Proyecto 2 - 2D Path Tracing

Yuen Law

Semestre I, 2020

Figura 1: https://twitter.com/fu5ha

Objetivos

1. Aplicar conceptos de algoritmos probabilísticos.
2. Conocer conceptos básicos sobre gráficos por computador.
3. Aplicar conocimientos adquiridos en optimización de algoritmos.

Descripción

Path tracing es una técnica de Monte Carlo para la creación de imágenes digitales tridimensionales. El algoritmo está basado en física óptica, por lo que trata de imitar el comportamiento real de la luz y es capaz de modelar iluminación global, tomando en cuenta fuentes de luz indirecta, como por ejemplo, luz que es reflejada por paredes y otras superficies. En general, el algoritmo crea y sigue múltiples rayos de luz que se reflejan y refractan por la escena para calcular la cantidad de luz que cada punto de a superficie visible recibe (cada pixel de la imagen). El algoritmo es recursivo, por lo que cada reflexión y refracción crean nuevos rayos de luz. Potencialmente se crearan cientos de miles de rayos, dependiendo del tamaño de la escena y de la imagen. Pero al ser Monte Carlo, en lugar de crear todos los posibles rayos, crea un número aleatorio de rayos que van en direcciones aleatorias.

Existen 2 formas para crear y seguir los rayos:

1. Crear los rayos desde las fuentes de luz y seguirlos hasta que terminen aleatoriamente en un pixel de la imagen.

2. Crear los rayos desde cada pixel y seguirlos hasta que lleguen a una fuente de luz, y desde devolverse para calcular cuánta luz llegará al pixel.

Es posible también utilizar una combinación de ambas técnicas también.

En este proyecto, ustedes deberán programar una versión bidimensional del algoritmo de path tracing, para obtener una imagen similar a la de arriba, en la que se ven varios objetos opacos y fuentes de luz.

La escena

La escena que usarán es 2D y consiste de 2 tipos de objetos: la geometría y una imagen de fondo. La geometría en la escena la usarán para determinar dónde se refleja la luz. La imagen de fondo la usarán como referencia para determinar el color de los pixeles. En la siguiente figura se muestra a la izquierda la imagen original y a la derecha, se ha combinado con segmentos que se alinean con las paredes de la habitación central. Dentro de la habitación hay dos fuentes de luz amarillas que crean sombras e iluminan con diferente intensidad los diferentes pixeles.

El algoritmo utilizado para crear esta imagen solo toma en cuenta la iluminación directa, es decir, solo se crean rayos de luz desde cada pixel hacia las fuentes de luz, sin considerar rebotes en la geometría.

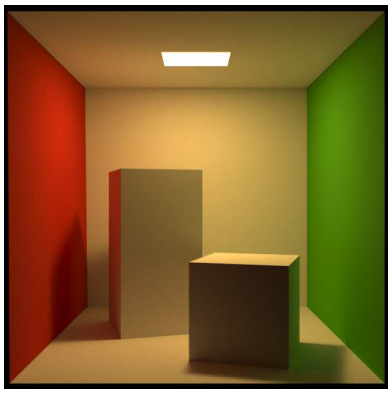

Figura 2: Color bleeding. Noten cómo el color de las paredes se refleja en los cubos.

Requerimientos funcionales

Iluminación global El algoritmo a implementar debe considerar además de la iluminación directa, al menos 1 rebote para incluir iluminación global. Además deben considerar el ´´color bleeding´´ que se da cuando la luz es reflejada por una superficie de color. Este efecto se puede apreciar en la imagen al margen. El color de la superficie (geeometría) lo pueden obtener de la imagen de referencia.

Geometría La geometría de la escena debe estar compuesta de al menos segmentos de línea recta. Además para cada segmento (o grupo de segmentos si aplica) se debe poder asignar la propiedad de especularidad. Es decir, si la superficie refleja la luz perfectamente como un espejo (especularidad=true) o si la superficie es difusa y refleja la luz en todas las direcciones (especularidad=false).

Puntos extra: incluir más tipos de geometría como círculos, o más propiedades como transparencia y refracción.

Fuentes de luz Deben incluir al menos 2 fuentes de luz puntuales (solo un punto). Las luces deben tener además intensidad y color. También deben considerar el radio de iluminación.

Puntos extra: incluir luces de diferente forma, por ejemplo, círculos o líneas.

La escena La imagen de fondo y los segmentos pueden ser fijos, pero deben ser seleccionados de forma que se puedan apreciar los efectos y optimización de su algoritmo.

Puntos extra: por creatividad.

Optimización Deben aplicar lo aprendido en clase hasta ahora, para tratar de optimizar el código y acelerar la ejecución sin afectar la calidad del resultado.

Puntos extra: si logran tiempos interactivos; poder por ejemplo mover una fuente de luz y ver el resultado inmediatamente.

Salida La imagen a reproducir debe tener una dimensión de al menos 500×500 pixeles.

Material de referencia

Pueden descargar un ejemplo básico, creado en Python. Tomen en cuenta que en el ejemplo solo se trata de un Raytracer, sin iluminación global ni ningún tipo de optimización. Este código fue utilizado para crear la figura 2 del documento. Utiliza un hilo aparte para crear la imagen, mientras el hilo principal va mostrando los resultados parciales. https://github.com/yuenlw/2DRaytracer.git

Este video explica un poco la matemática:
https://youtu.be/TOEi6T2mtHo

Entregables

1. Documentación PDF, en el formato IEEE para conferencias (máximo 8 páginas) con las siguientes secciones: Abstract, Introducción (planteamiento del problema), Trabajo relacionado, métodos (descripción de la solución), Resultados (análisis del tiempo de ejecución del algoritmo, comparado con la calidad de los resultados), Conclusión y Referencias.
2. Código en un repositorio de Git. El último commit realizado debe ser antes de la fecha y hora límite de entrega.

Aspectos administrativos

1. Fecha de entrega: viernes 26 de junio, hora: media noche
2. El pdf debe ser enviado por medio de Schoology
3. El proyecto debe realizarse en parejas
4. El proyecto puede ser desarrollado en el lenguaje/herramienta de su elección.
5. Las tareas entregadas con retraso tendrán 10 puntos menos de la nota obtenida, por cada día de retraso. El primer día cuenta a partir de los primeros 15 minutos después de la hora especificada.

Evaluación

Rubro	Valor
Documento PDF	(30)
- Abstract	2.5
- Introducción	2.5
- Trabajo relacionado	2.5
- Métodos	10
- Resultados	7.5
- Conclusión	2.5
- Referencias	2.5
Código	(70)
- Escena	10
- Propiedades de Geometría	10
- Iluminación directa	5
- Iluminación global	15
- Luces	5
- Salida	5
- Monte Carlo	10
- Otras optimizaciones	10
Total	100