en el Capítulo 2.

Año	**Método**	**Referencia**
1975	Binary Space Partitioning Tree	[Bentley 75]
1980	Jerarquía de volúmenes envolventes	[Rubin 80]
1984	Octree	[Glassner 84]
1986	Grid Rectangular	[Fujimoto 86, Amanatides 87]
1987	Kd-tree	[Subramanian 87, Fussel 88]
1989	Grid Recursivo	[Jevans 89]
1992	Rectilinear Binary Spatial Partitioning	[Sung 92]
1995	Jerarquía de Grids Uniformes	[Cazals 95]
1995	Octree-R	[Whang 95]
1997	Grid Adaptativo	[Klimansezewski 97]
1999	Grid Regular	[Havran 99]
2001	Coherent Ray Tracing	[Wald 01]
2004	Packet Ray Tracing	[Wald 04a]
2007	Recorrido Vectorizado	[Noguera 07, Noguera 09]
2007	Ray Stream Tracing	[Wald 07]

Cuadro 1.4: Métodos de Indexación Espacial.

1.6.1. Métodos clásicos

Existen varios métodos de indexación espacial muy usados y estudiados. Un Grid consiste en la división de una caja envolvente de forma uniforme en las coordenadas X, Y y Z, obteniéndose $x \times y \times z$ celdas. Se puede encontrar la caja que contiene un punto del espacio de forma inmediata, pero al no adaptarse a la escena puede desperdiciar mucho espacio en memoria.

Un Octree consiste en la subdivisión jerárquica de la caja envolvente; en cada nivel se divide el voxel en ocho partes iguales que resultan de colocar 3 planos en las coordenadas X, Y y Z que pasan por el centro, si el voxel está lo suficientemente ocupado. Hay varias formas de recorrerlo para averiguar en qué celda se encuentra un punto, o qué celdas atraviesa un línea, con diferentes parámetros de eficiencia en tiempo y memoria. Especialmente interesante es el método paramétrico de [Revelles 00] que permite recorrer un Octree con un rayo de forma muy eficiente. Primero se transforma el rayo para que sus componentes sean positivas, y tras ello se calculan los puntos de entrada y salida del octree en los planos X, Y y Z. Con esta información, se obtiene en un bucle el recorrido correcto del octree.

Un Rectilinear Binary Space Partitioning Tree (RBSPTree), también llamado kd-tree, divide de forma jerárquica el espacio, cada vez en dos vóxeles usando planos de corte en X, Y ó Z alternativamente, también siguiendo un criterio de ocupación.

1.6.2. Métodos de recorrido

El coste de intersecar las primitivas con una indexación espacial muchas veces depende del orden en que se recorren los nodos de la indexación. Un método de recorrido de una indexación espacial es un algoritmo que indica qué nodos y en qué orden se deben visitar para cada primitiva geométrica a intersecar.

Existen algoritmos que permiten usar técnicas SIMD para aumentar la eficiencia del recorrido. Coherent Ray Tracing [Wald 01] es un método que permite intersecar cuatro rayos a la vez con una indexación espacial. Se basa en usar instrucciones vectoriales para procesar en paralelo los cuatro rayos, y obtiene grandes mejoras en la eficiencia al realizar menos recorridos por el árbol, sobre todo en el caso de rayos coherentes. La técnica se extiende en [Wald 04a] (Packet Ray Tracing) para streams de rayos de mayor tamaño.

El método de Recorrido Vectorizado ([Noguera 07, Noguera 09]) describe la forma de intersecar intersecar un conjunto grande de rayos con una indexación espacial en un único recorrido. El método funciona muy bien incluso en el caso de rayos no coherentes, propiedad muy deseable para su aplicación en algoritmos de iluminación global.

1.7. Estudio teórico de la eficiencia de algoritmos

El estudio de la eficiencia de los algoritmos tiene su base en los trabajos de Church [Church 36] y Turing [Turing 36] sobre computabilidad. La teoría de la complejidad algorítmica estudia el tiempo que se tarda en resolver un problema y establece una jerarquía. Una introducción a esta teoría puede verse en [Sipser 06]. Existen problemas tan difíciles que no se pueden resolver con un algoritmo (no computables), otros a los que se les puede calcular una solución sólo para un subconjunto del dominio de los datos de entrada (semicomputables) y otros a los que siempre se les puede calcular una solución (computables).

Los problemas computables pueden dividirse por dificultad en función del tiempo que se tarda en obtener una respuesta. Los problemas NP se definen como aquellos que pueden ser resueltos en una máquina de turing no determinista en tiempo polinomial. Estos problemas requieren un tiempo exponencial en máquinas deterministas. Los problemas P son aquellos que pueden ser resueltos en tiempo polinomial en máquinas deterministas, y se consideran resolubles de forma eficiente (también existen problemas más difíciles que NP, pero se tardaría tanto en resolverlos que no se usan en la práctica).

Cuando se diseña un algoritmo es muy interesante conocer a priori cuánto tardarán en resolver una instancia del problema en función del tamaño del problema. Para ello se utiliza la notación $O()$. La función t que da el tiempo de cálculo del algoritmo pertenece a un conjunto de funciones denominado $O(f(n))$ (o lo que es lo mismo, el tiempo del algoritmo está en $O(f(n))$) si existe k tal que el algoritmo resuelve un problema de tamaño n en un tiempo $t < k * f(n)$. k es una constante arbitraria que depende de la implementación del algoritmo y del hardware. Utilizaremos esta notación para estudiar el rendimiento de los algoritmos expuestos en esta memoria de tesis, en función del número de rayos (complejidad de la iluminación) y del número de vértices (complejidad de la geometría).

Existe trabajo previo tanto en el estudio de la eficiencia en tiempo de algoritmos realistas de iluminación como en el estudio de sus parámetros de error y varianza. Esta sección contiene un compendio de estudios que han sido de utilidad para esta tesis.

1.7.1. Estudio teórico de indexación espacial y Ray Tracing

Havran [Havran 00b] estudia la eficiencia de diferentes métodos de indexación espacial usando un enfoque estadístico independiente de la implementación, con un conjunto de escenas de diferentes características. Hay estudios de las características de la escena que hacen que unas indexaciones espaciales sean más eficientes que otras. En [Havran 00a] y [Havran 03] se desarrolla un modelo computacional y de rendimiento de indexaciones espaciales que permite la comparación analítica de distintas técnicas para una escena y un conjunto de rayos dados.

Revelles, en [Revelles 03], caracteriza de forma teórica el concepto de optimizador (que engloba el de indexación espacial) y define una medida de eficiencia. Otros estudios usan una simulación rápida con pocos rayos para elegir el método de indexación más apropiado [Revelles Moreno 01]. También se han estudiado teóricamente modelos de costo que permiten mejorar los criterios de subdivisión [Goldsmith 87, MacDonald 90, Aronov 03].

Como la eficiencia de Ray Tracing está supeditada al recorrido eficiente de la estructura de datos que contiene la escena, los estudios de Ray Tracing se basan en técnicas similares. Reinhard [Reinhard 96] estudió el coste del algoritmo de ray tracing para distintas estructuras de indexación espacial teniendo en cuenta

la ocultación producida por los objetos a la hora de recorrer el árbol. Scherson [Scherson 87] estudió qué parámetros de la escena eran buenos predictores del costo de raytracing para esa escena. Szirmay-Kalos [Szirmay-Kalos 98] estudió teóricamente la eficiencia (en promedio y en el peor caso) de algoritmos de indexación espacial para el caso de Ray Tracing. Finalmente Aronov [Aronov 02] diseñó un predictor del coste de Ray Tracing, que funciona bien para octrees.

1.7.2. Estudio de la varianza de algoritmos de iluminación global

Existen algunos estudios de la varianza y el error de diferentes métodos de Iluminación Global. Arvo [Arvo 95b] hizo un análisis del error de los algoritmos de elementos finitos. En cuanto a métodos de Monte Carlo, Sbert estudió la varianza de Pathtracing en [Sbert 97a]; mientras que en [Sbert 97b] estudió la varianza del método de muestreo por importancias, para superficies difusas. Shirley estudió el algoritmo de Radiosidad de Monte Carlo en [Shirley 92]. Como ejemplo de estudio de algoritmos de líneas globales, [Sbert 96b] estudia el error y la variancia de *global multipath Monte Carlo*. La ecuacion general de la varianza para el caso no difuso fue estudiada en detalle por Carlos Ureña [Ureña Almagro 98], para raytracing, radiosidad y pathtracing.

Algoritmos más nuevos basados en Photon Maps han recibido menos atención. El capítulo 3 contiene un análisis de los algoritmos de Cuenta de Impactos, Photon Maps, DETP y Ray Maps.

1.8. Herramientas formales

Para estudiar teóricamente los algoritmos de Iluminación Global, son necesarias algunas nociones de probabilidad, geometría y análisis. Esta sección describe los conceptos relevantes.

1.8.1. Teoría de Probabilidad

La teoría de probabilidad es la rama de las matemáticas que estudia los fenómenos aleatorios. En esta sección se introducen algunos conceptos que serán útiles en el estudio teórico de las técnicas de Iluminación Global del capítulo 3. Nos basamos en el libro de Parzen [Parzen 92].

Dada una variable aleatoria discreta X, que toma valores $x_0, \ldots, x_n$ con probabilidad $p_0, \ldots, p_n$, se definen los conceptos de esperanza de la variable X como

$$E(X) = \sum_{i=0}^{n} x_i p_i \tag{1.2}$$

y de varianza de la variable X como

$$Var(X) = E((X - E(X))^2) = E(X^2) - E^2(X) \tag{1.3}$$

El concepto de esperanza está relacionado con el valor promedio de la variable, y la varianza está relacionada con la dispersión (esto es, cómo de probable es encontrar valores alejados de la media). La desviación típica σ se define como la raíz cuadrada de la varianza. Cuando estimamos la esperanza de una variable aleatoria tomando muestras, podemos calcular un intervalo de confianza, que es un intervalo al que el valor que buscamos pertenece con una probabilidad determinada.

En el caso de una variable aleatoria continua, existen un número infinito de posibles valores que ésta puede tomar. En este caso, se define una función de densidad de probabilidad f en el dominio de posibles valores D. Para un subconjunto posibles valores $S \subset D$, se puede calcular la probabilidad de que la variable tome un valor en S usando una integral:

$$P(x \in S) = \int_S f(s)ds \tag{1.4}$$

La esperanza y la varianza se definen de manera similar al caso continuo:

$$E(X) = \int xf(x)dx \qquad Var(X) = E((X - E(X))^2) \tag{1.5}$$

Son interesantes también los conceptos de vector aleatorio y función aleatoria: un vector aleatorio es una tupla en la que cada componente es una variable aleatoria, mientras que una función aleatoria es una función que a cada elemento del dominio asigna una variable aleatoria. Usaremos estos conceptos al estudiar los algoritmos estocásticos de iluminación global.

1.8.2. Ensayos de Bernoulli independientes

Un ensayo de Bernoulli es un experimento que sólo puede tener dos resultados, normalmente llamados éxito y fracaso. La probabilidad de éxito se denota p y la de fracaso $q = 1 - p$.

Si repetimos el experimento de forma independiente n veces, podemos estudiar la probabilidad de tener un número determinado de éxitos. La probabilidad de tener k éxitos (y por tanto $n - k$ fracasos) es

$$b(k; n, p) = \binom{n}{k} p^k q^{n-k} \tag{1.6}$$

La probabilidad de tener al menos k éxitos es:

$$\sum_{i=k}^{n} b(i; n, p) = 1 - \sum_{i=0}^{k-1} b(i; n, p) \tag{1.7}$$

Estas fórmulas se vuelven intratables cuando el número de ensayos se hace muy grande. En ese caso, existen las siguientes aproximaciones:

- Si $p < 0,1$, la distribución de probabilidad se puede aproximar por una distribución de Poisson:

$$b(k;n,p) \approx e^{-np}\frac{(np)^k}{k!} \tag{1.8}$$

- Si $npq \geq 10$ la distribución de probabilidad cumulativa se puede aproximar por una distribución normal:

$$\sum_{k=a}^{b} b(k;n,p) \approx \Phi\left(\frac{b-np+\frac{1}{2}}{\sqrt{npq}}\right) - \Phi\left(\frac{a-np-\frac{1}{2}}{\sqrt{npq}}\right) \tag{1.9}$$

1.8.3. Distribución binomial

La distribución binomial es la distribución de probabilidad del número de éxitos en ensayos de Bernoulli independientes. Sea X una variable aleatoria que sigue una distribución binomial:

$$X \rightsquigarrow b(n,p) \tag{1.10}$$

La esperanza de esta variable $E(X)$ es

$$E(X) = np \tag{1.11}$$

y la varianza es

$$Var(X) = np(1-p) \tag{1.12}$$

Si generamos una nueva variable aleatoria Y escalando X ($Y = aX$), los valores de esperanza y varianza de Y son:

$$E(Y) = aE(X) = anp \qquad Var(Y) = a^2 Var(X) = a^2 np(1-p) \tag{1.13}$$

Utilizaremos estas fórmulas en el estudio teórico de la eficiencia de Estimación de Densidades en el Plano Tangente, para calcular el número de rayos que hay que procesar en cada momento. El estudio de la varianza de los distintos métodos de estimación de densidades también usa estas fórmulas.

1.8.4. Estadísticos de orden

La estadística de orden trata de las propiedades y aplicaciones de las variables aleatorias que aparecen al ordenar las muestras tomadas de una variable aleatoria [David 03]. Dada una muestra $\{x_1, \ldots, x_n\}$ de una variable aleatoria X, el estadístico de orden k es el k-ésimo valor más pequeño, y es una variable aleatoria que se nota $X_{(k)}$. Si X tiene una función de densidad de probabilidad $f(x)$ con probabilidad acumulada $F(x)$, la densidad de probabilidad del estadístico de orden k es:

$$f_{X_{(k)}}(x) = \frac{n!}{(k-1)!(n-k)!}F(x)^{k-1}(1-F(x))^{n-k}f(x) \tag{1.14}$$

y la densidad de probabilidad acumulada es:

$$F_{X_{(k)}}(x) = \sum_{i=k}^{n} \binom{n}{i} F^i(x)[1 - F(x)]^{n-i} \tag{1.15}$$

Utilizaremos los estadísticos de orden para derivar el valor esperado y la varianza exacta de Photon Maps y DETP.

1.8.5. Cálculo exterior

El cálculo exterior es la rama de las matemáticas que formaliza el concepto de diferencial. Usaremos los conceptos definidos en esta sección en el estudio de las probabilidades geométricas de la geometría integral.

Dado un espacio vectorial $V(\mathbb{R})$, definimos un tensor como una aplicación de V^n en $\mathbb{R}$. Un tensor T es antisimétrico si cumple la siguiente propiedad:

$$T(v_0, \ldots, v_i, \ldots, v_j, \ldots, v_{n-1}) = -T(v_0, \ldots, v_j, \ldots, v_i, \ldots, v_{n-1}) \tag{1.16}$$

para todos los posibles valores de los parámetros. Esto es, intercambiar los valores de dos parámetros cualesquiera manteniendo el resto de parámetros constantes cambia el signo del resultado.

Dados dos tensores antisimétricos $T : V^n \to \mathbb{R}$ y $S : V^n \to \mathbb{R}$, y $2n$ vectores $t_0, \ldots, t_{n-1}, s_0, \ldots, s_{n-1}$ definimos el producto exterior de los tensores T y S como un nuevo tensor $T \wedge S$ de la siguiente forma:

$$T \wedge S : V^{2n} \to \mathbb{K} \tag{1.17}$$

$$(T \wedge S)(t_0, \ldots, t_{n-1}, s_0, \ldots, s_{n-1}) = \tag{1.18}$$

$$T(t_0, \ldots, t_{n-1})S(s_0, \ldots, s_{n-1}) - S(t_0, \ldots, t_{n-1})T(s_0, \ldots, s_{n-1}) \tag{1.19}$$

Sobre este conjunto de tensores se define una operación de diferenciación d (cálculo de una derivada), con las siguientes propiedades:

$$df = \nabla f = \sum_{i=1}^{n} \vec{e}_i \frac{\partial f}{\partial x_i} \tag{1.20}$$

$$d(x + y) = dx + dy \tag{1.21}$$

$$ddx = 0 \tag{1.22}$$

$$d(x \wedge y) = dx \wedge y + (-1)^k x \wedge dy \tag{1.23}$$

donde f es un tensor, $\vec{e}_i$ es la base estandar de V, x, y son formas diferenciales y k es el grado de la forma diferencial (los tensores tienen grado 0; diferenciar una forma de grado k genera una forma de grado $k + 1$).

1.8.6. Geometría integral

La geometría integral es el campo de la matemática que estudia conceptos relacionados con las probabilidades de intersección entre objetos geométricos. Utiliza resultados de estadística, geometría y análisis. En el campo de la Iluminación Global basada en métodos estocásticos, calcular el tiempo de cómputo de los algoritmos se ve dificultado por la naturaleza estadística de los algoritmos. Las fórmulas analíticas que proporciona la geometría integral para la probabilidad de intersección entre líneas y conjuntos convexos son aplicables a la intersección rayo-objeto y rayo-nodo en una indexación espacial. Una vez conocidas estas fórmulas, el tiempo de los algoritmos puede estimarse como $p * t_i + (1 - p) * t_{ni}$, donde p es la probabilidad de intersección, t_i es el tiempo en caso de intersección, y t_{ni} es el tiempo en caso de no intersección.

Para derivar fórmulas en geometría integral, se define una medida en el espacio n-dimensional que permite definir una integral en el espacio. A partir de esta integral se derivan los conceptos de longitud, superficie y volumen, e integrales de curvatura media. Con estos datos, se derivan fórmulas que permiten calcular las probabilidades de intersección e inclusión entre distintos elementos geométricos. Seguimos el libro de Santaló [Santaló 02] para definir los conceptos de conjunto convexo, medida de un conjunto de puntos, medida de un conjunto de líneas y exponer resultados usados en el estudio teórico de nuestros algoritmos.

Un conjunto de puntos K en el plano es convexo si para cada par de puntos $A, B \in K$, se cumple que $\overline{AB} \subset K$, donde $\overline{AB}$ es el segmento que une A y B. Definimos ∂K como la frontera de K. Definiremos la medida de un conjunto de puntos de la siguiente forma: Sean x, y coordenadas cartesianas en el plano, y P un conjunto de puntos del plano. La única medida invariante bajo transformaciones geométricas es (salvo escalados)

$$m(P) = \int_P dx \wedge dy \tag{1.24}$$

donde $\wedge$ denota el producto exterior. Definimos la densidad de puntos como $dP = dx \wedge dy$. La probabilidad p de que un elemento aleatorio de un conjunto X esté en un conjunto $Y \subset X$ es $p = \frac{m(Y)}{m(X)}$.

Una línea recta G en el plano queda determinada por el ángulo ϕ que la dirección perpendicular a G hace con respecto a una dirección fija ($0 \leq \phi \leq 2\pi$) y su distancia p al origen O ($0 \leq p$). La medida de un conjunto de líneas G invariante bajo transformaciones es $m(G) = \int_G dp \wedge d\phi$. Definimos la densidad de un conjunto de líneas como $dG = dp \wedge d\phi$.

1.8.7. Intersección entre líneas y conjuntos convexos

Sea G un conjunto de líneas y K un conjunto convexo en el plano. La medida de las líneas de G que intersecan a K es:

$$m(G; G \cap K \neq \emptyset) = \int_{G \cap K \neq \emptyset} dp \wedge d\phi = L \tag{1.25}$$

donde L es la longitud de ∂K.

Sean K_0, K_1 conjuntos convexos en E_n (el espacio euclídeo de dimensión n) tal que $K_1 \subset K_0$. La probabilidad de que un r-plano L_r ($r = 1, \ldots, n-1$), elegido de acuerdo con una distribución aleatoria invariante bajo rotación y traslación, que corta a K_0 también corte a K_1 es:

$$p(L_r \cap K_1 \neq \emptyset) \;=\; \frac{M_{r-1}(\partial K_1)}{M_{r-1}(\partial K_0)} \tag{1.26}$$

donde ∂K es la frontera de K y M_{r-1} son las integrales de curvatura media.

Especializando para $n = 3$ (espacio tridimensional) y $r = 1$ (hiperplanos de una dimensión: líneas) la probabilidad resultante es:

$$p(L_0 \cap K_1 \neq \emptyset) = \frac{M_0(\partial K_1)}{M_0(\partial K_0)} \tag{1.27}$$

Particularizando para cuerpos convexos K de E_3, las integrales de curvatura media son $M_0 = F$ (área de ∂K), como dice la sección III.13.6 de [Santaló 02]. Por tanto,

$$p(L_0 \cap K_1 \neq \emptyset) \;=\; \frac{F_1}{F_0} \tag{1.28}$$

Es decir, la probabilidad p de que una línea recta que corta a un conjunto convexo también corte a otro conjunto convexo incluído en él es el cociente entre las superficies de los conjuntos. Esta propiedad es fundamental, ya que constituye la base del estudio teórico de los algoritmos presentados en esta memoria de tesis.

Al estudiar los algoritmos de Iluminación Global, supondremos que la distribución de los rayos sigue una distribución uniforme (densidad invariante bajo translación y rotación), y usaremos las fórmulas anteriores para estimar las probabilidades de intersección y los tiempos de cálculo.

En la verificación de las predicciones, puede ser necesario generar rayos (líneas) con una distribución uniforme. Para generar un rayo que siga esta distribución, se deben generar dos puntos distribuidos uniformemente en la superficie de la esfera. Usamos el método presentado en [Sbert 96a] y [Tobler 98]. Luego, la línea que une los dos puntos sigue una distribución uniforme en la esfera, como muestra [Rovira 05]. El algoritmo puede verse en la figura 1.7.

1.8.8. Transformada de Fourier

La transformada de Fourier permite estudiar el comportamiento de funciones periódicas complejas mediante su descomposición en sumas ponderadas de funciones trigonométricas simples. Usaremos esta transformada en la demostración de teoremas necesarios para el estudio teórico de nuestros algoritmos.

La definición formal de la transformada es la siguiente: Dada una función $f : \mathbb{R} \to \mathbb{C}$, integrable Lebesgue ($f \in L^1(\mathbb{R})$) definimos la transformada de Fourier de f como:

$$F(\xi) := \int_{-\infty}^{\infty} f(x)\; e^{-2\pi i x \xi}\, dx, \tag{1.29}$$

Algoritmo 1.8.1: PointOnSphereSurface()

$Z \leftarrow U[-1, 1]$
$\phi \leftarrow U[0, 2\pi]$
$\theta \leftarrow \arcsin(z)$
$X \leftarrow \cos(\theta)\cos(\phi)$
$Y \leftarrow \cos(\theta)\sin(\phi)$
return $(Point(X, Y, Z))$

Algoritmo 1.8.2: LineInSphere()

$a \leftarrow$ PointOnSphereSurface()
$b \leftarrow$ PointOnSphereSurface()
return $(Ray(a, b - a))$

Figura 1.7: Pseudocódigo que genera rayos uniformemente en la esfera unidad.

La transformada de Fourier inversa se define como:

$$f(x) := \int_{-\infty}^{\infty} F(\xi)\ e^{2\pi i x \xi}\, d\xi, \tag{1.30}$$

En física, estas transformadas permiten pasar del dominio del tiempo al dominio de la frecuencia y viceversa. Si f es una función periódica de periodo $2T$, se puede definir la serie de Fourier asociada a f:

$$f(t) \sim \sum_{n=-\infty}^{\infty} c_n\, e^{\pi i \frac{n}{T} t} \tag{1.31}$$

donde

$$c_n = \frac{1}{2T} \int_{-T}^{T} f(t)\, e^{-\pi i \frac{n}{T} t}\, dt. \tag{1.32}$$

son los coeficientes de Fourier.

1.8.9. Teorema de muestreo de Nyquist–Shannon

En el campo de Teoría de la Información, el teorema de muestreo de Nyquist-Shannon [Nyquist 28, Shannon 49] indica a qué frecuencia (espacial o temporal) se debe muestrear una señal para poder reconstruirla. Formalmente:

Teorema 1.8.1 *Si una función $x(t)$ no contiene frecuencias mayores o iguales que B muestras por unidad de espacio o tiempo, está completamente determi-*

nada por sus ordenadas en una serie de puntos espaciados a $1/(2B)$ *unidades de distancia.*

Demostración Sea $F(\omega)$ el espectro de $f(t)$. Entonces

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\,d\omega = \frac{1}{2\pi}\int_{-2\pi W}^{2\pi W} F(\omega)e^{i\omega t}\,d\omega \tag{1.33}$$

ya que se asume que $F(\omega)$ es cero fuera de la banda W. Si definimos

$$t = \frac{n}{2W} \tag{1.34}$$

donde n es cualquier entero positivo o negativo, obtenemos

$$f\left(\frac{n}{2W}\right) = \frac{1}{2\pi}\int_{-2\pi W}^{2\pi W} F(\omega)e^{i\omega\frac{n}{2W}}\,d\omega \tag{1.35}$$

A la izquierda están los valores de $f(t)$ en los puntos de muestreo. La integral de la derecha es el coeficiente n-ésimo de la expansión en serie de Fourier de la función $F(\omega)$, usando el intervalo $[-W, W]$ como periodo fundamental. Por tanto los valores de las muestras $f(\frac{n}{2W})$ determinan los coeficientes de Fourier en la expansión en serie de $F(\omega)$, y por tanto determinan $F(\omega)$, ya que $F(\omega)$ es cero para frecuencias mayores que W, y para frecuencias menores $F(\omega)$ queda determinada si sus coeficientes de Fourier son conocidos. Pero $F(\omega)$ determina la función original $f(t)$ completamente, ya que una función está completamente determinada si su espectro es conocido. Por tanto las muestras originales determinan la función $f(t)$ completamente. ∎

Este teorema resulta fundamental en cualquier aplicación que necesite reconstruir funciones a partir de muestras discretas, como de hecho ocurre en estimación de densidades, donde es especialmente relevante debido a que impone una cota inferior al tamaño de las variaciones visibles de la función de radiancia que se pueden reconstruir al generar una imagen.

Corolario 1.8.2 *Si una función tiene componentes en frecuencia mayores que 2 veces el periodo de muestreo, no puede ser reconstruída sin pérdida.*

Utilizaremos este teorema en el cálculo del valor óptimo del radio del disco de Estimación de Densidades en el Plano Tangente (sección 3.8.2).

Capítulo 2

Indexación de Discos

En este capítulo estudiaremos la Indexación de Discos, un algoritmo que permite mejorar los tiempos del algoritmo de Estimación de Densidades en el Plano Tangente (DETP) . Las publicaciones relacionadas son [García 05] y [García 06]. Comenzaremos comentando la razón de ser del nuevo método. Tras ello se describe en detalle su implementación y se compara con el algoritmo de Caché de Esferas. Finalmente se resumen los resultados y las conclusiones obtenidas.

2.1. Objetivos

Los tiempos de cálculo de Estimación de Densidades en el Plano Tangente y la Caché de Esferas, descritos en la sección 1.4.4 del primer capítulo (página 28), son demasiado altos para su uso en entornos interactivos complejos. Nuestro objetivo es investigar formas de calcular la irradiancia más eficientes en tiempo.

La Caché de Esferas recorre por orden todos los puntos donde se debe calcular la irradiancia y para cada punto busca rayos cercanos. Por tanto, su eficiencia es lineal en el número de puntos. Si insertamos los puntos en una estructura de datos de indexación espacial jerárquica, como las descritas en la sección 1.6 del capítulo anterior (página 35), podemos conseguir una eficiencia en tiempo logarítmica en el número de puntos de irradiancia. Denominamos al método Indexación de Discos. Una ventaja adicional de este método es que se pueden estimar nuevas muestras de irradiancia interpolando entre muestras cercanas, como si fuera la caché de irradiancias definida en [Ward 88], usando el método de interpolación descrito en esta referencia.

Aunque este método está indicado sólo para métodos de estimación de densidades de radio fijo, ya que en caso contrario no se conoce a priori la zona del espacio que afecta a cada punto, Havran, en [Havran 05b], intenta aplicar un método similar a éste para Photon Maps. Sin embargo, debido a que en Photon Maps un fotón afecta a puntos de irradiancia más o menos lejanos en función del resto de fotones, el método no es aplicable sin cambiar de forma radical la

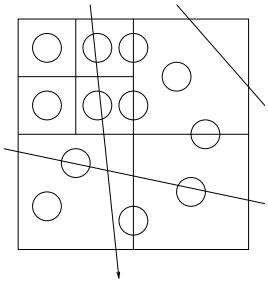


Figura 2.1: Indexación de Discos.

forma de hacer estimación de densidades.

2.2. Descripción del método

La técnica de Indexación de Discos crea una indexación espacial con los discos de la escena. La caja englobante de los discos se subdivide recursivamente y se crea una estructura de indexación espacial. Posteriormente, los rayos atraviesan el índice espacial añadiendo su contribución a los discos que intersecan. (la figura 2.1 es un ejemplo en 2D que muestra los discos y las trayectorias de los rayos). El rayo sólo ha de seguirse hasta la primera intersección con la escena real (o la segunda si hay superficies cóncavas); puede sin embargo atravesar un número indeterminado de discos a los que irá sumando su energía. El pseudocódigo puede verse en la figura 2.2. El índice espacial debe ser capaz de guardar discos y calcular eficientemente todas las intersecciones con un segmento (los extremos del segmento son el origen del rayo y la intersección con la escena real). Todos los algoritmos mencionados en la sección 1.6 pueden hacerlo.

La sección 1.7.1 del capítulo anterior (página 38) describe algunos estudios que permiten elegir la mejor técnica de indexación espacial para intersecar la escena real con los rayos. Ya que la posición de los discos sigue la superficie de los objetos, esta investigación es aplicable a esta técnica también. La sección 1.6 contiene una descripción de algoritmos útiles de indexación espacial. Especialmente interesantes para Indexación de Discos son los algoritmos que atraviesan el árbol con varios rayos a la vez (como [Wald 01, Wald 04a]), o incluso una única vez (como [Noguera 07]). Este último algoritmo es muy útil en Iluminación Global ya que funciona bien para rayos poco coherentes, lo que es el caso en Indexación de Discos.

El método de Indexación de Discos tiene más rendimiento que la Caché de Esferas cuando los discos tienen un radio que es del orden de magnitud de la distancia media entre puntos en que se calcula la irradiancia, o menor, como veremos a continuación en la sección 2.3. Demostraremos teóricamente en la sección 3.12 que la distancia media entre puntos es el radio óptimo del disco para la técnica de DETP. Para discos mayores, la Caché de Esferas es mejor.

Algoritmo 2.2.1: SPATIALINDEXING.FOLLOW(Ray, I)

$DiscList\ DL$
$Node \leftarrow FirstNode(Ray)$
while $Node \neq NULL$
do $\begin{cases} DL.Add(Node.Intersections(Ray, I)) \\ Node \leftarrow NextNode(Node, Ray, I) \end{cases}$
return (DL)

Algoritmo 2.2.2: INITIALDETPRADIOSITYCALCULATION($Intersection\ I$)

$DiscList\ DL$
for each $Vertex$ in the scene
do $\begin{cases} disc \leftarrow Disc(Vertex.pos, Vertex.nor) \\ disc.Energy \leftarrow 0 \\ DL.Add(disc) \end{cases}$
$SpatialIndexingSI = CreateSI(DL)$
for each Ray in the scene
do $\begin{cases} DiscList\ Intersected = SI.Follow(Ray, I) \\ \textbf{for each } disc \text{ in } Intersected \\ \quad \textbf{do } \{disc.Energy+ = Ray.Energy \end{cases}$

Figura 2.2: Pseudocódigo para el cálculo inicial de la radiosidad.

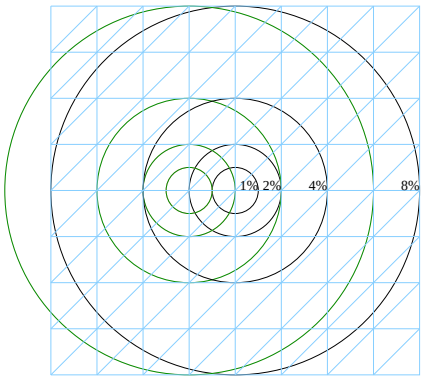


Figura 2.3: Discos en una malla típica.

Figura 2.4: Escena del árbol, 10 000 fotones.

La profundidad máxima de la indexación espacial debe ajustarse al cambiar el tamaño del disco, ya que al crecer los discos, se cortan entre sí. Dividir un voxel crea voxels donde la mayoría de los discos pertenecen a todos los voxels hijos, lo que hace que el tiempo de intersección con los hijos sea mayor que con el voxel original. Para ilustrar el problema, imagine que queremos calcular la irradiancia en cada uno de los vértices de la malla de la figura 2.3 (recuerde sin embargo que DETP es independiente de geometría). Hay un disco centrado en cada vértice de la escena. Al crecer el disco, se nota que una partición del espacio no ayuda.

2.3. Comparación empírica entre la Caché de Esferas y la Indexación de Discos

En esta sección se usará una escena con 71 966 triángulos y un objeto móvil de 500 triángulos (véase la figura 2.4). Esta escena se llama la primera Escena del Árbol. Se generó una segunda escena con un tipo distinto de árbol y de suelo, con triángulos más grandes, que se muestra en la figura 2.5.

Se usa un árbol BSP alineado a los ejes [Sung 92] en los primeros ejemplos de esta sección, y se usa un Octree en la comparación posterior, más completa. Los resultados de tiempos para el árbol BSP se muestran en el Cuadro 2.1 (E es la Escena, F es el número de fotones, y Radio corresponde al radio del disco en DETP). Se muestra el tiempo para calcular la fase de estimación de

Figura 2.5: Segunda Escena del Árbol, 20 000 fotones.

densidades para la escena correspondiente y el tamaño relativo del radio del disco. En este capítulo, todos los tiempos se expresan en segundos, y los radios como porcentajes del tamaño de la escena. El número óptimo de primitivas por voxel puede verse en el cuadro 2.2. Se puede observar claramente cómo al crecer el tamaño de los discos, es mejor un árbol de menor profundidad con mayor densidad de discos en las hojas, como comentamos en la sección anterior.

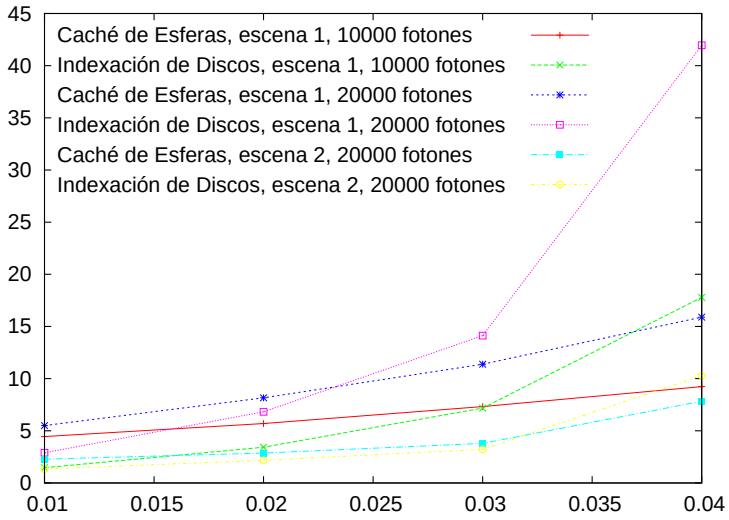
Los triángulos del suelo tienen aristas cuya longitud es aproximadamente el 2 % de la longitud de la escena y los triángulos del árbol tienen aristas de aproximadamente el 1 %. Puede verse que para tamaños de los discos por debajo del 4 % de la escena, la Indexación de Discos es más rápida que la cache de esferas. Para 20 000 fotones, los resultados sólo son mejores para tamaños del 1 % y 2 %.

Este cuadro muestra que el rendimiento de la Indexación de Discos disminuye al aumentar el area de los discos. Puede verse que para la segunda escena los resultados son mejores usando discos más grandes con respecto a la primera escena, debido al hecho de que el tamaño de los triángulos ha aumentado. Para discos grandes una subdivisión espacial de grano fino no es útil, ya que provoca muchas intersecciones repetidas con el disco al atravesar la estructura por los rayos.

Para hacer una comparación más completa entre las técnicas, se realizaron una serie de experimentos con la primera escena, usando un octree como base para la Indexación de Discos. Los experimentos llevados a cabo cambian las siguientes tres variables:

- Radio del disco de DETP (1 %, 2 %, 4 %, 8 % y 16 % de la escena)

Algoritmo	**E**	**F**	**Radio**			
			1 %	2 %	3 %	4 %
Cache de Esferas	1	10k	4,45	5,70	7,32	9,24
Indexación de Discos	1	10k	1,46	3,43	7,18	17,79
Cache de Esferas	1	20k	5,50	8,16	11,38	15,89
Indexación de Discos	1	20k	2,89	6,82	14,12	41,95
Cache de Esferas	2	20k	2,28	2,86	3,79	7,83
Indexación de Discos	2	20k	1,34	2,17	3,22	10,29



Cuadro 2.1: Tiempo de cómputo de la Caché de Esferas y la Indexación de Discos para distintas escenas. Tiempo como función del radio del disco.

Escena	**Fotones**	**Radio**			
		1 %	2 %	3 %	4 %
1	10 000	20	20	50	800
1	20 000	20	20	200	1600
2	20 000	40	80	160	32000

Cuadro 2.2: Mejor número de primitivas por voxel para la indexación de disco como función del radio del disco.

