Graphics and Imaging Lab

Trazado de rayos

Bloque 2 - Computación Gráfica, Entornos Inmersivos y Multimedia

Máster en Informática

Objetivo

- Diseñar un algoritmo de trazado de rayos simple.
- Comprender y aplicar la Ecuación de Render para luz directa.
- Comprender e implementar rayos de sombra como primera aproximación a la Iluminación Global.

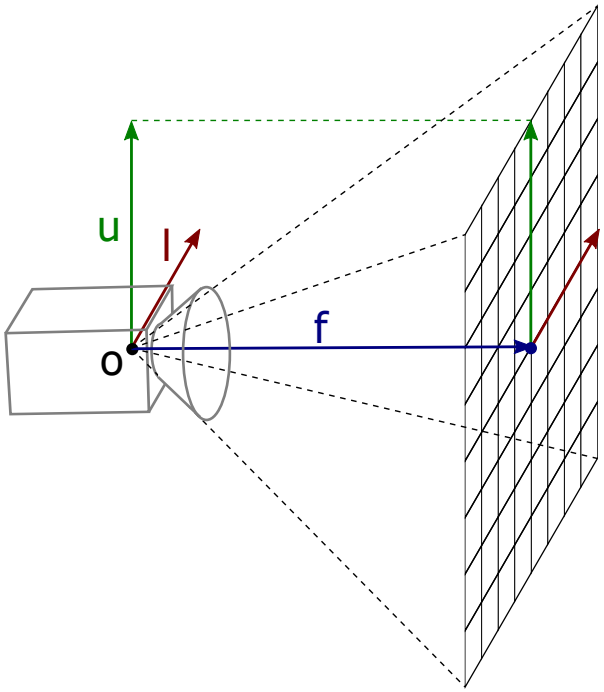
1 Problema a resolver

En las aplicaciones en tiempo real del bloque 1 de la asignatura has podido mostrar geometrías rasterizadas con *OpenGL* e iluminadas de forma simple. El resultado de dicho proceso puede ser visualmente más o menos correcto, pero se trata de iluminación local, en la que cada triángulo, cada objeto, tiene una apariencia que sólo depende de sí mismo.

En este trabajo implementarás un trazador de rayos en la que reproducirás visualizaciones similares pero añadiendo el primer elemento constituyente de una iluminación global: las sombras.

2 Camara

Para trazar rayos, se generarán desde una **cámara**. Tu modelo de cámara será un model *pinhole* modelado mediante un punto de origen y tres vectores directores perpendiculares entre sí:



Esto generará una matriz $\mathbf{M}$ de cambio de base entre coordenadas locales de la cámara a coordenadas globales del mundo (multiplicando por la izquierda):

$$\mathbf{M} = \begin{pmatrix} l_x & u_x & f_x & o_x \\ l_y & u_y & f_y & o_y \\ l_z & u_z & f_z & o_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

Dada esta matriz y un mapeo del plano de imagen con coordenadas en el eje horizontal $u \in [-1, 1]$ y en el eje vertical $v \in [-1, 1]$ se puede generar un punto $\mathbf{p}$ cualquiera en coordenadas del mundo el plano de proyección aplicando

$$\mathbf{p} = \mathbf{M} \begin{pmatrix} u \\ v \\ 1 \\ 1 \end{pmatrix} \tag{2}$$

Todos los rayos generados tendrán como origen el punto $\mathbf{o}$ de la cámara y como dirección $\mathbf{p} - \mathbf{o}$ donde $\mathbf{p}$ se genera eligiendo unas coordenadas u y v concretas. Específicamente,

lanzarás un único rayo en el centro de cada píxel (obteniendo las correspondientes u y v mediante regla de tres). Opcionalmente, podrás lanzar varios rayos aleatorios por pixel, obteniendo un efecto de *antialiasing*.

