PATH TRACER

INFORMÁTICA GRÁFICA 2017/2018

Escuela de Ingeniería y Arquitectura

Grado en Ingeniería Informática

Alba Clemente Villafranca – 652204

Alberto Milián Barquín - 681048

índice

[**1.-Introducción** 2](#_Toc523957116)

[**2.- Decisiones de implementación** 2](#_Toc523957117)

[**3.- Creación de la escena** 3](#_Toc523957118)

[3.1.- Cámara 3](#_Toc523957119)

[3.2.- Figuras 4](#_Toc523957120)

[**4.- Requerimientos básicos** 5](#_Toc523957121)

[4.1.- Puntos de luz 5](#_Toc523957122)

[4.2.- Luz Directa 5](#_Toc523957123)

[4.2.1.- Rayos de sombra 5](#_Toc523957124)

[4.2.2.- BRDF de Phong 5](#_Toc523957125)

[4.3.- Luz Indirecta 6](#_Toc523957126)

[4.3.1.- Condiciones de finalización de un path 6](#_Toc523957127)

[4.4.- Next Event Estimation 6](#_Toc523957128)

[4.5.- Antialiasing 6](#_Toc523957129)

[**5.- Requerimientos Opcionales** 6](#_Toc523957130)

[5.1.-Reflexión Ideal 7](#_Toc523957131)

[5.2.- Refracción 7](#_Toc523957132)

[5.2.1- Ley de Snell 7](#_Toc523957133)

[**6.- Conclusión** 7](#_Toc523957134)

# **1.-Introducción**

El presente proyecto consiste en la implementación de un Path Tracer capaz de generar (renderizar) imágenes simulando el transporte y la interacción de la luz con los objetos geométricos de una escena, así como la generación y estudio de dichas imágenes.

# **2.- Decisiones de implementación**

El lenguaje de programación elegido para la implementación del programa ha sido Java, debido al conocimiento que los componentes del equipo poseen del mismo.

Las geometrías que conforman la escena se encuentran implícitas en el código, lo cual quiere decir que para generar diferentes escenas se requiere una modificación del código directamente, no obstante, la sección de código que realiza esta función se encuentra bien señalizada mediante comentarios.

Las imágenes generadas son de formato png, se ha seleccionado este formato debido a que da una buena calidad de imagen en este caso y es compatible con la mayoría de plataformas y sistemas operativos.

Para la ejecución del Path Tracer, la línea de comendo debe seguir el siguiente esquema:

|  |
| --- |
| java -jar pathTracer.jar <antialiasing> <anchura> <altura> <nombreDeLaImagen> |

Tal que:

* <antialiasing>: es el número de rayos (o caminos, paths) que se lanzan por cada píxel de la imagen.
* <anchura> y <altura>: la anchura y altura en píxeles que tendrá la imagen generada.
* <nombreDeLaImagen>: El nombre de la imagen, sin añadir la extensión (.png), ya que eso lo hará el programa.

Un ejemplo de línea de comandos para una ejecución del programa sería:

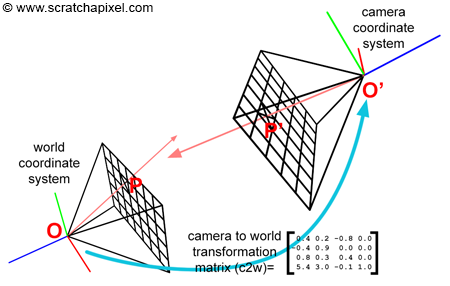
|  |
| --- |
| java -jar pathTracer.jar <100> <1080> <1080> <scene1> |

# **3.- Creación de la escena**

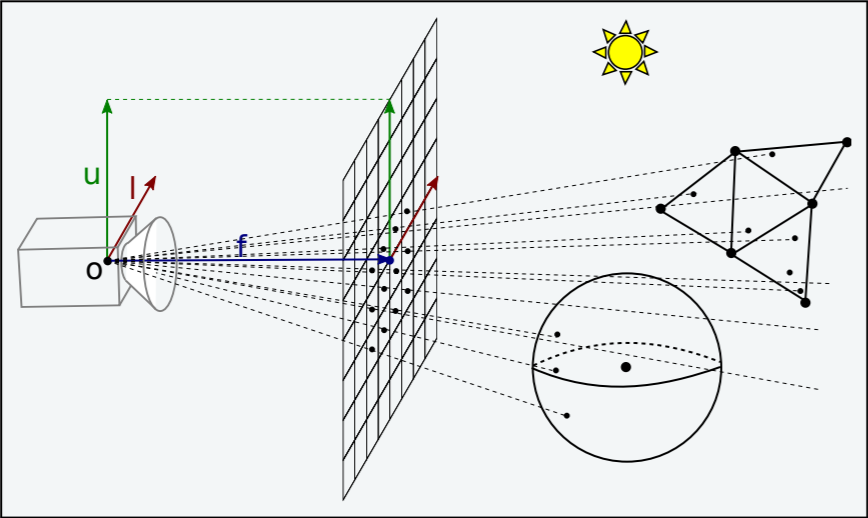
La escena esta compuesta de una serie de objetos geométricos, una cámara, los píxeles que componen la imagen y los puntos de luz que iluminan la escena.

## 3.1.- Cámara

La cámara está representada por un punto, que será el origen de los rayos (*paths*), y tres direcciones, perpendiculares entre sí, que conforman la base de coordenadas en base a la cual se generará la imagen.