- Profundidad máxima del octree (5, 6, 7 u 8 niveles)
- Número de fotones (entre 100 y 409 600)

Las muestras con los resultados de tiempos de todos los experimentos pueden verse en el apéndice A; en este capítulo mostraremos sobre todo gráficas resumen. Mostraremos gráficamente los costes de crear la Indexación de Discos para los distintos tamaños de disco y profundidad, y los compararemos con

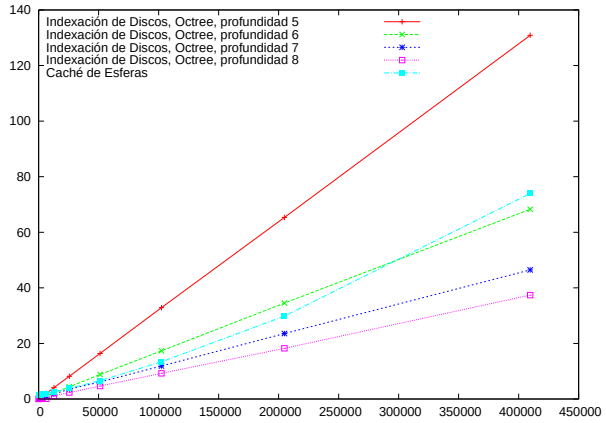


Figura 2.6: Tiempo para calcular la estimación de densidades. Radio 1 % de la escena. Tiempo en función del número de fotones.

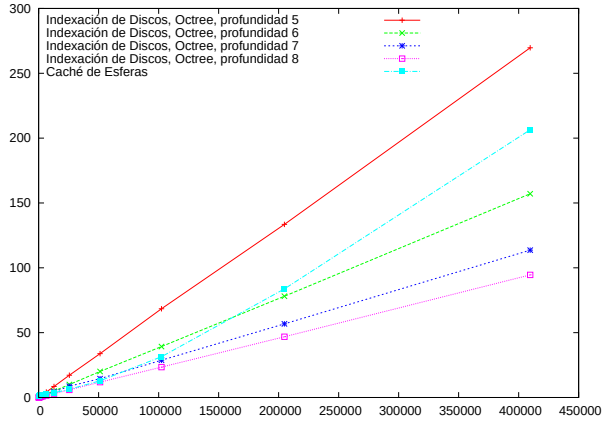


Figura 2.7: Tiempo para calcular la estimación de densidades. Radio 2 % de la escena. Tiempo en función del número de fotones.

la caché de esferas. También estudiaremos el coste de cada caso para diferentes números de rayos, y finalizaremos con una gráfica que muestra el número de fotogramas para amortizar el coste de la creación del árbol de discos (i.e. a partir de este número de fotogramas, el costo total de Indexación de Discos es menor que el de la caché de esferas). Un gráfico con el tiempo de cómputo para distintos tamaños de radio del disco puede verse en las figuras 2.6 (1 %), 2.7 (2 %), 2.8 (4 %), 2.9 (8 %) y 2.10 (16 %). Cada figura contiene los resultados de cinco ejecuciones: Indexación de Discos usando un Octree con profundidad máxima del árbol de 5, 6, 7 u 8 niveles, y Caché de Esferas.

Por otra parte, también es interesante ver cómo afecta el tiempo de creación

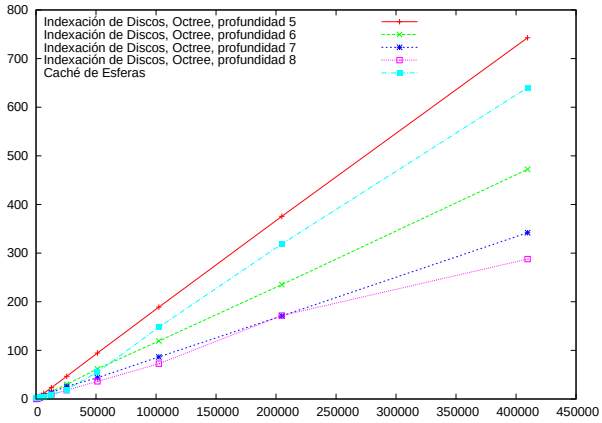


Figura 2.8: Tiempo para calcular la estimación de densidades. Radio 4 % de la escena. Tiempo en función del número de fotones.

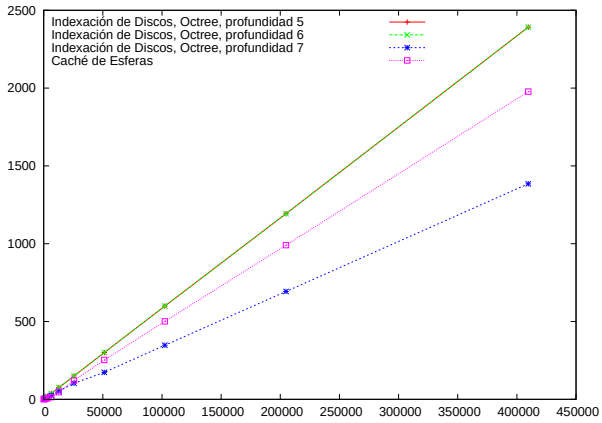


Figura 2.9: Tiempo para calcular la estimación de densidades. Radio 8 % de la escena. Tiempo en función del número de fotones.

del árbol. Las figuras 2.11 a 2.15 muestran el número de fotogramas necesarios para amortizar la creación del árbol, para un radio de disco entre el 1 % y el 16 % de la escena, en función del número de rayos trazados. Los huecos de las gráficas son puntos donde la Caché de Esferas es más eficiente. Se puede observar que para discos de tamaño pequeño el tiempo de creación del árbol puede amortizarse en unos 100 a 200 fotogramas (5 a 10 segundos de película a 20 fotogramas por segundo). Para tamaños de disco muy grandes el tiempo de amortización se dispara, o bien la Caché de Esferas es más eficiente.

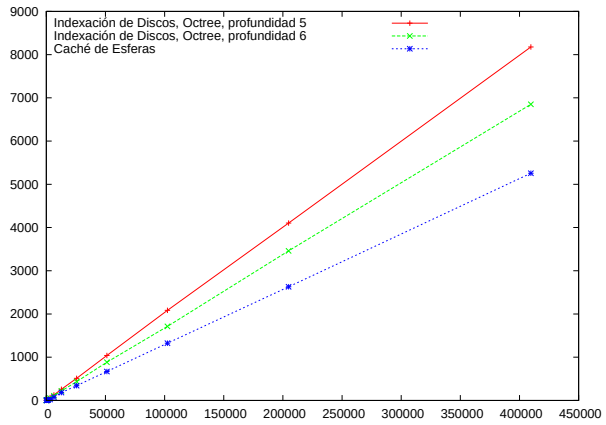


Figura 2.10: Tiempo para calcular la estimación de densidades. Radio 16 % de la escena. Tiempo en función del número de fotones.

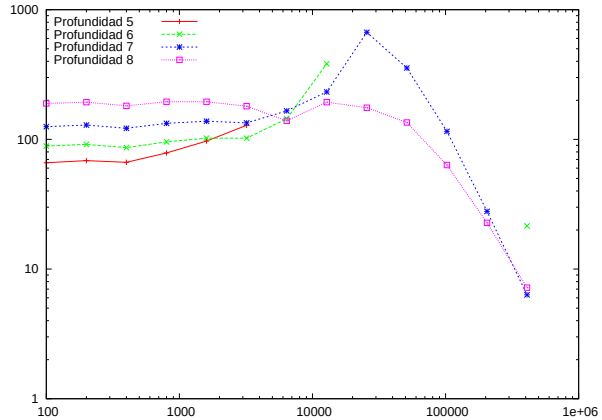


Figura 2.11: Fotogramas para amortizar la creación del árbol en Indexación de Discos, usando un Octree. Radio 1 % de la escena.

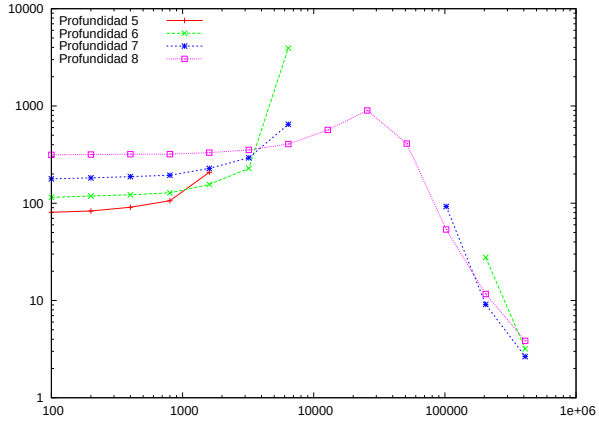


Figura 2.12: Fotogramas para amortizar la creación del árbol en Indexación de Discos, usando un Octree. Radio 2 % de la escena.

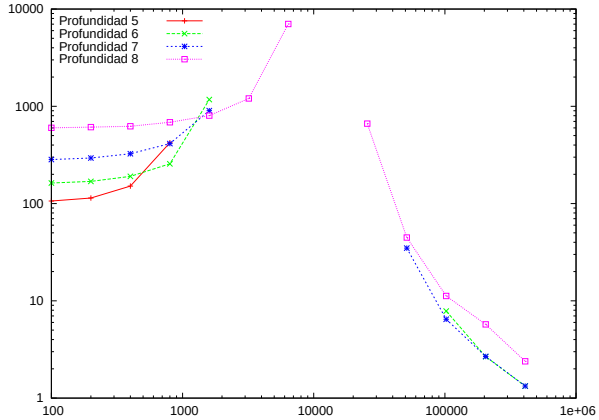


Figura 2.13: Fotogramas para amortizar la creación del árbol en Indexación de Discos, usando un Octree. Radio 4 % de la escena.

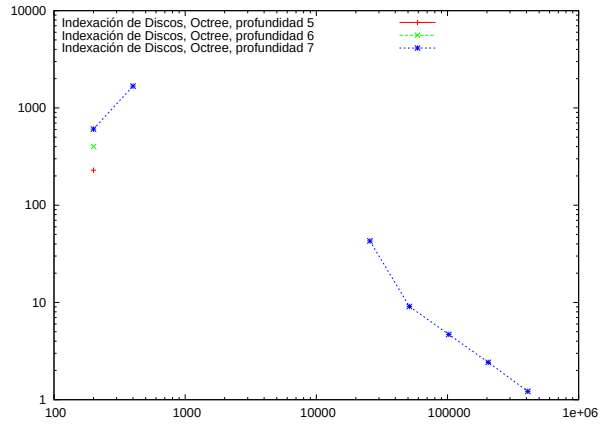


Figura 2.14: Fotogramas para amortizar la creación del árbol en Indexación de Discos. Radio 8 % de la escena.

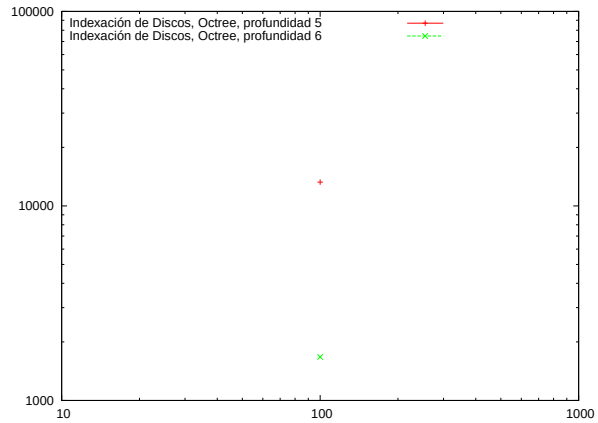


Figura 2.15: Fotogramas para amortizar la creación del árbol en Indexación de Discos. Radio 16 % de la escena. Se observa claramente que el método no es adecuado para discos tan grandes.

2.4. Resultados

Los resultados más importantes a destacar son:

- Para radios pequeños, la Indexación de Discos con la máxima profundidad es siempre mejor que la Caché de Esferas, y mejora los tiempos en gran medida (hasta 50 % para 409600 fotones). Por tanto, este método es muy beneficioso en la generación de imágenes de gran calidad, en la que el sesgo debe ser muy pequeño.
- El rendimiento de la Indexación de Discos baja más rápido que el de la Caché de Esferas conforme crece el tamaño del disco.
- Los requerimientos de memoria de la Indexación de Discos crecen muy rápido al crecer el tamaño del disco, si la profundidad del árbol se mantiene constante.
- La profundidad del octree debería disminuir al crecer el tamaño del disco, de modo que se evite la copia de discos en muchos nodos vecinos. Grandes tamaños de disco, por tanto, disminuyen la eficiencia de esta técnica, que deja de ser competitiva con la Caché de Esferas.
- En este ejemplo, para radios de tamaño pequeño y profundidades bajas, la Indexación de Discos es útil cuando el número de fotones es menos de 10 000.

Como resumen, este método es útil tanto para imágenes de muy alta calidad (con discos pequeños y potencialmente un número de rayos alto) como para simulaciones rápidas de baja calidad (con un número de rayos relativamente pequeño). Los casos intermedios se gestionan de forma más eficiente con la Caché de Esferas.

2.5. Conclusiones

Se ha diseñado e implementado una técnica de indexación espacial de los puntos de irradiancia para Estimación de Densidades en el Plano Tangente, que hemos llamado Indexación de Discos. Hemos comparado esta técnica con la Caché de Esferas, usando dos escenas de complejidad media. La nueva técnica obtiene mejores resultados que la Caché de Esferas para el caso de discos pequeños y un número relativamente bajo de rayos. Veremos en el capítulo 4 que estas características lo hacen útil para optimizar parte del cálculo incremental de la iluminación.

Se observa también una gran mejora de los tiempos para discos muy pequeños y un alto número de rayos; por tanto esta técnica es útil para imágenes de alta calidad. Finalmente, este método es compatible con la estimación de nuevas muestras de irradiancia por interpolación. Como desventaja, hay que tener en cuenta que la gestión de una indexación espacial basada en árboles ocupa bastante más memoria que la lista lineal de la Caché de Esferas, lo que limita su uso

en entornos de pocos recursos. Sin embargo, en el caso de algoritmos gráficos interactivos, las restricciones de tiempo son normalmente mucho más importantes que las de memoria, así que se suelen preferir algoritmos más rápidos aunque consuman más memoria.

Capítulo 3

Estudio teórico de la eficiencia de las técnicas

En este capítulo estudiaremos de forma teórica la varianza, el sesgo y la eficiencia de las distintas técnicas de iluminación global mencionadas anteriormente. Parte de este capítulo ha sido publicado previamente en [García 05, García 06].

3.1. Objetivos

Muy a menudo los algoritmos nuevos se comparan con algoritmos anteriores con un conjunto pequeño de escenas de prueba. Aunque existe un conjunto estandarizado de escenas de prueba [Lext 00], éste no siempre se usa, ya que muchos métodos están diseñados específicamente para escenarios con características especiales (cáusticas, fluorescencia, escenas de complejidad extremadamente alta, etc) que o bien no están representados en el estandar, o bien representan un porcentaje muy pequeño de las escenas. El estandar también está en evolución, añadiendo nuevas escenas con estas características.

Estamos interesados en comparar algoritmos en general, para cualquier tipo de escena. Esto nos lleva a un estudio teórico de los algoritmos. La sección 3.2 comenta cómo comparar entre sí algoritmos estocásticos. Aparte del tiempo de cómputo, en estos algoritmos es importante el error de la solución obtenida, así que hay que tener en cuenta ambas cosas. La medida de eficiencia definida en esa sección es una forma útil.

La sección 3.3 define los conceptos de sesgo y varianza, e introduce resultados útiles en el estudio de los algoritmos. En particular, se define una cota de la varianza para algoritmos de estimación de densidades, y se presenta un estimador de la varianza.

Tras esto se realizará un estudio de la convergencia, el sesgo, la varianza y la complejidad de las técnicas de iluminación global usando métodos estocásticos más utilizadas: Cuenta de Impactos, Photon Maps, Estimación de Densidades en

el Plano Tangente (DETP) y Ray Maps, seguido de un estudio de la eficiencia en tiempo de distintas variantes de Estimación de Densidades en el Plano Tangente, para distintos valores de los parámetros.

Los distintos algoritmos que tienen la misma técnica de estimación de densidades subyacentes calcularán la misma solución para cualquier escena y conjunto de rayos (aunque tardarán tiempos diferentes). En este caso, los algoritmos pueden compararse calculando el tiempo de cómputo en función del tamaño de la escena (número de discos y tamaño) y del número de rayos. Por tanto, en particular podemos comparar la eficiencia de los algoritmos que usan DETP entre sí sin considerar la varianza. El capítulo concluye con una sección de estimación automática de parámetros para DETP, que usa resultados teóricos del estudio para encontrar óptimos de los distintos parámetros. La notación usada en este capítulo se resume en el Cuadro 3.1.

Para comparar entre sí las otras técnicas de estimación de densidades (Cuenta de Impactos, Photon Maps, o los otros métodos de estimación de Ray Maps), se debe usar el estudio de varianza y error, mediante la medida de eficiencia de la sección siguiente.

El estudio de la complejidad de los algoritmos necesita abstraerse de los parámetros concretos de una simulación determinada. Las variables del estudio son el número de cálculos de irradiancia (complejidad de la escena) y el número de rayos (complejidad de la iluminación). Para calcular el tiempo promedio de los algoritmos es necesario una estimación del número de intersecciones rayo-disco, que dependen de la distribución de los rayos. Supondremos una distribución uniforme de rayos en el espacio, y calcularemos la medida del conjunto de rayos que intersecan un volumen finito. Esta suposición no realista es esencial para obtener fórmulas matemáticas para la probabilidad de intersección. Es una suposición común [Aronov 02, Aronov 03], también usada implícitamente en [Reinhard 96], y que funciona bien en la práctica.

Para integrar, necesitamos una medida para el espacio de rayos que sea invariante bajo rotación y translación [Santaló 02]. Esto significa que todas las posiciones y direcciones son igualmente importantes. La medida se llama μ_ℓ en [Aronov 03] y permite definir la densidad de los rayos; se usa implícitamente en las siguientes secciones.

Un estudio asintótico del rendimiento de los algoritmos en el caso promedio es útil para probar la escalabilidad, así que usaremos la notación $O()$ para ver el comportamiento de los distintos algoritmos en función de los parámetros.

3.2. Comparación teórica entre distintos algoritmos

Para comparar distintos algoritmos de estimación de densidades, a veces ejecutarlos en un número limitado de escenas preseleccionadas no es suficiente. Algunos algoritmos tienen ventajas e inconvenientes que dependen de la escena probada. Si las escenas no son conocidas a priori, las comparaciones entre los

Estudio de varianza	
$L, \widehat{L}, L_{max}$	Radiancia, estimador de radiancia y radiancia máxima
R_{max}	Reflectividad máxima
ϕ_T	Energía total
A	Área de la escena
A_0	Área de la envolvente convexa de la escena
K	Razón entre el área máxima y mínima de los parches
$P = \{P_i\}$	Conjunto de muestras de irradiancia
$n_P = \#P$	Número de muestras de irradiancia
d_P	Distancia promedio entre muestras de irradiancia
n_R	Número de rayos
p_1	Probabilidad de intersección de un rayo con una superficie
n_V	Número de vértices de la escena
T_i, A_{T_i}	Triángulos de la escena y área asociada
s	Número de triángulos que comparten un vértice en el mallado
k	Fotones buscados por Photon Maps
r_{PM}, V_{PM}	Distancia al impacto k y volumen que engloba a los fotones
Estudio de la complejidad	
Cantidades relacionadas con esferas	
$S = \{S_i\}$	Conjunto de esferas
r_i	Radio de S_i
r_0 =Envolvente de la escena; $\forall i > 0, r_i = r_{i-1}Q = r_0Q^i$	
n_i	Número de rayos que intersecan S_i
t_i	Costo de recalcular S_i
$V_i = \frac{4}{3}\pi r_i^3$	Volumen de S_i
m_i	Número de recálculos de la esfera S_i con ordenación de puntos.
k	Número de esferas
Cantidades relacionadas con el tiempo	
u	Tiempo de intersección Rayo-Disco
t	Tiempo de intersección Rayo-Esfera
T_R	Tiempo para recalcular las esferas con ordenación de puntos
T_I	Tiempo de intersección disco-esfera interna
$T = T_R + T_I$	Tiempo del algoritmo
Otros símbolos	
$0 <$Q< 1	Razón de radios entre dos esferas
d	Radio del disco en DETP
A_d	Área del disco en DETP
z	Multiplicidad del árbol en la Caché de Esferas Multilista
a	Arista de la caja englobante de la escena (para Ray Maps)
R_k	Rayos en un voxel a profundidad k (para Ray Maps)

Cuadro 3.1: Símbolos usados en el estudio teórico.

algoritmos se pueden hacer estudiando teóricamente su rendimiento en tiempo y varianza con respecto a la complejidad de la escena. Los algoritmos pueden compararse en ese caso usando la medida de eficiencia que definimos aquí:

$$\text{Eficiencia} = \frac{1}{\text{Tiempo} \quad \text{Varianza}} \tag{3.1}$$

donde Tiempo y Varianza se refieren al tiempo para calcular la fase de estimación de densidades de los algoritmos, y a la varianza de la solución obtenida (no se tiene en cuenta la fase de fotosimulación).

Otra ventaja de conocer la varianza de los algoritmos es ser capaz de estimar el error sin generar una costosa imagen de referencia, como se muestra en la siguiente sección.

3.3. Análisis del sesgo y de la variancia

Cuando se usa un algoritmo estocástico, la exactitud del algoritmo y su coste computacional están relacionados. En comparación con algoritmos no estocásticos, a cambio de un tiempo de cómputo bastante más pequeño, el algoritmo obtiene un estimador aleatorio de la solución. Si el estimador converge a la solución verdadera, el algoritmo no tiene sesgo. A menudo algoritmos con sesgo pueden producir soluciones mucho más rápido, con el inconveniente de disminuir la exactitud.

Los algoritmos estocásticos usan variables aleatorias durante su ejecución, y el resultado final del algoritmo puede verse también como una variable aleatoria. Las variables aleatorias tienen varianza. Esto significa que distintas ejecuciones de los algoritmos producen soluciones distintas, y la diferencia entre la solución real y una ejecución determinada es una variable aleatoria proporcional a la raíz cuadrada de la varianza (la desviación típica σ).

El sesgo de un algoritmo es la diferencia entre la solución correcta y la solución promedia que el algoritmo obtiene. Buenos algoritmos reducen el sesgo conforme los parámetros (número de rayos, etc) aumentan, y convergen a la solución correcta cuando los parámetros divergen. Formalmente, el sesgo en un punto P es

$$Bias_P = L(P) - E(S(P)) \tag{3.2}$$

donde L es la radiancia y $S(P)$ es el resultado del algoritmo.

Cuando se diseñan algoritmos dentro del campo de Gráficos por Ordenador, es conveniente compararlos con algoritmos existentes para demostrar que son mejores que el resto de técnicas en algún sentido. Normalmente se compara el coste y el error de la técnica nueva con repecto a las anteriores. El error de un algoritmo se calcula normalmente usando la media de los errores generados por el algoritmo en cada pixel, usando una imagen de referencia. Alternativamente, podemos usar un estudio de complejidad en tiempo de los algoritmos, sesgo y varianza para comparar los algoritmos.

La mayoría de los algoritmos de Estimación de Densidades (y en particular los algoritmos de Cuenta de Impactos, Photon Maps, DETP y Ray Maps, que estudiamos en este capítulo) funcionan encontrando muestras de energía, sumándolas, y dividiendo por el área ocupada. Otra forma de ver estos métodos de estimación es como un producto escalar en un espacio vectorial de funciones de cuadrado integrable: $E(\widehat{L}) = \langle k_P | L \rangle$, donde $\widehat{L}$ es el estimador de la radiancia y k_P es el núcleo de la estimación de densidades en un punto P (esta es la función que asigna pesos a las distintas muestras en función de su distancia al punto). El número de muestras de energía en una región sigue una distribución binomial. Si el núcleo de la estimación de densidades es constante (salvo Photon Maps, los mencionados arriba), el valor esperado de la distribución binomial

$$E = \sum_{k=0}^{k=n} b(k; n, p_1) \tag{3.3}$$

se está muestreando para cada estimación de la radiancia en un punto, donde n es el número de rayos generados n_R, p_1 es la probabilidad de que un rayo interseque (la intersección se define de forma diferente para los distintos métodos: impactos en superficies o trayectorias) en la vecindad del punto de cálculo de radiancia, y k es el número de rayos cerca del punto. Este valor E se multiplica después por la energía de cada fotón y se divide por el área muestreada. Los algoritmos con núcleo constante permiten obtener un límite superior de la variancia para escenas difusas con la siguiente fórmula ([Shirley 92]):

$$Var(\widehat{L}) \leq \frac{R_{max}}{(1 - R_{max})\pi K A} L_{max} \frac{\phi_T^2}{n_R} \tag{3.4}$$

donde R_{max} es la reflectividad máxima, A es el área de la escena, K es el cociente entre el área del parche más grande y el más pequeño, L_{max} es la radiancia máxima, y ϕ_T es la potencia. Las fórmulas para la variancia de los distintos métodos se pueden obtener usando estadísticos de orden o estimación de densidades, pero estas fórmulas dependen de la radiancia entrante, que es desconocida.

Debido a esto, aparte de acotar la varianza de los algoritmos, calcularemos una aproximación al valor de la varianza. Para poder calcular esta aproximación se asumirá una distribución uniforme de rayos y puntos de irradiancia. Para una escena dada, la varianza de los algoritmos sólo depende del número de muestras encontradas y del área. La varianza es directamente proporcional al cuadrado de la potencia de los fotones e inversamente proporcional al número de rayos trazados y al área donde se están buscando los fotones:

$$Var(\widehat{L}) \approx c_1 \times \frac{\phi_T^2}{n_R \, A} \tag{3.5}$$

donde $\widehat{L}$ es de nuevo el estimador de radiancia, c_1 es la constante que tiene en cuenta el resto de los efectos, n_R es el número de rayos, A es el área, y ϕ_T es la potencia de los fotones. Todos los fotones tienen la misma potencia, ya que

es más eficiente absorber o reflejar el fotón entero en las interacciones con las superficies.

En las siguientes secciones se estudiarán los distintos algoritmos siguiendo el siguiente esquema: primero una descripción del algoritmo, luego un estudio de la convergencia y el sesgo, un estudio de la varianza (se acotará primero la varianza de cada algoritmo en el caso general, y después se realizará una estimación de la radiancia suponiendo una distribución uniforme de rayos), y finalmente un estudio de la complejidad algorítmica.

3.4. Cuenta de Impactos

El algoritmo de Cuenta de Impactos (sección 1.4.1) divide la geometría de la escena en parches. La simulación del transporte de la luz consiste en tirar rayos desde las fuentes de luz, y reflejarlos o absorberlos en las superficies de acuerdo con la BRDF. En cada parche se va acumulando la energía de los fotones que lo golpean, para poder reconstruir la radiancia posteriormente.

Para evitar que los parches sean visibles en la imagen final, se realiza una interpolación dentro del parche. La mayoría de las escenas hoy en día se modelan mediante triángulos, así que en la práctica muchas veces cada triángulo es un parche. En este caso, a los vértices se les asigna un valor de radiancia promediando el valor de los triángulos a los que pertenece, y tras ello se hace una interpolación trilineal dentro del triángulo.

3.4.1. Convergencia y Sesgo

La primera fase de la Cuenta de Impactos (cálculo de radiancia en un parche completo) converge a la densidad de energía promedio en cada triángulo conforme el número de rayos aumenta. Sea $\{T_i\}$ el conjunto de los N triángulos de la escena.

$$E(\widehat{L}(T_i)) = \frac{1}{A(T_i)} \int_{p \in T_i} L(p) dA(p) \tag{3.6}$$

(Notaremos la radiancia con el símbolo L, y los estimadores de radiancia $\widehat{L}$). El triángulo T_i tiene un área $A(T_i)$. El estimador de radiancia en cada vértice se calcula promediando los triángulos que contienen el vértice. Si un vértice V pertenece a distintos triángulos T_i, entonces

$$\widehat{L}(V) = \frac{1}{n} \sum_{i=1}^{n} \widehat{L}(T_i) \tag{3.7}$$

Para calcular la radiancia dentro de un triángulo se hace una interpolación trilineal de los valores de radiancia que se han calculado en los vértices, con objeto de que la división en triángulos no sea visible. Formalmente, en un punto p con coordenadas baricéntricas p_0, p_1, p_2, perteneciente a un triángulo con vértices

v_0, v_1, v_2, el estimador es

$$\widehat{L}(p) = \sum_{i=0}^{2} p_i \widehat{L}(v_i) \tag{3.8}$$

La diferencia entre la radiancia en un punto y el valor esperado de la radiancia en el punto es el sesgo de la Cuenta de Impactos:

$$Bias_p = L(p) - E(\widehat{L}(p)) \tag{3.9}$$

donde $Bias_p$ es el sesgo de un punto dado p.

3.4.2. Varianza

Usaremos la cota de la varianza de algoritmos de radiosidad de Monte Carlo descrita por Shirley (sección 3.3) y aplicaremos los resultados a la Cuenta de Impactos. (en Cuenta de Impactos, cada triángulo es un parche de radiancia). Suponemos que existe una radiancia máxima L_{max}, y que el cociente entre las áreas del triángulo mayor y menor es K. La varianza en un triángulo está acotada por

$$var(\widehat{L}(T_i)) \leq \frac{NR_{max}L_{max}\phi_T^2}{(1-R_{max})\pi KAn_R} \tag{3.10}$$

donde N es el número de triángulos, R_{max} es la reflectividad máxima de la escena, ϕ_T es la potencia total, A es el área de la escena y n_R es el número de rayos. El estimador de radiancia de un vértice V_j es el promedio de los estimadores de los n_{v_j} triángulos que contienen el vértice, así que la varianza se reduce.

$$var(\widehat{L}(V_j)) \leq \frac{NR_{max}L_{max}\phi_T^2}{\pi(1-R_{max})n_{v_j}KAn_R} \tag{3.11}$$

El estimador final para un punto es el promedio de la radiancia de los vértices del triángulo. En el baricentro del triángulo, el área que se usa es el de todos los triángulos que lo rodean. El triángulo se usa tres veces, los triángulos que comparten una arista con el triángulo se usan dos veces, y el resto de los triángulos una única vez. El resultado para los vértices v_a, v_b, v_c es $n_{v_a} + n_{v_b} + n_{v_c} - 5$. La varianza es

$$var(\widehat{L}(V_j)) \leq \frac{NR_{max}L_{max}\phi_T^2}{(1-R_{max})\pi(n_{v_a} + n_{v_b} + n_{v_c} - 5)KAn_R} \tag{3.12}$$

La varianza en otros puntos del triángulo depende de las coordenadas del punto y cambia de forma suave de la ecuación 3.11 a la ecuación 3.12.

Si suponemos una distribución uniforme de rayos, podemos calcular una aproximación a la varianza en un triángulo. Sea A_T el área del triángulo, y s el número de triángulos a los que pertenece cada vértice. Como la contribucion de los impactos se suma en los triángulos y más tarde a los vértices se les asigna

una radiancia que es el promedio de la de los triángulos a los que pertenecen, el área usada es:

$$A_{used} \approx A_T(3s-5) \tag{3.13}$$

(suponemos por simplicidad que triángulos cercanos tienen un área similar en la triangulación). Si el resto de factores es el mismo, nuestro estimador de la varianza en el triángulo es

$$Var(\widehat{L}) \approx c_1 \times \frac{\phi_T^2}{n_R A_T(3s-5)} \tag{3.14}$$

con c_1 la constante que tiene en cuenta el resto de los efectos. (este resultado se obtiene de cambiar el área usada en la ecuación 3.5). Podemos observar que un mallado en el que los vértices se comparten por más triángulos produce una reducción de la varianza. Esta reducción está acompañada desde luego por un aumento del sesgo, debido a que se usa un área mayor para promediar. Resumiendo, podemos destacar que la varianza es inversamente proporcional al número de rayos y al área del triángulo.

3.4.3. Complejidad

El coste del algoritmo puede dividirse en las siguientes fases:

- Acumulación de la energía en los triángulos (una suma para cada rayo): $O(n_R)$
- Cálculo de la densidad de energía en cada triángulo: $O(N)$
- Interpolación de los vértices: $O(n_V\ s)$
- Interpolación dentro de los triángulos: $O(n_P)$

donde n_V es el número de vértices, y n_P el número de muestras de irradiancia. El coste total es $O(n_R + n_V\ s + n_P)$. Como el número de vértices es proporcional al número de triángulos ($n_V \propto N$), el coste puede también describirse como $O(n_R + N\ s + n_P)$ cuando esta formulación sea más útil.

3.5. Photon Maps

En la sección anterior vimos cómo el algoritmo de Cuenta de Impactos tiene una fase de fotosimulación seguida de una fase de estimación de densidades. La limitación del algoritmo es que sólo se pueden generar estimadores de la radiancia interpolados dentro de un triángulo; por el contrario, Photon Maps puede estimar la radiancia en cualquier punto de una superficie, aunque a costa de almacenar todos los fotones en una estructura de datos kd-tree (sección 1.4.3).

Photon Maps calcula la irradiancia en un punto p buscando los k impactos de fotones más cercanos, añadiendo la energía de los primeros $k-1$, y dividiendo por el área de un círculo con radio la distancia al impacto k-ésimo. Sin embargo,

si los fotones no se encuentran en el mismo plano que el punto, o si el punto está cerca de un borde y no hay superficie para que los fotones impacten del otro lado del borde, se introduce sesgo.

3.5.1. Convergencia de Photon Maps

Comenzaremos demostrando que Photon Maps converge para una función de radiancia constante en un disco; tras ello se indica la demostración general. Sea v el valor constante de la radiancia entrante por unidad de área en el disco unidad. La fase de fotosimulación impactará n rayos en el disco con una distribución uniforme, ya que la radiancia entrante es constante. El área del disco unidad es π, y el conjunto de impactos tiene una función de distribución

$$\overline{f}(x) = \begin{cases} \frac{1}{\pi} & \text{si } |x-p| \leq 1 \\ 0 & \text{en otro caso} \end{cases} \tag{3.15}$$

donde p es el centro del disco.

La función de distribución de la distancia al centro se puede calcular viendo que la medida de los puntos a una distancia d es la longitud de la circunferencia de radio d:

$$f(x) = a2\pi x \tag{3.16}$$

donde a es una constante para obligar a que f tenga integral 1 en el rango de distancias válidas [0,1]. Resolviendo queda $a\pi = 1$, así que

$$f(x) = 2x \tag{3.17}$$

La distribución acumulada es

$$F(x) = \int_0^x f(x')dx' = x^2 \tag{3.18}$$

Usando los conceptos de la sección 1.8.4, la distribución del estadístico de orden k (la distancia del impacto buscado por Photon Maps) es

$$f_{x_{(n)}}(x) = \frac{n!}{(k-1)!(n-k)!} F^{k-1}(x)(1-F(x))^{n-k} f(x) \tag{3.19}$$

El estimador de Photon Maps es

$$g(R) = \frac{(k-1)\phi}{\pi R^2} \tag{3.20}$$

donde R es la distancia del k-ésimo impacto más cercano y $\phi = \pi v/n$ es la energía de un fotón (el π es el área del disco).

La esperanza del estimador de Photon Maps es

$$E(g(R)) = \int_0^1 g(R) f_{x_{(n)}}(R)dR = v \tag{3.21}$$

La demostración general puede describirse de la siguiente forma: Sea v el valor de la radiancia en el punto de interés P (supongamos radiancia contínua en P). Si $v > 0$, existe una ϵ-bola donde todos los puntos tienen radiancia positiva. Conforme n diverge, el número esperado de impactos en la ϵ-bola diverge. Por tanto, $f_{x_{(n)}}$ tiende a una δ de Dirac con probabilidad uno. Tomando el límite cuando $\epsilon \to 0$, podemos considerar un diferencial de área alreadedor de P, donde la radiancia es constante, y usar el razonamiento anterior para demostrar que Photon Maps converge a v (el requerimiento de la existencia de la bola significa que los puntos en un borde puede que no converjan).

La descripción original de Photon Maps [Jensen 95] (y todas las referencias relacionadas conocidas por nosotros) indican que se debe usar la energía de los k impactos más cercanos. Sin embargo, en ese caso el estimador tiene un sesgo, ya que se produce una sobreestimación de $\frac{k}{k-1}$. En Photon maps, $k = 20$, así que el sesgo es del 5 % (también se realizó un test empírico para confirmar el resultado teórico). En algunos algoritmos, los resultados se normalizan dividiendo por el valor máximo obtenido, así que esto no es un problema. Sin embargo, dado que la fase de *final gathering* de Photon Maps combina distintos estimadores para la iluminación directa e indirecta [Jensen 01], este sesgo debería ser tenido en cuenta. También hay que tener cuidado al desarrollar nuevos algoritmos basados en Photon Maps.

3.5.2. Sesgo

Las fuentes de sesgo para Photon Maps pueden clasificarse en cuatro tipos: de borde, de proximidad, sobreestimación y topológico. Estudiaremos de forma secuencial los distintos tipos de sesgo.

Photon Maps calcula la radiancia promedio en una semiesfera que rodea al punto. Cuando el punto está en el extremo de una superficie el valor calculado tiene sesgo de borde [Havran 05a]. Photon Maps estima la densidad de energía sumando la energía de los impactos y dividiendo por el área del círculo. El resultado de esto es un estimador de la radiancia mucho menor que el correcto cerca de los bordes, ya que hay una zona inalcanzable. Hay dos soluciones para este problema. Una es tener en cuenta los rayos que no intersecan ninguna geometría pero intersecan el disco (el método de Estimación de Densidades en el Plano Tangente, que estudiaremos en la sección 1.4.4). La otra es dividir por el área de la zona alcanzable, como hacen Hey y Purgathofer [Hey 02]. En este caso se usa información de geometría en las cercanías del punto para evitar el sesgo de borde de Photon Maps mencionadao en esta sección, dividiendo por un área corregida. El área se estima usando *geometry feelers* para encontrar zonas inalcanzables.

El sesgo que se produce si no se implementa ninguna de las soluciones mencionadas es la subestimación de la radiancia por un factor que es la fracción de área alcanzable, tras proyectarla en el plano tangente. Conforme el número de impactos diverge, este sesgo tiende a cero en los puntos que no están situados en un borde.

El segundo sesgo a tener en cuenta es el sesgo de proximidad, que se refiere al

hecho de que el estimador tiende a la radiancia promedio en el círculo centrado en el punto, con radio la distancia al k-ésimo impacto. El sesgo de sobreestimación aparece si se añade por error la energía del k-ésimo fotón. Finalmente, el sesgo topológico se refiere al hecho de que si la superficie que contiene al punto no es plana, el estimador del área no será correcto.