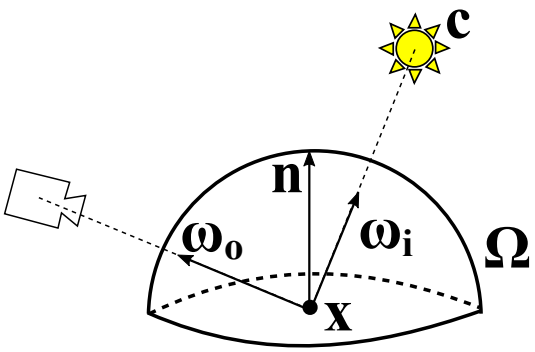
3 Intersección más cercana

Cada rayo intersectará con la geometría de la escena. Se deberá cargar un archivo `.obj` (o más, opcionalmente) con el código del bloque 1. Con ese archivo en memoria, se deberá encontrar el punto de intersección más cercano entre el rayo y el triángulo.

Para calcular la intersección con un único triángulo, se puede inspirarse en el código de `https://en.wikipedia.org/wiki/M%C3%B6ller%E2%80%93Trumbore_intersection_algorithm`, pero una vez lograda la intersección con un único triángulo habrá que recorrérselos todos buscando aquella más cercana al origen del rayo.

4 Iluminación y color

Para cada rayo se calcula la intersección más cercana, y para cada intersección más cercana hay que calcular la iluminación (color) resultante para el pixel.



Si aplicamos la Ecuación de Render para el caso particular de una única fuente de luz, tendremos

$$L_o(\mathbf{x}, \omega_\mathbf{o}) = \frac{p}{|\mathbf{c}-\mathbf{x}|^2} f_r\left(\mathbf{x}, \frac{\mathbf{c}-\mathbf{x}}{|\mathbf{c}-\mathbf{x}|}, \omega_\mathbf{o}\right) \left|\mathbf{n} \cdot \frac{\mathbf{c}-\mathbf{x}}{|\mathbf{c}-\mathbf{x}|}\right| \tag{3}$$

donde

- $\mathbf{x}$ es el punto de intersección con el rayo.
- $\mathbf{c}$ es el punto de la fuente de luz.

- $\mathbf{n}$ es la normal a la superficie en el punto de intersección $\mathbf{x}$.
- $\omega_\mathbf{o}$ es la dirección desde la que se está observando la superficie (el origen de la cámara para un rayo primario).
- p es la energía (color) de la fuente de luz.
- f_r es la BRDF que representa la apariencia del material del objeto correspondiente.
- L_o es la radiancia saliente, el color que queremos obtener como valor del pixel correspondiente.

El valor de esa ecuación sólo se considera si la luz llega a la superficie correspondiente. Para comprobarlo deberás trazar un rayo de sombra entre $\mathbf{p}$ y $\mathbf{c}$ para asegurar que no hay ningún objeto que ocluya dicha fuente de luz. Si la luz está ocluida, su contribución es nula.

Como BRDF puedes usar una BRDF muy sencilla difusa (Lambertiana) definida mediante el color del objeto k_d que sería $f_r(\mathbf{x}, \omega_\mathbf{i}, \mathbf{omega_o} = \frac{k_d}{\pi}$. Adicionalmente, para añadir reflejos especulares, puedes utilizar una BRDF de Phong, tal y como se enlaza desde Moodle.

A priori puedes fijar un único material y una única fuente de luz, pero recomendamos para apariencias más logradas varios objetos de distintos materiales y varias fuentes de luz diferentes.

5 Generación de la imagen

Guarda el resultado de la iluminación por cada píxel en una imagen en memoria conforme vayas calculando la iluminación para cada rayo, y después guarda el fichero resultante en un formato común de imágenes.

Te proporcionamos como archivo auxiliar a este trabajo `png.zip`, que contiene una cabecera muy sencillita que te permitirá cargar y guardar imágenes en formato `.png`. Deberás, para ello, tener instalada la biblioteca **libpng** en tu sistema.

Entrega

Deberás entregar tu trabajo a través de una entrega de Moodle en un único archivo comprimido `.zip`, que incluya código fuente (con instrucciones sencillas de compilación si es posible) y varias imágenes en formato `.png` generadas con tu trazador de rayos.