Los rayos iniciales tienen como origen el pinhole de la cámara y como destino, un punto aleatorio dentro de cada píxel de la imagen.



## 3.2.- Figuras

En cuanto a los objetos geométricos, la mayoría de las escenas tienen lugar en una *Cornell Box*, que consiste en 5 planos infinitos que forman entre sí un habitáculo con uno de los lados abiertos, en el cual está situada la cámara. La pared izquierda es de color rojo, la pared derecha verde, y el resto de paredes (superior, fondo e inferior) grises.

Para la implementación de las figuras, puesto que en el Path Tracer todas las figuras tienen una clase para calcular el punto de impacto de un rayo y las características de éste según su geometría y una serie de propiedades comunes, se ha decidido utilizar herencia, creando así una clase Java genérica denominada *Figura.*

La clase figura está compuesta por los siguientes atributos:

* Centro de la figura, representado mediante un objeto de la clase Punto
* Color
* El número real t, representa la distancia desde el origen del rayo hasta el punto de intersección con la figura
* Propiedades del material del objeto, concretamente los índices de refracción y reflexión y el coeficiente difuso
* La normal de la figura, representada por un objeto del tipo Dirección.
* Los booleanos para saber si la figura es una esfera (distinta forma de calcular la normal) y si es una figura luminosa.
* El método *intersect*, que calcula el valor de t, dado un punto origen y una dirección. Este método se sobreescribe en todas las clases hijas ya que depende completamente del tipo de geometría.

El Path Tracer puede generar esferas, planos infinitosy triángulos.

//IMAGEN FIGURAS

# **4.- Requerimientos básicos**

A continuación, se analizan los requerimientos básicos que cumple este Path Tracer.

## 4.1.- Puntos de luz

Para que se produzca una imagen visible la escena necesita tener al menos una fuente luminosa, con potencia suficiente como para iluminar las figuras. Más adelante, en la sección de la luz directa, se analizará como la distancia entre un punto y otro afecta a la energía luminosa que llega a dicho punto desde el origen.

En este Path Tracer las fuentes luminosas son puntos de luz , puede haber uno o más puntos de luz en la escena.

La mayoría de las escenas han sido renderizadas con dos puntos de luz, generados aleatoriamente cada uno dentro de un triángulo. Ambos triángulos están dispuestos en el plano superior de la escena y formando un rectángulo, de tal forma que simule ser una superficie rectangular luminosa.

## 4.2.- Luz Directa

La luz directa consiste en lanzar un rayo de sombra desde el punto donde un rayo a impactado hacia cada punto de luz de la imagen. A partir de este rayo de sombra y mediante la BRDF de Phong se calcula el color correspondiente al píxel que ha sido atravesado por el rayo.

//IMAGEN LUZ DIRECTA

* Puesto que la luz es aditiva, la iluminación de un punto determinado de la escena será el resultado de la combinación del flujo luminoso de las distintas fuentes de luz.

//IMAGENES DISTINTOS PUNTOS DE LUZ

### 4.2.1.- Rayos de sombra

### 4.2.2.- BRDF de Phong

En el c

* Luz directa: Obtención de la luz directa que se observa en el píxel correspondiente, calculada utilizando la técnica de BRDF de Phong para determinar radiancia (el color).

//IMAGEN LUZ DIRECTA

## 4.3.- Luz Indirecta

* Luz indirecta: La implementación de esta iluminación permite generar el efecto denominado *color bleeding*, entre otras. Cuando un rayo impacta contra un objeto y este recibe luz directa, refracta parte de la energía recibida con el color alterado en base al color del objeto y el color del rayo que lo impacta. Si este nuevo rayo de energía llega a impactar con otro objeto puede llegar a iluminar con ese nuevo color resultante dicho objeto, siempre y cuando la distancia no sea suficientemente grande como para que la energía del rayo sea insignificante.

//IMAGEN CON LUZ INDIRECTA

//IMAGEN SOLO CON INDIRECTA

### 4.3.1.- Condiciones de finalización de un path

Si por cada rayo que impacta sobre una superficie hubiera que generar un nuevo rayo el programa no acabaría nunca, por este motivo es necesario generar una condición que limite el número de rayos generados por la luz indirecta. Esto se consigue de tres maneras:

* Ruleta Rusa
* Máximo Número de Rebotes de un rayo (path)
* Cuando el rayo no impacta con ningún objeto

#### 4.3.1.1.- Ruleta Rusa

#### 4.3.1.2.- Rebotes (bounds)

## 4.4.- Next Event Estimation

* Next Event Estimation: En el fenómeno de la luz indirecta es muy difícil que un rayo aleatorio alcance una fuente de luz. Next Event Estimation consiste en dirigir un rayo de sombra desde el punto donde un rayo intersecta con un objeto a los puntos de luz de la escena, cada vez que un rayo impacta. Esta operación se realiza en cada vez que tiene lugar una recursividad de un rayo de luz indirecta.

## 4.5.- Antialiasing

# **5.- Requerimientos Opcionales**

Adicionalmente, este Path Tracer también puede generar imágenes que contengan los efectos de reflexión ideal (espejos ideales).

## 5.1.-Reflexión Ideal

## 5.2.- Refracción

### 5.2.1- Ley de Snell

# **6.- Conclusión**