3.5.3. Varianza

En general, la varianza de Photon Maps en el punto P es

$$Var(g_p(R)) = E(g_p^2(R)) - E^2(g_p(R)) \tag{3.22}$$

donde g_p se refiere al estimador en un punto dado p (la definición es la de g en la sección 3.5.1 pero la distribución de probabilidad de la distancia a los impactos depende del punto, así que lo indicamos explícitamente), mientras que el error cuadrático medio para un conjunto de muestras de irradiancia $\{P_i\}$ es

$$Error = \sum_i \left(L(P_i) - E(g_{P_i}(R))\right)^2 \tag{3.23}$$

y estas fórmulas no pueden simplificarse sin información adicional.

Es posible obtener una cota de la varianza usando estadísticos de orden. En general, la varianza de un estadístico de orden está acotado por la siguiente fórmula ([Papadatos 95]):

$$Var(X_{k:n}) \leq \sigma_n^2(k)\sigma^2 \tag{3.24}$$

donde $X_{k:n}$ es el estadístico de orden k de n variables aleatorias independientes igualmente distribuidas $X_1,\ldots,X_n$ con variancia σ^2, y σ_n^2 se define como

$$\sigma_n^2(k) = \sup_{0<x<1}\left(\frac{I_x(k, n+1-k)\cdot(1 - I_x(k, n+1-k))}{x(1-x)}\right) \tag{3.25}$$

I_x es la función beta incompleta

$$I_x(a,b) = \frac{1}{B(a,b)}\int_0^x u^{a-1}(1-u)^{b-1}du \tag{3.26}$$

La varianza σ^2 de la distribución subyacente de impactos está acotada por la ecuación 3.4 de la sección 3.3. Sustituyendo la ecuación 3.24 en la ecuación 3.4 obtenemos

$$Var(\widehat{L}) \leq \sigma_{n_R}^2(k)\frac{R_{max}}{(1-R_{max})\pi A}L_{max}\frac{\phi_T^2}{n_R} \tag{3.27}$$

En el caso de Photon Maps, k es 20 y n_R es varias órdenes de magnitud más grande que k. En estas circunstancias, $\sigma_{n_R}^2(k)$ tiene un valor muy pequeño. Conforme el número de rayos aumenta, $\sigma_{n_R}^2(k)$ disminuye. Cuando el número de rayos diverge, $\sigma_{n_R}^2(k)$ tiende a 1.

Si comparamos la ecuación 3.27 con la correspondiente de la Cuenta de Impactos (ecuación 3.12), podemos observar que el método de Photon Maps será mejor si

$$\frac{N}{(n_{v_a} + n_{v_b} + n_{v_c} - 5)K} > \sigma^2_{n_R}(k) \tag{3.28}$$

En escenas donde los triángulos sean de tamaños similares ($K \approx 1$), podemos ver que conforme el número de rayos aumenta, el método de Photon Maps tiene una reducción en varianza con respecto a la Cuenta de Impactos.

La cota descrita en la ecuación 3.27 está en muchos casos lejos del valor real. Si asumimos una distribución uniforme de n_R rayos, se pueden obtener algunas aproximaciones útiles de la variancia. Photon Maps busca los k fotones más cercanos en una esfera con radio creciente. Denotaremos aquí r_{PM} la distancia al impacto k-ésimo. V_{PM} es el volumen de la esfera correspondiente. V_0 y r_0 corresponden a la esfera envolvente de la escena (sección 3.3). Dado que

$$k = n_R \times \frac{V_{PM}}{V_0} = n_R \times \frac{r^3_{PM}}{r^3_0} \tag{3.29}$$

(como los impactos están distribuidos uniformemente, la fracción de impactos es el cociente entre los volúmenes) el radio r_{PM} de una esfera S_{PM} con k fotones es:

$$r_{PM} = \frac{\sqrt[3]{k} r_0}{\sqrt[3]{n_R}} \tag{3.30}$$

El área del círculo máximo de la esfera es

$$A = \pi r^2_{PM} = \frac{\pi \sqrt[3]{k^2} r^2_0}{\sqrt[3]{n^2_R}} \tag{3.31}$$

El estimador de variancia (usamos la ecuación 3.5) es

$$Var(\widehat{L}) \approx \frac{c_2 \sqrt[3]{n^2_R}}{\pi \sqrt[3]{k^2} r^2_0 n_R} = \frac{c_2}{\pi r^2_0 \sqrt[3]{n_R k^2}} \tag{3.32}$$

Los puntos afectados por la correción en el algoritmo desarrollado por Hey et al mencionado en la sección anterior tienen un incremento de varianza que es el cuadrado del cociente entre las áreas corregidas y sin corregir.

3.5.4. Complejidad

La complejidad de Photon Maps depende del coste de encontrar los fotones en la estructura de indexación espacial que los contiene (el kd-tree). El procedimiento se llama formalmente una búsqueda del vecino más cercano, y puede realizarse en tiempo logarítmico [Bentley 75]. Llamaremos t al coste de subir o bajar por la estructura un único paso. Como el árbol está balanceado, el coste del algoritmo de Photon Maps es como máximo $t\ k\ \log_2(n_R)$.

En la práctica, desplazarse a los nodos vecinos puede realizarse en tiempo constante. Si el árbol está densamente poblado, hay $2^{3z} - 1$ vecinos a una distancia z, que se pueden encontrar dando $2z$ pasos. Para k impactos, la distancia será $z = \log_2(k)/3$. Por tanto, el costo de una muestra de irradiancia es

$$T_1 = t(\lceil \log_2(n_R) \rceil + 2(k-1)\lceil \log_2(k-1)/3) \rceil \tag{3.33}$$

Para n_P muestras de irradiancia procesadas de forma iterativa, el coste es

$$T = n_P t(\lceil \log_2(n_R) \rceil + 2(k-1)\lceil \log_2(k-1)/3) \rceil \tag{3.34}$$

La complejidad es $O(n_P(\log n_R + k \log k))$.

3.6. Algoritmos basados en las trayectorias de los rayos

Los algoritmos explicados en las secciones anteriores usan los impactos de los rayos para estimar la radiancia, y descartan las trayectorias de los fotones para que el algoritmo sea más rápido y tenga menos coste en memoria. Sin embargo, las trayectorias dan información útil acerca de la distribución de la energía y pueden reducir la varianza de un algoritmo que las tenga en cuenta, especialmente en el caso de escenas complejas, que pueden tener objetos pequeños o de gran curvatura. La desventaja es que usar las trayectorias es algo más costoso en tiempo y en espacio.

Se han publicado dos algoritmos principales usando este esquema: Estimación de Densidades en el Plano Tangente y Ray Maps. La sección siguiente resume el método de Estimación de Densidades en el Plano Tangente (descrito en la sección 1.4.4) y explica su sesgo y su varianza. La sección 3.6.2 resume Ray Maps (sección 1.4.5) y detalla el sesgo, la varianza y la eficiencia de esta técnica. Secciones posteriores describen formas eficientes de implementar la Estimación de Densidades en el Plano Tangente y estudian la complejidad de cada algoritmo.

3.6.1. Estimación de Densidades en el Plano Tangente

La Estimación de Densidades en el Plano Tangente es una técnica que también se basa en usar los fotones generados por una fase de fotosimulación pero que añade el concepto de usar las trayectorias de los fotones cerca de los puntos de muestra, en lugar de usar los impactos de los fotones en las superficie como en la técnica de Photon Maps. Esto es ventajoso cuando existen pocos impactos cerca de las muestras de irradiancia, por ejemplo en escenas con superficies pequeñas. El esquema del método es el siguiente:

Se crea un disco centrado en el punto en el que la irradiancia debe calcularse y orientado de forma que sea tangente a la superficie; la energía de los rayos que intersecan al disco se suma y se calcula la densidad de energía en el disco,

que se usa como estimador de radiancia en el punto. En las superficies pequeñas puede apreciarse una gran reducción de la varianza.

El algoritmo es muy adecuado en escenas de geometría compleja (con muchos objetos de gran curvatura, distintos tamaños o bordes afilados). El algoritmo examina las trayectorias de los rayos cerca de los puntos donde se muestrea la irradiancia, y a pesar de que esto es más costoso que las búsquedas de impactos de otras técnicas, la disminución del error compensa el aumento del tiempo.

Sin embargo, en escenas más simples, con regiones grandes de baja curvatura y pocos bordes afilados, métodos más simples obtienen una solución similar (ligeramente peor) con menor complejidad computacional.

Para que el algoritmo fuera aplicable a un mayor rango de situaciones, se han presentado distintas optimizaciones para diminuir la complejidad del algoritmo usando indexación espacial, tarjetas graficas, reuso de datos de iluminación, etc.

Convergencia, Sesgo y Variancia

El estimador de radiancia para este método converge a la densidad de energía en el disco conforme el número de rayos diverge. Como este valor es posiblemente distinto del valor en el centro del disco, el algoritmo tiene sesgo de proximidad. Este sesgo tiende a cero cuando el tamaño del disco tiende a cero.

Cuando se permite que el radio del disco aumente al estilo de Photon Maps usando radio variable (por ejemplo cuando se encuentran pocos fotones), hay que tener cuidado para evitar el sesgo de sobreestimación mencionado en el apartado de Photon Maps.

Para discos de tamaño fijo, podemos usar la ecuación 3.4 para dar una cota de la varianza, ya que el núcleo de la estimación de densidades es fijo. Sea d el radio del disco. El área del disco es πd^2. La cota de la varianza es

$$Var(\widehat{L}) \leq \frac{R_{max}}{(1 - R_{max})\pi^2 d^2} L_{max} \frac{\phi_T^2}{n_R} \tag{3.35}$$

El estimador de la varianza para una distribución uniforme de rayos depende del tamaño de los discos.

$$A = A_d = \pi d^2 \tag{3.36}$$

El estimador de la varianza es

$$\frac{Var(\widehat{L}) \approx c_2}{\pi d^2 n_R} \tag{3.37}$$

donde c_2 de nuevo tiene en cuenta el resto de los efectos.

Podemos aprovechar que los discos son conjuntos convexos para obtener un estimador más preciso del sesgo, la varianza y el error relativo, suponiendo una distribución uniforme de rayos. Comprobar si un rayo interseca con un disco puede verse como un ensayo de Bernoulli. Definamos una distribución binomial, $Ray(X)$, con el número de rayos que intersecan el disco centrado en el punto:

$$Ray(X) \rightsquigarrow B(n_R, p_1) \tag{3.38}$$

donde p_1 es la probabilidad de intersección de un único rayo, que se puede calcular de acuerdo con la sección 1.8.7 (intersección de líneas y conjuntos convexos en el espacio 3D) ya que el rayo puede verse como una línea recta y el disco es un conjunto convexo.

$$p_1 = \frac{A_d}{A_0} \tag{3.39}$$

donde A_0 es el área de la envolvente convexa de la escena.

Siguiendo la sección 1.8.3 (donde se explica la distribución binomial y sus propiedades), la esperanza de $Ray(X)$ es

$$E(Ray(X)) = n_R\ p_1 = n_R \frac{A_d}{A_0} \tag{3.40}$$

y la varianza es

$$Var(Ray) = n_R\ p_1\ (1 - p_1) = n_R \frac{A_d}{A_0}\left(1 - \frac{A_d}{A_0}\right) \tag{3.41}$$

Sea $\phi = \frac{\Phi_T}{n_R}$ la energía de cada rayo. La distribución $Rad(X) = \frac{\phi Ray(X)}{A_d}$ obtenida como un escalado de $Ray(X)$ es el estimador de la radiancia en el punto de acuerdo con el método DETP. Su esperanza es

$$E(\widehat{L}) = \frac{\phi E(Ray)}{A_d} = \frac{\phi n_R}{A_0} \tag{3.42}$$

y su varianza es

$$Var(\widehat{L}) = \frac{\phi^2}{A_d^2} Var(Ray) = \frac{\phi^2}{A_d^2} n_R \frac{A_d}{A_0}\left(1 - \frac{A_d}{A_0}\right) = \frac{\phi^2 n_R}{A_d A_0}\left(1 - \frac{A_d}{A_0}\right) \tag{3.43}$$

Simplificando (expandiendo ϕ) queda:

$$Var(\widehat{L}) = \frac{\phi_T^2}{A_d A_0 n_R}\left(1 - \frac{A_d}{A_0}\right) \tag{3.44}$$

La varianza de DETP no es exactamente inversamente proporcional al área del disco (es algo menor debido al factor $1 - \frac{A_d}{A_0}$), aunque para valores pequeños del área del disco, que son los más usuales en iluminación global para evitar sesgo, el factor es muy cercano a la identidad.

La desviación típica es la raiz cuadrada de la varianza:

$$\sigma = \sqrt{\frac{\phi_T^2}{A_d A_0 n_R}\left(1 - \frac{A_d}{A_0}\right)} \tag{3.45}$$

Finalmente podemos calcular el error relativo medio dividiendo la desviación típica por la esperanza:

$$Error = \frac{\sigma}{E(\widehat{L})} = \frac{\sqrt{\frac{\phi_T^2}{A_d A_0 n_R}\left(1 - \frac{A_d}{A_0}\right)}}{\frac{\phi n_R}{A_0}} = \sqrt{\frac{A_0 - A_d}{A_d n_R}} \tag{3.46}$$

DETP tiene un trade-off óptimo entre exactitud y varianza cuando el radio del disco (que es una constante definida por el usuario d) es aproximadamente la distancia entre los puntos d_P en que se muestrea la irradiancia, cuando los puntos de muestreo están predefinidos. La distancia promedio entre muestras limita el tamaño de las características de la iluminación que pueden ser reconstruidas. Los valores de iluminación entre estas muestras se crearán usando interpolación, así que las características más pequeñas que esta distancia se perderán.

Formalmente, el teorema de muestreo de Nyquist–Shannon (sección 1.8.9) demuestra que una señal puede reconstruirse muestreando al doble de su frecuencia máxima. Al aplicar este teorema al dominio espacial, muestrear a una distancia d_P significa que las características de tamaño menor que $2d_P$ se distorsionarán o se perderán.

La varianza de DETP depende del número de rayos n_R y el área del disco A_d (ecuación 3.35 de la página anterior). Véase la figura 3.1, izquierda. Hay

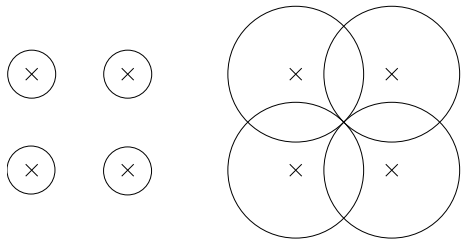


Figura 3.1: Muestras en un grid y discos asociados para dos radios diferentes.

espacio libre fuera de los discos, así que los rayos que atraviesan ese espacio no se están usando. El radio del disco puede incrementarse al mismo tiempo que se disminuye el número de rayos para que la varianza se mantenga constante, obteniéndose un descenso en el costo del algoritmo ya que hay que generar menos rayos (figura 3.1, derecha).

El suavizado dentro del disco corresponde a un kernel de radio d. Mientras que el radio del disco sea menor que la distancia entre muestras, la información perdida por este suavizado se perdería de todas formas debido a la interpolación entre muestras consecuencia del teorema de Nyquist-Shannon.

El radio del disco no debería ser mayor que la distancia entre muestras, porque en ese caso el promedio de la energía para todos los fotones del disco elimina detalles que se podrían reconstruir en caso contrario, ya que este suavizado comienza a dominar. Por tanto, el valor óptimo de d es d_P.

Complejidad

Para calcular el tiempo que se necesita para la estimación de densidades en n_P muestras de irradiancia, sin usar optimizaciones, intersecamos el disco centrado en cada punto con los n_R rayos. Por tanto, el tiempo es $n_P\ n_R\ u$, donde u es el tiempo de intersección rayo-disco.

La eficiencia de DETP básica claramente es $O(n_R n_P)$. Como este coste es excesivamente alto, se han desarrollado distintas optimizaciones para conseguir algoritmos eficientes. Estudiaremos en la sección 3.8 la Caché de Esferas y en la sección 3.11 la Indexación de Discos.

3.6.2. Ray Maps

El algoritmo de Ray Maps, descrito en la sección 1.4.5 (página 33) almacena las trayectorias de los fotones en un kd-tree construido bajo demanda. Ray Maps permite que el usuario elija diferentes criterios para el dominio de la intersección (disco, hemisferio, esfera, caja englobante) y para los criterios de búsqueda de vecino más cercano (distancia de la intersección rayo-plano tangente, distancia al segmento del rayo, distancia a la línea soporte del rayo). La varianza de cada selección posible es diferente. La enumeración de la varianza de las diferentes posibilidades se encuentra fuera del ámbito de esta memoria; sin embargo el lector interesado puede usar las fórmulas descritas en las secciones anteriores para derivar la varianza de su combinación de parámetros preferida. Como ejemplo, usar un disco como dominio de intersección junto con la distancia a la intersección del rayo corresponde con DETP (sección anterior), mientras que usar el hemisferio obtiene una solución parecida a la de Photon Maps (sección 3.5).

Número de rayos en cada nodo del kd-tree

El procedimiento descrito antes puede ser usado para estudiar las características de los Ray Maps de Havran. Primeramente, se puede estimar el número de rayos en cada nodo del kd-tree. Si hay n_R rayos distribuidos uniformemente, dividir la superficie de los nodos por la superficie del nodo raíz da la fracción de los rayos. Si suponemos que el kd-tree tiene una razón de aspecto de 1:1:1, podemos obtener fórmulas razonablemente simples. A profundidad k, el número de rayos es:

$$Rays(k) = \begin{cases} \left(\frac{1}{4}\right)^{k/3\rfloor} n_R & \text{if } k \equiv 0 \text{ (modulo 3)} \\ \left(\frac{1}{4}\right)^{k/3\rfloor} \frac{2}{3} n_R & \text{if } k \equiv 1 \text{ (modulo 3)} \\ \left(\frac{1}{4}\right)^{k/3\rfloor} \frac{5}{12} n_R & \text{if } k \equiv 2 \text{ (modulo 3)} \end{cases} \tag{3.47}$$

Con esto, se puede demostrar que los tres criterios de subdivisión de un nodo del kd-tree de Ray Maps son equivalentes.

Los criterios de subdivisión son:

- Rays (k) $> c_R$
- Diagonal (k) $> c_d$ * Diagonal (0)
- k $< c_k$

donde c_R es una constante con el mínimo número de rayos en un voxel, c_d es otra constante con el cociente de las diagonales del voxel mínimo y el primer

voxel, c_k es la profundidad máxima, y Diagonal(k) es la longitud de la diagonal de una celda a profundidad k. [Havran 05a] usa $c_R = 32$, $c_d = 0,1\,\%$ y $c_k = 30$.

La ecuación 3.47 puede usarse para cambiar entre el primer y el tercer criterio de terminación: $c_R = Rays(c_k)$. Si expresamos ahora la diagonal como una función de la profundidad:

$$diagonal(k) = \begin{cases} \left(\frac{1}{2}\right)^{k/3\rfloor} \sqrt{3} & \text{if } k \equiv 0 \text{ (modulo 3)} \\ \left(\frac{1}{2}\right)^{k/3\rfloor} \frac{3}{2} & \text{if } k \equiv 1 \text{ (modulo 3)} \\ \left(\frac{1}{2}\right)^{k/3\rfloor} \sqrt{\frac{3}{2}} & \text{if } k \equiv 2 \text{ (modulo 3)} \end{cases} \tag{3.48}$$

el segundo y el tercer criterio de subdivisión pueden ser igualados. Con esto se demuestra que los tres criterios son equivalentes.

Complejidad

Con estas fórmulas, el coste de Ray Maps puede ser estimado. Si se usa Ray Maps en modo Photon Maps, las fórmulas describen el número de rayos en cada nodo. El número de nodos que hay que atravesar puede encontrarse calculando $\frac{n}{Rays(k)}$, donde notamos n el número de fotones a buscar. Tras ello, los razonamientos de la sección 3.5.4 pueden usarse para calcular el coste.

Si usamos Ray Maps en modo DETP, en cada punto de irradiancia hay que crear un disco de radio d y buscar los vóxeles que intersecan el disco. Tras ello, se interseca el disco con los rayos contenidos en los vóxeles. Igual que vimos en indexación de discos, la profundidad máxima del árbol debe ser tal que el tamaño de un voxel sea del mismo tamaño aproximado que el radio del disco. Así el número de voxels está acotado entre uno y ocho. El número de rayos en un voxel de ese tamaño (R_k) se puede calcular de forma análoga a la caché de esferas (ahora es el cociente entre las superficies de los voxeles y el árbol completo). Sea a la arista del cubo que engloba la escena.

$$R_k = n_R * \frac{d^2}{a^2}$$

El coste de intersecar los discos con un voxel es $n_P\ u\ R_k$. A esto hay que añadir el coste $O(k)$ de atravesar el árbol para cada punto de irradiancia, y el porcentaje de discos que abarcan varios vóxeles:

$$n_P \left(u\ n_R \frac{d^2}{a^2} + O(k)\right) \leq T \leq 8\ n_P \left(u\ n_R \frac{d^2}{a^2} + O(k)\right)$$

donde T es el coste final (se ha expandido el valor de r_k). El algoritmo finalmente es $T = O(n_R * n_P)$ con una constante oculta $\frac{d^2}{a^2}$.

3.7. Comparación de las técnicas con respecto a su varianza

La varianza de la Cuenta de Impactos depende del tamaño de los triángulos. Conforme la complejidad de las escenas aumenta y la triangulación se vuelve más fina, los triángulos se vuelven más pequeños y la varianza de la Cuenta de Impactos aumenta con respecto a la de técnicas más avanzadas.

Photon Maps y DETP tienen la misma varianza para escenas simples (localmente planas) ya que los impactos se encuentran en el plano tangente. Sin embargo, en geometría compleja el área efectiva de Photon Maps disminuye ya que sólo encuentra los impactos en las superficies reales; no puede aprovecharse de rayos que intersecan el plano tangente fuera del borde de los objetos. Esto significa que para escenas complejas, la varianza de DETP será más pequeña en bordes y similar en puntos interiores, dando una reducción de varianza. Esto es especialment importante para escenas con muchos objetos pequeños.

También es de destacar que la varianza de Photon Maps puede disminuirse incrementando el número de fotones buscados mientras que DETP puede ser ajustada aumentando el tamaño del disco (a costa de aumentar el sesgo obtenido tanto en un método como en el otro). Se puede decir que Photon Maps promedia sobre energía constante (los fotones) mientras que DETP promedia sobre el área. Excepto por los problemas de los bordes mencionados arriba, la varianza y el sesgo deberían ser los mismos cuando los fotones y el área son los mismos.

3.8. Estimación del tiempo promedio de la Caché de Esferas

Primeramente introduciremos algunas fórmulas generales útiles en el estudio de la Caché de Esferas. Tras ello se estudia la caché de esferas con y sin ordenación de puntos. Finalmente se obtienen fórmulas simplificadas para la eficiencia.

La caché de esferas (sección 1.4.4) organiza los rayos en una lista de esferas de tamaño descendiente. Si el disco a intersecar se encuentra en la esfera más pequeña, se calculan las intersecciones rayo-disco sólamente con los rayos que se encuentran en dicha esfera. En caso contrario, se produce un fallo de caché, y se recalculan las esferas necesarias, con un nuevo centro en el punto de muestra de irradiancia.

Para la caché de esferas sin ordenación de puntos hemos de suponer una distribución uniforme de puntos para calcular la irradiancia. Para calcular la fracción promedio de rayos que intersecan una esfera, usaremos algunos resultados de geometría integral. Los conceptos necesarios fueron introducidos en la sección 1.8.7.

La probabilidad de que un rayo (una línea) que interseca a una esfera K_0

interseque también a una esfera interior K_i es:

$$p(L_0 \cap K_i \neq \emptyset) \;=\; \frac{F_i}{F_0} \;=\; \frac{4\pi r_i^2}{4\pi r_0^2} \;=\; \frac{r_i^2}{r_0^2} \tag{3.49}$$

(recordemos que F denota el área y r el radio).

Si la distribución de rayos es uniforme, el número de rayos que interseca la esfera interior es independiente de la posición de la misma, y es proporcional al cociente del cuadrado del radio de las esferas. Recordemos que n_R es el número de rayos. Llamaremos S_0 a la primera esfera que engloba la escena y contiene todos los rayos. A partir de esta esfera se van construyendo esferas S_i con un radio cada vez menor. El radio de cada esfera se calcula multiplicando el radio de la esfera anterior por el factor de radios Q, que es un número real, parámetro del algoritmo de estimación de densidades, con $0 < Q < 1$.

El número de rayos que intersecan S_i es entonces:

$$n_i \;=\; n_R \frac{r_i^2}{r_0^2} \;=\; n_R Q^{2i} \tag{3.50}$$

El costo de recalcular una esfera S_i una vez es el número de rayos en la esfera que la rodea multiplicado por el tiempo de intersección rayo-esfera:

$$t_i = t * n_{i-1} \tag{3.51}$$

donde t es el tiempo de intersección rayo-esfera. La esfera 0 nunca se recalcula, porque todos los discos están contenidos en ella. En las siguientes dos secciones se hace un analisis separado de la caché de esferas básica y de la caché de esferas con ordenación de puntos.

3.8.1. Tiempo promedio usando la caché de esferas sin ordenación de puntos

Como no hay ordenación de puntos, supondremos que los vértices siguen una distribución aleatoria uniforme en el espacio. En este caso, la probabilidad de que una esfera no sea recalculada es la probabilidad de que un vértice nuevo esté dentro de la esfera antigua, que es el volumen de la esfera dividido por el volumen de la escena. S_0 es un volumen envolvente útil para la escena. Matemáticamente, $Scene_V \leq V_0$. Para n_V vértices, una esfera S_i se recalcula un número medio m_i de veces, con

$$m_i = n_V \left[1 - \frac{V_i}{Scene_V}\right] \leq n_V \left[1 - \frac{V_i}{V_0}\right] = n_V[1 - Q^{3i}] \tag{3.52}$$

Ahora necesitamos saber cuántas esferas hay. Las esferas se crean con un radio decreciente, hasta que el radio de la esfera está justo por encima del radio del disco. (i.e. la siguiente esfera tendría un radio menor que el radio del disco).

Sea k el número de esferas. Para calcular k, usaremos d como el radio del disco. k debería cumplir las siguientes ecuaciones:

$$r_k = Q^k r_0 \geq d \tag{3.53}$$

$$r_{k+1} < d \tag{3.54}$$

Por tanto k puede calcularse como:

$$k = \left\lfloor \log_Q \left(\frac{d}{r_0} \right) \right\rfloor \tag{3.55}$$

El coste T_R de recalcular las esferas debido a fallos de caché es

$$T_R = \sum_{i=1}^{k} m_i * t_i \tag{3.56}$$

Un límite superior más simple puede calcularse expandiendo la ecuación 3.56:

$$T_R = \sum_{i=1}^{k} n_V \left[1 - \frac{V_i}{V_0} \right] [t * n_{i-1}] =$$

$$n_V t \sum_{i=1}^{k} \left[1 - \frac{V_i}{V_0} \right] n_{i-1} =$$

$$n_R n_V t \sum_{i=1}^{k} \left[1 - \frac{V_i}{V_0} \right] Q^{2(i-1)} =$$

$$n_R n_V t \sum_{i=1}^{k} (1 - Q^{3i}) Q^{2(i-1)} \tag{3.57}$$

Como

$$(1 - Q^{3i}) Q^{2(i-1)} = Q^{2i-2} - Q^{3i+2i-2} =$$

$$Q^{2i-2} - Q^{5i-2} = \frac{1}{Q^2} (Q^{2i} - Q^{5i}) \tag{3.58}$$

y

$$\sum_{i=1}^{k} (Q^{2i} - Q^{5i}) = \sum_{i=1}^{k} Q^{2i} - \sum_{i=1}^{k} Q^{5i} = \sum_{i=1}^{k} (Q^2)^i - \sum_{i=1}^{k} (Q^5)^i =$$

$$\frac{Q^2 (1 - Q^{2\lfloor \log_Q (d/r_0) \rfloor})}{1 - Q^2} - \frac{Q^5 (1 - Q^{5\lfloor \log_Q (d/r_0) \rfloor})}{1 - Q^5} \lesssim$$

$$\frac{Q^2 (1 - (d^2/r_0^2))}{1 - Q^2} - \frac{Q^5 (1 - (d^5/r_0^5))}{1 - Q^5} \tag{3.59}$$

podemos usar las ecuaciones 3.57 y 3.59, para obtener una cota de T_R:

$$T_R \leq n_R n_V t \frac{1}{Q^2} \left[\frac{Q^2(1-(d^2/r_0^2)}{1-Q^2} - \frac{Q^5(1-(d^5/r_0^5)}{1-Q^5} \right] \tag{3.60}$$

El coste de recalcular la esfera interior, T_I, es

$$T_I = u\ n_k\ n_V = u\ n_R\ Q^{2k}\ n_V = \tag{3.61}$$

$$u\ n_V\ n_R\ Q^{2\left\lfloor \log_Q\left(\frac{d}{r_0}\right)\right\rfloor} \lesssim u\ n_V\ n_R\ \frac{d^2}{r_0^2} \tag{3.62}$$

Finalmente, el coste T de usar la caché de esferas es el coste de recalcular las esferas más el tiempo de intersecar el disco con los rayos de la esfera interior, para cada vértice

$$T = T_R + T_I \tag{3.63}$$

que se puede expandir para obtener un límite superior usando las ecuaciones 3.60 y 3.61:

$$T \leq n_R n_V \left[t \frac{1}{Q^2} \left(\frac{Q^2(1-(d^2/r_0^2)}{1-Q^2} - \frac{Q^5(1-(d^5/r_0^5)}{1-Q^5} \right) + u\ \frac{d^2}{r_0^2} \right] \tag{3.64}$$

La figura 3.2 muestra que este valor es mayor que no usar ninguna optimización. (No usar optimizaciones corresponde a un plano horizontal a una altura de 1) Por tanto este método no es útil si no hay coherencia en la posición de los vértices; esto está de acuerdo con experimentos prácticos. Puede probarse trivialmente que el algoritmo es $O(n_R\ n_V)$.

Esto hace que la caché de esferas sin ordenación de puntos no obtenga mejoras en el tiempo. Se puede demostrar que el algoritmo es $O(n_R\ n_P)$, con una constante oculta ligeramente superior a la unidad.

3.8.2. Tiempo promedio usando la caché de esferas con ordenación de puntos

Partimos de la ecuación 3.50, que indica cuantos rayos atraviesan cada esfera.

$$n_i \ = \ n_R Q^{2i} \tag{3.65}$$

Recordemos (ecuación 3.51) que el coste de recalcular la esfera S_i una vez es el número de rayos en la esfera envolvente por el tiempo de intersección rayo-esfera,

$$t_i \ = \ t\ n_{i-1} \tag{3.66}$$

donde t es el tiempo de intersección rayo-esfera. Como vimos antes, la esfera 0 nunca se recalcula, porque todos los discos están contenidos dentro.

Ahora averigüemos cuántas esferas hay. Las esferas se crean con un radio decreciente hasta que el radio de la esfera está justo por encima del radio del disco (i.e. el radio de la siguiente esfera sería menor que el del disco).

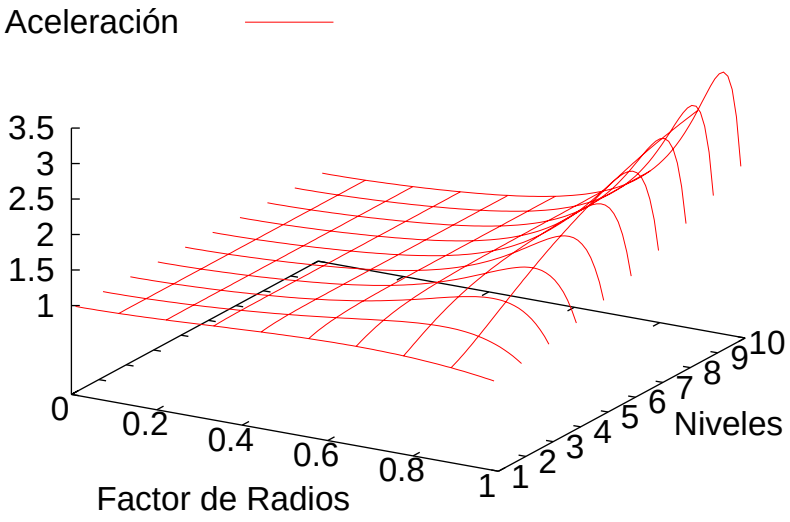


Figura 3.2: Tiempo de la caché de rayos sin ordenación de puntos en unidades de intersecciones por vértice y por rayo, en función del número de esferas y el factor de radios.

Sea k la profundidad máxima de la lista de esferas. Para calcular k, usaremos d como el radio del disco. Recordemos (sección 1.4.4) que el cociente entre radios de dos esferas adyacentes es Q, y que las esferas se construyen hasta que su radio está justo por encima del radio del disco d. k debería por tanto satisfacer las siguientes ecuaciones:

$$r_k = Q^k r_0 \geq d \quad ; \quad r_{k+1} < d \tag{3.67}$$

Por lo tanto k se puede calcular así:

$$k = \left\lfloor \log_Q \left(\frac{d}{r_0} \right) \right\rfloor \tag{3.68}$$

El coste de intersecar el disco con los rayos de la esfera interna, T_I, es:

$$T_I = un_k n_P = un_R Q^{2k} n_P = \tag{3.69}$$

$$un_P n_R Q^{2\left\lfloor \log_Q \left(\frac{d}{r_0} \right) \right\rfloor} \lesssim un_P n_R \frac{d^2}{r_0^2} \tag{3.70}$$

Dado que la caché de esferas no es útil a no ser que la localización de los puntos en que se calcula la irradiancia sean coherentes, una curva que rellena el espacio se usa para ordenar los puntos. Este algoritmo usa la curva de Lebesgue.

El algoritmo de ordenación divide el cubo en 2^{48} celdas. Cada celda tiene una coordenada X, Y y Z con 16 bits cada una. A cada punto centro de un disco se le asignan las coordenadas enteras de la celda donde está. Estas 3 coordenadas se concatenan para formar un número de 48 bits. Una función reordena los bits del número de la siguiente forma: los 3 bits menos significativos de cada coordenada corresponde con los 3 bits menos significativos del nuevo número, y se repite el proceso hasta llegar al bit más significativo. Un ejemplo con 3 bits por coordenada sería:

- Original: $x_2x_1x_0y_2y_1y_0z_2z_1z_0$
- Reordenado: $x_2y_2z_2x_1y_1z_1x_0y_0z_0$

Ahora las celdas se visitan desde la de índice 0 hasta la de índice $2^{48}-1$. Cambiar los tres bits menos significativos significa mover el objeto como máximo la diagonal de las celdas, $2*\sqrt{3}*2^{-16}$, y como mínimo la arista de las celdas, $2*2^{-16}$. Una esfera con un radio de ese orden de magnitud se recalcularía aproximadamente 2^{48} veces. Es fácil ver que una esfera de radio $2^{-m}r_0$ se recalcula 2^{3m} veces. Por tanto, ya que:

$$r_i = Q^i r_0 = 2^{-m} r_0 \tag{3.71}$$

$$m = -\log_2 Q^i \tag{3.72}$$

la esfera S_i se recalcula $m_i = 2^{3*(-\log_2 Q^i)} = Q^{-3i}$ veces.

Éste es el máximo de veces que se reconstruye la esfera, y es independiente del número de puntos en que se calcula la estimación de densidades. Hay un límite independiente del máximo número de veces, dado por el número de puntos de irradiancia. La esfera S_i se recalcula como máximo n_P veces. El coste de recálculo en este caso es:

$$T_R = \sum_{i=1}^{k} min(n_P, m_i) t_i \tag{3.73}$$

y el coste total es:

$$T = T_R + T_I \tag{3.74}$$

T puede usarse para averiguar qué Q debe usarse. Vea las figuras 3.3, 3.4 y 3.5. La figura 3.3 muestra el coste de recalcular las esferas por rayo. (i.e. 1 significa n_R intersecciones, equivalente a no optimizar), en función del radio del disco y Q. Un Q pequeño crea pocas esferas, así que el tiempo es pequeño. El radio del disco influye en la longitud de la lista de esferas, pero las esferas más pequeñas tienen muy pocos rayos y por tanto añaden poco tiempo, así que las diferencias en tiempo son pequeñas.

La figura 3.4 da una gráfica del tiempo de intersección de los discos con los rayos de la esfera interna. Este tiempo sólo depende del radio de la esfera, que depende de Q: el radio es r_k tal que $r_k = r_0 Q^k \geq d$ y $r_{k+1} < d$. La gráfica muestra que hay un infinito número de Q óptimos. Sin embargo, si la inestabilidad numérica hace que sea difícil obtener los valores mínimos, Q debería estar

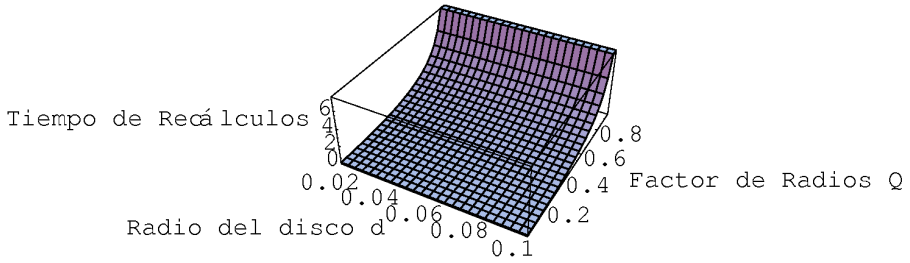


Figura 3.3: Número de recálculos de la caché de esferas en función de Q y el radio del disco.

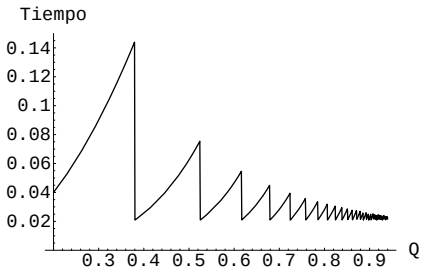


Figura 3.4: Tiempo de intersección del disco con los rayos de la esfera interna, en función de Q.

tan cerca como sea posible de 1. Veremos posteriormente que si no tenemos en cuenta el tamaño del disco, se puede obtener un óptimo para Q.

Un Q alto da esferas internas que envuelven firmemente a los discos, aunque el número de recálculos es alto. Un Q bajo hace que los discos tengan más espacio alrededor; habrá muchos rayos en la esfera interior y aumentará el tiempo de cálculo. Los picos de la gráfica corresponden a esferas internas que envuelven perfectamente a los discos, i.e., $d = r_k$. Los mínimos locales de la función tienen el mismo valor, que se corresponde con el tiempo para intersecar a los rayos en una esfera de radio d: $un_R \frac{d^2}{r_0^2}$.

La figura 3.5 muestra la integración de las dos gráficas anteriores. Tiene forma de U, como era de esperar de las gráficas componentes, y muestra cómo cambiar el radio del disco hace que el valor óptimo de Q cambie suavemente para crear una esfera interna que envuelva el disco. Cuando Q se acerca a 1, el número de esferas aumenta exponencialmente y el tiempo sube asintóticamente a infinito en $Q = 1$, que significa un número infinito de esferas.

La figura muestra que los valores de Q pertenecientes al intervalo $0,6 - 0,7$ son un buen compromiso, porque el número de recálculos está por debajo de 1 en la figura 3.3 y cerca del punto donde la pendiente se vuelve importante, y

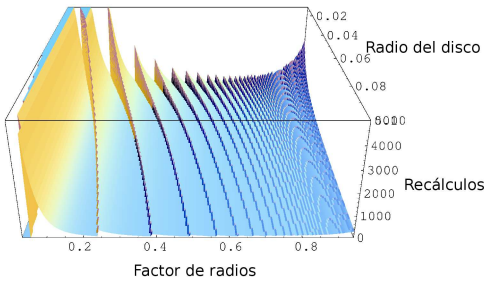


Figura 3.5: Tiempo de recálculo de la caché de esferas en función de Q y el radio del disco.

el valor en la gráfica de la figura 3.4 también es bastante bajo. El resultado es coherente con experimentos prácticos [Lastra 02c].

Se vio antes que para un n_P lo suficientemente grande, el número de recálculos es fijo y el tiempo sólo depende de n_R. El tiempo para intersecar los discos en la esfera interna, por el contrario, depende tanto de n_R como de n_P y es por tanto $O(n_R\ n_P)$. La constante oculta, $\frac{d^2}{r_0^2}$, puede hacer que este algoritmo sea bastante eficiente en la práctica.

Existe un radio de disco óptimo, que puede ser calculado a partir del número de muestras. Utilizar este valor óptimo permite simplificar las fórmulas obtenidas hasta ahora. La sección 3.6.1 demuestra que la distancia promedio entre muestras es el valor óptimo para el radio del disco.

Si el radio del disco es aproximadamente igual a la distancia entre muestras de irradiancia, se puede probar que la eficiencia del algoritmo aumenta. En lo que resta de sección, supondremos que el radio del disco es aproximadamente igual a la distancia entre muestras. Para n_P pequeño (para no llegar al límite de recálculos), ya que el radio del disco es aproximadamente la distancia entre muestras, se cumple que:

$$k \approx \log_Q \left(\frac{1}{\sqrt[3]{n_P}} \right) = \log_Q \left(n_P^{-1/3} \right) = -\frac{1}{3} \log_Q n_P = -\frac{1}{3} \frac{\log n_P}{\log Q} = c_Q \log n_P \tag{3.75}$$

donde c_Q sólo depende de Q.

Ahora, para calcular el orden de eficiencia del algoritmo, se muestra el valor de algunas cantidades importantes, y finalmente se expande el tiempo del algoritmo completo y se calcula la eficiencia. El valor de n_i, el número de rayos en la esfera i, es $n_i = n_R Q^{2i}$, como se vio en la ecuación 3.50. El tiempo para recalcular la esfera i es:

$$t_i = t\ n_R\ Q^{2i-2} \tag{3.76}$$

El número de recálculos de la esfera i:

$$m_i = Q^{-3i} \tag{3.77}$$

El tiempo para recalcular la esfera i para todo el algoritmo es:

$$m_i \; t_i = t \; n_R \; Q^{-i-2} \tag{3.78}$$

Finalmente, el tiempo del algoritmo completo debido a fallos de caché es por tanto:

$$T_R = \sum_{i=1}^{k} m_i \; t_i = \frac{t n_R}{Q^2} \left(\frac{\sqrt[3]{n_P} - 1}{1 - Q} \right) \tag{3.79}$$

y por tanto, vemos que T_R está en el orden $O(n_R \; n_P^{1/3})$.

En cuanto a T_I, supongamos que los puntos se distribuyen en un grid regular. La distancia entre los puntos es d y existen $\sqrt[3]{n_P}$ puntos a lo largo de los laterales del grid. El radio de la esfera circumscrita al grid es

$$r_0 = \frac{\sqrt{3} d \sqrt[3]{n_P}}{2} \tag{3.80}$$

Resolviendo en d:

$$d = \frac{2 r_0}{\sqrt{3} \sqrt[3]{n_P}} \tag{3.81}$$

y expandiendo d en la ecuación 3.69 resulta en

$$T_I = \frac{4}{3} \; u \; n_R \sqrt[3]{n_P} \tag{3.82}$$

El tiempo total del algoritmo es:

$$T = T_R + T_I = \frac{t n_R}{Q^2} \left(\frac{\sqrt[3]{n_P} - 1}{1 - Q} \right) + \frac{4}{3} \; u \; n_R \sqrt[3]{n_P} \tag{3.83}$$

Esta ecuación no tiene en cuenta el hecho de que los discos no están envueltos exactamente por las esferas de menor tamaño, y por tanto no presenta los picos mencionados en la sección anterior. Sin embargo, nos permite calcular teóricamente un valor óptimo de Q. Para calcular el valor óptimo de Q, se calcularán la derivada de esta función con respecto de Q:

$$\frac{\partial T}{\partial Q} = \frac{t n_R (\sqrt[3]{n_P} - 1)}{(1 - Q)^2 Q^2} - \frac{2 t n_R (\sqrt[3]{n_P} - 1)}{(1 - Q) Q^9} \tag{3.84}$$

y se resolverá la ecuación $dT/dQ = 0$, obteniendo $Q = 2/3$. El signo de la segunda derivada indica si el valor encontrado es un máximo o un mínimo.

$$\frac{\partial^2 T}{\partial Q^2} = \frac{6 t n_R (\sqrt[3]{n_P} - 1)}{(1 - Q) Q^4} - \frac{4 t n_R (\sqrt[3]{n_P} - 1)}{(1 - Q)^2 Q^9} + \frac{2 t n_R (\sqrt[3]{n_P} - 1)}{(1 - Q)^9 Q^2} \tag{3.85}$$

$$\left(\frac{\partial^2 T}{\partial Q^2} \right) \left(\frac{2}{3} \right) = \frac{729}{8} t n_R (\sqrt[3]{n_P} - 1) \tag{3.86}$$

Como t, n_R y $\sqrt[3]{n_P} - 1$ son todos valores positivos, la segunda derivada en el punto $2/3$, $(\partial^2 T / \partial Q^2)(2/3)$ también es positiva; por tanto la función tiene un mínimo en $Q = 2/3$. El resultado concuerda plenamente con los resultados anteriores y con los experimentos [Lastra 02c].

3.8.3. Estudio del límite de subdivisión

Hasta este momento se ha usado la geometría integral para calcular el número promedio de rayos. Sin embargo, la Caché de Esferas tiene un parámetro que indica que las esferas que tienen pocos rayos no deberían dividirse, ya que el coste de mantener estas esferas es superior al de intersecar directamente los discos con los rayos de la esfera padre. Este parámetro es el límite de subdivisión: el número mínimo de rayos que debe tener una esfera para dividirse. Para tener esto en cuenta, se deben usar las distribuciones de probabilidad definidas en las secciones 1.8.2 y 1.8.3 del primer capítulo (página 40).

Comprobar si un rayo interseca un conjunto convexo puede verse como un ensayo de Bernoulli. La probabilidad de que exactamente k rayos intersequen el conjunto es

$$p(n = k) = b(k; n_R, p_1) \tag{3.87}$$

donde n es el número de rayos que intersecan el conjunto, $b(k; n_R, p_1)$ es la distribución binomial, p_1 es la probabilidad de que un rayo interseque con el conjunto, y n_R es el número total de rayos. La probabilidad de que menos de r_{min} rayos intersequen el conjunto es

$$p = \sum_{k=0}^{r_{min}-1} b(k; n_R, p_1) \tag{3.88}$$

Recordemos que Q es el cociente entre el radio de dos esferas adyacentes S_i y S_{i+1}. Si la esfera S_i tiene R rayos, el número promedio de rayos de S_{i+1} será Q^2R, ya que la fracción de rayos es proporcional al cociente de las superficies, y por tanto al cuadrado de los cocientes de los radios (ecuación 3.49). Sea r_{min} el mínimo número de rayos que una esfera debe tener para que se la subdivida (en nuestros experimentos, hemos usado $r_{min} = 100$, ya que heurísticamente es un valor óptimo).

El número de esferas en cada nivel puede ser dividida en dos partes: las esferas creadas debidas a un fallo de cache en ese nivel, y las esferas creadas debido a un fallo en niveles superiores. Como los rayos se distribuyen uniformemente, comprobar si un rayo está en una esfera puede ser considerado un ensayo de Bernoulli. Para los n_R rayos, tenemos n_R ensayos de Bernoulli independendientes repetidos. La probabilidad de que una esfera en el nivel i tenga k rayos es (usando como base la ecuación 3.87)

$$P(n = k) = b(k; n_R, Q^{2i}) \tag{3.89}$$

Para saber si una esfera en el nivel i va a subdividirse, hay que calcular la probabilidad de que tenga menos de r_{min} rayos. La probabilidad es:

$$p_i = \sum_{k=0}^{r_{min}-1} b(k; n_R, Q^{2i}) = \sum_{k=0}^{r_{min}-1} \binom{n_R}{k} Q^{2ik}(1-Q)^{2i(n_R-k)} \tag{3.90}$$

Esto significa que si tenemos m_i esferas en el nivel i, existirán en promedio $m_i(1 - p_i)$ esferas con más de r_{min} rayos, cada una con un descendiente en el

nivel $i+1$. Las esferas del nivel $i+1$ son entonces $m_i(1-p_i)$ más el número de fallos de caché en el nivel $i+1$, y el procedimiento debe repetirse para el resto de los niveles. También es de destacar que la desviación estándar del número de rayos también es conocida: $\sigma = n_R Q^{2i}$. Esto significa que no sólo se pueden hacer los estudios clásicos de rendimiento en el peor caso y en el caso promedio. Es fácil también calcular el tiempo que el algoritmo tardará con un nivel de confianza dado.

El valor de la probabilidad de subdivisión es muy cercano a uno en los niveles superiores con muchos rayos ($p_i \approx 1$), y muy cercano a cero en los niveles inferiores con aproximadamente r_{max} rayos ($p_i \approx 0$). Como la forma analítica de p_i es compleja, hay aproximaciones estándar. La aproximación de Poisson puede usarse si la probabilidad del suceso de Bernoulli es menor que 0,1 (sección 1.8.2). (La probabilidad es $Q^{2i} = 0,1$, por tanto para niveles $i = \frac{\log_Q(0,1)}{2}$ e inferiores).

$$p_i \approx e^{-n_R Q^{2i}} \sum_{k=0}^{r_{min}-1} \frac{(n_R Q^{2i})^k}{k!} \tag{3.91}$$

Para los niveles superiores, se puede usar la aproximación a una distribución normal:

$$p_i \approx \Phi\left(\frac{r_{min} - 1 - n_R Q^{2i} + \frac{1}{2}}{\sqrt{n_R Q^{2i}(1-Q^{2i})}}\right) - \Phi\left(\frac{-n_R Q^{2i} - \frac{1}{2}}{\sqrt{n_R Q^{2i}(1-Q^{2i})}}\right) \tag{3.92}$$

donde Φ es la función de distribución normal ($\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-1/2y^2} dy$).

Estas aproximaciones son esenciales para evaluar las expresiones, dado que el gran número de rayos en las soluciones de Iluminación Global hacen que la evaluación de la función binomial sea intratable.

Usando el hecho de que el número de rayos en una esfera sigue una distribución binomial, como se vio antes, se puede conseguir una aproximación mejor del número de rayos promedio en una esfera que la de la ecuación 3.50. Dado que sólo las esferas con más de r_{min} rayos se subdividen, podemos calcular la esperanza del número de rayos en las esferas con más de r_{min} rayos usando la definición de probabilidad condicionada.

La probabilidad de que una esfera tenga menos de r_{min} rayos es p_i (definida en la sección anterior). La probabilidad de que tenga k rayos, condicionada a que tenga más de r_{min} rayos, es

$$P(r = k | r \geq r_{min}) = \frac{P(r = k \cap r > r_{min})}{P(r \geq r_{min})} \tag{3.93}$$

Obviamente,

$$P(r = k \cap r \geq r_{min}) = \begin{cases} P(r=k) & \text{si } k \geq r_{min} \\ 0 & \text{en otro caso} \end{cases} \tag{3.94}$$

y

$$P(r \geq r_{min}) = 1 - P(r < r_{min}) = 1 - p_i \tag{3.95}$$

La esperanza del número de rayos para la esferas de más de r_{min} rayos es

$$E_i = \sum_{k=0}^{n_R} k\ P(r = k/r \geq r_{min}) = \frac{\sum_{k=r_{min}}^{n_R} k\ b(k; n_R, Q^{2i})}{1 - p_i} =$$

$$\frac{n_R Q^{2i} - \sum_{k=0}^{r_{min}-1} k\ b(k; n_R, Q^{2i})}{1 - p_i} \tag{3.96}$$

donde hemos usado el valor esperado del número de rayos (ecuación 3.50) y separado la sumatoria del valor esperado de la siguiente forma:

$$E(n) = \sum_{k=0}^{n_R} k\ b(k; n_R, Q^{2i}) = \sum_{k=0}^{r_{min}-1} k\ b(k; n_R, Q^{2i}) + \sum_{k=r_{min}}^{n_R} k\ b(k; n_R, Q^{2i}) \tag{3.97}$$

(El uso de las aproximaciones mencionadas anteriormente es muy útil aquí también). Por tanto, en el siguiente nivel, el número promedio de rayos será $Q^2 E_i$ (ya que Q^2 es la razón entre los rayos de dos esferas consecutivas).

El costo de la Caché de Esferas se puede obtener de las ecuaciones 3.57 y 3.61, substituyendo el valor del número de rayos en cada nivel $n_R Q^{2i}$ por su nuevo estimador mejorado E_i. El resultado es el siguiente:

$$T = n_P \left(u E_k + t \sum_{i=1}^{\log_Q \frac{d}{r_0}} (1 - Q^{3i}) E_{i-1} \right) \tag{3.98}$$

3.9. Validación de los supuestos teóricos

Esta sección contiene una comparación entre el error observado heurísticamente y el error predicho desde el punto de vista teórico para escenas reales, en las que la distribución de los rayos no es uniforme. Se usaron dos escenas de tamaño medio para probar los resultados teóricos. La primera, Patio, se puede ver en la figura 3.6. La segunda, Expo, se puede ver en la figura 3.7. La sección 3.9.1 compara la predicción teórica de la densidad de rayos (suponiendo una distribución uniforme) con la densidad real que se observa en las escenas de prueba. Posteriormente la sección 3.9.2 estudia el efecto de distintas formas de ordenar los puntos de irradiancia en el número de fallos de caché.

3.9.1. Número promedio de rayos en las esferas

En el estudio teórico de la sección 3.8, se hizo la suposición de que la distribución de los rayos era uniforme. Sin embargo, en escenas reales la distribución depende de la posición e intensidad de las fuentes de luz y de los objetos. En la práctica, las fuentes de luz se colocan de modo que den una iluminación suficiente en la parte interesante de las escenas. Esto incrementa la densidad de los fotones en la zona donde estamos calculando la radiancia. Se realizaron dos

Figura 3.6: Escena Patio.

Figura 3.7: Escena Expo.

simulaciones, una con la escena Patio y otra con la excena Expo. En cada escena se simularon un millón de rayos primarios y sus correspondientes reflexiones por la escena. Para cada nivel de la caché de esferas, se calculó empíricamente el número promedio de rayos y se comparó con la predicción teórica. Los resultados se pueden ver en las figuras 3.8 y 3.9, donde se muestra la diferencia porcentual entre el número de rayos promedio real y la predicción teórica, para cada nivel.

A pesar de que el estudio teórico subestima el número de rayos, la predicción es bastante cercana a la real en los niveles medios y bajos de la lista de esfera, que corresponden a la mayoría del tiempo del algoritmo. Por tanto, podemos decir que el estudio teórico obtiene una buena aproximación al número de rayos reales, y por tanto puede calcular una buena aproximación al coste de los algoritmos.

Aunque los rayos no están distribuidos uniformemente, conforme las esferas se vuelven más pequeñas, la densidad de rayos se vuelve más uniforme, y por tanto la discrepancia entre la escena real (con densidad variable) y la escena teórica (con densidad fija) disminuye.

Para aumentar la precisión de la estimación para escenas en las que la distribución de rayos no sea uniforme, se puede usar un enfoque híbrido. El algoritmo

puede ejecutarse sólo en los niveles más altos de la lista de esferas, teniendo en cuenta sólo los fallos de caché y sin hacer estimación de densidades. Tras ello, los niveles inferiores se pueden estimar con precisión usando el enfoque teórico, ya que los rayos tienden a ser más uniformes conforme el volumen se hace más pequeño. Además se puede dividir el número de rayos por una constante grande, y luego multiplicar los resultados por esta constante para disminuir el tiempo de cómputo de la estimación, como sugiere Revelles en [Revelles 03] y [Revelles Moreno 01].

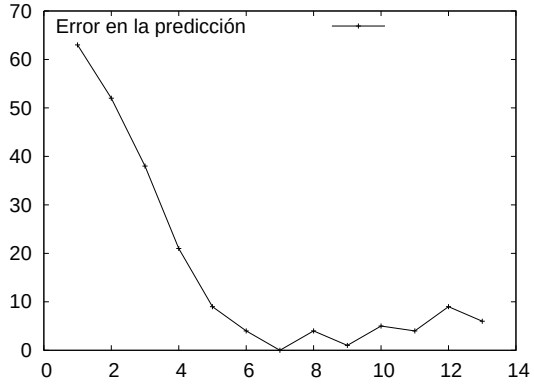


Figura 3.8: Error percentual en la predicción teórica de la escena Patio, para cada nivel.

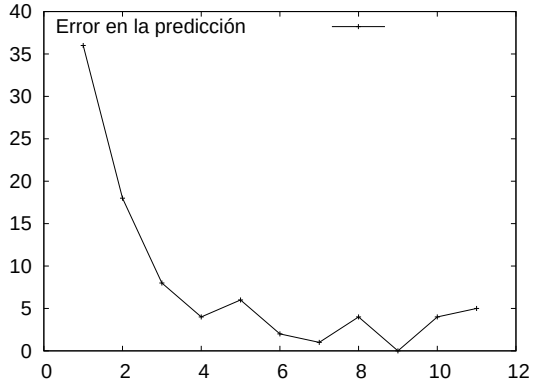


Figura 3.9: Error percentual en la predicción teórica de la escena Expo, para cada nivel.

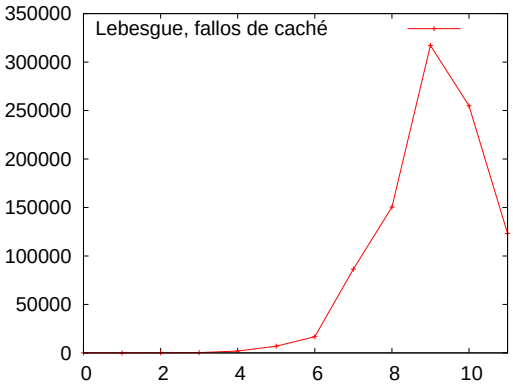


Figura 3.10: Fallos de caché en cada nivel. Ordenación de Lebesgue. $Q = 0,6$.

3.9.2. Fallos de caché en distribuciones uniformes de rayos

Al modelar la caché de esferas teniendo en cuenta el límite de subdivisión, empieza a ser importante cómo se ordenan los puntos para conseguir coherencia espacial. Los puntos se ordenan siguiendo una curva que rellene el espacio y que tenga coherencia espacial. En la caché de esferas, las posibilidades son la curva de Lebesgue o la de Hilbert. Para estudiar cómo afecta la elección del tipo de curva a los fallos de caché, se diseñó un programa que crea un conjunto de rayos siguiendo una distribución uniforme en la esfera unidad, siguiendo el algoritmo descrito en la figura 1.7 de la página 45.

Tras ello se creó un conjunto de puntos con una distribución uniformemente dentro de la esfera unidad, y finalmente se asignó un conjunto de normales distribuidas uniformemente a cada punto y se generó un disco centrado en cada punto, con un radio del 1 %, orientado según la normal del punto. Este conjunto de discos se intersecó con los rayos usando la caché de esferas.

Resultados

La figura 3.10 contiene una gráfica con el número de fallos de caché en cada nivel. Se usa la ordenación de Lebesgue, con un factor de radios de 0,6, y el límite de subdivisión por defecto de 100 rayos. Se usaron 1024*1024 rayos. La gráfica muestra un incremento en fallos de caché conforme el nivel aumenta, hasta que se llega al nivel 9. Tras ello hay una disminución rápida. De acuerdo con la teoría, el nivel 9 tiene 94,6 rayos en media, así que la probabilidad de subdivisión es muy pequeña (no es cero porque la distribución de los rayos no es totalmente uniforme) La disminución de golpe del número de fallos de caché en la figura 3.10 en el nivel 10 se debe al hecho de que el número promedio de rayos es tan pequeño que pocas esferas son subdivididas. También es interesante ver el comportamiento de los fallos de caché para un factor de radios alto.

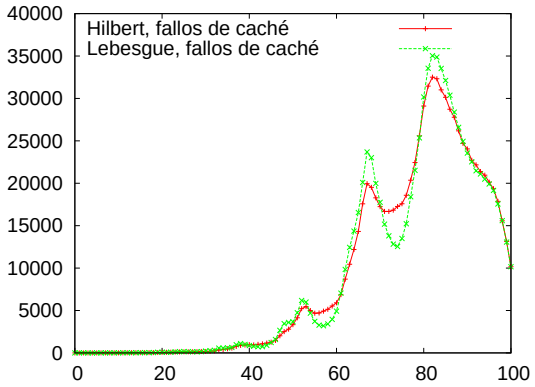


Figura 3.11: Fallos de caché a cada nivel para distintos métodos de ordenación. $Q = 0,95$.

La figura 3.11 muestra los fallos de caché en cada nivel para un factor de radios de 0,95, para la ordenación de Lebesgue y la de Hilbert. Puede verse que la de Hilbert tiene picos mucho menores. Esto se debe a que a pesar de que los resultados de coherencia espacial en el peor caso obtenidos teóricamente son mejores para la curva de Lebesgue, la curva de Hilbert tiene más coherencia espacial que la de Lebesgue en el caso promedio [Hungershöfer 02]. En esta gráfica aparecen una serie de máximos locales seguidos de mínimos locales. Estos puntos están situados en niveles de la caché de esferas que corresponden con un tamaño de 2^{-k} de la escena ($k = 1, 2, ...$), y están causados por la interacción entre la caché de esferas y la ordenación de puntos siguiendo la curva que rellena el espacio. Pueden observarse picos secundarios exactamente en el centro entre dos picos, también debidos al mismo motivo aunque menos acusados. Si se calcula el nivel cuyo número promedio de rayos es 100, se obtiene 90,24. En la gráfica puede observarse el cambio en la forma de la gráfica a partir del nivel 90, ya no dominado por la ordenación de puntos sino por el número de esferas creadas.

3.10. Caché de Esferas Multilista

La caché de esferas multilista (descrita en la sección 1.4.4) organiza las esferas en forma de árbol, de modo que un fallo de caché no descarte necesariamente esferas que puedan resultar de utilidad en el futuro. Hay dos posibilidades: tener un número fijo de descendientes por esfera, o bien indicar un número máximo de esferas que pueden existir a la vez.

3.10.1. Complejidad de la Caché de Esferas Multilista

Estudiaremos primero el caso sin ordenación de puntos. En el caso de tener un número fijo de descendientes por esfera, la lista de esferas se transforma en un árbol de multiplicidad z (z es un parámetro entero, $z > 0$, fijado a priori). La Caché de Esferas original puede verse como el caso particular $z = 1$. La probabilidad de un éxito de caché en el nivel l es la probabilidad de que un punto en la esfera $l-1$ también esté en la esfera l:

$$p_1 = \frac{V_l}{V_{l-1}} = Q^3 \tag{3.99}$$

Si hay más de una esfera, el nuevo volumen estará acotado entre V_l y sV_l. El valor esperado será $sV_1 - V_{int}$ donde V_{int} es la fracción esperada de las esferas que se intersecan entre sí. Dada una esfera S_1 en el nivel l, y dos esferas S_2, $\hat{S}_2$ en el nivel $l+1$, incluídas en S_1, la probabilidad de que un punto en S_1 pertenezca a S_2 es Q^3. Para cada punto de $\hat{S}_2$, la probabilidad de que este punto esté en S_2 es también Q^3. El volumen promedio de la intersección de S_2 y $\hat{S}_2$ es por tanto el volumen de S_2 por la probabilidad de intersección Q^3. Para dos esferas, el volumen promedio de la unión de ambas es

$$V_{l,2} = V_l(1 + 1 - Q^3) \tag{3.100}$$

Este procedimiento puede repetirse para obtener fórmulas para distintos valores de z.

$$V_{l,z} = V_{l,z-1} + V_l - Q^3(V_{l,z-1}) \tag{3.101}$$

La probabilidad de un fallo de caché sin ordenación de puntos (distribución uniforme de puntos) en el nivel l es por tanto

$$P_l = 1 - \frac{V_{l,z}}{V_{l-1}} \tag{3.102}$$

que sólo depende de l, z y Q. La razón entre las áreas de $V_{l,z}$ y V_{l-1} puede usarse en lugar de Q^2 para calcular el coste del método siguiendo el procedimiento de la sección 3.8.1. El coste promedio en tiempo (T) del algoritmo es, por tanto:

$$T = n_R n_P \left(u\frac{d^2}{r_0^2} + t \sum_{i=1}^{\log_Q \frac{d}{r_0}} \left[1 - \frac{V_{i,z}}{V_0}\right] Q^{2i-2} \right) \tag{3.103}$$

Si queremos tener en cuenta también el efecto del límite de subdivisión estudiado en la sección 3.8.3, el resultado queda

$$T = n_P \left(uE_k + t \sum_{i=1}^{\log_Q \frac{d}{r_0}} \left[1 - \frac{V_{i,z}}{V_0}\right] E_{i-1} \right) \tag{3.104}$$

donde E_i es el número promedio de rayos en las esferas del nivel i cuando se tiene en cuenta el límite de subdivisión, tal y como se define en la ecuación 3.8.3 de la sección 3.8.3.

Si por el contrario elegimos fijar el número máximo de esferas (M_S) y descartar la esfera menos recientemente utilizada cuando se desee crear una nueva, el comportamiento esperado es que en promedio, el número de hijos de una esfera (z) dependa de la profundidad k de la lista de esferas, ya que tener más niveles significa tener menos esferas por nivel para un M_S fijo:

$$M_S = \sum_{i=0}^{k} z^i = \frac{1 - z^{k+1}}{1 - z} \tag{3.105}$$

que puede resolverse numéricamente. Una vez que se encuentre una solución para z, las ecuaciones 3.103 ó 3.104 se pueden usar para calcular el coste. En la práctica, como las probabilidades de tener un fallo de caché a distintos niveles son distintas, el valor de z será distinto en cada nivel.

Si las muestras de irradiancia se ordenan siguiendo una curva que rellena el espacio, las esferas almacenadas en la cola sólo se usarán cuando la curva vuelva a una parte del espacio que ya haya sido visitada. Sin embargo, la curva de Lebesgue (y aun más la de Hilbert) tiende a explorar completamente una región antes de moverse a otra región. Esto significa que la probabilidad de reusar las esferas es muy pequeña, y el esfuerzo de mantener la lista de esferas no proporciona reducción de tiempos.

3.11. Indexación de discos

Estudiaremos ahora de forma teórica los parámetros y el rendimiento de la técnica de indexación de discos. Esta técnica, descrita en el capítulo 2, crea una indexación espacial con los discos necesarios para las muestras de irradiancia. En lugar de procesar los discos iterativamente, se procesan los rayos iterativamente intersecándolos con la indexación espacial y añadiendo la contribución de los rayos a cada disco que intersequen.

El diámetro del disco es un tamaño mínimo útil para los voxels de la indexación espacial de esta técnica, ya que como se vio en la sección 2.2, dividir un voxel con un disco de un tamaño aproximado del voxel replica el voxel en los hijos, aumentando el tiempo de recorrido.

El orden de complejidad del método depende de la técnica usada, ya que la indexación de discos se puede usar con distintas técnicas de indexación espacial, Sin embargo, en [Reinhard 96] se prueba que el orden de complejidad para un grid, un árbol binario y un octree es la raíz cúbica del número de celdas para las tres técnicas. En [Szirmay-Kalos 98] se estudia la complejidad media de ray tracing para otras técnicas. La evidencia sugiere que la mayoría de las técnicas de indexación tienen el mismo orden de eficiencia en el caso promedio. Usaremos un octree como técnica de indexación en nuestro análisis dadas sus prestaciones en escenarios genéricos.

El tamaño relativo del lado de un voxel a profundidad k con respecto al lado de la escena completa es 2^{1-k}. Si fijamos el diámetro del disco como lado del voxel más pequeño, obtenemos:

$$2^{1-k} = 2\ d \tag{3.106}$$

$$k = \lfloor -\log_2 d \rfloor \tag{3.107}$$

Hay por tanto 8^k voxels. Una distribución uniforme de las muestras de irradiancia significa que hay $n_P/8^k$ muestras por voxel, y por tanto, el tiempo de intersección entre un rayo y un voxel es $n_P\ u/8^k$. Como cada rayo atraviesa una línea de voxels (2^k voxels) y el origen se encuentra recorriendo el árbol (k pasos), el tiempo promedio para este método es:

$$T = u\ k\ n_R\ n_P/4^k \tag{3.108}$$

El tiempo de cómputo puede variar dependiendo de la distribución de las muestras de irradiancia. Varios autores [Scherson 87, Szirmay-Kalos 98, Aronov 03, Havran 03] han estudiado este efecto, pero el resultado final es que la complejidad en notación $O()$ se conserva. Por tanto, el tiempo queda

$$T \in O(k\ n_R\ n_P/4^k) \tag{3.109}$$

Si el tamaño de voxel es similar al del disco, $k \in O(\log_2 \sqrt[3]{n_P})$. El algoritmo está por tanto en $O(n_R \sqrt[3]{n_P} \log n_P)$.

Si usamos una estructura de partición espacial balanceada, de acuerdo con [Aronov 02], llegar a un nodo vecino puede hacerse en O(1). Entonces el rendimiento de la indexación de discos queda $O(n_R \sqrt[3]{n_P})$.

La eficiencia es mayor que el $O(n_R n_P)$ de la caché de esferas para n_P altos. Para n_P pequeños, en que la distancia entre muestras de irradiancia es similar al radio del disco, la eficiencia es la misma que la de la caché de esferas.

3.12. Estimación automática de parámetros para DETP

El método DETP tiene los siguientes parámetros: número de rayos n_R, número de muestras n_P, radio del disco d y si se debe usar *artifact control* o no. El número de rayos depende de la calidad deseada de la solución. El número de muestras depende de la geometría, pero este parámetro, en general, no es independiente de cómo se modele la escena ni de cómo se haga la interpolación de valores de radiancia.

Los valores del resto de parámetros de DETP básica son los siguientes: el valor óptimo del radio del disco es la distancia promedio entre muestras (sección 3.8.2). *Artifact control* debería usarse cuando existan superficies cóncavas.

La caché de esferas introduce algunos otros parámetros: el factor de radios Q y la posibilidad de ordenar las muestras. La sección 3.8.2 también muestra que el valor óptimo de Q es $2/3$. Los estudios de eficiencia de la caché de esferas

n_R	Número de rayos
n_V	Número de vértices
n_P	Número de muestras de irradiancia
s	Número de triángulos a los que pertenece cada vértice en el mallado de la escena
d	Ancho de banda del kernel de la estimación de densidades
a	Longitud de la escena
r_0	Radio de la esfera englobante
n	Número de fotones que Photon Maps busca
a	Arista de la caja englobante de la escena

Cuadro 3.2: Notación

con y sin ordenación de puntos, validados empíricamente [García 04, García 05, García 06], indican que la ordenación debería usarse siempre. De las dos ordenaciones posibles (Lebesgue y Hilbert), la curva de Hilbert garantiza coherencia y obtiene resultados ligeramente mejores que la de Lebesgue.

La indexación de discos introduce los siguientes parámetros: profundidad del árbol k y método de indexación. La profundidad deberia ser tal que el lado de un nodo es igual al radio del disco. El método de indexación que contiene los discos debería ser balanceado, y una buena elección es aquél que se eligió para indexar espacialmente las primitivas que describen la escena real, porque, salvo en el caso de efectos volumétricos, las muestras de irradiancia se toman sobre la superficie de los objetos, así que es muy probable que los discos y las primitivas de la escena sigan la misma distribución en el espacio.

3.13. Conclusiones

Se ha estudiado teóricamente la convergencia, el sesgo, la varianza y la complejidad de las técnicas de Cuenta de Impactos, Photon Maps, DETP y Ray Maps. Los distintos algoritmos se han comparado entre sí con respecto a la varianza. El resultado principal es que para escenas complejas, con mallado fino y grandes curvaturas, el algoritmo DETP es mejor, seguido de Photon Maps. Cuenta de Impactos obtiene resultados inferiores. El resultado de Ray Maps depende del tipo de estimación elegida. En cuanto a la complejidad, se han derivado fórmulas para cada algoritmo. Usando la notación del cuadro 3.2 obtenemos los siguientes resultados de coste de tiempo de los algoritmos:

- El coste del algoritmo de Cuenta de Impactos está en $O(n_R + n_V\ s + n_P)$.
- El coste de Photon Maps está en $O(n_P(\log n_R + n \log n))$.
- El coste de Ray Maps está en $O(n_R * n_P)$ con una constante oculta $\frac{d^2}{a^2}$.

Posteriormente se ha realizado un estudio teórico de la eficiencia en tiempo de las distintas variantes de DETP. A continuación se presenta el coste en tiempo de cada algoritmo.

- DETP básica y la Caché de Esferas tienen una complejidad en tiempo de $O(n_R\ n_P)$.
- La Caché de Esferas con ordenación de puntos, aunque $O(n_R\ n_P)$, tiene una constante oculta proporcional a $\frac{d^2}{r_0^2}$ (la fracción de rayos en una esfera cuyo radio es el de los discos), que hace que el algoritmo sea bastante rápido en la práctica.
- Para escenas pequeñas, puede probarse que el coste en tiempo de este último algoritmo está en $O(n_R \sqrt[3]{n_P})$.
- Se ha realizado también un estudio de la complejidad de la Caché de Esferas Multilista.
- En cuanto al método de Indexación de Discos introducido en el capítulo 2, éste tiene complejidad en tiempo $O(n_R \sqrt[3]{n_P} \log n_P)$ para árboles no balanceados y $O(n_R \sqrt[3]{n_P})$ para balanceados.

Finalmente se ha usado el estudio teórico para encontrar valores óptimos de los distintos parámetros del algoritmo DETP. En cuanto a los parámetros de la Caché de Esferas, se han obtenido los siguientes resultados:

- Se ha demostrado que el factor de radios debe estar entre 0,6 y 0,7, y simplificando se ha obtenido un valor óptimo de $2/3$.
- Se ha estudiado cómo afecta el hecho de no subdividir una esfera de menos de r_{min} rayos, y se ha comparado el estudio teórico con los resultados de escenas reales.
- Se han tenido en cuenta los distintos métodos de ordenación de puntos (Hilbert y Lebesgue).

Es interesante señalar que el orden de complejidad de la caché de esferas con ordenación de puntos y de Ray Maps es muy similar. Aunque no existe un software unificado que permita comparar directamente la Caché de Esferas con Ray Maps, los resultados publicados comparando por una parte la Caché de Esferas con Photon Maps y por otra parte Ray Maps con Photon Maps, indican que los resultados de tiempo son similares para ambas técnicas, lo que está de acuerdo con los resultados teóricos.

Capítulo 4

Cálculo incremental de la iluminación para escenas quasi-estáticas

En este capítulo estudiaremos cómo, a partir de una escena con información de iluminación global, se puede actualizar de forma rápida y eficiente la iluminación tras el movimiento de un objeto de la escena. Parte del trabajo expuesto en este capítulo ha sido publicado previamente en [García 04, García 07].

4.1. Objetivos

Queremos resolver el problema de calcular la Iluminación Global en una escena quasi-estática con tiempos interactivos. Definimos una escena quasi-estática como la combinación de un escenario estático y un objeto dinámico, cuando la complejidad del objeto dinámico es mucho menor que la complejidad del escenario. Este tipo de escenas es muy usual, sobre todo en videojuegos, que son las aplicaciones interactivas gráficas más comunes. Hay que tener en cuenta dos hechos:

- Estamos creando una animación. Cada fotograma se ve solamente durante aproximadamente 40 milisegundos (a 25 fotogramas por segundo, que es la tasa de refresco de la televisión en Europa). Por tanto no se necesita la solución exacta; es suficiente con una aproximación, incluso en el caso hipotético de que no hubiera requerimientos de tiempo real.

- Queremos tasas de refresco interactivas. Esto significa que sólo tenemos 40 ms de tiempo de cómputo por fotograma, así que habrá que hacer algunas simplificaciones para ajustarnos a los requerimientos de rendimiento.

Estos dos hechos nos llevan a un cálculo aproximado de la iluminación.

El método presentado aquí puede aplicarse a escenas totalmente dinámicas también. En el caso de que todos los objetos se muevan en un fotograma determinado, el algoritmo degenera en los algoritmos de cálculo completo de toda la iluminación. Sin embargo, si sólo parte de la escena cambia en un fotograma determinado (esta parte puede ser distinta en cada fotograma) nuestro método obtiene una aceleración del orden del cociente entre el área de la escena y el área envolvente de los objetos móviles en ese fotograma.

Este algoritmo está diseñado suponiendo fuentes de luz estáticas. Para calcular la iluminación nueva en el caso de que una o varias fuentes de luz se muevan durante parte de la animación, se recomienda el algoritmo de reuso de caminos de Sbert, descrito en la sección 1.5.3, para obtener la iluminación en esos fotogramas.

4.2. Cálculo y actualización del conjunto de rayos

Algoritmo 4.2.1: CálculoInicialDeFotosimulación(*Fotones*)

```
Sea numLights el número de luces de una escena.
Sea fotones el número de fotones para simular.
Calcular el número de fotones para cada luz,
de acuerdo con el percentaje de energía emitida desde esa luz
for Light ← 1 to numLights
do { for i ← 1 to número de fotones para Light
     do { Ray ← Lights[Light].GetPhoton()
          Insertar Ray en RayList
          while Ray no sea absorbido
          do { IntersectedTriangle ←Primera intersección
               de Ray con la escena
               Ray ← IntersectedTriangle.Reflect(Ray)
```

La función GetPhoton() crea un fotón aleatorio en la superficie de la fuente de luz.
La función Reflect(Ray) refleja el rayo de acuerdo con la BRDF de la superficie y devuelve un nuevo rayo, o NULL si el fotón se absorbe.

Figura 4.1: Pseudocódigo para el cálculo inicial de fotosimulación.

El algoritmo calcula una aproximación a la iluminación global usando una fase de fotosimulación seguida de una fase de estimación de densidades de las descritas en la sección 1.4 del primer capítulo (página 25).

El primer paso del algoritmo es la generación de los rayos y el almacenamiento de los puntos de intersección de cada rayo con la escena, siguiendo un esquema de fotosimulación como el descrito en la sección 1.4.1 (página 27). El pseudocódigo puede verse en la figura 4.1.

En los siguientes fotogramas, el conjunto de rayos se actualiza de la siguiente forma: conocemos qué triángulo intersecó al rayo en el fotograma anterior. Si este triángulo pertenece al objeto móvil, el rayo es recalculado, teniendo en cuenta la nueva posición del objeto móvil. Si el triángulo no pertenece al objeto móvil (o si no había intersecado en el fotograma anterior) el rayo se interseca con la posición actual del móvil. Si hay intersección, se recalculan sus reflexiones. En caso contrario, el rayo es aún válido, pero el procedimiento ha de repetirse para sus reflexiones. Es de destacar que para la mayoría de los rayos, no se necesita un test de intersección con la escena, que es donde está la mayor complejidad.

Para implementar este enfoque eficientemente, se usa una lista de rayos de dos niveles: El primer nivel contiene los rayos primarios; estos rayos permiten acceder a una lista simplemente enlazada que contiene los rayos secundarios, terciarios, etc, fruto de la reflexión del rayo en los objetos de la escena. Recalcular un rayo implica guardar una copia del rayo (arrastrando una referencia al resto de las reflexiones de este rayo) en una lista auxiliar, a la que nos referiremos como "Rayos antiguos" a partir de ahora, y comprobar de nuevo el rayo original con la escena y el objeto móvil. Esto generará una lista de nuevas reflexiones, que se guardan en el lugar en que estaba la referencia a las reflexiones antiguas. También han de guardarse en otra lista auxilar, "Rayos nuevos". Con este enfoque, tenemos tres listas de rayos: Rayos antiguos, rayos nuevos y la lista de rayos. Estas listas se usan después para calcular valores de radiosidad en los vértices. El pseudocódigo del algoritmo puede verse en la figura 4.2.[1]

4.3. Cálculo de la radiosidad en el primer fotograma

El cálculo inicial de la radiosidad es, en nuestro algoritmo, independiente del método de estimación de densidades elegido. La primera vez que se ejecuta el algoritmo de radiosidad, hay que tener en cuenta todos los rayos de la escena. El procedimiento se muestra en la figura 4.3. Este algoritmo se expresa con un nivel de abstracción muy alto, ya que el algoritmo específico (Cuenta de Impactos, Photon Maps o Estimación de Densidades en el Plano Tangente) se pasa como argumento a la función.

El algoritmo de actualización de la radiosidad, al contrario del cálculo inicial, depende tanto del tipo de vértice (estático o dinámico) como del método de iluminación global. Las siguientes dos secciones describen cómo proceder en cada caso.

[1] Un vídeo de ejemplo puede verse en el fichero movil.mpeg, que se puede encontrar en http://giig.ugr.es/~rgarcia/tesis/movil.mpeg.

Algoritmo 4.2.2: FOLLOWRAY(*Ray*)

```
while Ray no sea absorbido
  do { Intersecar Ray con la escena
       Ray.next ←siguiente reflexión de Ray
       Ray ← Ray.next
```

Algoritmo 4.2.3: RECÁLCULODEFOTOSIMULACIÓN()

```
OldRayList ← EmptyList
NewRayList ← EmptyList
for i ← 1 to fotones
  do { Ray ← RayList[i]
       while Ray no sea absorbido
         do { if Ray intersecaba el objeto móvil or
              Ray interseca ahora el objeto móvil
              then { Insertar Ray en OldRayList
                     FOLLOWRAY(Ray)
                     Insertar Ray en NewRayList
                     break
              Ray ←siguiente reflexión de Ray
```

Figura 4.2: Pseudocódigo para el recálculo de la fotosimulación.

Algoritmo 4.3.1: INITIALRADIOSITYCALCULATION(DensityEstimatonAlgorithm *DE*)

```
for each Vertex in the scene
  do { DE.CalculateRadiosity(Vertex, RayList)
```

Figura 4.3: Pseudocódigo para el cálculo inicial de la radiosidad.

4.4. Actualización de la radiosidad en vértices estáticos

En el algoritmo de actualización de la radiosidad para fotogramas posteriores al primero, el objeto móvil se encuentra en una nueva posición. Los rayos que

intersecaban al objeto en el fotograma anterior han sido recalculados en la fase de fotosimulación. Obviamente, los rayos que no intersecaban antes y ahora lo hacen también han sido recalculados. Sin embargo, la mayoría de los rayos tiene la misma contribución que antes a la radiosidad en un punto dado.

Es por tanto interesante intentar conservar la información de radiosidad antigua que sigue siendo válida. Ésta es la expresión de la radiancia que deja un punto x en dirección ω_0:

$$L_r(x,\omega_0) = \int_\Omega f_r(x,\omega_0,\omega) L_i(x,\omega) cos(\theta) d\sigma(\omega) \tag{4.1}$$

Para superficies difusas, se puede simplicar a:

$$L_r(x,\omega_0) = \frac{\rho(x)}{\pi} \int_\Omega L_i(x,\omega) cos(\theta) d\omega = \frac{\rho(x)}{\pi} E(x) \tag{4.2}$$

donde $\rho(x)$ es la reflectividad en x y $E(x)$ la irradiancia en x.

Cuando tenemos una escena dinámica, el tiempo ha de tenerse en cuenta: Sea $E(x,t)$ la irradiancia en x en el instante t. Sea $\tilde{E_R}(x;S)$ la irradiancia en x debido al conjunto de rayos S. Este conjunto es resultado de la fase de fotosimulación, y los rayos contienen la información necesaria para realizar el método adecuado de estimación de densiades. Sea S^t el conjunto de rayos en el instante t.

$$\tilde{E}(x,t) = \tilde{E_R}(x;S^t) \approx E(x,t) \tag{4.3}$$

es nuestro estimador de radiancia, que aproxima el valor real $E(x,t)$.

Por tanto,

$$E(x,t) \approx \tilde{E}(x,t-1) + \tilde{E_R}(x;N^t) - \tilde{E_R}(x;O^t) \tag{4.4}$$

donde definimos N^t como:

$$N^t = S^t - S^{t-1} \tag{4.5}$$

(rayos nuevos) y O^t como:

$$O^t = S^{t-1} - S^t \tag{4.6}$$

(rayos antiguos).

Esta fórmula se extrae del razonamiento siguiente: Sea I^t el conjunto de índices invariantes en un instante t:

$$I^t = S^t \cap S^{t-1} \tag{4.7}$$

Ya que la contribución de cada rayo a la radiosidad es aditiva (esto es una consecuencia directa de la linealidad de los operadores integrales en los que se basa la estimación de densidades usando Monte Carlo; la consecuencia es que se cumple $\tilde{E_R}(x;A\cup B) = \tilde{E_R}(x;A) + \tilde{E_R}(x;B) - \tilde{E_R}(x;A\cap B)$ para todo A y B)

$$\tilde{E_R}(x;S^t) = \tilde{E_R}(x;I^t \cup N^t) = \tilde{E_R}(x;(S^{t-1} - O^t) \cup N^t) \tag{4.8}$$

Como O^t está incluído en S^{t-1}, $\tilde{E_R}(x; S^{t-1} - O^t) = \tilde{E_R}(x; S^{t-1}) - \tilde{E_R}(x; O^t)$; además N^t y S^{t-1} son disjuntos y por tanto podemos concluir

$$\tilde{E_R}(x; S^t) = \tilde{E_R}(x; S^{t-1}) - \tilde{E_R}(x; O^t) + \tilde{E_R}(x; N^t) \tag{4.9}$$

Esta ecuación es 4.4 con los términos de la parte derecha en distinto orden.

Para recalcular la radiosidad, ejecutamos el algoritmo de estimación de densidades con el conjunto de rayos antiguos generados en el paso anterior. La radiancia calculada se resta del valor anterior de radiancia en cada vértice. Luego el algoritmo se ejecuta de nuevo con el nuevo conjunto de rayos, y la estimación de radiancia calculada se suma al valor en el vértice. El resultado es idéntico a calcular la estimación de la radiancia en el conjunto completo de rayos, pero mucho más rápido. Nótese que este cálculo no requiere el uso del conjunto completo de rayos.

4.5. Cómputo y actualización de la radiosidad en vértices dinámicos

El algoritmo para vértices dinámicos depende del método de estimación de densidades. Primero se explicará el algoritmo para Cuenta de Impactos. Después se explicará el algoritmo para Photon Maps y Estimación de Densidades en el Plano Tangente.

4.5.1. Cuenta de Impactos

Para actualizar la radiosidad usando Cuenta de Impactos, hemos de conocer qué triángulos han recibido nuevos impactos o ya no tienen los impactos que tenían. Afortunadamente, la fase de trazado de fotones calcula esta información, así que podemos eliminar la contribución de los rayos que ya no cortan los triángulos y añadir los impactos nuevos.

La radiosidad en un vértice finalmente se calcula como el promedio de las radiosidades de los triángulos a los que pertenece. Por tanto, el algoritmo de recálculo de los objetos móviles con cuenta de impactos es idéntica al recálculo de la escena estática. El pseudocódigo puede verse en la figura 4.4.

4.5.2. Photon Maps y Estimación de Densidades en el Plano Tangente

Dado que la actualización de la radiosidad usando estos métodos depende de información de impactos en puntos no locales (los discos en DETP son desconocidos para la fase de trazado de fotones; y los vértices a los que afecta un impacto en Photon Maps son desconocidos a no ser que se haga una búsqueda específica), el algoritmo usado en Cuenta de Impactos no puede usarse para estos métodos.

Algoritmo 4.5.1: RadiosityRecalculationForImpactCount()

for each *Ray* in *OldRayList*
do $\begin{cases} TriangleIntersected = Ray.GetHit() \\ TriangleIntersected.Substract(Ray) \end{cases}$
for each *Ray* in *NewRayList*
do $\begin{cases} TriangleIntersected = Ray.GetHit() \\ TriangleIntersected.Add(Ray) \end{cases}$

Figura 4.4: Pseudocódigo para el recálculo de la radiosidad para el método de Cuenta de Impactos.

Para calcular el estimador de la radiosidad en los puntos del objeto móvil, descartamos el valor de radiancia que se obtuvo en el frame anterior, y ejecutamos el algoritmo inicial, comprobando todos los rayos de la escena. Para cada muestra de irradiancia, se calcula la contribución de los rayos a la muestra, siguiendo el algoritmo de estimación de densidades elegido. Para referencia, véase la figura 4.3, donde se muestra el pseudocódigo para el cálculo inicial de la radiosidad. Nótese que este algoritmo es diferente del usado en [Dmitriev 02], ya que Dmitriev descarta los fotones antiguos aunque sigan siendo válidos.

4.5.3. Comparación entre las distintas formas de actualizar la radiosidad

Hemos visto que el algoritmo de Cuenta de Impactos es intrínsicamente más rápido que el de Photon Maps o DETP, ya que sólo hay que tener en cuenta los rayos recalculados. El problema es que los parámetros de ruido y varianza son mucho mayores. Es, por tanto, necesario incrementar el número de fotones, y el tiempo final de iluminación para una escena determinada es más alto.

Aunque se ha descrito en la literatura que los parámetros de ruido y varianza son mucho más alto en Cuenta de Impactos que en Photon Maps ([Jensen 01], página 52) o DETP ([Lastra 02b], sección 2), se ha estudiado el tiempo para el cálculo completo de la irradiancia.

Como hemos visto en las secciones anteriores, al hacer recálculo con Cuenta de Impactos podemos reusar información de la iluminación del móvil, mientras que en Photon Maps y DETP debemos descartarla. Es necesario responder a la pregunta de si la Cuenta de Impactos, que usa un algoritmo más eficiente en el recálculo del móvil, puede compensar el incremento de fotones necesario para obtener una solución con un error comparable a los otros métodos.

En este caso mediremos el tiempo de recálculo de la escena completa en Photon Maps o DETP, y el tiempo de recalcular los rayos que intersecan el objeto móvil sólamente en Cuenta de Impactos. Como ejemplo usaremos la

escena compleja estudiada en detalle en la sección 4.9.

Figura 4.5: Estimación de Densidades en el Plano Tangente, 50 000 fotones.

Figura 4.6: Estimación de densidades usando Cuenta de Impactos, 1 000 000 fotones.

Usaremos la diferencia de luminancia promedio entre una imagen de referencia (Estimación de Densidades en el Plano Tangente, radio 0,01, cien millones de fotones) y las imágenes generadas que se muestran abajo. Se pueden comparar las figuras 4.5 y 4.6 para ver la mejor calidad de DETP. El cuadro 4.1 muestra el

Algoritmo	Fotosimulación	Iluminación	Total	Error
DETP	0,05	0,80	0,85	16,10 %
ImpactCount	1,09	0,02	1,11	139,54 %

Cuadro 4.1: Comparación entre DETP y Cuenta de Impactos.

tiempo en segundos para el recálculo de los rayos (fotosimulación), la estimación de densidades (iluminación), y el tiempo total. Finalmente se muestra el error. Es fácil ver que aunque el tiempo de simulación es similar (ligeramente más bajo para DETP), el error es un orden de magnitud mayor para cuenta de impactos.

4.6. Reuso de la ordenación de puntos

El rendimiento de la Caché de esferas depende de la ordenación de los puntos. Esto se debe al hecho de que se mantiene una estructura de datos jerárquica lineal, compuesta de esferas de radio decreciente. Estas esferas tienen una estructura de datos asociada que guarda los rayos que las atraviesan. Cuando se hace estimación de densidades en un punto que está dentro de la esfera actual (en realidad cuando el disco completo que se usa en la estimación de densidades está en la esfera), encontrar los rayos que afectan su radiancia es rápido. Cuando el punto está fuera de la esfera (o al menos parte del disco), la esfera se descarta y la esfera padre en la estructura se comprueba hasta encontrar una esfera válida. Entonces se reconstruye parte de la jerarquía o lista de esferas, tal y como se describe en la sección 1.4.4 del primer capítulo (página 30).

Para disminuir lo máximo posible el número de esferas eliminadas, se usa un esquema de preordenación que obtiene el máximo rendimiento del algoritmo disminuyendo el número de veces que hay que reconstruir las esferas (sección 1.4.4). Debido a la naturaleza estática de la mayoría de la escena y el hecho de que el móvil es rígido, el orden óptimo para un mínimo de fallos de caché se puede estudiar en un primer paso. En las siguientes pasadas, el orden se reutiliza.

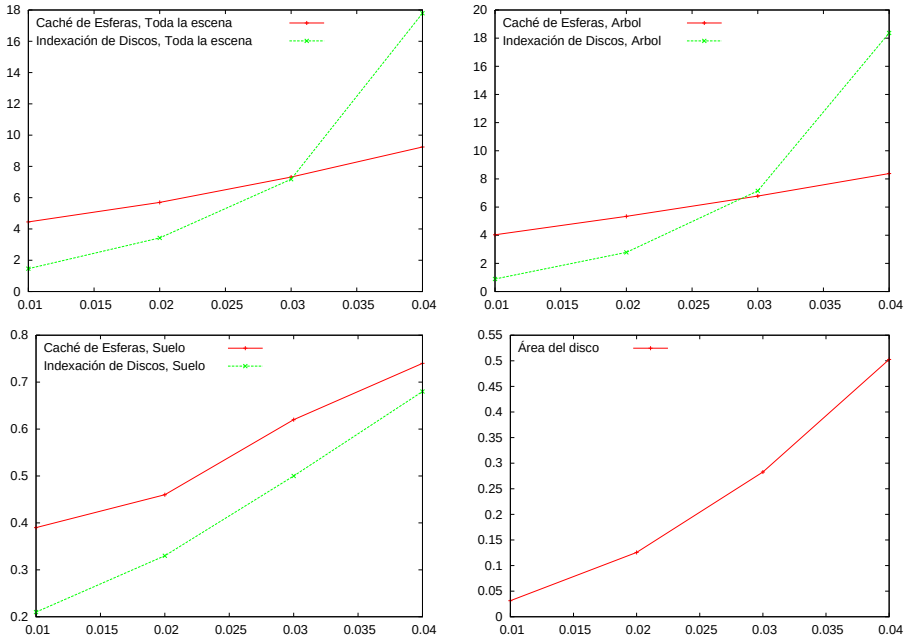
4.7. Indexación de Discos

Se integró la técnica de indexación espacial para los discos de Estimación de Densidades en el Plano Tangente (capítulo 2) en el algoritmo de recálculo de iluminación. El algoritmo calcula los discos de los vértices de la escena, y los ordena usando un método de indexación espacial. La radiancia se calcula intersecando cada rayo contra los discos, usando la indexación espacial para acelerar el proceso.

Este método tiene mayor rendimiento que el método original de intersección con la Caché de Esferas cuando los discos tienen un radio de tamaño similar al de las aristas de los triángulos, o bien si los discos son grandes, para un número pequeño de rayos. En otras situaciones, el rendimiento de la Caché de Esferas es superior.

El número máximo de primitivas por voxel debería ajustarse conforme cambia el tamaño del disco, ya que cuando los discos crecen, intersecan unos con otros; dividir el voxel crea voxels en los que la mayoría de los discos pertenecen a ambos voxels. Esto hace que el tiempo de intersección con los nuevos voxels sea más alto que el tiempo de usar el voxel original que combina ambos subvoxeles, como se indicó en la sección 2.2 (página 48).

a) Comparación entre la Caché de Esferas y la Indexación de Discos para toda la escena, sólo el árbol y sólo el suelo. Tiempo como función del radio del disco. Se muestra también el área de cada disco, como porcentaje del área del suelo.



b) Mejor número de primitivas por voxel para Indexación de Discos como función del radio del disco.

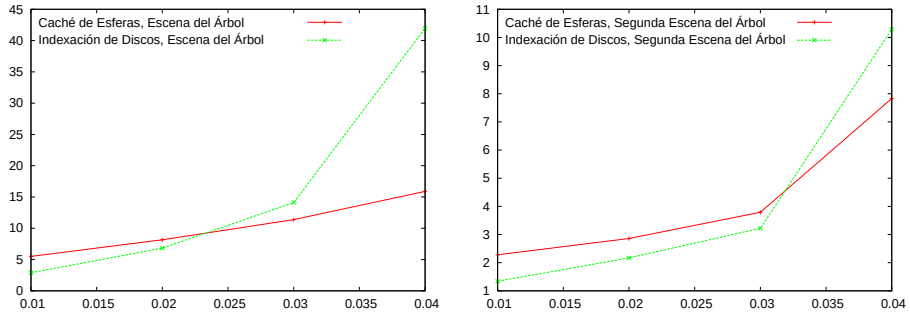
Escena \ Radio	0,01	0,02	0,03	0,04
Escena completa	20	20	50	800
Árbol	20	10	100	1600
Suelo	20	80	120	160

Figura 4.7: Tiempo de recálculo (en segundos) para 10 000 fotones, Escena del Árbol.

Las gráficas de tiempos y el número óptimo de primitivas por voxel se muestran en la figura 4.7 para 10 000 fotones. Los cuadros con los tiempos pueden

verse en el apéndice A. La figura 4.8, a) izquierda y b) tiene la información correspondiente para el caso de 20 000 fotones (el apéndice A tiene los datos correspondientes a esta figura).

a) Comparación entre Caché de Esferas e Indexación de Discos para la Escena del Árbol (izquierda) y la Segunda Escena del Árbol (derecha). Tiempo en función del radio del disco.



b) Mejor número de primitivas por voxel para Indexación de Discos como función del radio del disco. Escena del Árbol.

Escena \ Radio	0,01	0,02	0,03	0,04
Escena completa	20	20	200	1600

c) Mejor número de primitivas por voxel para la Indexación de Discos como función del radio del disco. Segunda Escena del Árbol.

Escena \ Radio	0,01	0,02	0,03	0,04
Escena completa	40	80	160	32000

Figura 4.8: Tiempo de recálculo de la radiosidad para 20 000 fotones para la Escena del Árbol y la Segunda escena del Árbol.

Los triángulos del suelo tienen un tamaño de 0,02 x 0,02, donde las aristas de la caja englobante de la escena miden una unidad. Los triángulos del árbol tienen un tamaño aproximado de 0,01 x 0,01. El radio del disco se mide en tanto por uno de la arista de la caja envolvente de la escena. El área del disco se mide como porcentaje del área del suelo (1x1). Puede verse que para tamaños inferiores a 0,04, el nuevo método es más rápido que la Caché de Esferas. Como los triángulos del suelo son mayores que los del árbol, la escena que sólo tiene el suelo tiene mejores resultados también para un radio del disco de 0,04.

Para 20 000 fotones, los resultados sólo son mejores para tamaños de 0,01 y 0,02. El problema es que este método es lineal en el número de rayos.

Se generó otra escena con un tipo diferente de árbol y de suelo, con triángulos mayores, que se muestra en la figura 4.9. La información de tiempos para esta escena puede verse en la figura 4.8, a) derecha y c).

Estos cuadros muestran la disminución del rendimiento de la indexación de discos conforme el área de los discos se incrementa. Puede verse que los resultados son mejores usando discos más grandes que en la escena anterior, debido al hecho de que el tamaño de los triángulos se ha incrementado.

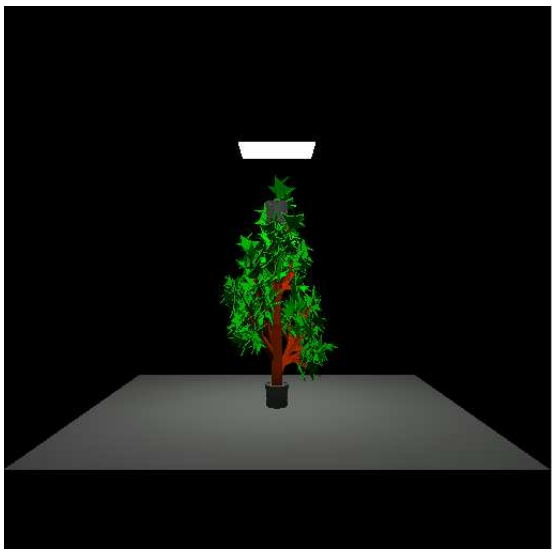


Figura 4.9: Segunda Escena del Árbol, 20 000 fotones.

Para discos grandes, el procedimiento normal de dividir los voxels hasta que queden pocos objetos en el voxel o se alcance una profundidad máxima no funciona. Como los discos son tan grandes, pertenecen a la mayoría o a todos los nodos recién creados. Esto hace que el algoritmo tenga que iterar por la lista de todos los discos para cada nodo, con lo que el tiempo resultante es más alto en lugar de más bajo. Para evitar esto, la profundidad máxima debe disminuir conforme el radio del disco aumenta, como se vio en la sección 2.2.

Una ventaja del método de Indexación de Discos es una fácil paralelización, ya que el árbol de indexación de los discos es estático, y puede accederse sin problemas de contención. Por otra parte, sincronizar el acceso y la actualización de la Caché de Esferas es complejo, y disminuye el rendimiento de la cache.

4.8. Estudio de tests de pre-intersección

Cuando se intersecan entre sí muchos objetos con geometría compleja, a veces es útil envolver cada objeto con otro objeto de geometría simple, y hacer un test de intersección entre los objetos envolventes simples para evitar hacer el test de intersección entre los objetos reales, si estos están alejados entre sí. Esto se denomina un pre-test de intersección, y puede mejorar los tiempos de cálculo si la probabilidad de intersección entre los objetos es pequeña.

En el caso de intersecciones rayo-disco, existen algunos tests estandar que envuelven el disco en un triángulo. Se estudiaron los tests de pre-intersección de Badouel [Badouel 90], Plücker [Teller 92] y Möller-Trumbore [Möller 97] con respecto del test de intersección rayo-círculo. Se crea un triangulo equilátero tangente al disco, y los rayos se intersecan primero con el triángulo, y si lo atraviesan, con el círculo. Aunque estos tests por sí solos aumentan el rendimiento un 34 %, en combinación con la Caché de Esferas sólo incrementan el rendimiento aproximadamente un 1 %.

El pretest de intersección pierde utilidad en combinación con indexación espacial o caché de esferas porque al usar esas técnicas se aumenta la probabilidad condicional de que haya intersección rayo-disco supuesto que se hace el test realmente (hemos visto que la mejora del la eficiencia depende de que esta probabilidad sea baja). Además, el rendimiento de los tests dependen del tamaño del disco, ya que para discos pequeños la mayoría de los rayos no intersecan ni el triángulo ni el disco, lo que significa que el pre-test aumentará la velocidad del cálculo, y para discos grandes una fracción significativa de los discos intersecan tanto el triángulo como el disco, con lo que el pre-test sólo será un estorbo. Para escenas pensadas para tiempo real, nos interesan los discos grandes, ya que eso hace posible tener un número más pequeño de rayos. Por desgracia, eso significa que el aumento de rendimiento del pre-test es casi inapreciable. Los test de pre-intersección mencionados fueron probados en la escena que se muestra en la sección 4.9. Los resultados para los diferentes tests pueden verse en la figura 4.10 (tiempos en segundos).

Los tests muestran que los tests de pre-intersección de Badouel y Möller-Trumbore son un poco más rápidos que no hacer pre-test. Plücker es más lento porque necesita que se almacenen las coordenadas de Plücker. La mejora de rendimiento es en el mejor caso (Möller-Trumbore) del 1,81 %.

4.9. Comparación entre Photon Maps y DETP

Ahora compararemos el tiempo que se necesita para actualizar la información de radiosidad para los métodos de estimación de densidades Photon Maps y Estimación de Densidades en el Plano Tangente.

Usaremos una escena con 71 966 triángulos y un objeto móvil de 500 triángulos. (Véase la figura 4.11) Esta escena también se ha usado en [Lastra 02a], aunque no había objeto móvil en ese artículo y para estudiar el método de Indexación de Discos en el capítulo 2. Llamaremos a ésta la Escena del Árbol.

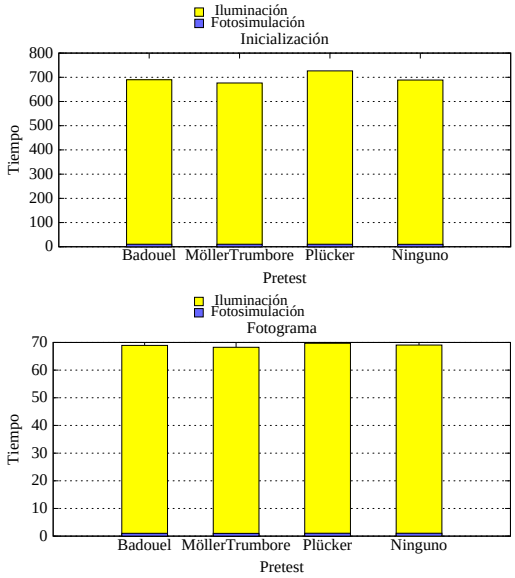


Figura 4.10: Tiempo de inicialización y de fotograma para simular la Escena del Árbol usando distintos tests de pre-intersección.

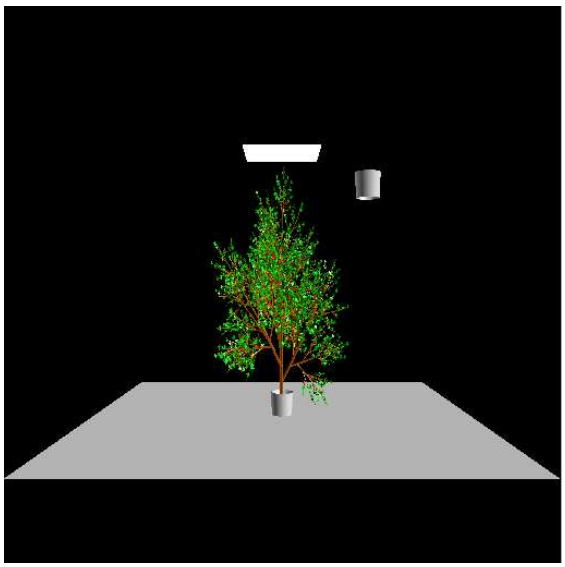


Figura 4.11: Render en color plano de la Escena del árbol.

Usaremos como referencia la imagen creada usando cien millones de rayos con el método de estimación DETP (el radio es el 1 % de la escena). En esta sección se da la información resultante de tiempos y error.

Figura 4.12: Estimación de Densidades en el Plano Tangente, 1 000 fotones.

4.9.1. Gráficas de rendimiento

Se han probado los algoritmos de Estimación de Densidades en el Plano Tangente, usando dos simulaciones de 1 000 fotones y 10 000 fotones respectivamente, y Photon Maps, con tres simulaciones de 10 000, 50 000 y 200 000 fotones. Las gráficas de error en función del tiempo pueden verse en la figura 4.17 (los cuadros con los tiempos pueden verse en el apéndice A).

La figura 4.12, contiene una imagen con el resultado de aplicar el algoritmo de DETP con 1 000 fotones. Se puede ver que activar la Caché de Esferas y la ordenación de las muestras de irradiancia permiten obtener 9 fotogramas por segundo, lo que puede considerarse tiempo real. El error de la imagen es bastante alto debido al pequeño número de fotones.

Al ejecutar DETP con 10 000 fotones, obtenemos la figura 4.13. Para este número de rayos, tenemos una tasa de refresco de dos fotogramas por segundo, que es interactiva, pero no tiempo real. El error se reduce a la mitad con respecto a la imagen anterior, y se debe casi por completo al sesgo del algoritmo debido al tamaño grande del disco.

En cuanto al algoritmo de Photon Maps, podemos observar el resultado de la simulación en las figuras 4.14 (10 000 fotones, página 116), 4.15 (50 000 fotones, página 117), y 4.16 (200 000 fotones, página 117). Para Photon Maps la tasa de

Figura 4.13: Estimación de Densidades en el Plano Tangente, 10 000 fotones.

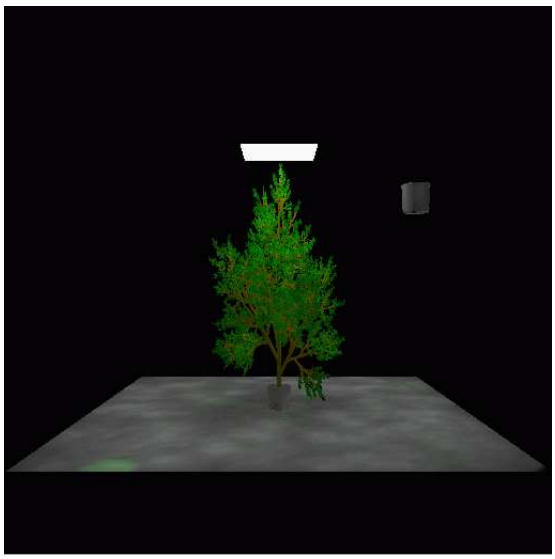


Figura 4.14: Estimación de Densidades Photon Maps, 10 000 fotones.

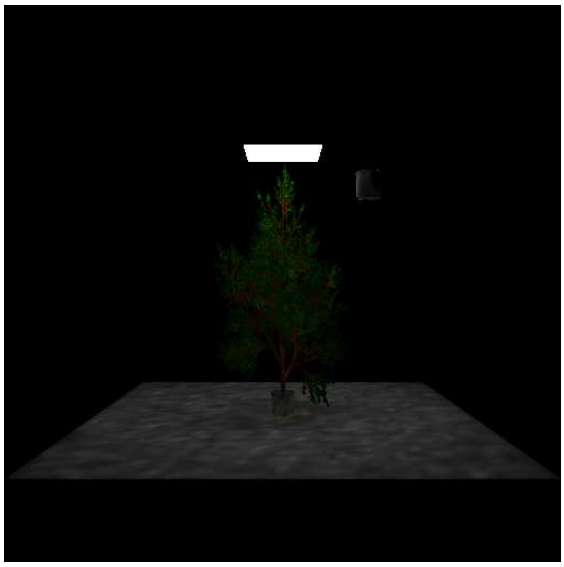


Figura 4.15: Estimación de Densidades Photon Maps, 50 000 fotones.

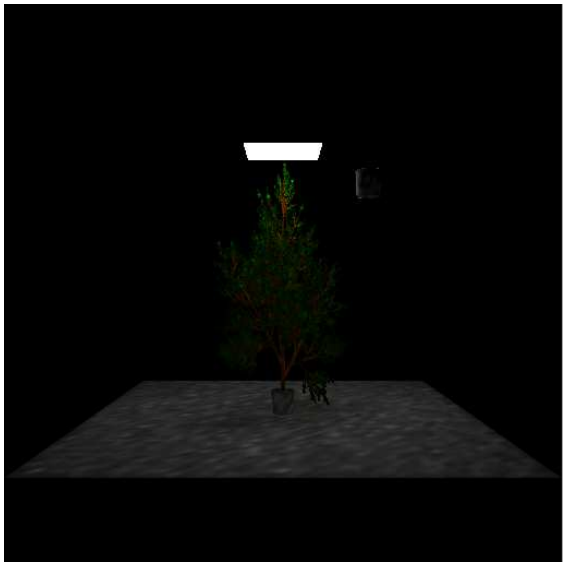


Figura 4.16: Estimación de Densidades Photon Maps, 200 000 fotones.

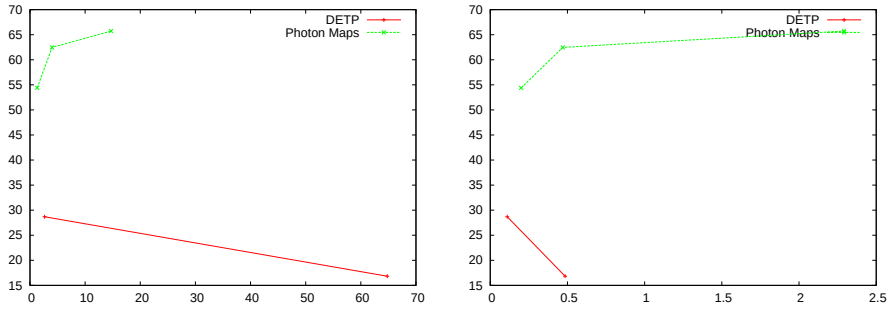


Figura 4.17: Comparación de Photon Maps y DETP. Error porcentual en función del tiempo. Inicialización (izquierda) y tiempo de fotograma (derecha).

refresco es de 5 fps (10 000 fotones), 2 fps (50 000 fotones) y 0,43 fps (200 000 fotones). Observamos que los parámetros de error son mucho más altos que los de DETP.

Los ficheros de referencia son una imagen generada con Photon Maps con cinco millones de fotones, usada para la columna Ruido, y una imagen con DETP, también con cinco millones de fotones, para la columna Total. La columna de Sesgo es la diferencia entre la imagen de referencia de Photon Maps y la imagen de referencia de DETP. Puede verse que al aumentar el número de fotones, la imagen tiende a la imagen de referencia de Photon Maps, divergiendo de la imagen de referencia de DETP. Esto es debido a que el sesgo de Photon Maps es más facil de ver conforme disminuye el tamaño de la esfera. La calidad de las imagenes para el número correspondiente de frames por segundo de DETP es mucho más baja en Photon Maps, así que nuestro método preferido para aplicaciones de tiempo real es Estimación de Densidades en el Plano Tangente.

4.10. Estudio teórico

En esta sección realizaremos un estudio teórico del algoritmo de recálculo utilizando como base las técnicas descritas en el capítulo 3. Comenzaremos calculando el error esperado para la escena de la sección anterior, usando las fórmulas de la sección 3.6.1. Tras ello estudiaremos la eficiencia de los algoritmos, siguiendo la sección 3.8.

4.10.1. Error teórico de DETP

Usaremos la sección 3.6.1 para calcular el error esperado a priori en la escena de la sección anterior. Esta sección desarrolla una fórmula para el error que se obtuvo a partir de la estimación de la esperanza y la varianza del algoritmo

DETP (ecuación 3.46, página 75):

$$Error = \sqrt{\frac{A_0 - A_d}{A_d n_R}} \tag{4.10}$$

donde A_0 es el área de la envolvente convexa de la escena, y A_d es el radio del disco.

Los datos necesarios de la escena son:

- Radio del disco $d = 0{,}1$.
- Número de rayos $n_R = 10\,000$, $n_R = 1\,000$.

Como calcular el área envolvente del árbol es muy complejo, usaremos la siguiente simplificación: calcularemos el área de la caja envolvente del árbol. El árbol mide $0,350115 \times 0,667508 \times 0,34873$; mientras que el suelo mide 1×1. El área total de la escena es de $A_0 = 2,05507$.

Los errores esperados son: $Error = 8,02\,\%$ (10 000 fotones) y $Error = 25,38\,\%$ (1 000 fotones). Los datos empíricos son (Cuadros A.13 y A.12, Error de Ruido) $Error = 8,57\,\%$ (10 000 fotones) y $Error = 24,64\,\%$ (1 000 fotones). Por tanto, en este caso la diferencia entre la predicción teórica del error y el valor real se encuentra entre el 3 % y el 6 %.

4.10.2. Estudio de eficiencia

Símbolos relacionados con las esferas	
$r_i = r_{i-1}Q = r_0Q^i$	Radio de la esfera i
r_D	Radio de la esfera envolvente del objeto móvil
Símbolos relacionados con los rayos	
n_R	Número de rayos
$\tilde{n}_R = n_R \frac{r_D^2}{r_0^2}$	Número de rayos que tocan el objeto móvil
Cantidades relacionadas con el tiempo	
u	Tiempo de intersección Rayo Disco
t	Tiempo de intersección Rayo Esfera
T_{DI}	Tiempo de Indexación de Discos
T_{SC}	Tiempo de Caché de Esferas
T_{Rec}	Tiempo de recálculo
A	Aceleración del algoritmo al usar recálculo
Otros símbolos	
$0 < Q < 1$	Razón de los radios de dos esferas
n_D	Muestras en objetos dinámicos
n_S	Muestras en objetos estáticos
$n_P = n_D + n_S$	Muestras de irradiancia

Cuadro 4.2: Símbolos usados en el estudio de eficiencia del recálculo.

El Cuadro 4.2 tiene un resumen de los símbolos usados en esta sección. Nos basaremos en las fórmulas obtenidas en el capítulo 3 que mostraban la complejidad de los diferentes algoritmos, y las aplicaremos al caso de cálculo incremental de escenas quasi-estáticas.

Si llamamos n_S al número de puntos estáticos y n_D al número de puntos dinámicos ($n_P = n_S + n_D$), y r_D al radio de la esfera envolvente del objeto dinámico, puede demostrarse que el número de rayos recalculados es:

$$n_R^{new} \approx n_R^{old} \approx n_R \frac{r_D^2}{r_0^2} =_{def} \tilde{n}_R \tag{4.11}$$

donde n_R es el número de rayos.

Puntos estáticos

El costo de la cache de esferas en este caso es

$$T = \frac{t\tilde{n}_R}{Q^2} \frac{\sqrt[3]{n_S} - 1}{1 - Q} + \frac{4}{3} u\tilde{n}_R \sqrt[3]{n_S} \tag{4.12}$$

donde el primer sumando corresponde a los fallos de cache y el segundo al coste de la esfera interior.

El valor óptimo del factor de radios entre esferas Q corresponde a $2/3$, como se vio en la sección 3.8.2, y teniendo en cuenta que el tiempo de intersección rayo-disco y rayo-esfera son similares ($t \approx u$), y que en el límite, para un n_S grande, $\sqrt[3]{n_S} - 1 \approx \sqrt[3]{n_S}$, el coste es 8,08 $t\ \tilde{n}_R \sqrt[3]{n_S}$. La Indexación de Discos cuesta 2 $t\ \tilde{n}_R \sqrt[3]{n_S}$, claramente la opción más rápida.

Puntos dinámicos

Teniendo en cuenta los resultados previos, el coste de la cache de esferas es

$$8{,}08\ t\ \tilde{n}_R \sqrt[3]{n_D} = 8{,}08\ t\ n_R \frac{r_D^2}{r_0^2} \sqrt[3]{n_D} \tag{4.13}$$

y el de la Indexación de Discos es $t\ n_R \sqrt[3]{n_D}$. Si los dos costes se usan como lados de una ecuación, y se resuelve con respecto a n_D, los valores de n_D para los cuales cada algoritmo es óptimo pueden ser deducidos.

$$8{,}08\ t\ n_R \frac{r_D^2}{r_0^2} \sqrt[3]{n_D} = t\ n_R \sqrt[3]{n_D} \tag{4.14}$$

Simplificando, (dividiendo entre $tn_R \sqrt[3]{n_D}$)

$$8{,}08 \frac{r_D^2}{r_0^2} = 1 \tag{4.15}$$

Tomando la raíz cuadrada y resolviendo para r_D obtenemos

$$r_D = 0{,}35 r_0 \tag{4.16}$$

Esto significa que si el tamaño de los objetos dinámicos es del 35 % de la escena, ambos algoritmos son igualmente rápidos en promedio. Comprobar el coste de los métodos para objetos de tamaño menor y mayor del 35 % de la escena muestra que la cache de esferas es más rápida para los objetos pequeños; mientras que la Indexación de Discos es más rápida para los objetos grandes.

Por tanto, un algoritmo de recálculo óptimo debería usar Indexación de Discos para los puntos estáticos y Caché de Esferas para los puntos dinámicos, que por la definición de escenas quasi-estáticas, son menores del 35 % de la escena.

Influencia del tamaño del móvil en la aceleración

La aceleración obtenida en el recálculo depende del número de muestras del móvil, del tamaño del móvil, y en general de los distintos parámetros de los algoritmos. Si, aparte de las suposiciones de la sección anterior, añadimos que la densidad de muestras en el móvil es la misma que en el resto de la escena, podemos observar cómo cambia la aceleración en función del tamaño del móvil.

Sea ρ la densidad de muestras. Se pueden relacionar los valores n_S y n_D de la siguiente forma:

$$n_S = \rho r_0^3 \qquad n_D = \rho r_D^3 \tag{4.17}$$

Despejando ρ y substituyendo obtenemos

$$\rho = \frac{n_S}{r_0^3} \qquad n_D = n_S \frac{r_D^3}{r_0^3} \tag{4.18}$$

Usando la ecuación 3.83 y

$$n_P = n_S + n_D = n_S + n_S \frac{r_D^3}{r_0^3} = n_S \left(1 + \frac{r_D^3}{r_0^3}\right) \tag{4.19}$$

así como las simplificaciones de la sección anterior ($t \approx u$, $\sqrt[3]{n_P} - 1 \approx \sqrt[3]{n_P}$ y $Q = 2/3$) obtenemos un tiempo para Caché de Esferas de

$$T_{SC} = \frac{97}{12} t n_R \sqrt[3]{n_S (1 + \frac{r_D^3}{r_0^3})} \tag{4.20}$$

Usando Indexación de Discos obtenemos:

$$T_{DI} = t n_R \sqrt[3]{n_S (1 + \frac{r_D^3}{r_0^3})} \tag{4.21}$$

claramente más rápido. El coste del recálculo es la suma de los costes de la parte estática y dinámica:

$$T_{Rec} = 2t\tilde{n}_R \sqrt[3]{n_S} + 8{,}08\ t\ n_R \frac{r_D^2}{r_0^2} \sqrt[3]{n_D} \tag{4.22}$$

La aceleración es el cociente entre el coste de la ecuación 4.21 y 4.22:

$$A = \frac{tn_R \sqrt[3]{n_S(1+\frac{r_D^3}{r_0^3})}}{2tn_R \frac{r_D^2}{r_0^2} \sqrt[3]{n_S} + 8{,}08\ t\ n_R \frac{r_D^2}{r_0^2} \sqrt[3]{n_S \frac{r_D^3}{r_0^3}}} \tag{4.23}$$

donde hemos usado la definición de $\tilde{n}_R$ y n_D. Simplificando queda:

$$A = \frac{\sqrt[3]{1+\frac{r_D^3}{r_0^3}}}{2\frac{r_D^2}{r_0^2} + 8{,}08\frac{r_D^3}{r_0^3}} \tag{4.24}$$

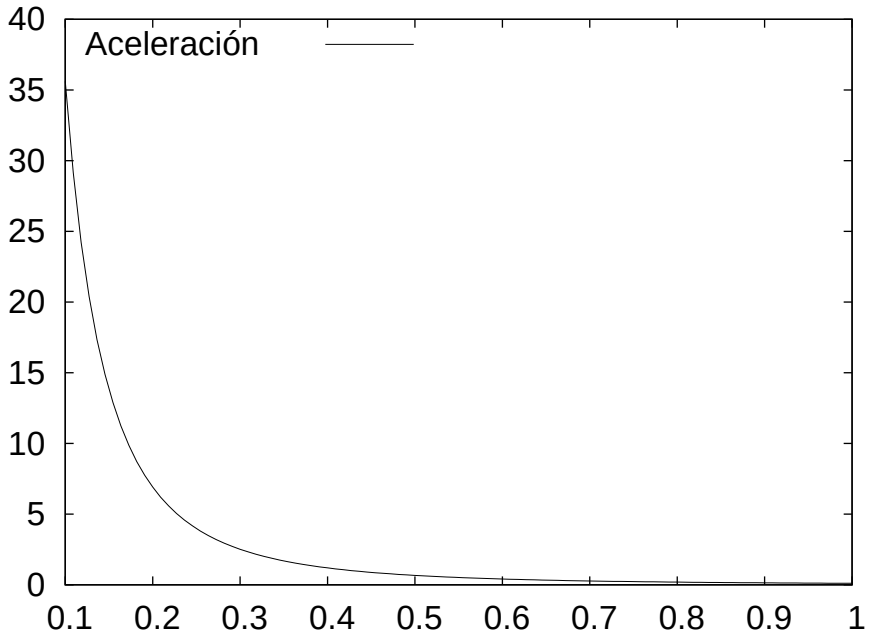


Figura 4.18: Gráfica de la aceleración teórica del recálculo, como función del tamaño del móvil (el tamaño del móvil se mide en tanto por uno del tamaño de la escena). La aceleración es el cociente entre el tiempo del algoritmo que no reusa información y el algoritmo de recálculo.

Si llamamos C al cociente entre el tamaño del móvil y el tamaño de la escena ($C = \frac{r_D}{r_0}$), vemos que la aceleración sólo depende ya de este cociente. Una gráfica puede verse en la figura 4.18.

En el campo de videojuegos, es muy usual tener dos densidades de mallado distintas, una para la escena, y otra para los personajes móviles, dado que a

menudo los personajes están cerca del protagonista y se requiere mayor calidad. En este caso, la ecuación 4.18 puede modificarse para tener este hecho en cuenta, y obtener un estimador adecuado de la aceleración.

Resultados experimentales

Para probar el estudio teórico, se diseñó una escena sencilla (la caja de Cornell) y un móvil también sencillo (una esfera). Se ha calculado la aceleración del cálculo de la iluminación global para distintos tamaños del móvil. La escena

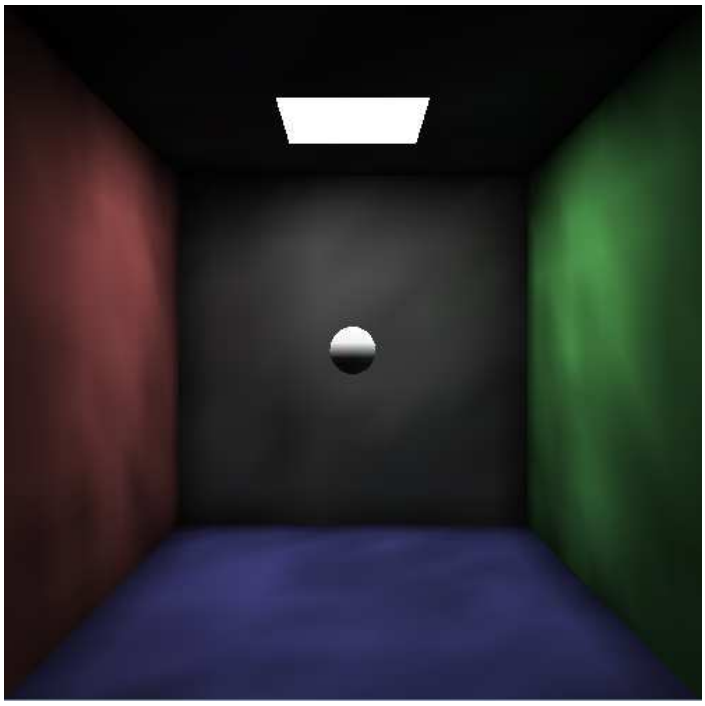


Figura 4.19: Caja de Cornell con móvil.

puede verse en la figura 4.19 para un ejemplo de móvil del 10 % de la escena. La figura 4.20 tiene los resultados empíricos.

Observamos que las gráficas de las figuras 4.18 (aceleración teórica) y 4.20 (aceleración empírica) son similares, a pesar de que muchos de los supuestos teóricos no se cumplen (la distribución de los puntos de irradiancia y de los rayos no es uniforme, y la densidad de puntos en el móvil y la escena es distinta, ya que por simplicidad hemos escalado el móvil sin cambiar la resolución de la malla). Si observamos el valor de las gráficas para un móvil con un tamaño del 30 % de la escena, vemos que la predicción teórica es de que el recálculo debería ser 2,5 veces más rápido, mientras que en la práctica es 7,5 veces más rápido.

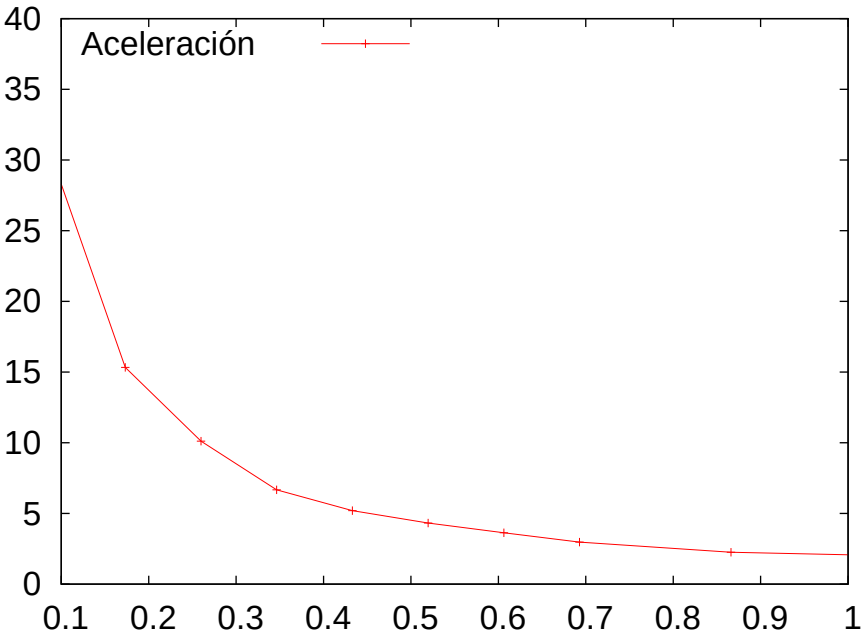


Figura 4.20: Aceleración empírica del cálculo de iluminación global, como función del tamaño del móvil (el tamaño del móvil se mide en tanto por uno del tamaño de la escena). La aceleración es el cociente entre el tiempo del algoritmo que no reusa información y el algoritmo de recálculo.

Esto se debe a que al no cambiar la resolución de la malla, el número de puntos donde se hace estimación de densidades es mucho menor que el predicho por la teoría. Con un móvil con un tamaño del 10 % de la escena, en la que el tamaño del mallado es similar entre la escena y el móvil, la aceleración en ambos casos es de un recálculo 30 veces más rápido. Podemos por tanto concluir que el estudio es útil para aproximar el comportamiento real de los algoritmos.

4.11. Recálculo en superficies no difusas

El recálculo de superficies no perfectamente difusas (o *glossy*) presenta algunos problemas adicionales que no existen en superficies difusas, que hemos analizado e implementado en nuestro sistema de iluminación global, y que describimos a continuación.

Los cambios en la posición de la cámara no modifican el estimador de radiancia con superficies difusas, pero sí con superficies *glossy*. Por tanto, hay que llamar al método de estimación cuando la cámara se mueve.

Además, el método de recálculo también se ve afectado, ya que en el caso *glossy*, un número que indique el estimador de radiancia del fotograma anterior no es suficiente. Hay que guardar el conjunto de rayos que afectaron al punto. Luego, estos rayos deben usarse junto con la nueva posición de la cámara para obtener el nuevo estimador de radiancia.

Cuando el objeto se mueve, los rayos nuevos y los antiguos que fueron usados para actualizar el estimador de radiancia en el caso difuso, han de comprobarse con la lista de rayos del vértice, y la lista de rayos debe actualizarse y calcularse un nuevo estimador de la radiancia.

Esto es bastante costoso; las escenas que podemos recalcular de forma interactiva con superficies *glossy* son bastante más sencillas que en el caso de superficies totalmente difusas.

4.12. Conclusiones

Se ha diseñado un algoritmo que calcula de forma incremental la iluminación de una escena para el caso de objetos móviles. Este algoritmo se ha aplicado a los métodos de estimación de densidades Cuenta de Impactos, Photon Maps y Estimación de Densidades en el Plano Tangente, incluyendo las dos optimizaciones de éste, y se ha comparado el rendimiento de los tres métodos. Estimación de Densidades en el Plano Tangente obtiene el resultado con menor error para un tiempo de cómputo dado. Se ha estudiado el efecto de usar pre-tests de intersección en la eficiencia del algoritmo. Para DETP básica, el aumento del rendimiento es del 34 %. Para la Caché de Esferas, sin embargo, el incremento del rendimiento es sólo del 1 %.

También se ha estudiado teóricamente el error del algoritmo de DETP y se ha comparado con los resultados empíricos. A pesar de que no se cumplen los supuestos teóricos, la diferencia entre el valor predicho del error y el valor empírico es de aproximadamente un 5 %.

Finalmente se ha estudiado teóricamente la eficiencia en tiempo del algoritmo de recálculo. La optimización de Caché de Esferas debe usarse para los objetos dinámicos y la Indexación de Discos para la parte estática, con objeto de minimizar el tiempo de cómputo. La influencia del tamaño del móvil en el tiempo del algoritmo ha sido cuantificada de forma teórica y empírica. Por último, el algoritmo de recálculo se ha aplicado también a superficies no difusas.

Capítulo 5

Software desarrollado

Para la obtención de los resultados descritos en esta memoria, se han desarrollado diversos componentes software, que han sido usados para validar la corrección de los métodos y para medir la eficiencia de los algoritmos. Este capítulo describe la estructura y funcionalidad del software.

5.1. Zeus

El sistema Zeus consiste en un software de simulación de la iluminación que incluye la siguiente funcionalidad:

- Cálculo de Iluminación Global usando métodos de Monte Carlo (Cuenta de Impactos, Photon Maps y Estimación de Densidades en el Plano Tangente con sus distintas variantes).
- Cálculo incremental de la iluminación en presencia de un móvil.
- Gran variedad de BRDFs para la definición de los materiales, que permiten entornos tanto difusos como *glossy*. Las BRDFs soportadas están descritas en profundidad en [Montes Soldado 08].
- Gran variedad de indexaciones espaciales, como octrees, kd-trees y grids, entre otros.
- Generación de imágenes de forma interactiva o por lotes.
- Soporte del formato GRanada File (GRF) tanto para la entrada (descripción de escenas y móvil) como para la salida de datos.

Este sistema implementa el algoritmo de Indexación de Discos, descrito en el capítulo 2, que mejora la velocidad del cálculo de la iluminación global cuando se usa el método DETP. También permite calcular incrementalmente la iluminación de acuerdo con lo expuesto en el capítulo 4.

Tecla	Significado
Generales	
'1'	Escribir un fichero GRF de salida
'3'	Recalcular la escena
Movimiento de cámara	
'y'	Arriba
'n'	Abajo
'g'	Izquierda
'j'	Derecha
'h'	Hacia el observador
'H'	En dirección contraria al observador
Rotación de la escena	
'a'	Derecha
'q'	Izquierda
's'	Arriba
'w'	Abajo
Movimiento del móvil	
'9'	Arriba
'l'	Abajo
'i'	Izquierda
'p'	Derecha
'o'	Hacia el observador
'O'	En dirección contraria al observador
Rotación del móvil	
'6'	En el eje X
'7'	En el eje Y
'8'	En el eje Z
Intensidad de la luz	
'm'	Aumentar
coma	Disminuir

Cuadro 5.1: Teclas del sistema Zeus.

Los parámetros de entrada del sistema se incluyen en un fichero de configuración de texto ASCII que permite elegir el método de indexación espacial, el método de cálculo de iluminación, ficheros de entrada y de salida de escena y móvil, etc. También permite elegir entre un modo interactivo y un modo para la generación de animaciones, en proceso por lotes. En el modo interactivo se puede controlar el movimiento del móvil, la cámara, etc usando teclas (Cuadro 5.1). En el modo de procesado por lotes, se le indica al sistema mediante texto los movimientos adecuados, y en su caso si se deben grabar a disco las descripciones de la escena con la iluminación global calculada, que pueden visualizarse con un programa externo. El apéndice B contiene cuadros que explican las diferentes opciones, aunque un ejemplo sencillo puede verse en el Cuadro 5.2.

```
DETP_UseFixedDisc 1
DETP_UseRayCache 1
DETP_SortQueries 1
DETP_ArtifactControl 1

DETP_PreTest_Use Plücker

DETP_Radius 100
DETP_RayCache_RadiusFactor 0.6
DETP_PluckerEdges 3

DE_Nphotons 1000000

FS_DiffuseLight 1

FS_Photons 2000000
FS_Scene 1patio.grf
FS_Mobile movil.grf
FS_Output_File 1patio.DETP.octree
FS_ShowPercentageAt 10

FS_EstimationMethod DETPR
FS_RecalculateWholeScene 1
FS_ColorCoefficientCorrector 1

FS_DensityEstimation 1
FS_Passes 2

OPT_Optimizer Octree
OPT_MaxDepth 8
OPT_MaxObjNode 25

Program_Interactive 0
Program_WalkKeys 1
Program_Seed 1
```

Cuadro 5.2: Ejemplo de fichero OP para Zeus.

El sistema permite mostrar las trayectorias de los rayos, lo que permite ver fácilmente qué rayos son recalculados en cálculo incremental de la iluminación (figura 5.1). Desafortunadamente, el número de rayos necesario para una iluminación realista (miles) hace que sea inviable observar al mismo tiempo iluminación realista y rayos usados. También se permite almacenar los datos de la indexación espacial, tanto para el cálculo eficiente de distintos algoritmos de iluminación para la misma escena, como para su visualización para propósitos

docentes usando un programa externo.

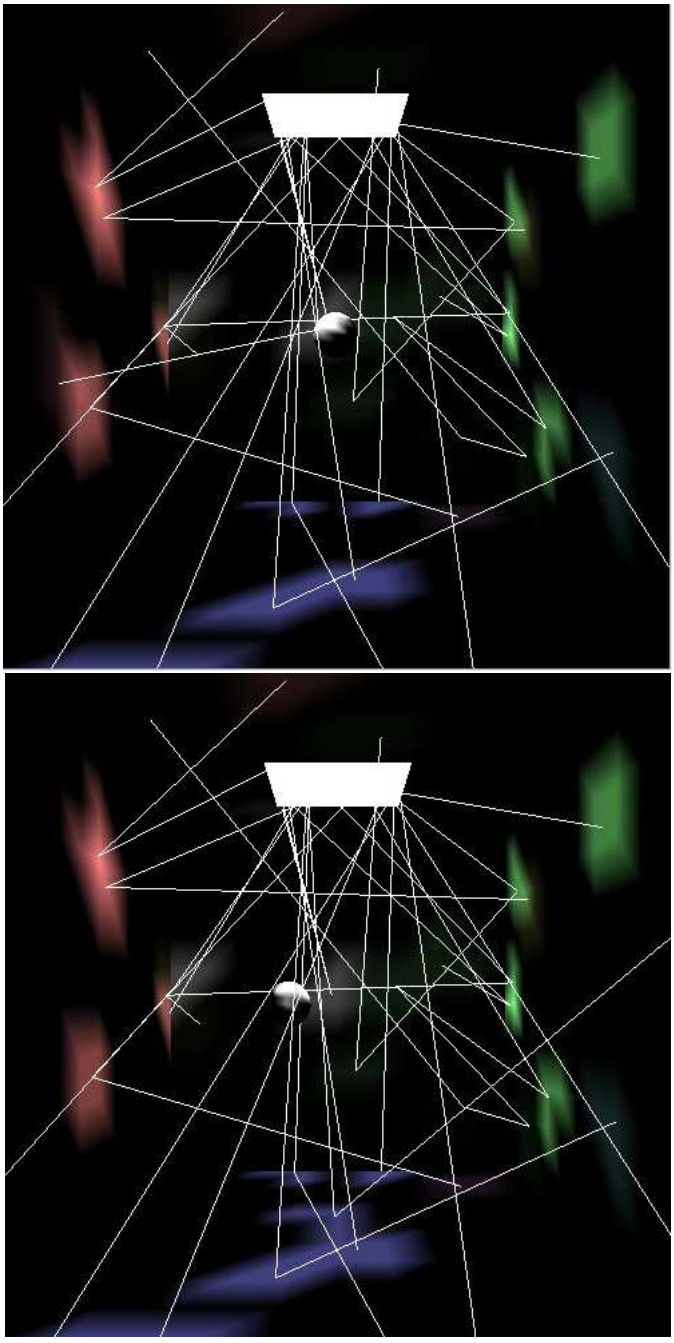


Figura 5.1: Dos instantáneas del sistema Zeus, mostrando el recálculo de los rayos, accesibles en http://giig.ugr.es/~rgarcia/tesis/zeus[12].gif. El video completo puede verse en http://giig.ugr.es/~rgarcia/tesis/movil.mpeg.

Lista	Grid 3D jerárquico	Binary Space Partitioning Tree
Grid 3D	Grid 3D recursivo	KD-Tree
Octree	Octree jerárquico	

Cuadro 5.3: Indexaciones espaciales soportadas en Zeus.

Beards-Maxwell	Ward	Strauss	Perfectamente difusa
Cook-Torrance	Phong	Blinn	Perfectamente especular
Poulin-Fournier	Lewis	Lafortune	He-Torrance-Sillion-Greenberg
Torrance-Sparrow	Slick	Minnaert	Datos Tabulados
Ashikhmin-Shirley	Shirley	Oren-Nayar	Combinaciones de las descritas

Cuadro 5.4: BRDFs soportadas en Zeus.

5.2. Arquitectura del sistema

El sistema ha sido implementado en C++ haciendo uso de la librería GLUT de utilidades de OpenGL. La arquitectura del sistema puede verse en la figura 5.2. Existen cuatro bloques principales, que contienen la funcionalidad de cálculo de iluminación global usando trazado de rayos desde la fuente de luz, materiales y reflexión de la luz en los objetos, estimación de densidades (*final gathering*), y métodos de indexación espacial para aceleración de los cálculos. Otros bloques contienen algoritmos eficientes de pre-intersección de primitivas, el código de entrada/salida, incluyendo lectura del formato de escena, parámetros, salida de resultados, e interfaz gráfica interactiva. La figura 5.3 contiene un diagrama de clases del sistema, simplificado. A continuación describimos los módulos principales, que se han implementado como *namespaces*.

- El módulo de indexación espacial contiene código que, a partir de una lista de primitivas gráficas, crear una estructura de datos que las gestiona de forma eficiente. En particular, el cálculo de la intersección de una línea con un conjunto de primitivas (triángulos y discos en el caso de este sistema) se optimiza creando una estructura recursiva que divide el espacio. Las estructuras implementadas para triángulos pueden verse en el cuadro 5.3. Cada indexación se implementa con una clase, que hereda de una clase abstracta Optimizer con la funcionalidad común. En uno de los métodos de iluminación global, es necesario repartir la energía de los rayos a los discos que atraviesan. Este módulo se encarga también de este cometido, usando el módulo de pre-intersección si es necesario. En este caso, las posibilidades son un octree o un Rectilinear Binary Space Partitioning Tree (aparte de estas posibilidades, el módulo permite generar otras estructuras de datos, aunque el sistema no las usa por el momento). La descripción completa de las indexaciones posibles puede encontrarse en [Revelles Moreno 01].

- El módulo de BRDFs contiene información sobre cómo reflejar la luz sobre

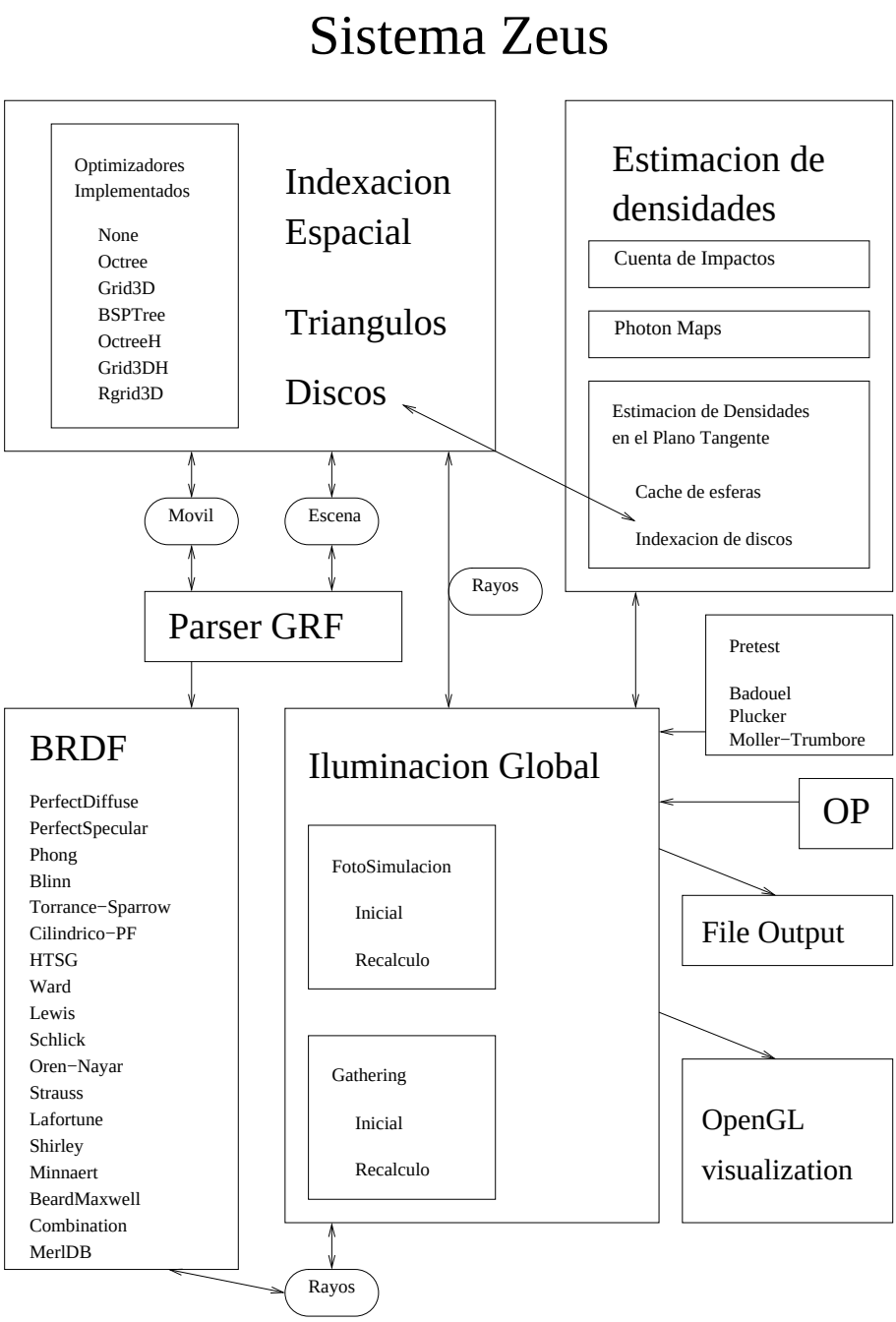


Figura 5.2: Diagrama de módulos de Zeus.

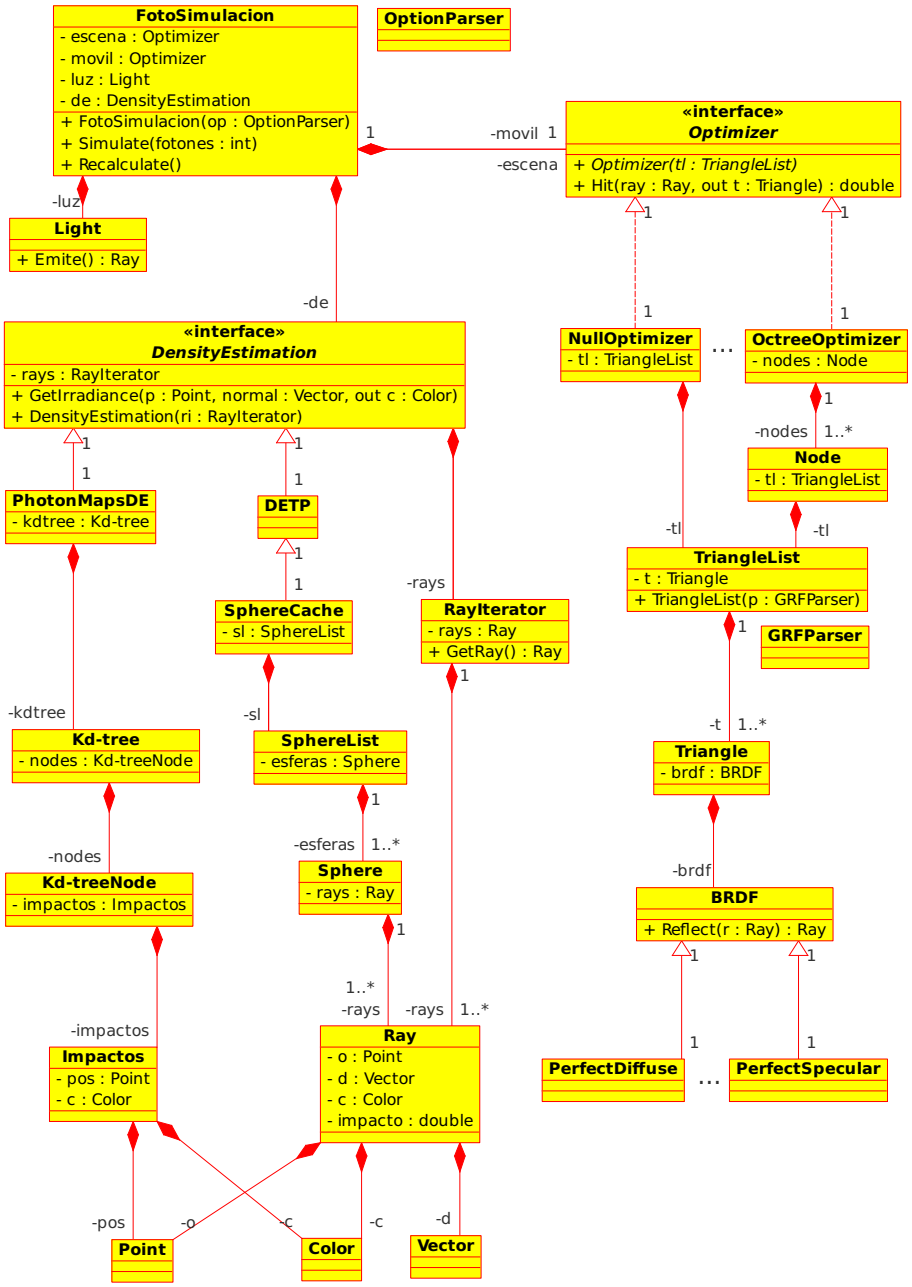


Figura 5.3: Diagrama de clases de Zeus (simplificado).

Cuenta de Impactos	Caché de Esferas, Hilbert	Indexación de Discos
Photon Maps	Caché de Esferas, Lebesgue	
DETP básica	Caché de Esferas Multilista	

Cuadro 5.5: Métodos de estimación de densidades soportados por Zeus.

distintos tipos de materiales. La reflexión de la luz se describe como una función de distribución de reflectancia bidireccional (en inglés BRDF). Las BRDFs contempladas por este módulo pueden verse en el cuadro 5.4. Cada BRDF se implementa también como una clase, que hereda de una clase abstracta BRDF. A partir de un rayo de entrada, este módulo calcula la dirección de salida del rayo (o su absorción) de acuerdo con el material en cuestión. [Montes Soldado 08] describe en detalle el proceso.

- El módulo de estimación de densidades contiene el código para obtener una estimación de la radiancia en un punto a partir de los rayos e impactos de fotones en la cercanía del punto. Las distintas posibilidades para el algoritmo de estimación pueden verse en el cuadro 5.5. La descripción de los algoritmos puede verse en [Lastra Leidinger 04] y en el capítulo 2.

- Finalmente el módulo de Iluminación Global se encarga de la simulación del transporte de la luz, y por tanto genera los fotones en las fuentes de luz, calcula el lugar donde cada fotón interacciona con un material (usando el módulo de indexación espacial) y refleja los fotones (usando el módulo de BRDFs). Tras la simulación, usa el módulo de estimación de densidades para calcular la radiancia. Este módulo también es responsable de actualizar la radiancia de modo incremental en caso necesario tras el movimiento del objeto móvil. Esta actualización usa de nuevo los módulos de indexación espacial, BRDFs y estimación de densidades.

El programa principal usa los módulos de entrada/salida para obtener los parámetros de la simulación y guardar los resultados de la simulación en disco, el módulo GRF para obtener la escena, y llama al módulo de indexación espacial para gestionar la escena y al de iluminación global para hacer la simulación. La visualización OpenGL muestra la escena y recoge la interacción con el usuario, en caso necesario (la interacción puede obtenerse también en modo batch a partir del fichero de configuración).

La figura 5.4 muestra un diagrama de secuencia con la ejecución de Zeus para una escena y un móvil, en la que se usa indexación de discos. La interacción se realiza a través de OpenGL. Por simplicidad, se supone una única interacción en la que se cambia la posición del móvil. Tras la visualización de la actualización de la iluminación, el programa termina.

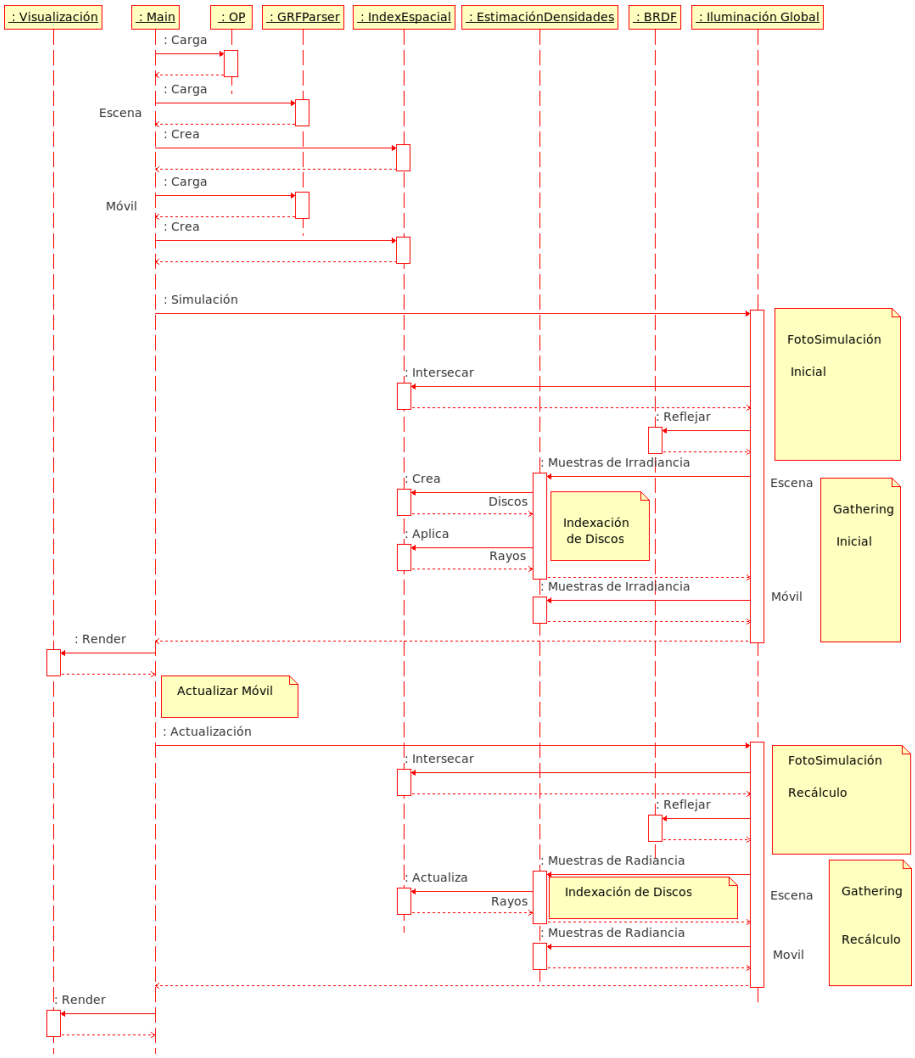


Figura 5.4: Diagrama de secuencia de Zeus.

5.3. Uso del sistema

La sintaxis del programa es:

```
zeus <fichero de configuración>
```

Una ejecución típica del sistema comienza con la lectura del fichero de configuración. Si se desea una simulación interactiva, el sistema inicializa la interfaz gráfica; en caso contrario se obtiene la lista de operaciones a realizar.

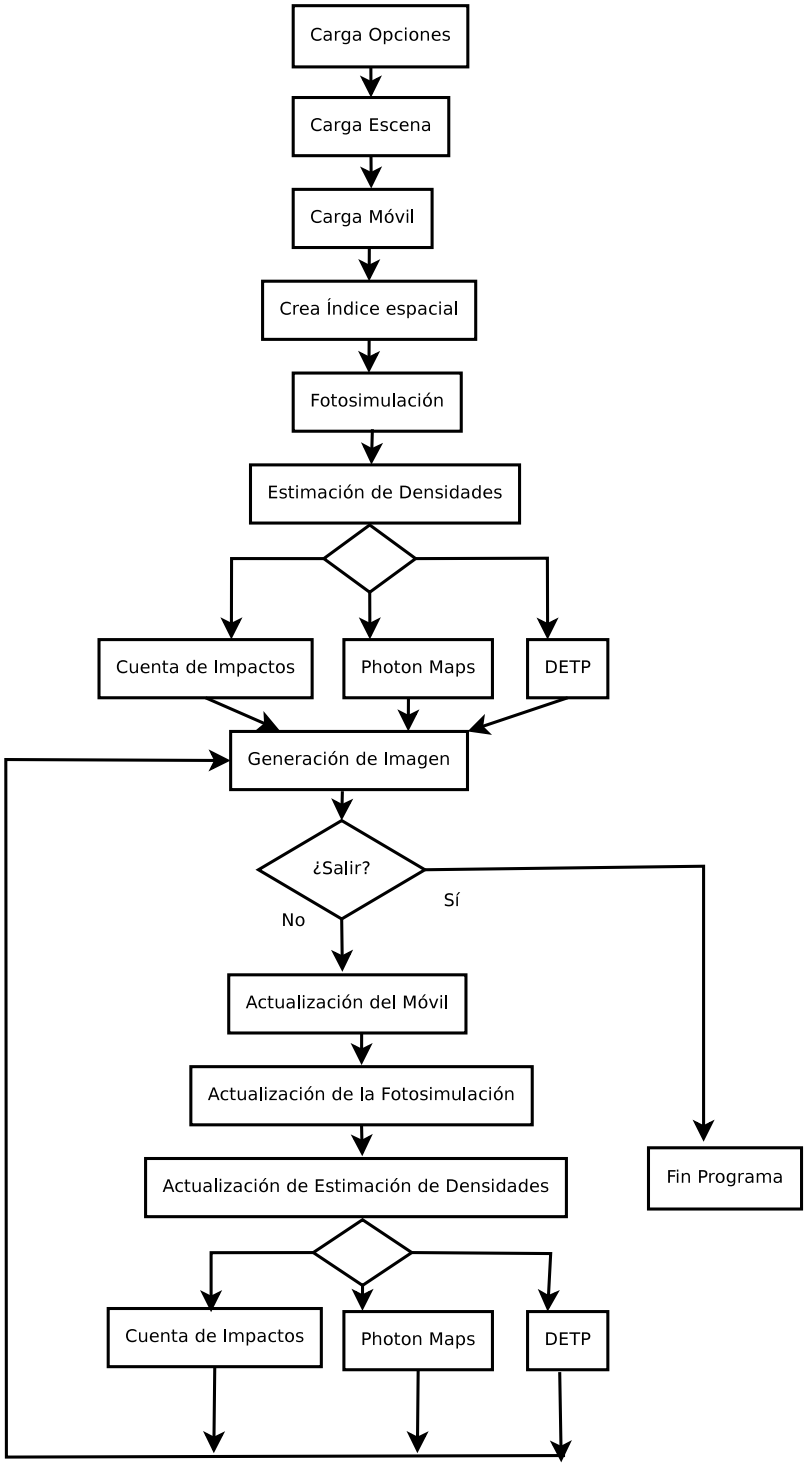


Figura 5.5: Diagrama de flujo de Zeus.

A continuación se carga la escena y se genera la indexación espacial requerida, y se produce una primera fotosimulación, seguida de la estimación de densidades adecuada.

Un bucle procesa las peticiones del usuario (movimiento del móvil y recálculo correspondiente de la iluminación, movimiento de cámaras, generación de imágenes de salida, etc.) hasta que se desee terminar la simulación. La figura 5.5 contiene un diagrama de flujo simplificado.

Si no se desea recálculo, se pueden generar imágenes usando más rayos de los que cabrían en memoria aprovechando el hecho de que la energía en los puntos de irradiancia es independiente del orden en que se procesen los rayos. En este caso se producen lotes de rayos que se procesan por separado añadiendo su contribución a los puntos de irradiancia. Esto puede considerarse una forma muy básica de procesamiento out of core.

El sistema, a petición del usuario, genera una serie de ficheros de salida con formato IGRF (GRF con información de irradiancia). Estos ficheros contienen una cabecera con los parámetros de la simulación. Tener un formato definido permite leer fácilmente mediante software estos ficheros, por ejemplo para generar automáticamente gráficas leyendo un conjunto de ficheros resultantes de la ejecución de un algoritmo con distintos parámetros. El formato puede verse en los cuadros 5.6 y 5.7.

5.4. Trabajo futuro

Para mejorar la eficiencia de este sistema se pretenden explorar los siguientes campos:

- Paralelización del sistema usando las librerías MPI para obtener un software de memoria distribuida ejecutable en un cluster.
- Implementación de los métodos de iluminación global en hardware gráfico avanzado usando o bien Cg o bien CUDA.
- Integración de ambos aspectos para la ejecución en un cluster de GPUs.
- Añadir soporte para guardar y cargar conjuntos de rayos en disco para permitir recálculo en situaciones que requieran algoritmos out of core.
- Integrar el algoritmo eficiente de muestreo para BRDFs descrito en [Montes 08a, Montes 08b].

```
//Timing data
//Illumination time: <float>
//Irradiance time: <float>
//Program Parameters
//Program_MobilePosition: <vector>
//Program_Interactive: [0|1]
//PhotoSimulagion parameters
//FS_DiffuseLight: [0|1]
//FS_Passes: <int>
//FS_Photons per pass: <int>
//Total Rays per pass (actually last pass): <int>
//FS_ShowPercentageAt: <float>
//FS_EstimationMethod: <método de estimación de densidades>
//RecalculateWholeScene: [0|1]
//FS_ColorCoefficientCorrector: <float>
//FS_Output_File: <fichero de salida>
//FS_Scene: <fichero de entrada de escena>
//FS_Mobile: <fichero de entrada del móvil>
//Density Estimation options
//DE_Nphotons: <int>
//DE_Radius: <float>
<opciones DETP si se usó el método DETP>
//Spatial Indexing Optimizing options
//OPT_Optimizer: <optimizador>
```

Cuadro 5.6: Cabecera de un fichero IGRF generado por Zeus.

```
//DETP_UseFixedDisc: [0|1]
//DETP_MaxRadius: <float>
//DETP_ArtifactControl: [0|1]
//DETP_PretestUse: <Plücker|Möller|Badouel|None>
//DETP_UseRayCache: [0|1]
//DETP_RayCache_RadiusFactor: <float>
//DETP_RayCache_RaysMax: <int>
//DETP_SortQueries: [0|1]
//DETP_UnSortQueries: [0|1]
//DETP_SortQueriesHilbert: [0|1]
//DETP_RayCache_Width: <int>
//DETP_RayCache_MaxLevels: <int>
//DETP_UseSphereLimitedRayCache: [0|1]
//DETP_SphereSortingMethod: Basic
//DETP_SphereLimit: <int>
//DETP_IrradianceSamplesOnGlossy: <int>
//DETP_UseRayDiscretization: [0|1]
//Ray Cache results <using Hilbert sorting|using Lebesgue sorting|
            |using explicit unsorting|without sorting>
   max. width == <int>
   max. levels == <int>
   max. spheres simult. == <int>
   num. queries == <int>
   num. cache faults == <int>
   prop. faults/queries == <float> %
   avg. rays in balls == <float
   faults, spheres, mean number of rays and sum of rays when
            fault at each level ==
      Level 0: <int>,<int>,<int>.<float>,<int>
                        ⋮
      Level <int>: <int>,<int>,<int>.<float>,<int>
```

Cuadro 5.7: Opciones DETP.

Capítulo 6

Conclusiones, aportaciones y trabajo futuro

6.1. Conclusiones y aportaciones

Las principales aportaciones de esta tesis son:

- El estudio tanto teórico como empírico de distintas técnicas de iluminación global basadas en métodos estocásticos (Cuenta de Impactos, Photon Maps, DETP y Ray Maps). El objeto del estudio ha sido por una parte los parámetros de sesgo o error, y por otra parte la eficiencia en tiempo.

- El diseño, implementación y estudio de nuevas optimizaciones de los algoritmos (Indexación de Discos).

- El diseño, implementación y estudio de formas eficientes para el cálculo incremental de la iluminación para escenas quasi-estáticas.

Como resultados concretos, podemos destacar lo siguiente:

- El capítulo uno contiene un resumen de los métodos de iluminación global más importantes, con especial énfasis en algoritmos de Monte Carlo eficientes.

- El capítulo dos describe el método de Indexación de Discos, que usa técnicas de indexación espacial de las muestras de irradiancia para mejorar el rendimiento de la Estimación de Densidades en el Plano Tangente, a costa de incrementar el uso de memoria. Las ventajas del método son las siguientes:

    - Gran mejora de tiempos para imágenes de alta calidad con sesgo muy reducido.

  - Mejora de tiempos también cuando el número de rayos es pequeño (por ejemplo para los casos en que se desee obtener un boceto rápido y de baja calidad de la iluminación global de una escena).
  - Permite estimar nuevas muestras de irradiancia por interpolación.

- El capítulo tres aporta un estudio teórico de la convergencia, el sesgo, la variancia y la complejidad de las técnicas de Cuenta de Impactos, Photon Maps, DETP y Ray Maps.

  - Se ha obtenido una cota de la varianza de los distintos algoritmos.
  - Se ha derivado el coste en tiempo de los distintos algoritmos usando la notación $O()$. Las distintas variantes de DETP (Caché de Esferas, Caché de Esferas Multilista, e Indexación de Discos) han sido también estudiadas.
  - Se han encontrado teóricamente valores óptimos para los distintos parámetros del algoritmo DETP y de la Caché de Esferas.

  Este estudio teórico, no sólo permite a un desarrollador comprobar a priori cuál de las técnicas estudiadas en esta memoria es más acorde con sus necesidades, sino que proporciona un marco de trabajo para estudiar y comparar otros algoritmos, ya existentes o que se desarrollen en un futuro, tanto en sus parámetros de error y varianza, como en los de su eficiencia en tiempo. Específicamente, es de destacar que el estudio teórico de Photon Maps permitió descubrir y corregir un sesgo en el método descrito en la literatura, en el algoritmo de estimación de la radiancia.

  El estudio empírico de las técnicas valida la comparación teórica entre los distintos algoritmos y demuestra que son útiles en aplicaciones reales. Ambos enfoques son complementarios.

- El capítulo cuatro presenta un algoritmo de cálculo incremental de la iluminación que permite reducir los tiempos en gran medida para escenas con móviles pequeños. El algoritmo de Indexación de Discos mencionado antes es útil en la actualización de la iluminación de la parte estática de la escena.

  Se ha realizado un estudio teórico de este algoritmo, obteniéndose los siguientes resultados:

  - Se ha estudiado teóricamente la influencia del tamaño del móvil en la aceleración.
  - Se ha comparado la predicción teórica del error del algoritmo DETP con el valor empírico.

  El cálculo incremental de iluminación se puede utilizar para aumentar el realismo de aplicaciones interactivas. Por ejemplo, en diseño gráfico es muy común trabajar incrementalmente, añadiendo objetos uno a uno y

moviéndolos a su posición final. La ganancia en tiempo del cálculo incremental de iluminación permite utilizar un cálculo correcto de iluminación global para aumentar el realismo durante el proceso de modelado, consiguiéndose una mayor facilidad de uso del software. Esto es especialmente importante en el caso de diseño en arquitectura, ya que la iluminación correcta de interiores es fundamental en este campo.

Otra aplicación muy extendida son los juegos en primera persona. Existe un fuerte crecimiento del grado de realismo de las imágenes de este tipo de juegos. Este método también está muy indicado en este caso, ya que los personajes suelen ser pocos y estar incluídos en un escenario de complejidad alta, lo que se corresponde perfectamente con el algoritmo.

- Finalmente el capítulo cinco documenta el sistema software de cálculo de iluminación global que implementa los conceptos descritos en esta memoria.

6.2. Trabajo futuro

Durante el desarrollo de esta tesis han aparecido distintos temas de trabajo que merecen ser estudiados con detalle. En esta sección comentamos brevemente los más importantes.

Implementación SIMD y GPU de las técnicas propuestas
Existe una implementación que usa paralelismo de datos para mejorar el rendimiento de la Estimación de Densidades en el Plano Tangente. Esta implementación usa las instrucciones SSE de Intel para los cálculos que se realizan en la CPU y código Cg para los cálculos que se realizan en la tarjeta gráfica, y permite comparar el rendimiento de la CPU y la GPU [Lastra 08].

Usando esta implementación como base, se ha diseñado un sistema que integra las ventajas de SIMD y GPU con las mejoras algorítmicas presentes en esta tesis, que se encuentra en fase de implementación. Usar SSE duplica la velocidad del código, y las tarjetas gráficas programables pueden ejecutar la estimación de densidades cinco veces más rápido que la versión SSE, obteniendo una ganancia global de 10x. Ésto, unido a la mejora debida al cálculo incremental, resulta en una reducción de tiempos de dos órdenes de magnitud. Por tanto, esperamos obtener Iluminación Global en tiempo real para escenas de complejidad grande.

Implementación de la indexación espacial y la fotosimulación en la GPU
Los algoritmos vectoriales usados en la indexación espacial y la reflexión de los rayos en las superficies son fácilmente portables a una arquitectura de tarjetas gráficas programables. Es bastante probable que el rendimiento del algoritmo mejore sustancialmente en esta arquitectura, si el resto de programas portados es una buena indicación.

Además, el hecho de tener el sistema de iluminación implementado entero en la GPU elimina muchas transferencias CPU-GPU que ralentizan el algoritmo. La indexación espacial propuesta por Noguera et al [Noguera 07, Noguera 09], que funciona muy bien con los rayos no coherentes que se generan en los algoritmos de iluminación global y que es fácilmente portable a la GPU, sería una buena base.

Mejoras del algoritmo de recálculo
Existen varios posibilidades para mejorar el algoritmo de recálculo. Miguel Lastra expone en su tesis doctoral un método para resumir los rayos que permite mejorar en gran medida los tiempos de cómputo para escenas con superficies *glossy* [Lastra Leidinger 04]. Integrar este método con nuestro método de cálculo incremental de la iluminación permitiría poder usar escenas glossy con tiempos de respuesta interactivos. Es necesario sin embargo estudiar cómo recalcular los rayos resumidos ya no válidos y cómo añadir rayos nuevos, para evitar la pérdida de calidad del resumen de rayos.

Por otra parte, hemos visto que el algoritmo de recálculo reusa la ordenación de puntos. En el caso de un móvil deformable, se debería estudiar si el tiempo de preproceso para recalcular el orden es menor que el incremento en fallos de caché al reusar el orden, y no reusar la ordenación de puntos si no resulta rentable (Es de notar que la deformación del móvil siguen reglas que conservan la cercanía de las partes; una persona que camina es un ejemplo útil).

Finalmente, sería interesante implementar un algoritmo híbrido que combine el algoritmo de recálculo descrito en el capítulo 4 con el de reuso de caminos para animación de fuentes de luz [Sbert 04], descrito en la sección 1.5.3, que tenga las ventajas de ambos (posibilidad de objetos y fuentes de luz móviles).

Aplicación a escenas exteriores con vegetación procedural
Los estudios comparativos de DETP y Photon Maps demuestran que DETP está especialmente indicado en geometrías complejas con objetos que tengan grandes curvaturas y con una fracción significativa de bordes. Este tipo de geometría aparece en el modelado de árboles y otros objetos naturales. El hecho de que DETP sea independiente de geometría lo hace especialmente indicado para el caso de árboles procedurales generados usando gramáticas, ya que la geometría se genera en tiempo de ejecución y no existe explícitamente.

Dentro de esta línea de trabajo se ha diseñado un sistema que integra los algoritmos de generación de vegetación usando sistemas de Lindenmayer mencionados en [González 01], y se ha comenzado la implementación de un prototipo.

Estudio teórico
Cuando se diseñan algoritmos dentro del campo de Gráficos por Ordenador, es conveniente compararlos con algoritmos existentes para demostrar que son mejores que el resto de técnicas en algún sentido. Normalmente se compara el coste y el error de la técnica nueva con repecto a las anteriores. El error de un algoritmo se calcula normalmente usando la media de los errores generados por

el algoritmo en cada píxel, usando una imagen de referencia. El problema de este enfoque es que el coste de calcular la imagen de referencia es normalmente dos o más órdenes de magnitud más alto que el coste del algoritmo.

En otros campos de la física, el procedimiento estandar es calcular un valor para la desviación estandar σ además del resultado del algoritmo μ, y devolver tres valores: $\mu - k\sigma$, μ y $\mu + k\sigma$, donde k se elige para que la solución real esté en el intervalo $[\mu - k\sigma, \mu + k\sigma]$ con una probabilidad determinada (el intervalo de confianza). Con este enfoque, no se necesita calcular una imagen de referencia, así que se evita un gran coste; pero ha de determinarse la varianza de los distintos algoritmos.

Aunque este procedimiento puede usarse para cualquier algoritmo gráfico estocástico, nos gustaría estudiar si puede usarse la estimación de la varianza de los algoritmos de iluminación global presentada en el estudio teórico para calcular las imágenes $\mu - k\sigma$, μ y $\mu + k\sigma$ para mostrarlas como una medida del error del algoritmo.

Por otra parte, el estudio teórico supone una distribución uniforme de rayos y puntos de irradiancia que no son ciertos en la práctica. Hemos visto que aunque los supuestos no se cumplan, las predicciones teóricas resultan bastante acertadas en la práctica. Sería interesante intentar obtener resultados teóricos para otras distribuciones de rayos y puntos.

Finalmente, siguiendo los pasos del capítulo 3 es posible estudiar los parámetros de convergencia, sesgo, varianza y complejidad de otros algoritmos de estimación de densidades que se vayan publicando en el futuro. Esto conllevará una comprensión más profunda de las ventajas e inconvenientes de cada algoritmo.

Apéndice A

Cuadros de tiempos

En este apéndice se presentan los cuadros de tiempos que corresponden a los capítulos 2 (Indexación de Discos) y 4 (Cálculo incremental de la iluminación).

A.1. Comparativa de tiempos entre Indexación de Discos y Caché de Esferas

Aquí se presentan los cuadros con los tiempos de cómputo de indexación de discos y la Caché de Esferas para distintas profundidades del octree, para distintos tamaños de disco y distinto número de fotones. El tiempo de ordenación de puntos para la Caché de Esferas es de 0.03 s. El tiempo de inicialización para Indexación de Discos puede verse en el cuadro A.1. El tiempo para calcular la iluminación usando las distintas técnicas puede verse en los cuadros A.2, A.3, A.4, A.5 y A.6. Cada cuadro contiene los datos para un disco de tamaño distinto: 1 %, 2 %, 4 %, 8 % y 16 % de la escena. Las posiciones que contienen las siglas OOM (*Out Of Memory*) indican que el método se quedó sin memoria durante su ejecución. Estos cuadros corresponden a las figuras 2.6 a 2.10 (páginas 53 a 55).

Radio	**Profundidad**			
	5	**6**	**7**	**8**
1 %	91,18	122,56	174,11	263,12
2 %	108,18	157,46	246,09	431,02
4 %	139,20	223,07	396,84	842,32
8 %	198,96	349,20	720,43	OOM
16 %	291,68	572,25	OOM	OOM

Cuadro A.1: Tiempo de creación del árbol de discos usando un octree.

Fotones	Indexación de Discos				Caché de Esferas
	5	**6**	**7**	**8**	
100	0,05	0,05	0,04	0,04	1,43
200	0,07	0,06	0,05	0,04	1,40
400	0,15	0,10	0,09	0,07	1,52
800	0,29	0,17	0,14	0,10	1,45
1600	0,60	0,34	0,28	0,19	1,54
3200	1,08	0,59	0,49	0,33	1,79
6400	2,02	1,10	0,90	0,06	1,95
12800	4,10	2,22	1,79	1,18	2,54
25600	8,16	4,40	3,57	2,33	3,83
51200	16,38	8,81	6,12	4,66	6,61
102400	32,88	17,35	11,88	9,24	13,39
204800	65,34	34,50	23,56	18,24	29,81
409600	130,89	68,27	46,46	37,38	73,97

Cuadro A.2: Tiempo de Indexación de Discos y Caché de Esferas para un radio de 1 % de la escena.

Fotones	Indexación de Discos				Caché de Esferas
	5	**6**	**7**	**8**	
100	0,09	0,06	0,05	0,06	1,43
200	0,12	0,09	0,07	0,06	1,42
400	0,28	0,18	0,16	0,12	1,47
800	0,55	0,34	0,30	0,22	1,57
1600	1,22	0,73	0,66	0,44	1,74
3200	2,19	1,30	1,15	0,77	1,99
6400	4,23	2,49	2,15	1,47	2,53
12800	8,47	4,98	4,34	2,95	3,71
25600	17,14	10,03	8,71	5,86	6,34
51200	33,79	20,04	14,56	11,82	12,87
102400	68,42	39,30	28,78	23,41	31,44
204800	133,48	78,06	56,75	46,81	83,74
409600	269,69	157,11	113,64	94,51	206,46

Cuadro A.3: Tiempo de Indexación de Discos y Caché de Esferas para un radio de 2 % de la escena.

Fotones	Indexación de Discos				Caché de Esferas
	5	6	7	8	
100	0,19	0,13	0,10	0,10	1,50
200	0,29	0,19	0,16	0,13	1,51
400	0,72	0,47	0,42	0,29	1,64
800	1,50	0,96	0,87	0,60	1,83
1600	3,32	2,16	1,91	1,3	2,35
3200	5,99	3,88	3,37	2,39	3,09
6400	11,52	7,36	6,63	4,47	4,59
12800	23,39	15,01	13,02	9,07	8,40
25600	46,71	29,98	26,03	18,12	19,39
51200	94,54	61,86	43,76	36,34	55,14
102400	188,91	119,27	86,44	72,61	147,74
204800	375,55	235,28	170,81	171,66	318,71
409600	742,83	472,22	341,86	287,64	640,38

Cuadro A.4: Tiempo de Indexación de Discos y Caché de Esferas para un radio de 4 % de la escena.

Fotones	Indexación de Discos				Caché de Esferas
	5	6	7	8	
100	0,60	0,60	0,33	OOM	1,66
200	0,87	0,87	0,55	OOM	1,74
400	2,18	2,18	1,59	OOM	2,02
800	4,64	4,64	3,3	OOM	2,64
1600	10,30	10,30	7,34	OOM	4,01
3200	18,99	18,99	13,30	OOM	6,34
6400	37,11	37,11	25,25	OOM	13,83
12800	74,77	74,77	51,84	OOM	46,02
25600	150,01	150,01	103,62	OOM	120,42
51200	300,56	300,56	173,33	OOM	252,62
102400	599,06	599,06	347,36	OOM	500,89
204800	1193,04	1193,04	693,22	OOM	990,66
409600	2391,72	2391,72	1385,03	OOM	1977,53

Cuadro A.5: Tiempo de Indexación de Discos y Caché de Esferas para un radio de 8 % de la escena.

Fotones	**Indexación de Discos**				Caché de Esferas
	5	**6**	**7**	**8**	
100	2,00	1,68	OOM	OOM	2,02
200	3,28	2,69	OOM	OOM	2,26
400	7,61	6,49	OOM	OOM	3,31
800	15,92	13,65	OOM	OOM	5,20
1600	33,87	29,09	OOM	OOM	9,08
3200	65,83	55,75	OOM	OOM	22,24
6400	126,77	107,92	OOM	OOM	72,92
12800	257,67	218,84	OOM	OOM	182,15
25600	510,24	434,50	OOM	OOM	340,86
51200	1039,69	884,14	OOM	OOM	669,33
102400	2083,97	1712,97	OOM	OOM	1324,89
204800	4103,47	3459,76	OOM	OOM	2629,59
409600	8176,12	6851,63	OOM	OOM	5256,20

Cuadro A.6: Tiempo de Indexación de Discos y Caché de Esferas para un radio de 16 % de la escena.

A.2. Comparación entre Indexación de Discos y Caché de Esferas en Recálculo

El cuadro A.7 contiene los tiempos de recálculo para la escena del árbol (figura 4.5, página 108), para los algoritmos de Caché de Esferas e Indexación de Discos. El radio del disco toma los valores del 1 %, 2 %, 3 % y 4 % de la escena, y se ha distinguido entre el tiempo de la escena completa, sólo el árbol y sólo el suelo, con una simulación de 10 000 fotones. Este cuadro corresponde a la figura 4.7 de la página 110. Los cuadros A.8 y A.9 contienen los tiempos correspondientes usando 20 000 fotones, para la Escena del Árbol y la Segunda Escena del Árbol, respectivamente, y corresponden a la figura 4.8 de la página 111.

Algoritmo \ Radio	0,01	0,02	0,03	0,04
Caché de Esferas (Toda la escena)	4,45	5,70	7,32	9,24
Indexación de Discos (Toda la escena)	1,46	3,43	7,18	17,79
Caché de Esferas (Árbol)	4,03	5,34	6,78	8,38
Indexación de Discos (Árbol)	0,89	2,78	7,15	18,37
Caché de Esferas (Suelo)	0,39	0,46	0,62	0,74
Indexación de Discos (Suelo)	0,21	0,33	0,50	0,68
Área del disco	1,57	6,28	14,14	25,13

Cuadro A.7: Comparación entre la Caché de Esferas y la Indexación de Discos para diferentes escenas. Tiempo como función del radio del disco. 10 000 fotones.

Algoritmo \ Radio	0,01	0,02	0,03	0,04
Caché de Esferas (Escena completa)	5,50	8,16	11,38	15,89
Indexación de Discos (Escena completa)	2,89	6,82	14,12	41,95

Cuadro A.8: Comparación entre Caché de Esferas e Indexación de Discos en el algoritmo de recálculo. Escena del Árbol, 20 000 fotones. Tiempos en segundos.

Algoritmo \ Radio	0,01	0,02	0,03	0,04
Caché de Esferas (Escena completa)	2,28	2,86	3,79	7,83
Indexación de Discos (Escena completa)	1,34	2,17	3,22	10,29

Cuadro A.9: Comparación entre Caché de Esferas e Indexación de Discos en el algoritmo de recálculo. Segunda Escena del Árbol, 20 000 fotones. Tiempos en segundos.

A.3. Tablas de tiempos relacionadas con el cálculo incremental de la iluminación

El cuadro A.10 contiene el tiempo para la inicialización y el recálculo de iluminación para los tests de pre-intersección de Badouel, Möller-Trumbore y Plücker, y el tiempo cuando no se usa pre-test, para la escena del árbol. Este cuadro corresponde a la figura 4.10 de la página 114.

El cuadro A.11 contiene los tiempos de fotosimulación y estimación de densidades para el algoritmo de Photon Maps, tanto para la inicialización como para los fotogramas sucesivos. También se muestra el error obtenido en cada simulación. Este cuadro corresponde a las figuras 4.14, 4.15 y 4.16 (página 116). Los cuadros A.12 y A.13 contienen los datos para el método de Estimación de Densidades en el Plano Tangente, para 1 000 y 10 000 fotones, respectivamente. Se distinguen las distintas optimizaciones del método y se muestra también el error obtenido. Estos cuadros corresponden a las figuras 4.12 y 4.13 de la página 115. Estos datos corresponden a la gráfica de la figura 4.17 de la página 118.

Finalmente el cuadro A.14 contiene el tiempo para calcular la iluminación de la escena completa y el tiempo de recálculo para distintos tamaños de móvil. Se usó un móvil esférico, cuyo radio es el porcentaje indicado del ancho de la escena. El cuadro corresponde a la figura 4.20 de la página 124.

a) Tiempo para simular la Escena del Árbol usando el test de pre-intersección de Badouel

Fase	Inicialización	Fotograma
Fotosimulación	10,34	1,02
Iluminación	679,44	67,90
Total	689,78	68,92

b) Tiempo para simular la Escena del Árbol usando el test de pre-intersección de Möller-Trumbore

Fase	Inicialización	Fotograma
Fotosimulación	10.29	0.98
Iluminación	666.34	67.27
Total	676.63	68.25

c) Tiempo para simular la Escena del Árbol usando el test de pre-intersección de Plücker

Fase	Inicialización	Fotograma
Fotosimulación	10,40	1,05
Iluminatión	716,47	68,66
Total	726,87	69,71

d) Tiempo para simular la Escena del Árbol sin usar tests de pre-intersección

Phase	Inicialización	Fotograma
Fotosimulación	10,38	1,05
Iluminación	678,75	68,00
Total	689,13	69,05

Cuadro A.10: Tiempo para simular la Escena del Árbol usando distintos tests de pre-intersección.

Fotones	Fotosimulación		Estimación de Densidades	
	Inicialización	Fotograma	Inicialización	Fotograma
10 000	0,66	0,02	0,66	0,18
50 000	3,14	0,13	0,86	0,34
200 000	12,51	0,50	2,20	1,79

Fotones	Ruido	Sesgo	Total
10 000	172,5 %	62,80 %	54,39 %
50 000	93,2 %	62,80 %	62,46 %
200 000	62,2 %	62,80 %	65,72 %

Cuadro A.11: Photon Maps.

Tiempo de fotosimulación (primer fotograma) 0,09
Actualización de fotosimulación 0,00

DETP básica		Caché de esferas			
		Sin Ordenación		Con Ordenación	
Inicialización	26,14	Inicialización	2,56	Inicialización	2,56
Fotograma	0,29	Fotograma	0,26	Fotograma	0,11

Error de Sesgo	Error de Ruido	Error Total
14,403 %	24,64 %	28,6912 %

Cuadro A.12: DETP, 1 000 fotones.

Tiempo de Fotosimulación (primer fotograma) 0.8 seconds
Actualización de Fotosimulación 0,02-0,03 segundos

DETP básica		Caché de Esferas			
		Sin Ordenación		Con Ordenación	
Inicialización	465,46	Inicialización	66,15	Inicialización	64,0
Fotograma	3,17	Fotograma	0,61	Fotograma	0,46

Error de Sesgo	Error de Ruido	Error Total
14,40 %	8,57 %	16,82 %

Cuadro A.13: DETP, 10 000 fotones.

Tamaño del móvil	Tiempo (Escena)	Tiempo (Recalculo)
1,25 %	0,90	0,03
2,50 %	0,92	0,03
5,00 %	0,92	0,06
7,50 %	0,91	0,09
10,00 %	1,00	0,15
12,50 %	1,04	0,20
15,00 %	1,08	0,25
17,50 %	1,09	0,30
20,00 %	1,16	0,39
25,00 %	1,26	0,56
50,00 %	1,87	1,63

Cuadro A.14: Tiempo de recálculo para distintos tamaños del móvil.

Apéndice B

Documentación del sistema Zeus

Aquí se presentan una serie de cuadros con las diferentes opciones de configuración del sistema Zeus, clasificadas de acuerdo a la parte del software a la que afectan. El cuadro B.1 contiene las opciones relacionadas con el programa principal. El cuadro B.2 contiene las opciones relacionadas con la fotosimulación. El cuadro B.3 contiene parámetros que afectan a la estimación de densidades en general, mientras que los cuadros B.4 y B.5 comentan las opciones relacionadas con Estimación de Densidades en el Plano Tangente. Finalmente el cuadro B.6 muestra las opciones relacionadas con optimizadores para indexación espacial y el cuadro B.7 contiene las opciones para generar imágenes usando path tracing.

Opción	**Tipo**	**Valor por defecto**
Descripción		
Program_MobilePosition	Vector	(0,0,0)
Coordenadas del origen del móvil con respecto al origen de la escena		
Program_DrawRays	Boolean	false
Indica si se debe dibujar la trayectoria de los fotones		
Program_Interactive	Boolean	false
Indica si se debe mostrar la ventana del visor y permitir interacción con la escena		
Program_Seed	Entero	time(0)
Semilla para el generador de números aleatorios del programa		

Cuadro B.1: Program: Opciones relacionadas con el programa principal.

Opción	Tipo	Por defecto	Otros valores
Descripción			
FS_DensityEstimation	Boolean	true	
Realizar la estimación de densidades			
FS_DiffuseLight	Boolean	true	
Usar luces difusas. Cuando está a false se usan luces direccionales			
FS_Photons	Entero	20	
Número de fotones para la fotosimulación			
FS_ShowPercentageAt	Double	10	
Cada cuánto mostrar información de estado			
FS_EstimationMethod	String	DETPR	PhotonMap ImpactCount BPT
Método de estimación de densidades			
FS_RecalculateWholeScene	Boolean	true	
Recalcular escena completa o sólo móvil			
FS_ColorCoefficientCorrector	Double	1	
Este valor se multiplica por el color resultado de la fotosimulación			
FS_Output_File	String	dump	
A esta cadena se le añade el número de fotograma (con 4 cifras) y extensión .igrf para conseguir el fichero de salida del programa			
FS_Scene	String	empty.grf	
Fichero GRF con la parte fija de la escena. El fichero empty.grf es un fichero temporal con una escena vacía			
FS_Mobile	String	empty.grf	
Fichero GRF con la parte móvil de la escena			
FS_Passes	entero	1	
Número de pasadas a realizar			

Cuadro B.2: FS: Opciones relacionadas con la fotosimulación.

Opción	Tipo	Valor por defecto
Descripción		
DE_Nphotons	Entero	13
En Photon Maps: número de fotones más cercanos al punto que deben encontrarse. En DETP con Radio Variable: número de rayos a intersecar, siempre entre el radio mínimo y máximo de DETP		
DE_Radius	Double	0.1
En Photon Maps: Distancia máxima para buscar impactos de fotones. En DETP: Radio del disco		

Cuadro B.3: DE: Opciones relacionadas con la estimación de densidades.

Opción	Tipo	Por defecto	Otros valores
Descripción			
DETP_UseFixedDisc	boolean	false	
Activa o desactiva el uso de radio fijo en el método DETP			
DETP_MaxRadius	Double	DE_Radius	
En Detp con Radio Variable: distancia máxima a la que se buscarán rayos si no se han encontrado suficientes a distancia DE_Radius			
DETP_ArtifactControl	boolean	true	
Activa o desactiva el control de errores en el método de estimación de densidades DETP			
DETP_PreTest_Use	string	None	Pluecker o Plücker Moeller o Möller Badouel
Indica el tipo de pre-test de intersección a utilizar			
DETP_PlueckerEdges	Entero	3	
Indica el número de lados que tendrá el polígono utilizado por el pre-test de intersección de coordenadas de Plücker			
DETP_UseRayCache	boolean	false	
Activa o desactiva el uso de la cache de esferas en el método DETP			
DETP_RayCache_RadiusFactor	Double	0.6	
Indica la relación entre el tamaño de cada esfera y la esfera padre de la cache de esferas			
DETP_SortQueries	boolean	false	
Activa o desactiva la ordenación de los puntos donde se va a calcular la iluminación usando la curva de Lebesgue			
DETP_IrradianceSamplesOnGlossy	Entero	0	
Rayos a usar para resumir la irradiancia (redondeado a múltiplo de 4 por exceso) 0: todos			
DETP_Optim_Use	String	"None"	"DiscSpatialIndexing"
Indica si usar Indexación de Discos			
DETP_SortQueriesHilbert	boolean	false	
Activa o desactiva la ordenación de los puntos donde se va a calcular la iluminación usando la curva de Hilbert			
DETP_UnsortQueries	boolean	false	
Desordena aleatoriamente los puntos donde se evalúa la irradiancia (para eliminar la coherencia inherente a mallas)			
DETP_RayCache_Width	Entero	1	
Número de descendientes máximos de cada esfera			
DETP_RayCache_MaxLevels	Entero	100	
Nivel máximo de profundidad de las esferas de la cache			
DETP_UseSphereLimitedRayCache	Entero	0	1
Hace que cada esfera de la ray cache pueda tener un número arbitrario de descendientes que se van creando bajo demanda. Si se supera un valor máximo, se eliminan una (o varias) para poder crear las siguientes. Las esferas se ordenan por un valor de prioridad y en cada momento se elimina la de mayor prioridad			

Cuadro B.4: DETP: Opciones relacionadas con Estimación de Densidades en el Plano Tangente.

Opción	Tipo	Por defecto	Otros valores
Descripción			
DETP_SphereLimit	Entero	10	
Valor que indica el número máximo de esferas que pueden existir simultaneamente en la ray cache si se activa DETP_UseSphereLimitedRayCache			
DETP_RayCache_RaysMax	Entero	100	
Mínimo número de rayos en una esfera de la ray cache para que sea susceptible de tener descendientes			

Cuadro B.5: DETP: Opciones relacionadas con Estimación de Densidades en el Plano Tangente. Continuación.

Opción	Tipo	Por defecto	Otros valores
Descripción			
OPT_Optimizer	String	Octree	None, Grid3D, BSPTree, BSPTreeOld, OctreeH, Grid3DH, RGrid3D, KDTree
Indexación espacial a utilizar			
OPT_TraversalMethod	String	Parametric (Octree), Amanatides (Grid)	Gargantini
Parámetros que afectan a un Octree			
OPT_Discretization	String	Partitioning	BoundBox
Discretización de los triángulos			
Parámetros que afectan a un BSPTree.			
OPT_MaxDepth	Entero	20	
Profundidad máxima			
OPT_ModoCorte	String	XYZ (BSP Tree), MenorCoste (KD Tree)	
Parámetros que afectan a un Grid3D			
OPT_Subdivision_X	Entero	10	
OPT_Subdivision_Y	Entero	10	
OPT_Subdivision_Z	Entero	10	
Número de subdivisiones en cada eje			

Cuadro B.6: OPT: Parámetros relacionados con la indexación espacial (optimizadores gráficos).

Opción	Tipo	Valor por defecto	Otros
Descripción			
PT_Render	string	Pixel	Vertex
Metodo de rendering utilizado en el path-tracer. En Vertex, el método se asocia a la geometría y el path-tracer es llamado para cada vértice. En Pixel, se implementa el modelo de cámara y se estima el color de cada píxel en la imagen			
PT_Samples	Entero	100	
Número de muestras empleadas en la estimación de radiancia usando el algoritmo de path-tracing			
PT_Depth	Entero	4	
Profundidad máxima del path-walk en el cálculo recursivo			
PT_Viewer	Vector	Observador de GRF	
Posición del observador en la escena			
PT_VLook	Vector	(0,0,-1)	
Vector LookAt del modelo de cámara			
PT_VUp	Vector	(0,1,0)	
Vector Up del modelo de cámara			
PT_VFov	double	60	
Ángulo en grados para el parámetro FieldOfView del modelo de cámara			
PT_Render_Resolution	Vector2D	(300,300)	
Resolución de la imagen generada, en PT_Render Pixel			
PT_Render_SampleRes	Vector2D	(1,1)	
Resolución de muestreado del área del píxel, en PT_Render Pixel			
PT_Output_Image	String	snapshot.tif	
Nombre de la imagen de salida para el caso PT_Render Pixel. Se le antecede tres valores separados por _ correspondientes a los parámetros PT_Samples y PT_Render_Resolution. El usuario puede elegir la extensión entre: tga, tif y ppm			
PT_Output_File	String	ptFile.igrf	
Nombre del fichero igrf, si el método utilizado es PT_Render Vertex			
PT_Autofinish	boolean	true	
En el caso verdadero, genera el fichero de salida y termina la aplicación. En el otro caso una ventana nos muestra la imagen resultado			
PT_Pdf_Normalized	boolean	true	
En el caso verdadero, la PDF está normalizada y el albedo es siempre uno. En el otro caso el albedo cumple que es menor o igual a uno			

Cuadro B.7: BPT: Opciones relacionados con pathtracing.

Bibliografía

[Alber 03] Jochen Alber & Rolf Niedermeier. *On Multidimensional Curves with Hilbert Property.* Theory Comput. Systems. Ed. Springer-Verlag, New York., vol. 33, páginas 295–312, 2003.

[Amanatides 87] John Amanatides & Andrew Woo. *A Fast Voxel Traversal Algorithm for Ray Tracing.* En EUROGRAPHICS'87, páginas 3–10, Amsterdam, 1987.

[Aronov 02] Boris Aronov, Hervé Brönnimann, Allen Y. Chang & Yi-Jen Chiang. *Cost prediction for ray shooting.* En SCG '02, páginas 293–302. ACM Press, 2002.

[Aronov 03] Boris Aronov, Hervé Brönnimann, Allen Y. Chang & Yi-Jen Chiang. *Cost-Driven Octree Construction Schemes: An experimental Study.* En SCG '03, páginas 227–236. ACM Press, 2003.

[Arvo 86] James Richard Arvo. *Backward Ray Tracing.* En ACM SIGGRAPH '86 Course Notes - Developments in Ray Tracing, volumen 12, páginas 259–263, 1986.

[Arvo 95a] James Arvo. *The Role of Functional Analysis in Global Illumination.* En Rendering Techniques '95, páginas 115–126. Springer-Verlag, 1995.

[Arvo 95b] James Richard Arvo. *Analytic Methods for Simulated Light Transport.* Tesis Doctoral, Yale University, 1995.

[Aupperle 93] Larry Aupperle & Pat Hanrahan. *Importance and Discrete Three Point Transport.* En Michael Cohen, Claude Puech & François Sillion, editores, Rendering Techniques '93, Eurographics, páginas 85–94. Consolidation Express Bristol, 1993. Proc. 4th Eurographics Rendering Workshop, Paris, France, June 14–16, 1993.

[Badouel 90] Didier Badouel. *An Efficient Ray-Polygon Intersection.* En Andrew Glassner, editor, Graphics Gems, páginas 390–393. Academic Press, 1990.

[Benthin 03] Carsten Benthin, Ingo Wald & Philipp Slusallek. *A Scalable Approach to Interactive Global Illumination.* En Computer Graphics Forum, volumen 22. Eurographics, Blackwell, September 2003.

[Bentley 75] Jon L. Bentley. *Multidimensional binary search trees used for associative searching.* Communications of the ACM, vol. 18(9), páginas 509–517, 1975.

[Buckalew 89] Chris Buckalew & Donald Fussell. *Illumination networks: fast realistic rendering with general reflectance functions.* SIGGRAPH Comput. Graph., vol. 23, nº 3, páginas 89–98, 1989.

[Cazals 95] Frederic Cazals, George Drettakis & Claude Puech. *Filtering, Clustering and Hierarchy Construction: A New Solution for Ray-Tracing Complex Scenes.* En EUROGRAPHICS'95, páginas 371–382, 1995.

[Christensen 94] Per H. Christensen, Eric J. Stollnitz, David H. Salesin & Tony D. DeRose. *Wavelet Radiance.* En Sakas et al. [Sakas 94], páginas 295–309. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Church 36] Alonzo Church. *An Unsolvable Problem of Elementary Number Theory.* American Journal of Mathematics, vol. 58, nº 2, páginas 345–363, 1936.

[Cohen 85] Michael F. Cohen & Donald P. Greenberg. *The Hemi-Cube: A Radiosity Solution for Complex Environments.* En SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques, volumen 19, nº 3, páginas 31–40, New York, NY, USA, 1985. ACM.

[Cohen 93] Michael F. Cohen, John Wallace & Pat Hanrahan. Radiosity and realistic image synthesis. Academic Press Professional, Inc., San Diego, CA, USA, 1993.

[Cook 84] Robert L. Cook, Thomas Porter & Loren Carpenter. *Distributed ray tracing.* SIGGRAPH Comput. Graph., vol. 18, nº 3, páginas 137–145, 1984.

[David 03] Herbert A. David & Haikady N. Nagaraja. Order statistics. Wiley-Interscience, 2003.

[Dmitriev 02] Kirill Dmitriev, Stefan Brabec, Karol Myszkowski & Hans-Peter Seidel. *Interactive Global Illumination using Selective Photon Tracing.* 13^{th} Eurographics Workshop on Rendering, páginas 25–36, 2002.

[Drettakis 97] George Drettakis & François X. Sillion. *Interactive update of global illumination using a line-space hierarchy.* En SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, páginas 57–64, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[Dutré 94] Philip Dutré & Yves D. Willems. *Importance-Driven Monte Carlo Light Tracing.* En Sakas et al. [Sakas 94], páginas 188–200. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Fujimoto 86] Akira Fujimoto, Takayuki Tanaka & Kansei Iwata. *ARTS: Accelerated ray-tracing system.* IEEE Computer Graphics & Applications, vol. 6, nº 4, páginas 16–26, 1986.

[Fussel 88] Donald S. Fussel & K.R. Subramanian. *Fast Ray Tracing Using K-D Trees.* Report Interno TR-88-07, U. of Texas, Austin, Dept. Computer Science, March 1988.

[García 03] Rubén J. García, Carlos Ureña & Miguel Lastra. *LSI-2003-1 Real Time Global Illumination for Quasi-Static Scenes.* Report interno, Departamento de Lenguajes y Sistemas Informáticos. Universidad de Granada, 2003.

[García 04] Rubén J. García, Carlos Ureña, Miguel Lastra, Rosana Montes & Jorge Revelles. *Interactive Global Illumination for Quasi-Static Scenes.* En Proceedings of the CGI2004, páginas 128–131, June 2004.

[García 06] Rubén J. García, Carlos Ureña, Jorge Revelles, Miguel Lastra & Rosana Montes. *Density estimation optimizations for global illumination.* En WSCG'2006 Short Communications Proceedings, páginas 125–132, 2006.

[García 05] Rubén J. García, Carlos Ureña, Miguel Lastra, Rosana Montes & Jorge Revelles. *Optimizaciones en estimación de densidades para iluminación global.* En Proceedings of the first Spanish Conference on Informatics (CEDI 2005), Congreso Español de Informática Gráfica, páginas 33–42. CEDI'05, Thomson, September 2005.

[García 07] Rubén J. García, Carlos Ureña, Rosana Montes, Miguel Lastra & Jorge Revelles. *A study of incremental update*

of global illumination algorithms. En WSCG'2007 Short Communications Proceedings, páginas 7–14, 2007.

[Glassner 84] Andrew S. Glassner. *Space Subdivision for Fast Ray Tracing.* IEEE Computer Graphics & Applications, vol. 4(10), páginas 15–22, 1984.

[Goldsmith 87] Jeffrey Goldsmith & John Salmon. *Automatic Creation of Object Hierarchies for Ray Tracing.* Computer Graphics and Applications, IEEE, vol. 7, nº 5, páginas 14–20, May 1987.

[González 01] Pilar González & Isidro Verdú. *An Efficient Technique for Ray Tracing of a DOL-System.* En SCCG '01: Proceedings of the 17th Spring conference on Computer graphics, páginas 165–172, Washington, DC, USA, 2001. IEEE Computer Society.

[Gortler 94] Steven Gortler, Michael F. Cohen & Philipp Slusallek. *Radiosity and Relaxation Methods: Progressive Refinement is Southwell Relaxation.* IEEE Comput. Graph. Appl., vol. 14, nº 6, páginas 48–58, 1994.

[Granier 01] Xavier Granier & George Drettakis. *Incremental Updates for Rapid Glossy Global Illumination.* En Eurographics 2001, 2001.

[Greger 98] Gene Greger, Peter Shirley, Philip M. Hubbard & Donald P. Greenberg. *The Irradiance Volume.* IEEE Computer Graphics and Applications, vol. 18(2), páginas 32–43, 1998.

[Günther 04] Johannes Günther, Ingo Wald & Peter Slusallek. *Realtime Caustics using Distributed Photon Mapping.* En Proceedings of the Eurographics Symposium on Rendering, 2004.

[Hanrahan 91] Pat Hanrahan, David Salzman & Larry Aupperle. *A rapid hierarchical radiosity algorithm.* volumen 25, páginas 197–206, New York, NY, USA, 1991. ACM.

[Havran 99] Vlastimil Havran & Filip Sixta. *Comparison of Hierarchical Grids.* Ray Tracing News, vol. 12(1), páginas 1–4, 1999.

[Havran 00a] Vlastimil Havran & Werner Purgathofer. *Comparison Methodology for Ray Shooting Algorithms.* Report interno, Czech Technical University, Vienna University of Technology, 2000.

[Havran 00b] Vlastimil Havran, Jan Přikryl & Werner Purgathofer. *Statistical Comparison of Ray-Shooting Efficiency Schemes.* Report Interno TR-186-2-00-14, Institute of Computer Graphics and Algorithms, Vienna University of Technology, may 2000.

[Havran 03] Vlastimil Havran & Werner Purgathofer. *On Comparing Ray Shooting Algorithms.* Computer and Graphics, vol. 27, Issue 4, páginas 593–604, August 2003.

[Havran 04] Vlastimil Havran, Jiří Bittner & Hans-Peter Seidel. *Ray maps for global illumination.* En SIGGRAPH '04: ACM SIGGRAPH 2004 Sketches, página 77, New York, NY, USA, 2004. ACM.

[Havran 05a] Vlastimil Havran, Jiří Bittner, Robert Herzog & Hans-Peter Seidel. *Ray Maps for Global Illumination.* 16th Eurographics Symposium on Rendering, páginas 43–54, 2005.

[Havran 05b] Vlastimil Havran, Robert Herzog & Hans-Peter Seidel. *Fast Final Gathering via Reverse Photon Mapping.* En Marc Alexa & Joe Marks, editores, The European Association for Computer Graphics 26th Annual Conference : EUROGRAPHICS 2005, volumen 24 of *Computer Graphics Forum*, páginas 323–333, Dublin, Ireland, August 2005. Blackwell.

[Heckbert 90] Paul S. Heckbert. *Adaptive radiosity textures for bidirectional ray tracing.* En SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques, páginas 145–154, New York, NY, USA, 1990. ACM Press.

[Hey 02] Heinrich Hey & Werner Purgathofer. *Advanced radiance Estimation for Photon Map Global Illumination.* Computer Graphics Forum, vol. 21, nº 3, páginas 541–546, 2002.

[Hungershöfer 02] Jan Hungershöfer & Jens-Michael Wierum. *On the Quality of Partitions Based on Space-Filling Curves.* En ICCS '02: Proceedings of the International Conference on Computational Science-Part III, páginas 36–45, London, UK, 2002. Springer-Verlag.

[Immel 86] David S. Immel, Michael F. Cohen & Donald P. Greenberg. *A Radiosity Method for Non-Difusse Environments.* En Computer Graphics. ACM Siggraph'86 Conference Proceedings, volumen 20(4), páginas 133–142, 1986.

[Jensen 95] Henrik Wann Jensen & Niels Jørgen Christensen. *Photon maps in bidirectional Monte Carlo ray tracing of complex objects.* Computers & Graphics, vol. 19, nº 2, páginas 215–224, Marzo 1995.

[Jensen 96] Henrik W. Jensen. *Global Illumination Using Photon Maps.* En Rendering Techniques'96, páginas 21–30. Springer-Verlag, 1996.

[Jensen 01] Henrik Wann Jensen. Realistic image synthesis using phothon mapping. AK Peters, 2001.

[Jevans 89] David Jevans & Brian Wyvill. *Adaptive Voxel Subdivision for Ray-Tracing.* En Proceedings of Graphics Interface'89, páginas 164–72, Junio 1989.

[Kajiya 86] James T. Kajiya. *The rendering equation.* En SIGGRAPH '86 Conference Proceedings, páginas 143–150, 1986.

[Keller 01] Alexander Keller & Wolfgang Heidrich. *Interleaved Sampling.* En Proceedings of the 12th Eurographics Workshop on Rendering Techniques, páginas 269–276, London, UK, 2001. Springer-Verlag.

[Klimansezewski 97] Kryzsztof S. Klimansezewski & Thomas W. Sederberg. *Faster Ray Tracing Using Adaptive Grids.* IEEE Computer Graphics and Applications, vol. 17(1), páginas 42–51, 1997.

[Lafortune 94] Eric P. Lafortune & Yves D. Willems. *A Theoretical Framework for Physically Based Rendering.* Computer Graphics Forum, vol. 13, nº 2, páginas 97–107, June 1994.

[Lastra Leidinger 04] Miguel Lastra Leidinger. *Stochastic Rendering Techniques for Complex Environments.* Tesis Doctoral, Universidad de Granada, 2004.

[Lastra 02a] Miguel Lastra & Carlos Ureña. *Density estimation on the tangent plane for radiosity.* III Jornadas Regionales de Informática Gráfica, páginas 133–145, 2002.

[Lastra 02b] Miguel Lastra, Carlos Ureña, Jorge Revelles & Rosana Montes. *A Density Estimation Technique for Radiosity.* 1st Ibero-American Symposium in Computer Graphics (SIACG'2002), 2002.

[Lastra 02c] Miguel Lastra, Carlos Ureña, Jorge Revelles & Rosana Montes. *A Particle-Path Based Method for Monte-Carlo Density Estimation.* Poster at: 13th EUROGRAPHICS Workshop on Rendering, 2002.

[Lastra 08] Miguel Lastra, Carlos Ureña, Jiří Bittner, Rubén García & Rosana Montes. *Estimación de Densidades usando GPUs.* En CEIG 2008 Congreso Español de Informática Gráfica, volumen 1, páginas 37–46. Eurographics Association, 2008.

[Lext 00] Jonas Lext, Ulf Assarsson & Tomas Möller. *BART: A benchmark for animated ray tracing.* Report interno, Dept. of Computer Engineering, Chalmers University of Technology, Goeteborg, 2000.

[MacDonald 90] J. David MacDonald & Kellog S. Booth. *Heuristics for Ray Tracing Using Space Subdivision.* The Visual Computer, vol. 6, nº 6, páginas 153–166, 1990.

[Malley 88] Thomas J. Malley. *A Shading Method for Computer Generated Images.* Master's thesis, Dept. of Computer Science, University of Utah, 1988.

[Martínez 09] Roel Martínez, Adrià Forés & Ignacio Martín. *Parallel implementation of a global line Monte Carlo radiosity.* En GRAPP 2009 Fourth International Conference on Computer Graphics Theory and Applications, páginas 164–169, 2009.

[Möller 97] Tomas Möller & Ben Trumbore. *Fast, minimum storage ray-triangle intersection.* Journal of Graphics Tools, vol. 2, nº 1, páginas 21–28, 1997.

[Montes Soldado 08] Rosana Montes Soldado. *Un algoritmo de muestreo por importancias para BRDFs arbitrarias aplicado a técnicas de Iluminación Global.* Tesis Doctoral, Universidad de Granada, 2008.

[Montes 08a] Rosana Montes, Carlos Ureña, Rubén García & Miguel Lastra. *GENERIC BRDF SAMPLING A sampling method for Global Illumination.* En Proceedings of the Third International Conference on Computer Graphics Theory and Applications, páginas 191–198, 2008.

[Montes 08b] Rosana Montes, Carlos Ureña, Miguel Lastra & Rubén García. *Un algoritmo de muestreo exacto para BRDFs arbitrarias.* En CEIG 2008 Congreso Español de Informática Gráfica, volumen 1, páginas 199–208. Eurographics Association, 2008.

[Neumann 95] László Neumann. *Monte Carlo Radiosity.* Computing, vol. 55, nº 1, páginas 23–42, 1995.

[Noguera 07] José M. Noguera & Carlos Ureña. *Un algoritmo de recorrido vectorizado para Ray-Tracing.* En Actas del XVII Congreso Español de Informática Gráfica (CEIG 2007), páginas 41–50. Thomson, 2007.

[Noguera 09] José María Noguera, Carlos Ureña & Rubén Jesús García. *A vectorized traversal algorithm for Ray Tracing.* En GRAPP 2009 Fourth International Conference on Computer Graphics Theory and Applications, páginas 58–63, 2009.

[Nyquist 28] Harry Nyquist. *Certain topics in telegraph transmission theory.* Transactions of the AIEE, vol. 47, páginas 617–644, 1928.

[Papadatos 95] Nickos Papadatos. *Maximum variance of order statistics.* Annals of the Institute of Statistical Mathematics, vol. 47, nº 1, páginas 185–193, January 1995.

[Parzen 92] Emanuel Parzen. Modern probability theory and its applications. Wiley-Interscience, wiley classics library edition, 1992.

[Pattanaik 92] Sumanta N. Pattanaik & Sudhir P. Mudur. *Computation of Global Illumination by Monte Carlo Simulation of the Particle Model of Light.* Proceedings of 3rd Eurographics Rendering Workshop, Bristol, páginas 71–83, 1992.

[Pattanaik 93] Sumanta N. Pattanaik & Sudhir P. Mudur. *Efficient potential equation solutions for global illumination computation.* Computers & Graphics, vol. 17, nº 4, páginas 387–396, 1993.

[Pattanaik 94] Sumanta N. Pattanaik & Kadi Bouattouch. *Haar Wavelet: A Solution to Global Illumination with General Surface Properties.* En Sakas et al. [Sakas 94], páginas 281–294. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Reinhard 96] Erik Reinhard, Arjan J. F. Kok & Frederik W. Jansen. *Cost Prediction in Ray Tracing.* En Rendering Techniques '96, páginas 41–50. Springer-Verlag, June 1996.

[Revelles Moreno 01] Jorge Revelles Moreno. *Formalización y evaluación de métodos de optimización para síntesis de imágenes.* Tesis Doctoral, Universidad de Granada, 2001.

[Revelles 00] Jorge Revelles, Carlos Ureña & Miguel Lastra. *An Efficient Parametric Algorithm for Octree Traversal.* Journal

of WSCG (UNION Agency-Science Press), vol. 8(2), páginas 212–219, 2000.

[Revelles 03] Jorge Revelles, Miguel Lastra, Rosana Montes & Pedro Cano. *A Formal Framework Approach for Ray-Scene Intersection Test Improvement.* En WSCG'2003, 2003.

[Rouselle 99] François Rouselle & Christophe Renaud. *Group Accelerated Shooting Methods for Radiosity.* 10^{th} Eurographics Workshop on Rendering, 1999.

[Rovira 05] Jordi Rovira, Peter Wonka, Francesc Castro & Mateu Sbert. *Point Sampling with Uniformly Distributed Lines.* En Point-Based Graphics (PBG'05) Proceedings, páginas 109–118, 2005.

[Rubin 80] Steven M. Rubin & Turner Whitted. *A Three-Dimensional Representation for Fast Rendering of Complex Scenes.* Computer & Graphics, vol. 14(3), páginas 110–116, 1980.

[Rubinstein 81] Reuven Y. Rubinstein. Simulation and the monte carlo method. John Wiley & Sons, Inc., New York, NY, USA, 1981.

[Sagan 94] Hans Sagan. Space-filling curves. Springer Verlag, 1994.

[Sakas 94] Georgios Sakas, Peter Shirley & Stefan Müller, editores. Photorealistic rendering techniques, Eurographics. Springer-Verlag Berlin Heidelberg New York, 1994. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Santaló 02] Luis Santaló. Integral geometry and geometric probability. Cambridge University Press, 2 edition, October 2002.

[Sbert 93] Mateu Sbert. *An Integral Geometry Based Method for Fast Form Factor Computation.* En Computer Graphics Forum. Eurographics '93 Conference Proceedings, páginas 409–420, 1993.

[Sbert 96a] Mateu Sbert. *The Use of Global Random Directions to compute Radiosity. Global Montecarlo Techniques.* Tesis Doctoral, Universitat Politècnica de Catalunya, 1996.

[Sbert 96b] Mateu Sbert, Xavier Pueyo, László Neumann & Werner Purgathofer. *Global multipath Monte Carlo algorithms for radiosity.* The Visual Computer, vol. 12, nº 2, páginas 47–61, 1996.

[Sbert 97a] Mateu Sbert. *Error and Complexity of Random Walk Monte Carlo Radiosity.* IEEE Transactions on Visualization and Computer Graphics, vol. 3, nº 1, páginas 23–38, 1997.

[Sbert 97b] Mateu Sbert. *Optimal Source Selection in Shooting Random Walk Monte Carlo Radiosity.* Comput. Graph. Forum, vol. 16, nº 3, páginas 301–308, 1997.

[Sbert 04] Mateu Sbert, László Szécsi & László Szirmay-Kalos. *Real-time Light Animation.* Comput. Graph. Forum, vol. 23, nº 3, páginas 291–300, 2004.

[Scherson 87] Isaac D. Scherson & Elisha Caspary. *Data structures and the time complexity of ray tracing.* The Visual Computer, vol. 3, nº 4, páginas 201–213, 1987.

[Schröder 94] Peter Schröder & Pat Hanrahan. *Wavelet Methods for Radiance Computations.* En Sakas et al. [Sakas 94], páginas 310–328. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Shannon 49] Claude E. Shannon. *Communication in the Presence of Noise.* Proceedings of the IRE, vol. 37, nº 1, páginas 10–21, Jan. 1949.

[Shao 88] Min-Zhi Shao, Qun-Sheng Peng & You-Dong Liang. *A new radiosity approach by procedural refinements for realistic image sythesis.* SIGGRAPH Comput. Graph., vol. 22, nº 4, páginas 93–99, 1988.

[Shirley 92] Peter Shirley. *Time complexity of Monte Carlo radiosity.* Computers & Graphics, vol. 16, nº 1, páginas 117–120, 1992.

[Shirley 95] P. Shirley, B. Wade, P. Hubbard, D. Zadeski, B. Walter & D.P. Greenberg. *Global Illumination via Density Estimation.* En Hanrahan & Purgathofer, editores, Rendering Techniques'95, páginas 219–230. Springer-Verlag, 1995.

[Sillion 89] François X. Sillion & Claude Puech. *A general two-pass method integrating specular and diffuse reflection.* En SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques, páginas 335–344, New York, NY, USA, 1989. ACM.

[Sillion 91] François X. Sillion, James R. Arvo, Stephen H. Westin & Donald P. Greenberg. *A global illumination solution for general reflectance distributions.* En SIGGRAPH '91: Proceedings of the 18th annual conference on Computer

graphics and interactive techniques, páginas 187–196, New York, NY, USA, 1991. ACM.

[Sillion 95] François X. Sillion. *A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters.* IEEE Transactions on Visualization and Computer Graphics, vol. 1(3), páginas 240–254, 1995.

[Sipser 06] Michael Sipser. Introduction to the theory of computation. PWS Publishing Company, Boston, 2 edition, 2006.

[Smits 92] Brian E. Smits, James R. Arvo & David H. Salesin. *An importance-driven radiosity algorithm.* SIGGRAPH Comput. Graph., vol. 26, nº 2, páginas 273–282, 1992.

[Subramanian 87] K.R. Subramanian. *Fast Ray Tracing Using K-d Trees.* Tesis Doctoral, Department of Computer Sciences, The University of Texas at Austin, 1987.

[Sung 92] Kelvin Sung & Peter Shirley. *Ray Tracing with the BSP Tree.* En David Kirk, editor, Graphics Gems III, páginas 271–274. Academic Press, 1992.

[Szirmay-Kalos 98] Lázló Szirmay-Kalos & Gábor Márton. *Worst-case versus average case complexity of Ray-shooting.* Computing, vol. 61, nº 2, páginas 103–131, 1998.

[Szirmay-Kalos 99] László Szirmay-Kalos. *Monte-Carlo Global Illumination Methods State of the Art and New Developments.* En 15th Spring Conference on Computer Graphics, páginas 3–21, April 1999.

[Tawara 04] Takehiro Tawara, Karol Myszkowski & Hans-Peter Seidel. *Exploiting temporal coherence in final gathering for dynamic scenes.* Computer Graphics International, 2004. Proceedings, páginas 110–119, 16-19 June 2004.

[Teller 92] Seth J. Teller. *Computing the Antipenumbra of an Area Light Source.* En Siggraph '92, páginas 26:139–148, July 1992.

[Tobler 97] Robert F. Tobler, Alexander Wilkie, Martin Feda & Werner Purgathofer. *A Hierarchical Subdivision Algorithm for Stochastic Radiosity Methods.* En Eurographics Rendering Workshop'97. Conference Proceedings, páginas 193–204, 1997.

[Tobler 98] Robert Tobler, László Neumann, Mateu Sbert & Werner Purgathofer. *A new form factor analogy and its application to stochastic global illumination algorithms.* En Rendering Techniques '98, 1998.

[Tobler 06] Robert F. Tobler & Stefan Maierhofer. *Improved Illumination Estimation for Photon Maps in Architectural Scenes.* En WSCG 2006 Full Papers Proceedings, páginas 257–262, 2006.

[Turing 36] A. M. Turing. *On Computable Numbers, with an application to the Entscheidungsproblem.* Proc. London Math. Soc., vol. 2, nº 42, páginas 230–265, 1936.

[Ureña Almagro 98] Carlos Ureña Almagro. *Métodos de Monte-Carlo Eficientes para Iluminación Global.* Tesis Doctoral, Lenguajes y Sistemas Informáticos, Universidad de Granada, April 1998.

[Ureña 97] Carlos Ureña, Xavier Pueyo & Juan Carlos Torres. *A formalization and classification of global illumination methods.* Computers & Graphics, vol. 21, nº 2, páginas 225–236, 1997.

[Veach 94] Eric Veach & Leonidas J. Guibas. *Bidirectional Estimators for Light Transport.* En Sakas et al. [Sakas 94], páginas 145–167. Proc. 5th Eurographics Rendering Workshop, Darmstadt, Germany, June 13–15, 1994.

[Veach 95] Eric Veach & Leonidas J. Guibas. *Optimally combining sampling techniques for Monte Carlo rendering.* En SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, páginas 419–428, New York, NY, USA, 1995. ACM.

[Veach 97] Eric Veach & Leonidas J. Guibas. *Metropolis Light Transport.* En Computer Graphics. ACM Siggraph'97 Conference Proceedings, páginas 65–76, 1997.

[Veach 98] Eric Veach. *Robust monte carlo methods for light transport simulation.* Tesis Doctoral, Stanford, CA, USA, 1998. Adviser-Guibas,, Leonidas J.

[Wald 01] Ingo Wald, Philipp Slusallek, Carsten Benthin & Markus Wagner. *Interactive Rendering with Coherent Ray Tracing.* Computer Graphics Forum, vol. 3, nº 20, páginas 153–164, 2001.

[Wald 02a] Ingo Wald, Thomas Kollig, Carsten Benthin, Alexander Keller & Philipp Slusallek. *Interactive global illumination using fast ray tracing.* En EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering, páginas 15–24, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

[Wald 02b] Ingo Wald, Thomas Kollig, Carsten Benthin, Alexander Keller & Philipp Slusallek. *TR-2002-02: Interactive Global Illumination.* Report interno, Computer Graphics Group, Saarland University, 2002.

[Wald 04a] Ingo Wald. *Realtime Ray Tracing and Interactive Global Illumination.* Ph.d. thesis, Saarland University, 2004.

[Wald 04b] Ingo Wald, Johannes Günther & Peter Slusallek. *Balancing Considered Harmful – Faster Photon Mapping using the Voxel Volume Heuristic.* Computer Graphics Forum, vol. 22, nº 3, 2004.

[Wald 07] Ingo Wald, Christiaan P Gribble, Solomon Boulos & Andrew Kensler. *SIMD Ray Stream Tracing - SIMD Ray Traversal with Generalized Ray Packets and On-the-fly Re-Ordering.* Report Interno UUSCI-2007-012, 2007.

[Wallace 87] John R. Wallace, Michael F. Cohen & Donald P. Greenberg. *A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods.* SIGGRAPH Comput. Graph., vol. 21, nº 4, páginas 311–320, 1987.

[Walter 97] Bruce Walter, Philip M. Hubbard, Peter Shirley & Donald P. Greenberg. *Global Illumination Using Local Linear Density Estimation.* ACM Transactions on Graphics, vol. 16, nº 3, páginas 217–259, July 1997.

[Ward 88] George J. Ward, Francis M. Rubinstein & Robert D. Clear. *A ray tracing solution for diffuse interreflection.* En SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques, páginas 85–92, New York, NY, USA, 1988. ACM Press.

[Whang 95] Kyu-Young Whang, Ju-Won Song, Ji-Woong Chang, Ji-Yun Kim, Wan-Sup Cho, Chong-Mok Park & Il-Yeol Song. *Octree-R: An Adaptive Octree for Efficient Ray Tracing.* IEEE Transaction on Visualization and Computer Graphics, vol. 1(4), páginas 343–349, 1995.

[Yu 95] Yizhou Yu & Qunsheng Peng. *Multiresolution B-spline Radiosity.* Computer Graphics Forum, vol. 14, nº 3, páginas 285–298, August 1995. ISSN 1067-7055.

Anexo. Copia de publicaciones

A vectorized traversal algorithm for Ray Tracing

José María Noguera, Carlos Ureña, Rubén Jesús García

GRAPP 2009 Fourth International Conference on Computer Graphics Theory and Applications pp. 58-63. Lisboa, Portugal, 5-8 February, 2009 Published by INSTICC PRESS. ISBN: 978-989-8111-67-8 Depósito Legal: 285424/08 Printed in Portugal.

Indicios de calidad:

- Indexado por ISI Web of Knowledge, en el Conference Proceedings Citation Index.
- Indexado por Inspec.
- Indexado por DBLP (Digital Bibliography & Library Project)

Estimación de Densidades usando GPUs

M. Lastra, C. Ureña, J. Bittner, R. García, R. Montes

CEIG 2008 Congreso Español de Informática Gráfica pp. 37-46. Barcelona 3-5 September, 2008 Published by the Eurographics Association. ISBN: 978-3-905673-69-2

Un algoritmo de muestreo exacto para BRDFs arbitrarias

R. Montes, C. Ureña, M. Lastra, R. García

CEIG 2008 Congreso Español de Informática Gráfica pp. 199-208. Barcelona 3-5 September, 2008 Published by the Eurographics Association. ISBN: 978-3-905673-69-2

GENERIC BRDF SAMPLING A sampling method for Global Illumination

Rosana Montes, Carlos Ureña, Rubén García, Miguel Lastra
GRAPP 2008 Proceedings of the Third International Conference on Computer Graphics Theory and Applications pp. 191-198. Funchal, Madeira (Portugal) 22-25 January, 2008 Edited by José Braz, Nuno Jardim Nunes and João Madeiras Pereira. ISBN: 978-989-8111-20-3. Depósito Legal: 269138/07

Indicios de calidad:

- Indexado por ISI Web of Knowledge, en el Conference Proceedings Citation Index.
- Indexado por Inspec.
- Indexado por DBLP (Digital Bibliography & Library Project)

A study of incremental update of global illumination algorithms

R. García, C. Ureña, R. Montes, M. Lastra, J. Revelles

WSCG'2007 (15-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision) Short Communications Proceedings. pp. 7-14. Plzen (Czech Republic) 29-01-2007 a 02-02-2007. ISBN 978-80-86943-02-2 Electronic Version: In Journal of WSCG, Vol 15, 2007. Supplements, Short Papers. CD ROM - ISSN 1213-6980. On-line - ISSN 1213-6964

Indicios de calidad:

- Indexado por ISI Web of Knowledge, en el Conference Proceedings Citation Index.

Density estimation optimizations for global illumination

R. García, C. Ureña, J. Revelles, M. Lastra, R. Montes

WSCG'2006 (14-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision) Short Communications Proceedings. pp. 125-132. Plzen (Czech Republic) 30-01-2006 a 03-02-2006. ISBN 80-86943-05-4 Electronic Version: In Journal of WSCG, Vol 14, 2006. Supplements, Short Papers. CD ROM - ISSN 1213-6980. On-line - ISSN 1213-6964

Indicios de calidad:

- Indexado por ISI Web of Knowledge, en el Conference Proceedings Citation Index.

Optimizaciones en estimación de densidades para iluminación global

R. García, C. Ureña, M. Lastra, R. Montes, J. Revelles

I Congreso Español de Informática (CEDI 2005) / S3 XV Congreso Español de Informática Gráfica (CEIG 2005) pp. 33-42. Granada (Spain). 13-09-2005 a 16-09-2005. Published by Jordi Regincos, Domingo Martin (Editors). Thomson. ISBN 84-9732-431-5

Interactive Global Illumination for Quasi-Static Scenes

R.J. García, C. Ureña, M. Lastra, R. Montes, J. Revelles
CGI'2004 (Computer Graphics International) Proceedings. pp. 128-131. Crete (Greece). 16-06-2004 a 19-06-2004. Published by the IEEE Computer Society. ISBN 0-7695-2171-1, ISSN 1530-1052

Indicios de calidad:

- Aparece en el Australian Ranking of ICT Conferences con clasificación A
- Aparece en el Computer Science Conference Ranking con clasificación 0.96
- Indexado por Scopus e ISI Web of Knowledge